LONDON METROPOLITAN UNIVERSITY

Islington college
(इस्लिङ्टन कलेज)

CC5051NI Databases

100% Individual Coursework

Autumn 2024

Credit: 15 Semester Long Module

Student Name: Prabhat Lamichhane

London Met ID: 23056292

Assignment Submission Date: 1/23/2025

Word Count: 4371

I confirm that I understand my coursework needs to be submi5ed online via My Second Teacher Classroom under the relevant module page before the deadline 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.

Document 19.docx

Islington College,Nepal

Document Details

Submission ID
trn:oid:::3618:79880344

Submission Date
Jan 22, 2025, 11:48 PM GMT+5:45

Download Date
Jan 22, 2025, 11:51 PM GMT+5:45

File Name
Document 19.docx

File Size
33.9 KB

56 Pages

4,333 Words

27,718 Characters

24% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Match Groups

90 Not Cited or Quoted 23%
Matches with neither in-text citation nor quotation marks

2 Missing Quotations 0%
Matches that are still very similar to source material

0 Missing Citation 0%
Matches that have quotation marks, but no in-text citation

0 Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

Top Sources

8% Internet sources
1% Publications
20% Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

Table of Contents

Table of Figure:

Table of Table:

1. Introduction

The E-Classroom Platform was created by Mary, a businesswoman, to alter the educational encounter. A place to do everything online will change the way college teachers and students interact. Using technology, Mary hoped to tackle issues that are often encountered in conventional educational establishments, such as the inability to efficiently monitor student growth, the difficulty of overseeing several courses and the absence of centralized access to materials and evaluations.

The E-Classroom Platform is a comprehensive solution aimed at meeting the numerous requirements of educational establishments. It's a one-stop shop for handling student enrolment, program management, module assignment, assessment monitoring and evaluations. Module-specific examinations, real-time performance feedback and sequential resource distribution will be key elements of the platform. Teachers can use simplified tools to manage courses, make announcements and monitor their students' progress.

A wide range of academic programs can be used with the platform's scalable and flexible architecture. Sharing modules across disciplines such as databases and programming. Administrators, instructors and students can all work together with ease in a collaborative atmosphere that facilitates a more dynamic and interesting educational ecosystem. Business regulations and operational requirements are explored in this case study to create a reliable database system that supports Mary's concept for the online learning platform.

1.1 Aims and Objectives

The goal of the E-Classroom Platform database is to create an organized system that helps manage students, teachers, programs, modules, and assessments efficiently. It aims to provide a user-friendly digital learning environment where students and teachers can interact smoothly. The database will track student enrollment, progress and results while supporting flexible course structures by linking modules to different programs. It will also ensure a clear sequence for completing module resources, enabling structured learning.

To achieve this, the system will store detailed information about students and their programs, organize modules under each program, and allow modules to be shared across programs. Teachers will be assigned to specific modules, and their announcements will be linked to relevant subjects. The database will keep records of assessments, including their titles, deadlines, and weightage, and it will store students' scores to show their performance in each module. Furthermore, it will manage module resources in a sequence that students must follow, ensuring they progress step by step. This database will support the smooth functioning of the E-Classroom Platform, making it easier to manage and deliver education effectively.

1.2 Current Business Activities and Operations

The E-Commerce stage is planned to encourage and streamline the centre academic exercises of colleges, guaranteeing that all instructive forms are overseen proficiently. This stage coordinates different components of the scholastic framework, such as understudies, program, modules, instructors and appraisals into a cohesive structure. By digitizing these operations, the stage points to improve availability, progress communication and guarantee an organized learning way for understudies.

Instructors and directors can moreover use the platform's highlights to organize assists, track understudy advances and oversee scholarly exercises more viably. Underneath are the key commerce exercises and operations that the stage underpins.

- Business Activities and Operations

⇒ Program Administration

- Colleges cover different programs such as BSc in Computing, Organizing and Mixed media, each comprising of numerous required modules.
- Programs are outlined to cater to scholarly disciplines and are organized to meet educational modules prerequisites.

⇒ Student Enrolment

- Understudies enlist in a single program and are required to total its related modules and evaluations.
- Enrolment information is followed to guarantee exact record-keeping and program arrangement.

⇒ Module Administration

- Each program comprises different modules that characterize the educational programs.
- Modules incorporate components like assets, evaluations and declarations basic for organized learning.
- A few modules such as, Programming may be shared over different programs to optimize educational modules conveyance. Þ Tracking Assessment
- Modules incorporate evaluations outlined to assess understudy execution.

- Each appraisal is connected to a particular module and incorporates points of interest such as title, weightage, due date and interesting recognizable proof. Þ Resource Allocation and Management
- Modules contain different assets such as (recordings, reports, instructional exercises) bars for conveying substance.
- Assets must be completed successively, guaranteeing dynamic and organized learning.

⇒ Performance Evaluation

- Understudy evaluations are evaluated and point by point comes about are given to reflect execution in each module.
- Comes about incorporating data such as marks gotten, adding up to marks and assessment components.

⇒ Educator Exercises

- Instructors are mindful about conveying substances, regulating evaluations and posting declarations.
- Declarations are connected to individual modules and are utilized to communicate critical upgrades or information.

1.3 Business Rule

- A student can enroll in only one program at any given time.
- Every program should include a minimum of one module.
- A module can belong to multiple programs.
- Every module must be allocated to a single teacher.
- A teacher can teach multiple modules.
- Every module must include a minimum of one assessment.
- An assessment can only be associated with one module.
- Resources need to be finished in a specified order.
- A student may have multiple results, one for each module they are enrolled in.
- Every result is associated with a single student and a specific module.
- Announcements need to be connected to a specific module.
- Completion of resource completion is necessary for each student.

1.4 Assumptions

- Students will select their program at the time of enrollment and will not change programs until the following enrollment period.
- Programs are structured with a basic curriculum that includes essential modules.
- Modules are created to be adaptable and relevant across various programs to promote interdisciplinary education.
- Every module will have an assigned teacher responsible for its delivery and assessment.
- Teachers possess the credentials to teach different subjects and can manage multiple modules at once.
- Assessments are essential for measuring student performance and understanding of the module content.
- Every assessment is uniquely designed to evaluate the learning results of one specific module.
- The educational pathway is designed to ensure that students develop understanding step by step.
- Students will receive personal performance assessments for every module they finish.
- Results are specific to the student's performance in a specific module, allowing precise monitoring of academic development.
- Announcements pertain to the module's content and activities, guaranteeing that students obtain relevant information.
- The system will track resource completion to ensure that students follow the necessary learning path and to provide feedback on their progress.

2. Identification and Creation of Entities and Attributes

The data gathered from the case study of E-Commerce Platform gives the following entities and attributes.

2.1 Student

Table 1:Data of Student in Initial ERD

S. No	Attribute Name	Data Type	Size	Constraint
1	Student Id	Number	10	Primary Key, Unique
2	Student Name	Character	15	Not Null
3	Student Address	Character	12	Not Null
4	Student Contact Number	Number	10	Not Null
5	Student Gmail	Character	25	Not Null, Unique

2.2 Module

Table 2:Data of Module in Initial ERD

S. No	Attribute Name	Data Type	Size	Constraint
1	Module Id	Number	10	Primary key, Unique
2	Time Period	Character	5	NOT NULL
3	Result	Number	3	NOT NULL
4	Announcement	Character	45	NOT NULL
5	Assessment	Character	46	NOT NULL

2.3 Teacher

Table 3:Data of Teacher in Initial ERD

S. No	Attribute Name	Data Type	Size	Constraint
1	Teacher Id	Number	7	Primary key, Unique
2	Teacher Name	Character	13	NOT NULL
3	Teacher Address	Character	17	NOT NULL
4	Teacher Contact Number	Number	10	NOT NULL
5	Teacher Gmail	Character	22	Not Null, Unique

2.4 Program

Table 4:Data of Program in Initial ERD

S. No	Attribute Name	Data Type	Size	Constraint
1	Program Id	Number, Character	9	Primary key, Unique
2	Program Name	Character	15	NOT NULL
3	Time Duration	Number	5	NOT NULL

3. Initial ERD

An entity relationship diagram (ERD) visually depicts the structure of a database, illustrating how entities, attributes, and relationships are linked. It helps with designing databases by illustrating the data and their interactions in a clear and logical way. Entities represent real world objects while attributes describe their properties and relationships and how entities are linked. ERD are essential for planning databases, ensuring efficient data organization and maintaining consistency. They are widely used in database design and management processes. (GeeksforGeeks, 2025)

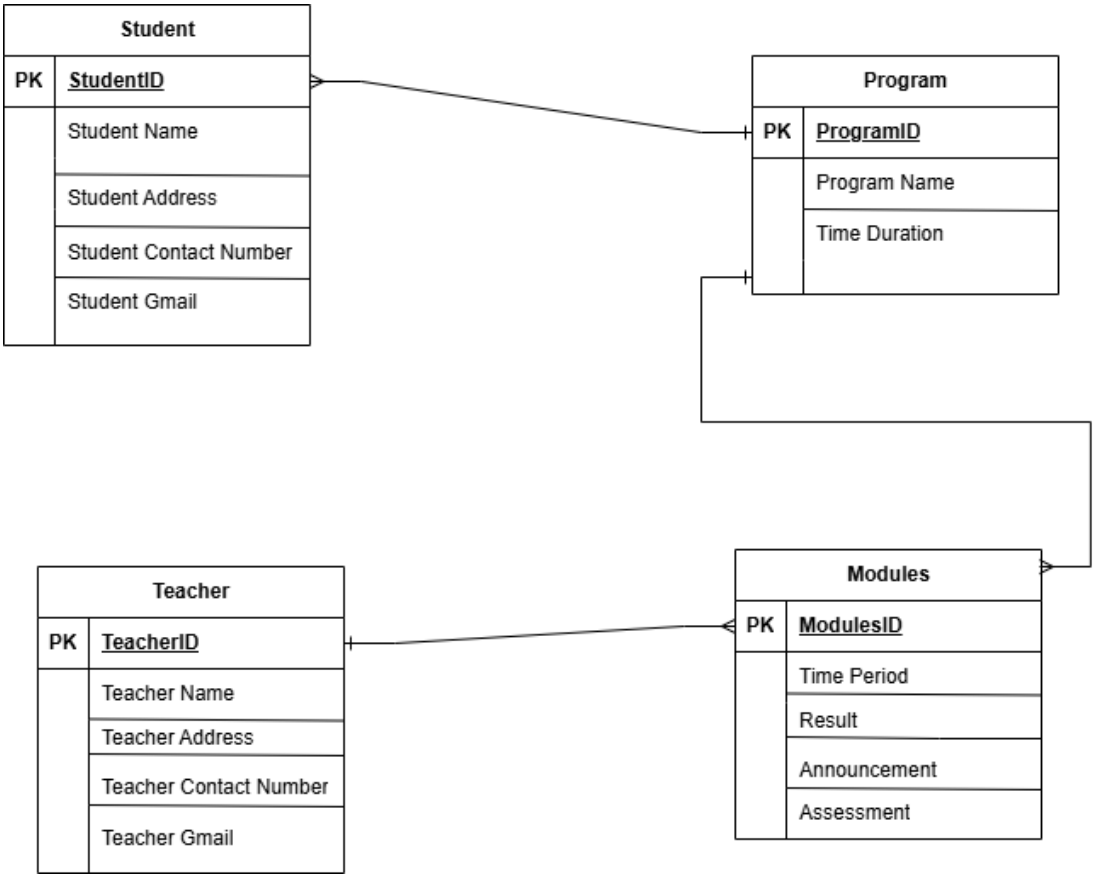

Figure 1:Initial ERD

4. Normalization

A database design technique called normalisation which eliminates unnecessary data and undesirable attributes including insertion, update, and deletion errors. Relationships connect larger tables together and normalisation rules separate them into smaller tables. Normalisation in SQL prevents duplicate or redundant data and ensures that data is saved correctly. (Learn, 2024)

4.1 UNF

A table in unnormalized form have arrays or repeating groups, which store several items in a single column. Inefficiencies and abnormalities in the data may result from this arrangement. By removing repetitive groupings, UNF tables are intended to become more structured.

Student Table :(Student ID, StudentName, ProgramID, ProgramName, {ModuleID, ModuleName {ResourceID, ResourceTitle, Type, Duration, Sequence}, {AssessmentID, AssessmentTitle, Deadline, Weightage, Total Mark}, {TeacherID, TeacherName, TeacherEmail, {AnnouncementID, AnnouncemetTitle, AnnouncementDate}}}).

4.2 1NF

In the First Normal Form (1NF), we verify that every table has unique rows and that every column contains atomic values:

1. **No Repeating Groups:** Eliminate nested or duplicate data and generate distinct rows or tables for them.
2. **Atomic Values:** Confirm that each column contains one, undivided value.
3. **Separate Tables for Entities:** Divide information into tables for every entity (e.g., Students, Modules, Assessments).
4. **Primary Keys:** Allocate distinct identifiers to every table for unambiguous recognition.
5. **Relationships:** Utilize foreign keys to connect associated tables.
   Here, all the condition want to follows:
   - **Student:** (StudentID, StudentName, ProgramID, ProgramName,).

- **Module:** (ModuleID, StudentID*, ModuleName).
- **Resource:** (ResourceID, ModuleID*, StudentID*, ResourceTitle, Type, Duration, Sequence).
- **Assessment:** (AssessmentID, ModuleID*, StudentID*, AssessmentTitle, Deadline, Weightage, Total Mark).
- **Teacher:** (TeacherID, StudentID*, ModuleID*, TeacherName, TeacherEmail).
- **Announcement:** (StudentID, ModuleID*, TeacherID*, AnnouncementID*, AnnouncemetTitle, AnnouncementDate).

4.3 2NF

Must be in 1NF Before Second Normal Form (2NF), we eliminate partial dependencies, where non-prime attributes depend solely on a portion of the composite key.

1. **Identify Primary Keys:** Locate the primary keys, particularly composite keys (multiple columns) within each table.
2. **Identify Partial Dependencies:** Determine if any non-prime attribute relies on a portion of a composite key.
3. **Create New Tables:** Transfer attributes that have partial dependencies to separate tables. Utilize the segment of the composite key they rely on as the main key.
4. **Update Original Tables:** Eliminate the partially dependent attributes from the original tables and introduce foreign keys to connect the tables.
5. **Repeat for All Tables:** Carry out the identical actions for each table containing composite keys.
6. **Verify:** Confirm that all non-prime attributes are completely reliant on the whole primary key.

➢ Here, we need follows all steps:

1. **Student:** (StudentID, StudentName, ProgramID, ProgramName).

▪ **Module:** (ModuleID, StudentID, ModuleName).

ModuleID, StudentID→

ModuleID→ ModuleName

StudentID→

Now, from Module table:

2. **Module:** (ModuleID, ModuleName).
3. **Student_Module:** (StudentID, ModuleID).

- **Resource:** (ResourceID, ModuleID, StudentID, ResourceTitle, Type, Duration, Sequence).

  ResourceID, ModuleID, StudentID→

  ModuleID, StudentID→

  ResourceID, StudentID→

  ResourceID→ ResourceTitle, Type, Duration, Sequence.

  ModuleID→

  StudentID→

  Now, from Resource table:

4. **Resource:** (ResourceID, ResourceTitle, Type, Duration, Sequence).
5. **Resource_Module_Student:** (ResourceID, ModuleID, StudentID).

- **Assessment:** (AssessmentID, ModuleID, StudentID, AssessmentTitle, Deadline, Weightage, Total Mark).

  AssessmentID, ModuleID, StudentID→

  AssessmentID, ModuleID→

  AssessmentID, StudentID→

  ModuleID, StudentID→

  AssessmentID→ AssessmentTitle, Deadline, Weightage, Total Mark.

  ModuleID→

  StudentID→

  Now, from Assessment table:

6. **Assessment:** (AssessmentID, AssessmentTitle, Deadline, Weightage, Total Mark).
7. **Assessment_Module_Student:** (AssessmentID, ModuleID, StudentID).

- **Teacher:** (TeacherID, StudentID, ModuleID, TeacherName, TeacherEmail).

TeacherID, StudentID, ModuleID→

TeacherID, StudentID →

TeacherID, ModuleID→

StudentID, ModuleID→

TeacherID→ TeacherName, TeacherEmail

StudentID→

ModuleID→

Now, from Teacher table:

8. **Teacher:** (TeacherID, TeacherName, TeacherEmail).
9. **TeacherID_StudentID_ModuleID:** (TeacherID, StudentID, ModuleID).

- **Announcement:** (AnnouncementID, StudentID, ModuleID, TeacherID, AnnouncemetTitle, AnnouncementDate).

StudentID, ModuleId, TeacherID, AnnouncementID→

StudentID, ModuleId, TeacherID →

StudentID, ModuleId, AnnouncementID→

StudentID, TeacherID, AnnouncementID→

ModuleId, TeacherID, AnnouncementID→

StudentID, ModuleId→

TeacherID, AnnouncementID→

StudentID, AnnouncementID→

StudentID, TeacherID→

Announcement→ AnnouncementTitle.

ModuleID→

TeacherID→

StudentID→

Now, from Announcement table:

10. **Announcement:** (AnnouncementID, AnnouncementTitle, AnnouncementDate).
11. **Student_Module_Teacher_Announcement:** (StudentID, ModuleID, TeacherID, AnnouncementID).

4.4 3NF

We need to follow for 3NF must be in 2NF. Then,

1. **Identify Transitive Dependencies:** Verify whether non-prime attributes rely on other non-prime attributes rather than directly depending on the primary key.
2. **Remove Transitive Dependencies:** Transfer the dependent attributes to separate tables.
3. **Update Relationships:** Modify foreign keys to ensure proper relationships among tables after relocating attributes.
4. **Verify:** Make sure that every non-prime attribute relies solely on the primary key and that there are no transitive dependencies left.

Here, all the 2NF table do not have transitive dependencies, they are already in Third Normal Form (3NF).

- **Student:** (StudentID, StudentName, ProgramID, ProgramName).
  StudentID→ ProgramID→ StudentName, programName. Here, occurs the transitive dependency in **ProgramID**. So, we need to remove transitive dependency break the different table:

1. **Student**: (StudentID, StudentName, ProgramID).
2. **Program:** (ProgramID, ProgramName).
3. **Module:** (ModuleID, ModuleName).

- Module → ModuleID. Here, is no transitive dependency. So, it's already in 3NF.

4. **Student_Module:** (StudentID, ModuleID).

- Student_Module → StudentID, ModuleID. Here, is no transitive dependency. So, it's already in 3NF.

5. **Resource:** (ResourceID, ResourceTitle, Type, Duration, Sequence).

- Resource → ResourceID. Here, is no transitive dependency. So, it's already in 3NF.

6. **Resource_Module_Student:** (ResourceID, ModuleID, StudentID).

- Resource_Module_Student → ResourceID, ModuleID, StudentID. Here, is no transitive dependency. So, it's already in 3NF.

7. **Assessment:** (AssessmentID, AssessmentTitle, Deadline, Weightage, Total Mark).

- Assessment→ AsssessmentID. Here, is no transitive dependency. So, it's already in 3NF.

8. **Assessment_Module_Student:** (AssessmentID, ModuleID, StudentID).

- Assessment_Module_Student→ AssessmentID, ModuleID, StudentID. Here, is no transitive dependency. So, it's already in 3NF.

9. **Teacher:** (TeacherID, TeacherName, TeacherEmail).

- Teacher →TeacherID. Here, is no transitive dependency. So, it's already in 3NF.

10. **Teacher_Student_Module:** (TeacherID, StudentID, ModuleID).

- Teacher_Student_Module→TeacherID, StudentID, ModuleID. Here, is no transitive dependency. So, it's already in 3NF.

11. **Announcement:** (AnnouncementID, AnnouncementTitle, AnnouncementDate).

- Announcement → AnnouncementID. Here, is no transitive dependency. So, it's already in 3NF.

12. **Student_Module_Teacher_Announcement:** (StudentID, ModuleID, TeacherID, AnnouncementID).

- Student_Module_Teacher_Announcement→StudentID, ModuleID, TeacherID, AnnouncementID. Here, is no transitive dependency. So, it's already in 3NF.

5. Data Dictionary

5.1 Student

Table 5:Data of Student in Final ERD

S. No	Attribute Name	Data Type	Size	Constraint
1	Student ID	Number	12	Primary key, Unique
2	Student Name	Character	15	NOT NULL
3	Program ID	Number	13	Foreign Key

5.2 Program

Table 6:Data of Program in Final ERD

S. No	Attribute Name	Data Type	Size	Constraint
1	Program ID	Number	12	Primary Key, Unique
2	Program Name	Character	16	NOT NULL

5.3 Module

Table 7:Data of Module in Final ERD

S. No	Attribute Name	Data Type	Size	Constraint
1	Module ID	Number	9	Primary Key, Unique
2	Module Name	Character	17	NOT NULL

5.4 Student_Module

Table 8:Data of Student_Module in Final ERD

S. No	Attributes	Data Type	Size	Constraint	Composite Constraint
1	StudentID	Number	10	Foreign Key	Primary Key
2	ModuleID	Number	10	Foreign Key	

5.5 Resource

Table 9:Data of Resource in Final ERD

S. No	Attribute Name	Data Type	Size	Constraint
1	Resource ID	Number	8	Primary key, Unique
2	Title	Character	10	NOT NULL, Unique
3	Type	Character	12	NOT NULL
4	Duration	Number	3	NOT NULL
5	Sequence	Number	6	NOT NULL

5.6 Resource_Module_Student

Table 10:Data of Resource_Module_Student in Final ERD

S. No	Attribute Name	Data Type	Size	Constraint	Composite Constraint
1	ResourceID	Number	13	Foreign Key	Primary key
2	ModuleID	Number	17	Foreign Key	
3	StudentID	Number	12	Foreign Key	

5.7 Assessment

Table 11:Data of Assessment in Final ERD

S. No	Attribute Name	Data Type	Size	Constraint
1	Assessment ID	Number	9	Primary key, Unique
2	Assessment Title	Character	19	NOT NULL, Unique
3	Deadline	Number	5	NOT NULL
4	Weightage	Number	7	NOT NULL
5	Total Marks	Number	12	NOT NULL

5.8 Assessment_Module_Student

Table 12:Data of Assessment_Module_Student in Final ERD

S. No	Attribute Name	Data Type	Size	Constraint	Composite Constraint
1	AssessmentID	Number	10	Foreign key	Primary key
2	ModuleID	Number	5	Foreign key	
3	StudentID	Number	11	Foreign Key	

5.9 Teacher

Table 13:Data of Teacher in Final ERD

S. No	Attribute Name	Data Type	Size	Constraint
1	TeacherID	Number	10	Primary key
2	TeacherName	Character	15	NOT NULL
3	TeacherEmail	Character	22	NOT NULL

5.10 Teacher_Student_Module

Table 14:Data of Teacher_Student_Module in Final ERD

S.NO	Attribute Name	Data Type	Size	Constraint	Composite Constraint
1	TeacherID	Number	12	Foreign key	Primary Key
2	StudentID	Number	13	Foreign key	
3	ModuleID	Number	14	Foreign key	

5.11 Announcement

Table 15:Data of Announcement in Final ERD

S.NO	Attribute Name	Data Type	Size	Constraint
1	AnnouncementID	Number	12	Primary Key
2	AnnouncementTitle	Character	18	NOT NULL
3	AnnouncementDate	DATE	30	NOT NULL

5.12 Student_Module_Teacher_Announcement

Table 16:Data of Student_Module_Teacher_Announcement in Final ERD

S.NO	Attribute Name	Data Type	Size	Constraint	Composite Constraint
1	StudentID	Number	13	Foreign key	Primary key
2	ModuleID	Number	14	Foreign key	
3	TeacherID	Number	12	Foreign key	
4	AnnouncementID	Number	15	Foreign key	

6. Final ERD

An Entity-Relationship Diagram (ERD) is a visual representation of data that illustrates how entities such as objects or concepts relate to each other within a system. ERDs are commonly used in database design to map out the structure and relationships of data, serving as a blueprint for constructing a database. By clearly defining entities, attributes, and relationships between them. ERDs help to ensure that the database is organized efficiently and can effectively support the required operations. (Geeksforgeeks, 2024)

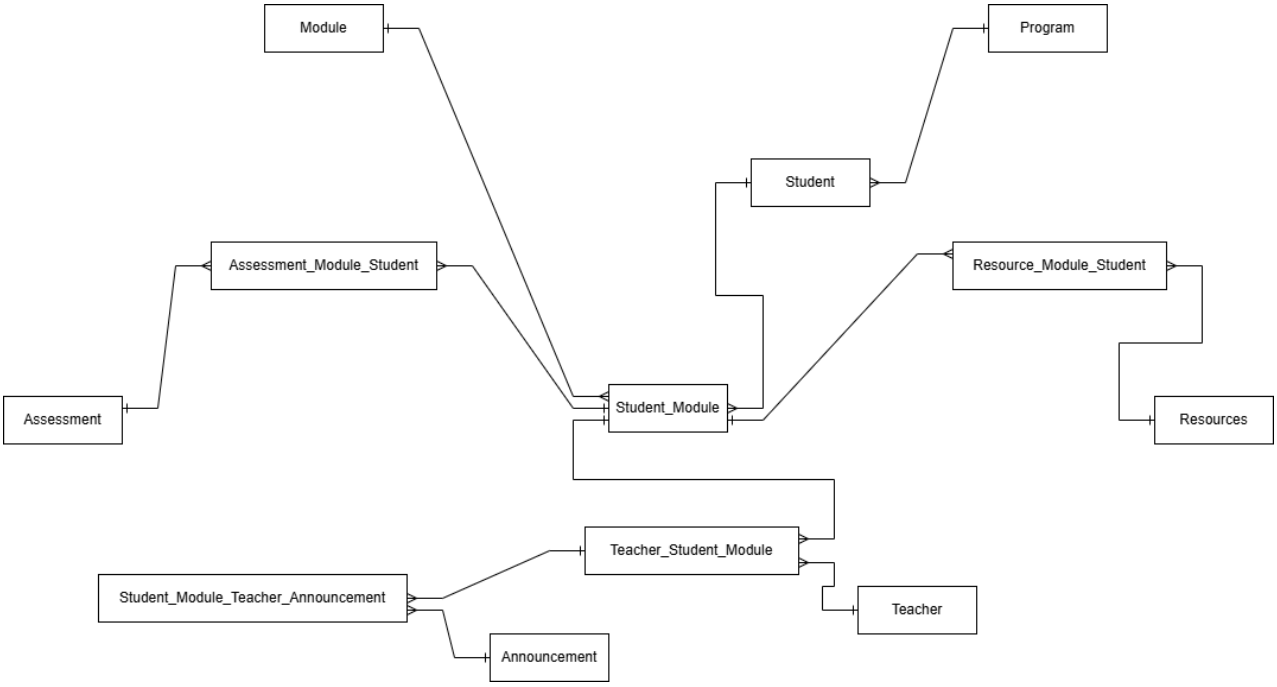

Figure 2:Final ERD

7. Implementation

7.1 Connecting System

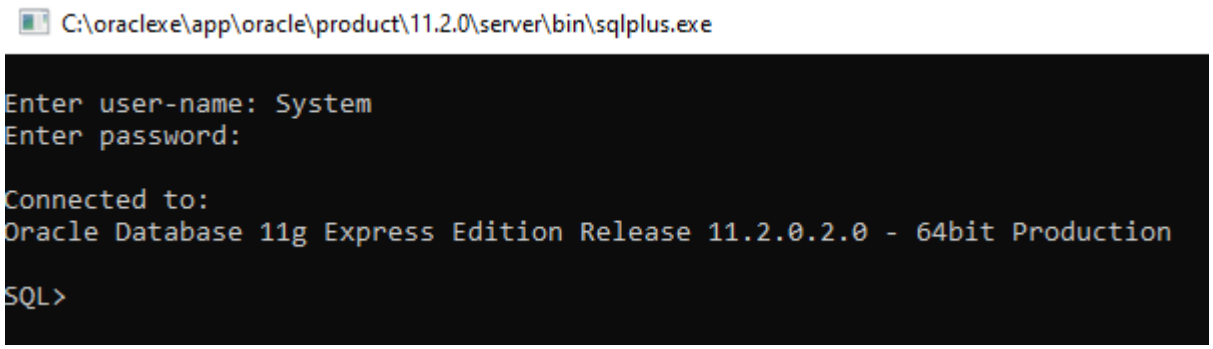

Figure 3:Connecting System

7.2 Creating User

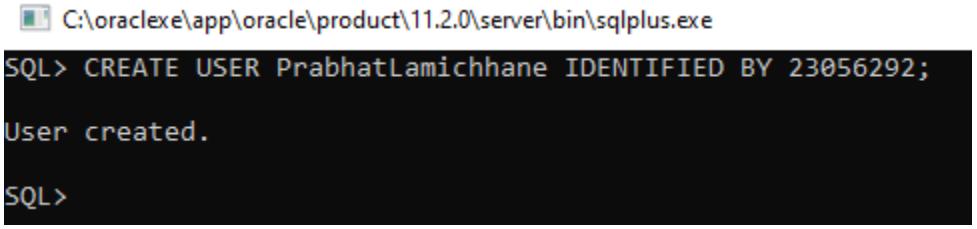

Figure 4:Creating User

7.3 Grant Connection

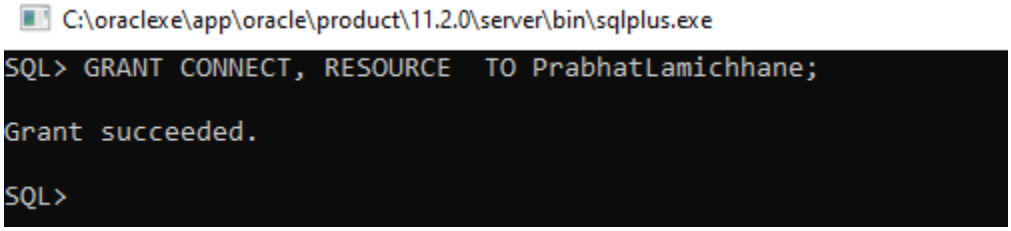

Figure 5:Grant Connection

7.4 Connecting User

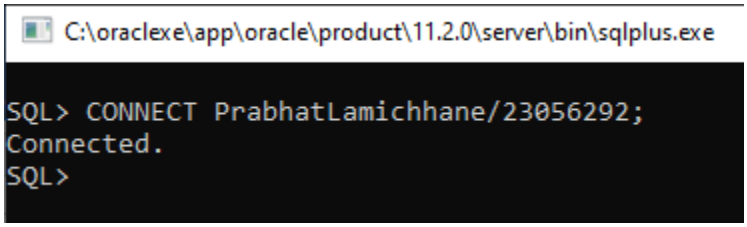

Figure 6:Connecting User

7.5 Creating Student Table

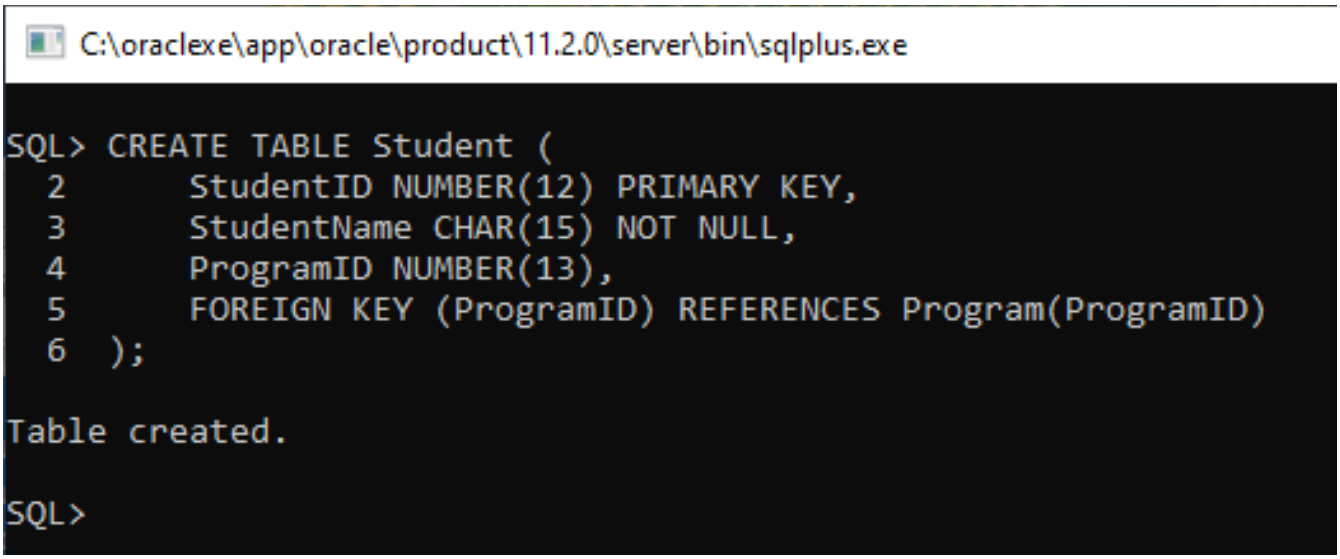

Figure 7:Creating Student Table

7.6 Describing Student

Figure 8: Describing Student

7.7 Creating Program Table

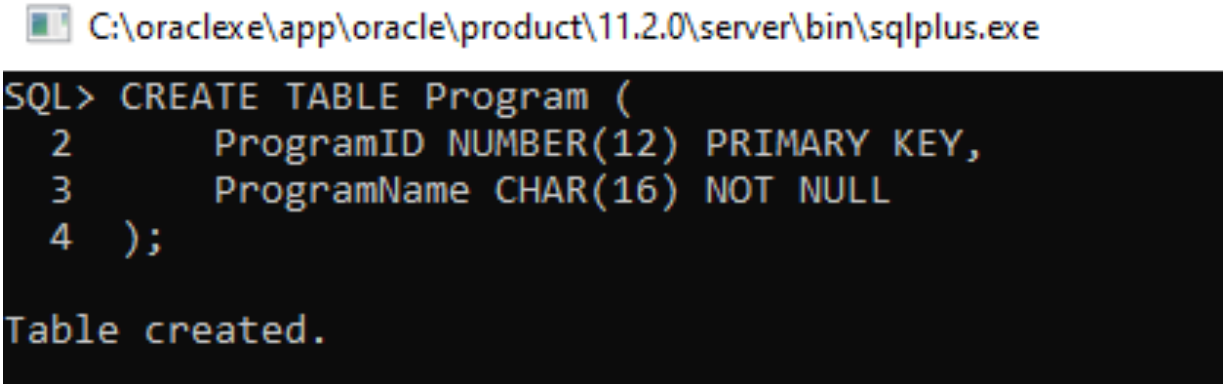

Figure 9:Creating Program Table

7.8 Describing Program

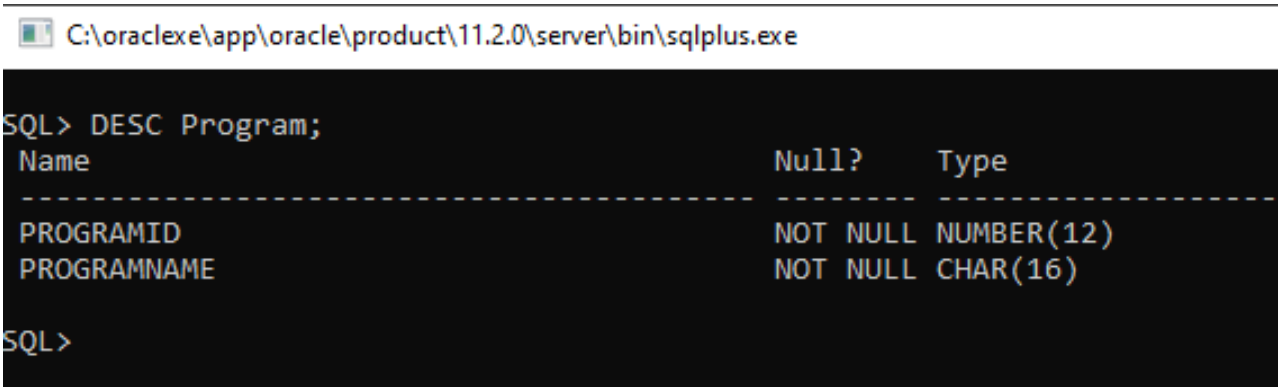

Figure 10:Describing Program

7.9 Creating Teacher Table

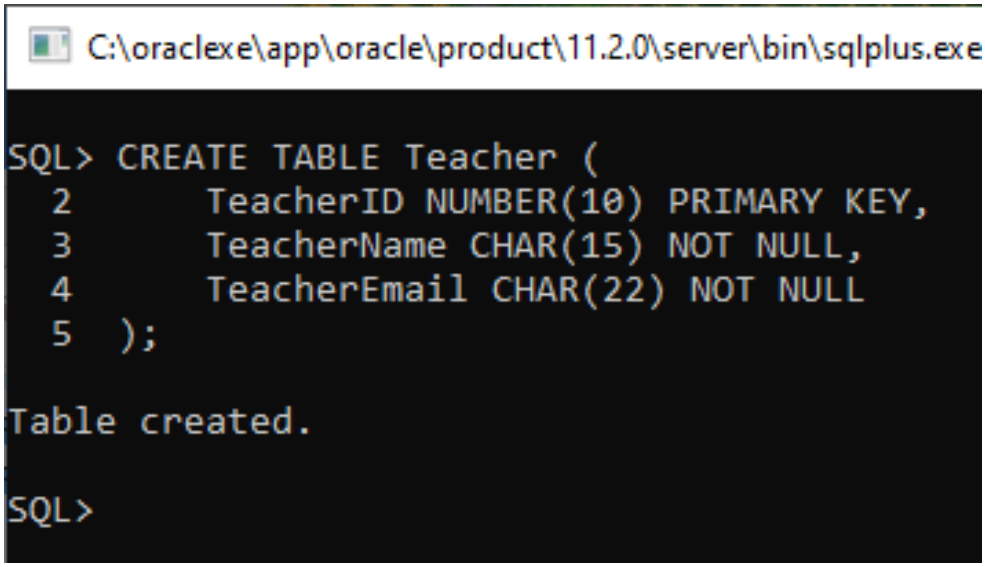

Figure 11:Creating Teacher Table

7.10 Describing Teacher

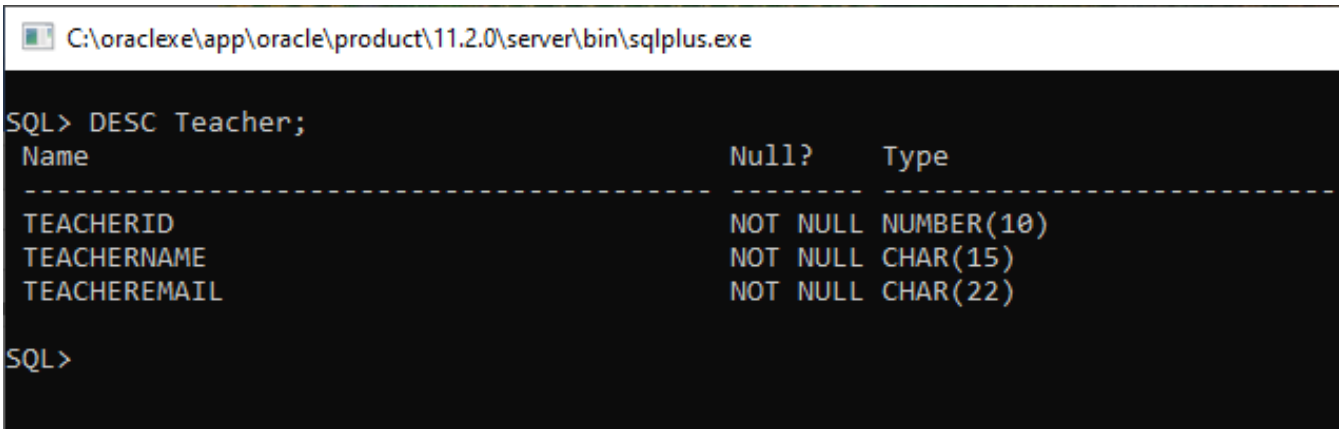

Figure 12:Describing Teacher

7.11 Creating Module Table

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> CREATE TABLE Module (
  2       ModuleID NUMBER(9) PRIMARY KEY,
  3       ModuleName CHAR(17) NOT NULL
  4   );

Table created.

SQL>
```

Figure 13:Creating Module Table

7.12 Describing Module

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> DESC Module;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 MODULEID                                  NOT NULL NUMBER(9)
 MODULENAME                                NOT NULL CHAR(17)

SQL>
```

Figure 14:Describing Module

7.13 Creating Student_Module Table

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> CREATE TABLE Student_Module (
  2       StudentID NUMBER(10),
  3       ModuleID NUMBER(10),
  4       PRIMARY KEY (StudentID, ModuleID),
  5       FOREIGN KEY (StudentID) REFERENCES Student(StudentID),
  6       FOREIGN KEY (ModuleID) REFERENCES Module(ModuleID)
  7   );

Table created.

SQL>
```

Figure 15:Creating Student_Module Table

7.14 Describing Student_Module

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe
SQL> DESC Student_Module;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------
 STUDENTID                                 NOT NULL NUMBER(10)
 MODULEID                                  NOT NULL NUMBER(10)

SQL>
```

Figure 16:Describing Student_Module

7.15 Creating Resource Table

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> CREATE TABLE Resources (
  2       ResourceID NUMBER(8) PRIMARY KEY,
  3       Title CHAR(10) NOT NULL UNIQUE,
  4       Type CHAR(12) NOT NULL,
  5       Duration NUMBER(3) NOT NULL,
  6       Sequence NUMBER(6) NOT NULL
  7  );

Table created.

SQL>
```

Figure 17:Creating Resource Table

7.16 Describing Resources

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> DESC  Resources;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------
 RESOURCEID                                NOT NULL NUMBER(8)
 TITLE                                     NOT NULL CHAR(10)
 TYPE                                      NOT NULL CHAR(12)
 DURATION                                  NOT NULL NUMBER(3)
 SEQUENCE                                  NOT NULL NUMBER(6)

SQL>
```

Figure 18:Describing Resources

7.17 Creating Resource_Module_Student Table

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> CREATE TABLE Resources_Module_Student (
  2      ResourceID NUMBER(13),
  3      ModuleID NUMBER(17),
  4      StudentID NUMBER(12),
  5      PRIMARY KEY (ResourceID),
  6      FOREIGN KEY (ResourceID) REFERENCES Resources(ResourceID),
  7      FOREIGN KEY (ModuleID) REFERENCES Module(ModuleID),
  8      FOREIGN KEY (StudentID) REFERENCES Student(StudentID)
  9  );

Table created.

SQL>
```

Figure 19:Creating Resource_Module_Student Table

7.18 Describing Resource_Module_Student

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> DESC Resources_Module_Student;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 RESOURCEID                                NOT NULL NUMBER(13)
 MODULEID                                           NUMBER(17)
 STUDENTID                                          NUMBER(12)

SQL>
```

Figure 20:Describing Resource_Module_Student

7.19 Creating Assessment Table

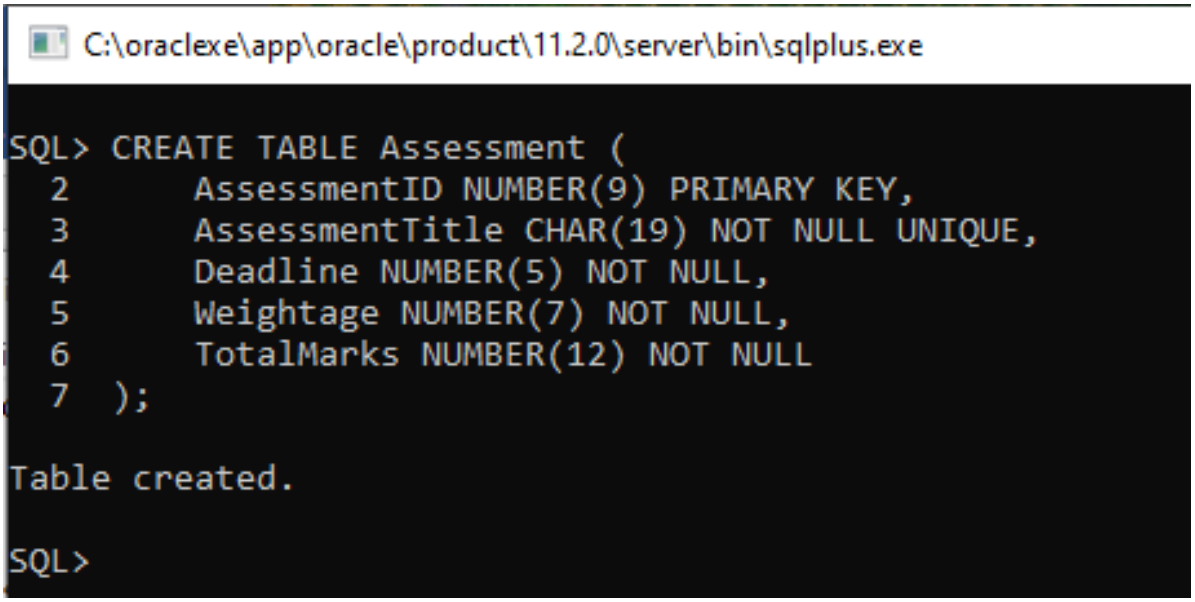

Figure 21:Creating Assessment Table

7.20 Describing Assessment Table

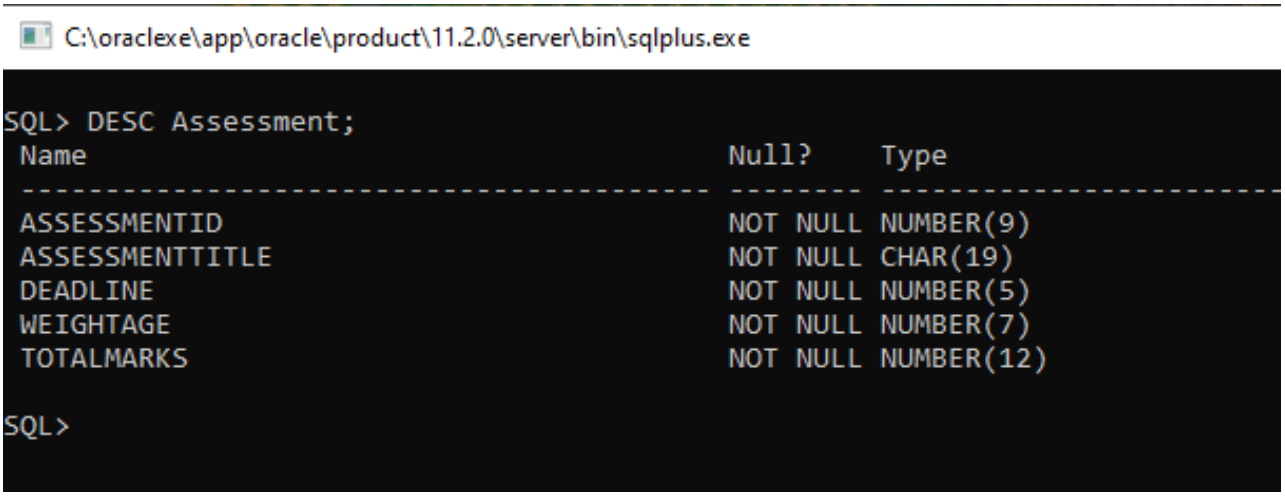

Figure 22:Describing Assessment Table

7.21 Creating Assessment_Module_Student Table

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> CREATE TABLE Assessment_Module_Student (
  2      AssessmentID NUMBER(10),
  3      ModuleID NUMBER(5),
  4      StudentID NUMBER(11),
  5      PRIMARY KEY (AssessmentID),
  6      FOREIGN KEY (AssessmentID) REFERENCES Assessment(AssessmentID),
  7      FOREIGN KEY (ModuleID) REFERENCES Module(ModuleID),
  8      FOREIGN KEY (StudentID) REFERENCES Student(StudentID)
  9  );

Table created.

SQL>
```

Figure 23:Creating Assessment_Module_Student Table

7.22 Describing Assessment_Module_Student.

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> DESC Assessment_Module_Student;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------
 ASSESSMENTID                              NOT NULL NUMBER(10)
 MODULEID                                           NUMBER(5)
 STUDENTID                                          NUMBER(11)

SQL>
```

Figure 24:Describing Assessment_Module_Student

7.23 Creating Teacher_Student_Module Table

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> CREATE TABLE Teacher_Student_Module (
  2      TeacherID NUMBER(12),
  3      StudentID NUMBER(13),
  4      ModuleID NUMBER(14),
  5      PRIMARY KEY (TeacherID),
  6      FOREIGN KEY (TeacherID) REFERENCES Teacher(TeacherID),
  7      FOREIGN KEY (StudentID) REFERENCES Student(StudentID),
  8      FOREIGN KEY (ModuleID) REFERENCES Module(ModuleID)
  9  );

Table created.

SQL>
```

Figure 25:Creating Teacher_Student_Module Table

7.24 Describing Teacher_Student_Module

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe
SQL> DESC Teacher_Student_Module;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------
 TEACHERID                                 NOT NULL NUMBER(12)
 STUDENTID                                          NUMBER(13)
 MODULEID                                           NUMBER(14)

SQL>
```

Figure 26:Describing Teacher_Student_Module

7.25 Creating Announcement Table

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> CREATE TABLE Announcement (
  2      AnnouncementID NUMBER(12) PRIMARY KEY,
  3      AnnouncementTitle CHAR(18) NOT NULL
  4  );

Table created.

SQL>
```

Figure 27:Creating Announcement Table

Figure 28:Screenshot of Altering announcement Date

7.26 Describing Announcement

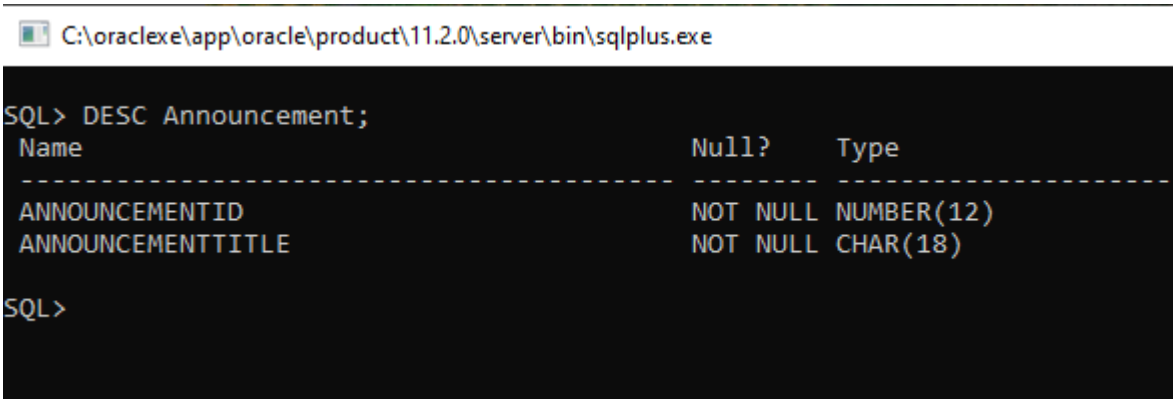

Figure 29:Describing Announcement

7.27 Creating Student_Module_Teacher_Announcement Table

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe
SQL> CREATE TABLE Stude_Modu_Teach_Announce (
  2      StudentID NUMBER(13),
  3      ModuleID NUMBER(14),
  4      TeacherID NUMBER(12),
  5      AnnouncementID NUMBER(15),
  6      PRIMARY KEY (StudentID, ModuleID, TeacherID, AnnouncementID),
  7      FOREIGN KEY (StudentID) REFERENCES Student(StudentID),
  8      FOREIGN KEY (ModuleID) REFERENCES Module(ModuleID),
  9      FOREIGN KEY (TeacherID) REFERENCES Teacher(TeacherID),
 10      FOREIGN KEY (AnnouncementID) REFERENCES Announcement(AnnouncementID)
 11  );

Table created.

SQL>
```

Figure 30:Creating Student_Module_Teacher_Announcement Table

7.28 Describing Student_Module_Teacher_Announcement

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> DESC Stude_Modu_Teach_Announce;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 STUDENTID                                 NOT NULL NUMBER(13)
 MODULEID                                  NOT NULL NUMBER(14)
 TEACHERID                                 NOT NULL NUMBER(12)
 ANNOUNCEMENTID                            NOT NULL NUMBER(15)

SQL>
```

Figure 31:Describing Student_Module_Teacher_Announcement

8. Inserting of Data

8.1 Inserting Student Data

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> INSERT INTO Student (StudentID, StudentName, ProgramID) VALUES (1, 'Prabhat', 201);
1 row created.
SQL> INSERT INTO Student (StudentID, StudentName, ProgramID) VALUES (2, 'Grish', 202);
1 row created.
SQL> INSERT INTO Student (StudentID, StudentName, ProgramID) VALUES (3, 'Susil', 203);
1 row created.
SQL> INSERT INTO Student (StudentID, StudentName, ProgramID) VALUES (4, 'Baruna', 204);
1 row created.
SQL> INSERT INTO Student (StudentID, StudentName, ProgramID) VALUES (5, 'Puja', 205);
1 row created.
SQL> INSERT INTO Student (StudentID, StudentName, ProgramID) VALUES (6, 'Karuna', 206);
1 row created.
SQL> INSERT INTO Student (StudentID, StudentName, ProgramID) VALUES (7, 'Rocky', 207);
1 row created.
SQL>
```

Figure 32:Inserting Student Data

8.2 Showing the Inserted Data of Student Detail

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> SELECT * FROM Student;

 STUDENTID STUDENTNAME      PROGRAMID
---------- --------------- ----------
         1 Prabhat                201
         2 Grish                  202
         3 Susil                  203
         4 Baruna                 204
         5 Puja                   205
         6 Karuna                 206
         7 Rocky                  207

7 rows selected.

SQL>
```

Figure 33:Showing the Inserted Data of Student Detail

8.3 Inserting Program Data

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe
SQL>
SQL> INSERT INTO Program (ProgramID, ProgramName) VALUES (201, 'Computer Eng');

1 row created.

SQL> INSERT INTO Program (ProgramID, ProgramName) VALUES (202, 'Information Tech');

1 row created.

SQL> INSERT INTO Program (ProgramID, ProgramName) VALUES (203, 'BBA');

1 row created.

SQL> INSERT INTO Program (ProgramID, ProgramName) VALUES (204, 'Cyber Security');

1 row created.

SQL> INSERT INTO Program (ProgramID, ProgramName) VALUES (205, 'MultiMedia');

1 row created.

SQL> INSERT INTO Program (ProgramID, ProgramName) VALUES (206, 'Maths');

1 row created.

SQL> INSERT INTO Program (ProgramID, ProgramName) VALUES (207, 'BSC Computing');

1 row created.

SQL>
```

Figure 34:Inserting Program Data

8.4 Showing the Inserted Data of Program Detail

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> SELECT * FROM Program;

 PROGRAMID PROGRAMNAME
---------- ----------------
       201 Computer Eng
       202 Information Tech
       203 BBA
       204 Cyber Security
       205 MultiMedia
       206 Maths
       207 BSC Computing

7 rows selected.

SQL>
```

Figure 35:Showing the Inserted Data of Program Detail

8.5 Inserting Module Data

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> INSERT INTO Module (ModuleID, ModuleName) VALUES (301, 'Databases');

1 row created.

SQL> INSERT INTO Module (ModuleID, ModuleName) VALUES (302, 'Frontend');

1 row created.

SQL> INSERT INTO Module (ModuleID, ModuleName) VALUES (303, 'Data Sci');

1 row created.

SQL> INSERT INTO Module (ModuleID, ModuleName) VALUES (304, 'Backend');

1 row created.

SQL> INSERT INTO Module (ModuleID, ModuleName) VALUES (305, 'Flutter');

1 row created.

SQL> INSERT INTO Module (ModuleID, ModuleName) VALUES (306, 'Ful Stack');

1 row created.

SQL> INSERT INTO Module (ModuleID, ModuleName) VALUES (307, 'Danjo');

1 row created.

SQL>
```

Figure 36:Inserting Module Data

8.6 Showing the Inserted Data of Module Detail

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> SELECT * FROM Module;

  MODULEID MODULENAME
---------- ----------------
       301 Databases
       302 Frontend
       303 Data Sci
       304 Backend
       305 Flutter
       306 Ful Stack
       307 Danjo

7 rows selected.

SQL>
```

Figure 37:Showing the Inserted Data of Module Detail

8.7 Inserting Student_Module Data

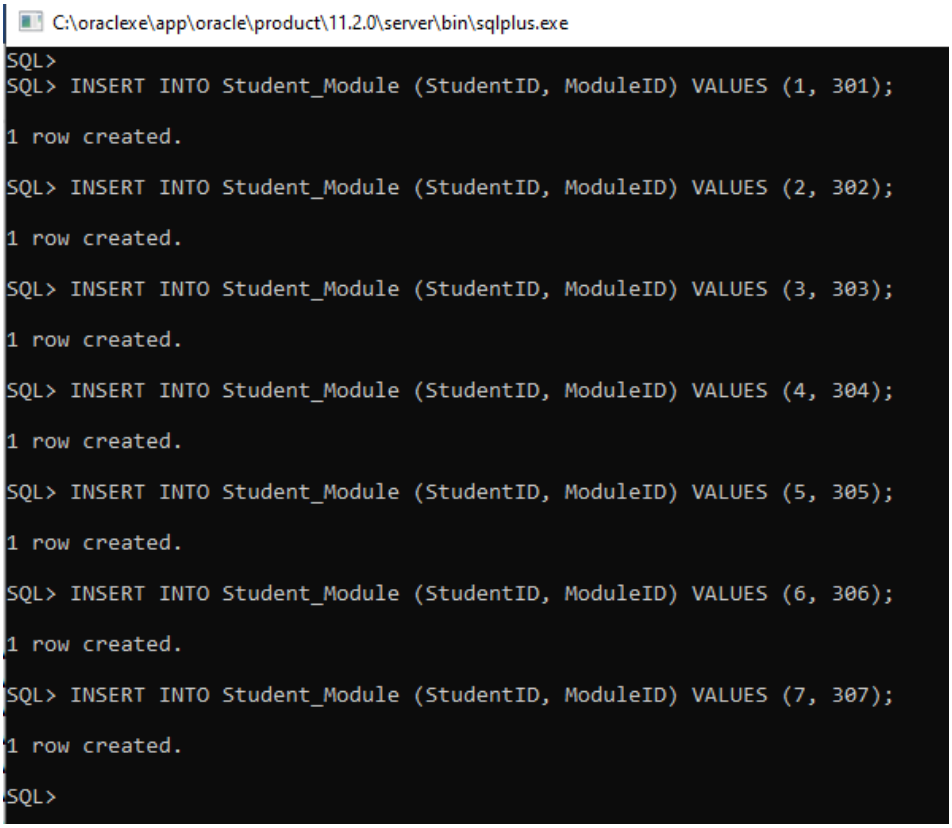

Figure 38:Inserting Student_Module Data

8.8 Showing the Inserted Data of Student_Module Detail

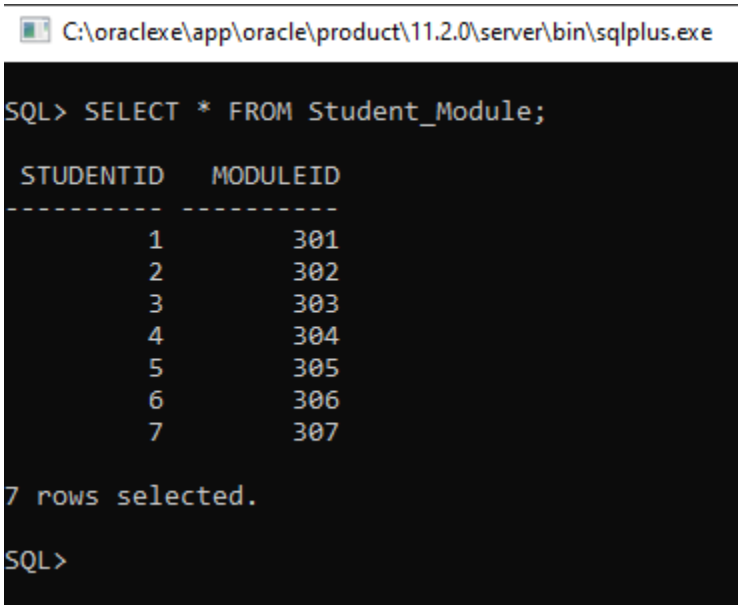

Figure 39:Showing the Inserted Data of Student_Module Detail

8.9 Inserting Resources Data

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> INSERT INTO Resources (ResourceID, Title, Type, Duration, Sequence) VALUES (401, 'Intro', 'Video', 120, 1);

1 row created.

SQL> INSERT INTO Resources (ResourceID, Title, Type, Duration, Sequence) VALUES (402, 'Basic', 'PDF', 10, 2);

1 row created.

SQL> INSERT INTO Resources (ResourceID, Title, Type, Duration, Sequence) VALUES (403, 'Hacking', 'Video', 150, 3);

1 row created.

SQL> INSERT INTO Resources (ResourceID, Title, Type, Duration, Sequence) VALUES (404, 'Quize', 'PDF', 20, 4);

1 row created.

SQL> INSERT INTO Resources (ResourceID, Title, Type, Duration, Sequence) VALUES (405, 'Javw', 'Video', 180, 5);

1 row created.

SQL> INSERT INTO Resources (ResourceID, Title, Type, Duration, Sequence) VALUES (406, 'Linux', 'PDF', 15, 6);

1 row created.

SQL> INSERT INTO Resources (ResourceID, Title, Type, Duration, Sequence) VALUES (407, 'React', 'Video', 100, 7);

1 row created.

SQL>
```

Figure 40:Inserting Resources Data

8.10 Showing the Inserted Data of Resources Detail

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> SELECT * FROM Resources;

RESOURCEID TITLE      TYPE           DURATION   SEQUENCE
---------- ---------- ------------ ---------- ----------
       401 Intro      Video               120          1
       402 Basic      PDF                  10          2
       403 Hacking    Video               150          3
       404 Quize      PDF                  20          4
       405 Javw       Video               180          5
       406 Linux      PDF                  15          6
       407 React      Video               100          7

7 rows selected.

SQL>
```

Figure 41:Showing the Inserted Data of Resources Detail

8.11 Inserting Resources_Module_Student Data

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> INSERT INTO Resources_Module_Student (ResourceID, ModuleID, StudentID) VALUES (401, 301, 1);

1 row created.

SQL> INSERT INTO Resources_Module_Student (ResourceID, ModuleID, StudentID) VALUES (402, 302, 2);

1 row created.

SQL> INSERT INTO Resources_Module_Student (ResourceID, ModuleID, StudentID) VALUES (403, 303, 3);

1 row created.

SQL> INSERT INTO Resources_Module_Student (ResourceID, ModuleID, StudentID) VALUES (404, 304, 4);

1 row created.

SQL> INSERT INTO Resources_Module_Student (ResourceID, ModuleID, StudentID) VALUES (405, 305, 5);

1 row created.

SQL> INSERT INTO Resources_Module_Student (ResourceID, ModuleID, StudentID) VALUES (406, 306, 6);

1 row created.

SQL> INSERT INTO Resources_Module_Student (ResourceID, ModuleID, StudentID) VALUES (407, 307, 7);

1 row created.

SQL>
```

Figure 42:Inserting Resources_Module_Student Data

8.12 Showing the Inserted Data of Resources_Module_Student Detail

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> SELECT * FROM Resources_Module_Student;

RESOURCEID   MODULEID  STUDENTID
---------- ---------- ----------
       401        301          1
       402        302          2
       403        303          3
       404        304          4
       405        305          5
       406        306          6
       407        307          7

7 rows selected.

SQL>
```

Figure 43: Showing the Inserted Data of Resources_Module_Student Detail

8.13 Inserting Assessment Data

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe
SQL>
SQL> INSERT INTO Assessment (AssessmentID, AssessmentTitle, Deadline, Weightage, TotalMarks) VALUES (501, 'Final Term', 30, 50, 100);

1 row created.

SQL> INSERT INTO Assessment (AssessmentID, AssessmentTitle, Deadline, Weightage, TotalMarks) VALUES (502, 'Project Sub', 60, 70, 100);

1 row created.

SQL> INSERT INTO Assessment (AssessmentID, AssessmentTitle, Deadline, Weightage, TotalMarks) VALUES (503, 'Coursework', 20, 30, 50);

1 row created.

SQL> INSERT INTO Assessment (AssessmentID, AssessmentTitle, Deadline, Weightage, TotalMarks) VALUES (504, 'Disscuss class', 10, 10, 20);

1 row created.

SQL> INSERT INTO Assessment (AssessmentID, AssessmentTitle, Deadline, Weightage, TotalMarks) VALUES (505, 'Extra class', 15, 15, 20);

1 row created.

SQL> INSERT INTO Assessment (AssessmentID, AssessmentTitle, Deadline, Weightage, TotalMarks) VALUES (506, 'Individual', 25, 40, 50);

1 row created.

SQL> INSERT INTO Assessment (AssessmentID, AssessmentTitle, Deadline, Weightage, TotalMarks) VALUES (507, 'Group Task', 90, 100, 100);

1 row created.

SQL>
```

Figure 44: Inserting Assessment Data

8.14 Showing the Inserted Data of Assessment Detail

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> SELECT * FROM Assessment;

ASSESSMENTID ASSESSMENTTITLE       DEADLINE  WEIGHTAGE TOTALMARKS
------------ -------------------- ---------- ---------- ----------
         501 Final Term                   30         50        100
         502 Project Sub                  60         70        100
         503 Coursework                   20         30         50
         504 Disscuss class               10         10         20
         505 Extra class                  15         15         20
         506 Individual                   25         40         50
         507 Group Task                   90        100        100

7 rows selected.

SQL>
```

Figure 45:Showing the Inserted Data of Assessment Detail

8.15 Inserting Teacher Data

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe
SQL>
SQL> INSERT INTO Teacher (TeacherID, TeacherName, TeacherEmail) VALUES (601, 'Ramesh', 'Ramesh@is.edu.np');

1 row created.

SQL> INSERT INTO Teacher (TeacherID, TeacherName, TeacherEmail) VALUES (602, 'Dipesh', 'Dipesh@Is.edu.np');

1 row created.

SQL> INSERT INTO Teacher (TeacherID, TeacherName, TeacherEmail) VALUES (603, 'Ashok', 'Ashok@Is.edu.np');

1 row created.

SQL> INSERT INTO Teacher (TeacherID, TeacherName, TeacherEmail) VALUES (604, 'Adesh', 'Adesh@Is.edu.np');

1 row created.

SQL> INSERT INTO Teacher (TeacherID, TeacherName, TeacherEmail) VALUES (605, 'Rakesh', 'Rakesh@Is.edu.np');

1 row created.

SQL> INSERT INTO Teacher (TeacherID, TeacherName, TeacherEmail) VALUES (606, 'Bishnu', 'Bishnu@Is.edu.np');

1 row created.

SQL> INSERT INTO Teacher (TeacherID, TeacherName, TeacherEmail) VALUES (607, 'Kamal', 'Kamal@Is.edu.np');

1 row created.

SQL>
```

Figure 46:Inserting Teacher Data

8.16 Showing the Inserted Data of Teacher Detail

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> SELECT * FROM Teacher;

 TEACHERID TEACHERNAME     TEACHEREMAIL
---------- --------------- --------------------
       601 Ramesh          Ramesh@is.edu.np
       602 Dipesh          Dipesh@Is.edu.np
       603 Ashok           Ashok@Is.edu.np
       604 Adesh           Adesh@Is.edu.np
       605 Rakesh          Rakesh@Is.edu.np
       606 Bishnu          Bishnu@Is.edu.np
       607 Kamal           Kamal@Is.edu.np

7 rows selected.

SQL>
```

Figure 47:Showing the Inserted Data of Teacher Detail

8.17 Inserting Announcement Data

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> INSERT INTO Announcement (AnnouncementID, AnnouncementTitle, AnnouncementDate) VALUES (701, 'First term', TO_DATE('20
24-03-01', 'YYYY-MM-DD'));

1 row created.

SQL> INSERT INTO Announcement (AnnouncementID, AnnouncementTitle, AnnouncementDate) VALUES (702, 'Deadline', TO_DATE('2024
-05-05', 'YYYY-MM-DD'));

1 row created.

SQL> INSERT INTO Announcement (AnnouncementID, AnnouncementTitle, AnnouncementDate) VALUES (703, 'Lab notice', TO_DATE('20
24-02-10', 'YYYY-MM-DD'));

1 row created.

SQL> INSERT INTO Announcement (AnnouncementID, AnnouncementTitle, AnnouncementDate) VALUES (704, 'Task Guidelines', TO_DAT
E('2024-05-15', 'YYYY-MM-DD'));

1 row created.

SQL> INSERT INTO Announcement (AnnouncementID, AnnouncementTitle, AnnouncementDate) VALUES (705, 'Vacation', TO_DATE('2024
-01-20', 'YYYY-MM-DD'));

1 row created.

SQL> INSERT INTO Announcement (AnnouncementID, AnnouncementTitle, AnnouncementDate) VALUES (706, 'Semester Start', TO_DATE
('2024-05-25', 'YYYY-MM-DD'));

1 row created.

SQL> INSERT INTO Announcement (AnnouncementID, AnnouncementTitle, AnnouncementDate) VALUES (707, 'Semester End', TO_DATE('
2024-04-28', 'YYYY-MM-DD'));

1 row created.

SQL>
```

Figure 48: Inserting Announcement Data

8.18 Showing the Inserted Data of Announcement Detail

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> SELECT * FROM Announcement;

ANNOUNCEMENTID ANNOUNCEMENTTITLE  ANNOUNCEM
-------------- ------------------ ---------
           701 First term         01-MAR-24
           702 Deadline           05-MAY-24
           703 Lab notice         10-FEB-24
           704 Task Guidelines    15-MAY-24
           705 Vacation           20-JAN-24
           706 Semester Start     25-MAY-24
           707 Semester End       28-APR-24

7 rows selected.

SQL>
```

Figure 49:Showing the Inserted Data of Announcement Detail

8.19 Inserting Assessment_Module_Student Data

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> INSERT INTO Assessment_Module_Student (AssessmentID, ModuleID, StudentID) VALUES (501, 301, 1);

1 row created.

SQL> INSERT INTO Assessment_Module_Student (AssessmentID, ModuleID, StudentID) VALUES (502, 302, 2);

1 row created.

SQL> INSERT INTO Assessment_Module_Student (AssessmentID, ModuleID, StudentID) VALUES (503, 303, 3);

1 row created.

SQL> INSERT INTO Assessment_Module_Student (AssessmentID, ModuleID, StudentID) VALUES (504, 304, 4);

1 row created.

SQL> INSERT INTO Assessment_Module_Student (AssessmentID, ModuleID, StudentID) VALUES (505, 305, 5);

1 row created.

SQL> INSERT INTO Assessment_Module_Student (AssessmentID, ModuleID, StudentID) VALUES (506, 306, 6);

1 row created.

SQL> INSERT INTO Assessment_Module_Student (AssessmentID, ModuleID, StudentID) VALUES (507, 307, 7);

1 row created.

SQL>
```

Figure 50:Inserting Assessment_Module_Student Data

8.20 Showing the Inserted Data of Assessment_Module_Student Detail

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> SELECT * FROM Assessment_Module_Student;

ASSESSMENTID   MODULEID  STUDENTID
------------ ---------- ----------
         501        301          1
         502        302          2
         503        303          3
         504        304          4
         505        305          5
         506        306          6
         507        307          7

7 rows selected.

SQL>
```

Figure 51:Showing the Inserted Data of Assessment_Module_Student Detail

8.21 Inserting Teacher_Student_Module Data

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> INSERT INTO Teacher_Student_Module (TeacherID, StudentID, ModuleID) VALUES (601, 1, 301);
1 row created.
SQL> INSERT INTO Teacher_Student_Module (TeacherID, StudentID, ModuleID) VALUES (602, 2, 302);
1 row created.
SQL> INSERT INTO Teacher_Student_Module (TeacherID, StudentID, ModuleID) VALUES (603, 3, 303);
1 row created.
SQL> INSERT INTO Teacher_Student_Module (TeacherID, StudentID, ModuleID) VALUES (604, 4, 304);
1 row created.
SQL> INSERT INTO Teacher_Student_Module (TeacherID, StudentID, ModuleID) VALUES (605, 5, 305);
1 row created.
SQL> INSERT INTO Teacher_Student_Module (TeacherID, StudentID, ModuleID) VALUES (606, 6, 306);
1 row created.
SQL> INSERT INTO Teacher_Student_Module (TeacherID, StudentID, ModuleID) VALUES (607, 7, 307);
1 row created.
SQL>
```

Figure 52:Inserting Teacher_Student_Module Data

8.22 Showing the Inserted Data of Teacher _Student_ Module Detail

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> SELECT * FROM Teacher_Student_Module;

 TEACHERID  STUDENTID   MODULEID
---------- ---------- ----------
       601          1        301
       602          2        302
       603          3        303
       604          4        304
       605          5        305
       606          6        306
       607          7        307

7 rows selected.

SQL>
```

Figure 53:Showing the Inserted Data of Teacher _Student_ Module Detail

8.23 Inserting Student_Module_ Teacher_Announcement Data

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe
SQL> INSERT INTO Stude_Modu_Teach_Announce (StudentID, ModuleID, TeacherID, AnnouncementID) VALUES (1, 301, 601, 701);
1 row created.
SQL> INSERT INTO Stude_Modu_Teach_Announce (StudentID, ModuleID, TeacherID, AnnouncementID) VALUES (2, 302, 602, 702);
1 row created.
SQL> INSERT INTO Stude_Modu_Teach_Announce (StudentID, ModuleID, TeacherID, AnnouncementID) VALUES (3, 303, 603, 703);
1 row created.
SQL> INSERT INTO Stude_Modu_Teach_Announce (StudentID, ModuleID, TeacherID, AnnouncementID) VALUES (4, 304, 604, 704);
1 row created.
SQL> INSERT INTO Stude_Modu_Teach_Announce (StudentID, ModuleID, TeacherID, AnnouncementID) VALUES (5, 305, 605, 705);
1 row created.
SQL> INSERT INTO Stude_Modu_Teach_Announce (StudentID, ModuleID, TeacherID, AnnouncementID) VALUES (6, 306, 606, 706);
1 row created.
SQL> INSERT INTO Stude_Modu_Teach_Announce (StudentID, ModuleID, TeacherID, AnnouncementID) VALUES (7, 307, 607, 707);
1 row created.
SQL>
```

Figure 54:Inserting Student_Module_ Teacher_Announcement Data

8.24 Showing the Inserted Data of Student_Module_Teacher_Annoucement Detail

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe
SQL>
SQL> SELECT * FROM Stude_Modu_Teach_Announce;

 STUDENTID   MODULEID  TEACHERID ANNOUNCEMENTID
---------- ---------- ---------- --------------
         1        301        601            701
         2        302        602            702
         3        303        603            703
         4        304        604            704
         5        305        605            705
         6        306        606            706
         7        307        607            707

7 rows selected.

SQL>
```

Figure 55:Showing the Inserted Data of Student_Module_Teacher_Annoucement Detail

9. Database Query

9.1 Information query

1. List the programs that are available in the college and the total number of students enrolled in each.

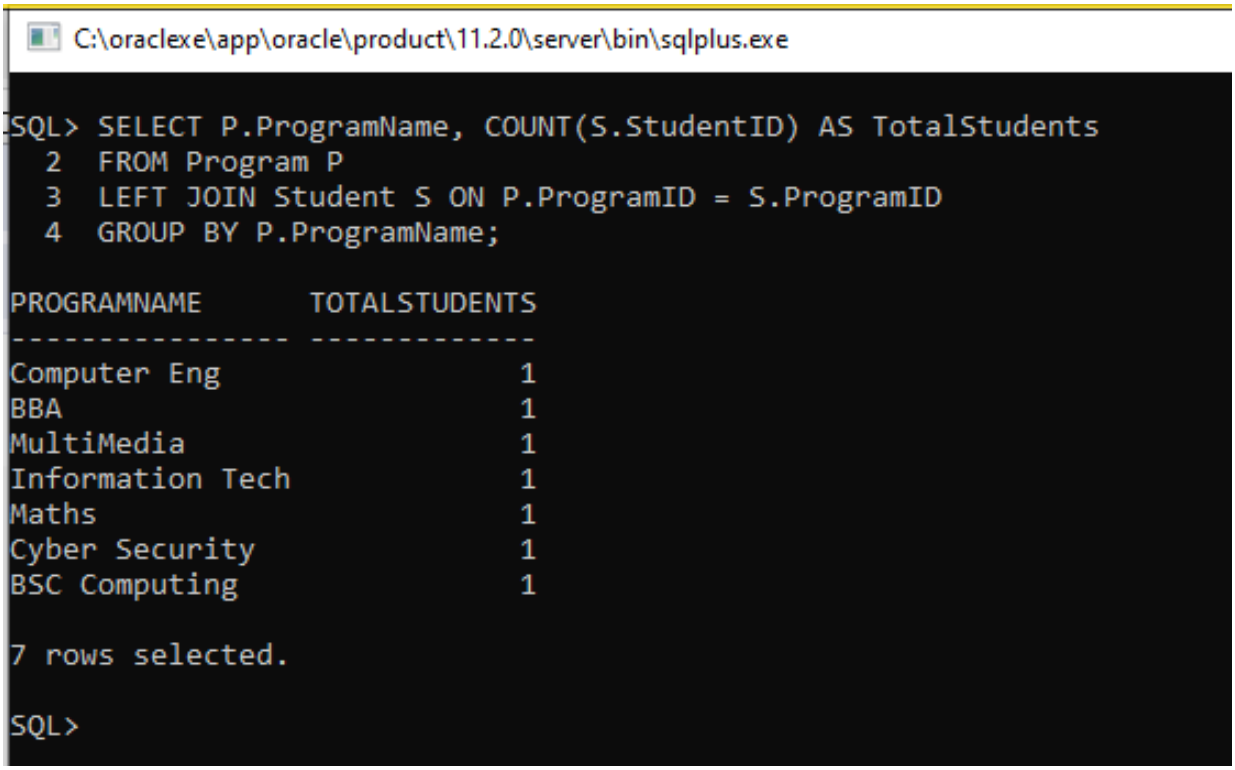

Figure 56:Screenshot of total number of students enrolled in each Program

2. List all the announcements made for a particular module starting from 1st May 2024 to 28th May 2024.

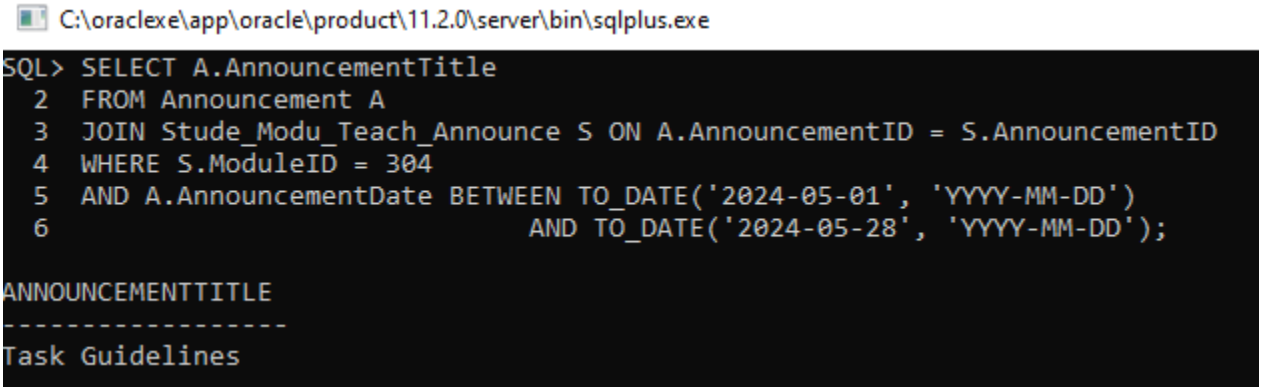

Figure 57:Screenshot of announcements made for a particular module starting from 1st May 2024 to 28th May 2024

3. List the names of all modules that begin with the letter 'D', along with the total number of resources uploaded for those modules.

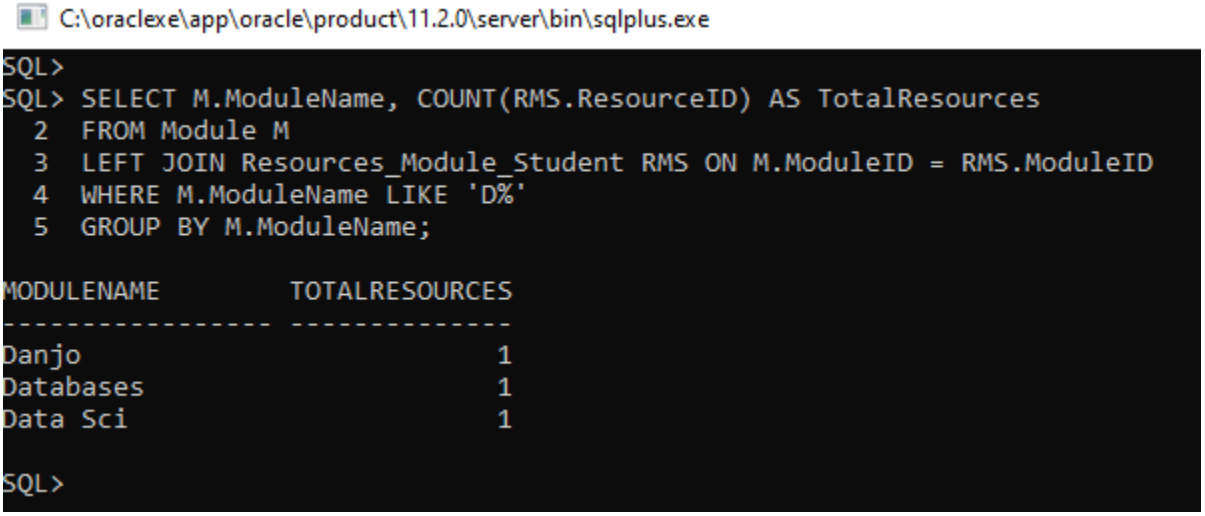

Figure 58:Screenshot of modules that begin with the letter 'D'

4. List the names of all students along with their enrolled program who have not submitted any assessments for a particular module.

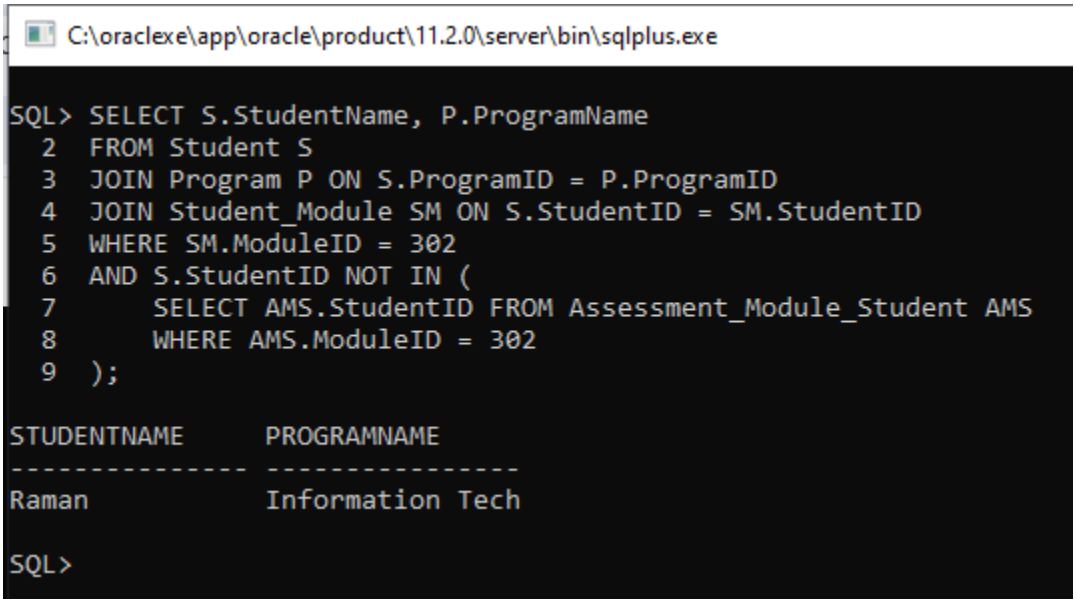

Figure 59:Screenshot of student who have not submitted any assessments for a particular module

5. List all the teachers who teach more than one module.

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> SELECT T.TeacherName, COUNT(TSM.ModuleID) AS TotalModules
  2  FROM Teacher T
  3  LEFT JOIN Teacher_Student_Module TSM ON T.TeacherID = TSM.TeacherID
  4  GROUP BY T.TeacherName
  5  HAVING COUNT(TSM.ModuleID) >= 1;

TEACHERNAME     TOTALMODULES
--------------- ------------
Ashok                      1
Rakesh                     1
Ramesh                     1
Dipesh                     1
Adesh                      1

SQL>
```

Figure 60:Screenshot of teachers who teach more than one module

9.2. Transaction query

1. Identify the module that has the latest assessment deadline.

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> SELECT M.ModuleName, A.Deadline
  2  FROM Assessment A
  3  JOIN Assessment_Module_Student AMS ON A.AssessmentID = AMS.AssessmentID
  4  JOIN Module M ON AMS.ModuleID = M.ModuleID
  5  WHERE ROWNUM = 1
  6  ORDER BY A.Deadline DESC;

MODULENAME          DEADLINE
---------------- ----------
Databases                30

SQL>
```

Figure 61:Screenshot of Identify the module that has the latest assessment deadline

2. Find the top three students who have the highest total score across all modules.

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> SELECT StudentName, TotalScore
  2  FROM (
  3      SELECT S.StudentName, SUM(A.TotalMarks) AS TotalScore
  4      FROM Student S
  5      JOIN Assessment_Module_Student AMS ON S.StudentID = AMS.StudentID
  6      JOIN Assessment A ON AMS.AssessmentID = A.AssessmentID
  7      GROUP BY S.StudentName
  8      ORDER BY TotalScore DESC
  9  )
 10  WHERE ROWNUM <= 3;

STUDENTNAME     TOTALSCORE
--------------- ----------
Rocky                  100
Prabhat                100
Grish                  100

SQL>
```

Figure 62:Screenshot of top three students who have the highest total score across all modules

3. Find the total number of assessments for each program and the average score across all assessments in those programs.

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe

SQL> SELECT P.ProgramName, COUNT(A.AssessmentID) AS TotalAssessments, AVG(A.TotalMarks) AS AvgScore
  2  FROM Program P
  3  JOIN Student S ON P.ProgramID = S.ProgramID
  4  JOIN Assessment_Module_Student AMS ON S.StudentID = AMS.StudentID
  5  JOIN Assessment A ON AMS.AssessmentID = A.AssessmentID
  6  GROUP BY P.ProgramName;

PROGRAMNAME      TOTALASSESSMENTS   AVGSCORE
---------------- ---------------- ----------
Computer Eng                    1        100
BBA                             1         50
MultiMedia                      1         20
Information Tech                1        100
Maths                           1         50
Cyber Security                  1         20
BSC Computing                   1        100

7 rows selected.

SQL>
```

Figure 63:Screenshot of total number of assessments for each program and the average score across all assessments in those programs

4. List the students who have scored above the average score in the ‘Databases’ module.

```
C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe
SQL>
SQL> SELECT S.StudentName, SUM(A.TotalMarks) AS TotalScore
  2  FROM Student S
  3  JOIN Assessment_Module_Student AMS ON S.StudentID = AMS.StudentID
  4  JOIN Assessment A ON AMS.AssessmentID = A.AssessmentID
  5  JOIN Module M ON AMS.ModuleID = M.ModuleID
  6  WHERE M.ModuleName = 'Databases'
  7  GROUP BY S.StudentName
  8  HAVING SUM(A.TotalMarks) >= (
  9      SELECT AVG(TotalMarks)
 10      FROM (
 11          SELECT A.TotalMarks
 12          FROM Assessment A
 13          JOIN Assessment_Module_Student AMS ON A.AssessmentID = AMS.AssessmentID
 14          JOIN Module M ON AMS.ModuleID = M.ModuleID
 15          WHERE M.ModuleName = 'Databases'
 16      )
 17  );

STUDENTNAME     TOTALSCORE
--------------- ----------
Prabhat                100

SQL>
```

Figure 64:Screenshot of students who have scored above the average score in the ‘Databases’ module

5. Display whether a student has passed or failed as remarks as per their total aggregate marks obtained in a particular module. (NOTE: Consider total aggregate marks equal to or above 40 is pass, below 40 is fail)

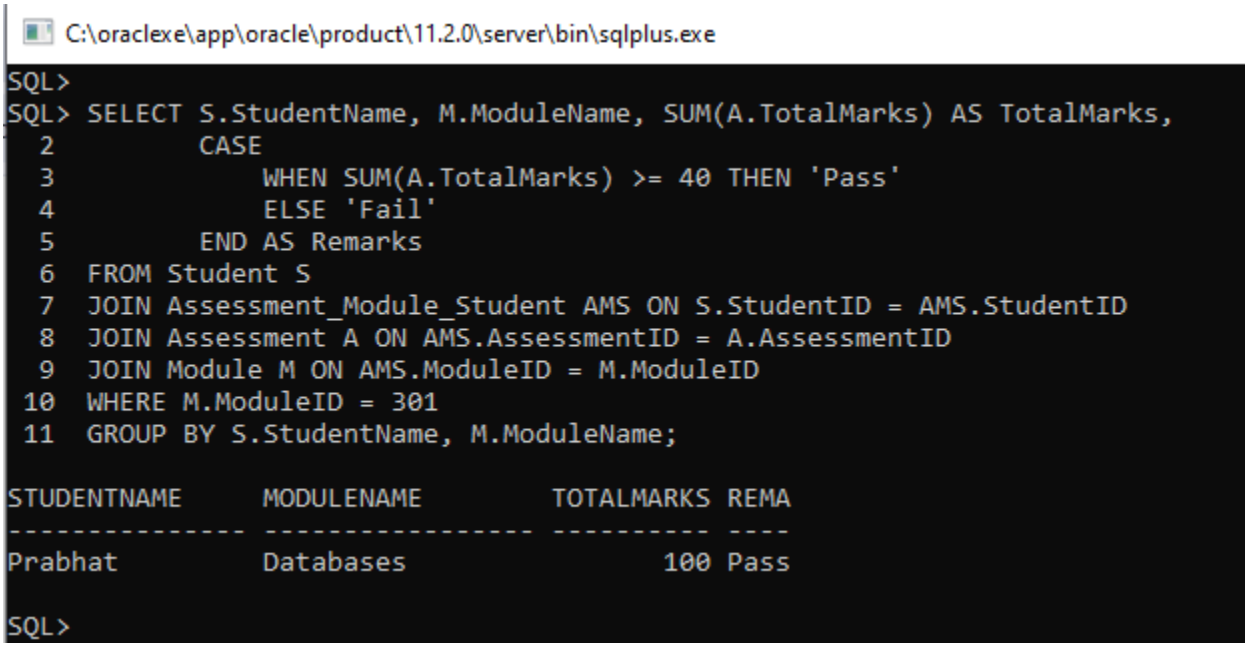

Figure 65:Screenshot of total aggregate marks equal to or above 40 is pass, below 40 is fail

9.3 Dump File Creation

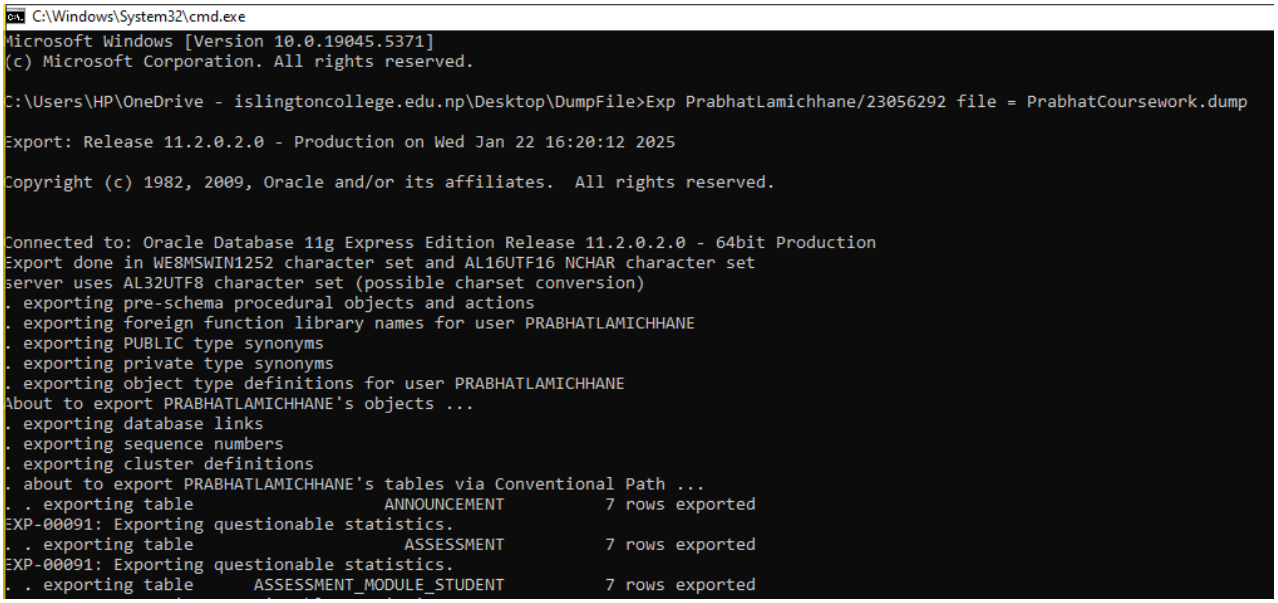

Figure 66:Dump File Creation

9.4 Dropping table

9.4.1 Dropping Student Table

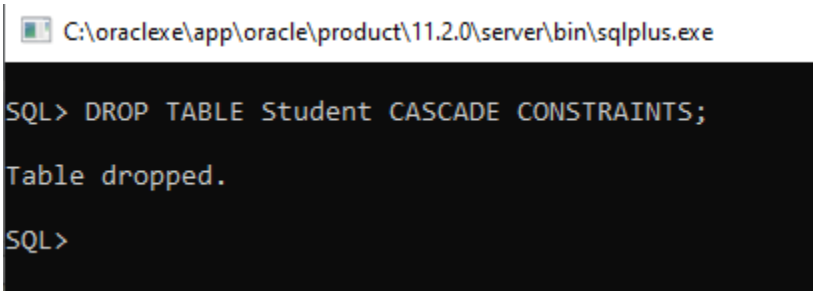

Figure 67:Dropping Student Table

9.4.2 Dropping Program Table

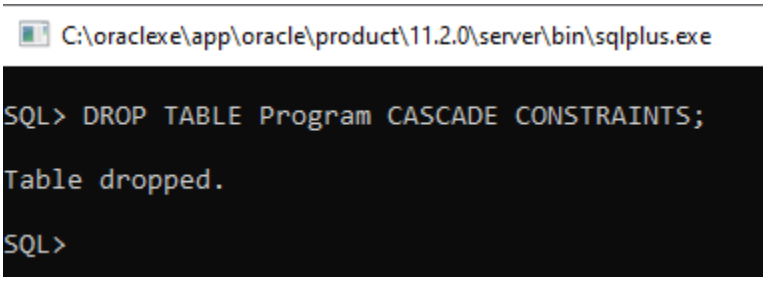

Figure 68:Dropping Program Table

9.4.3 Dropping Module Table

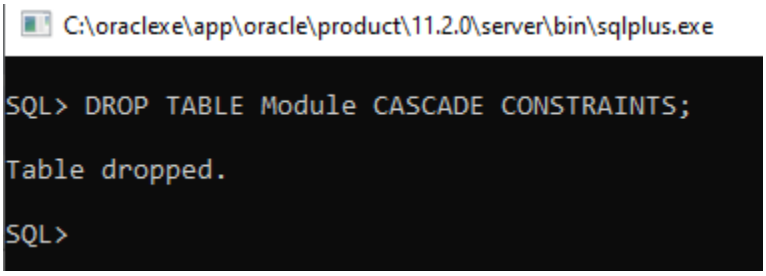

Figure 69: Dropping Module Table

9.4.4 Dropping Teacher Table

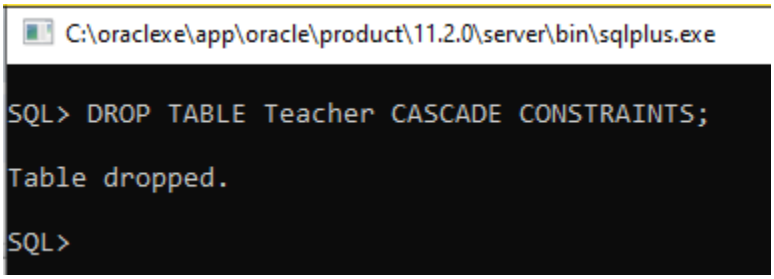

Figure 70:Dropping Teacher Table

9.4.5 Dropping Resources Table

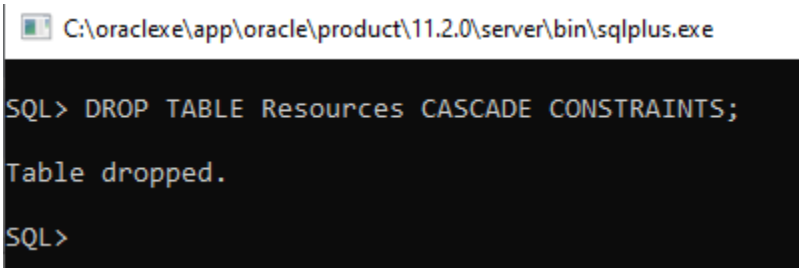

Figure 71:Dropping Resources Table

9.4.6 Dropping Assessment Table

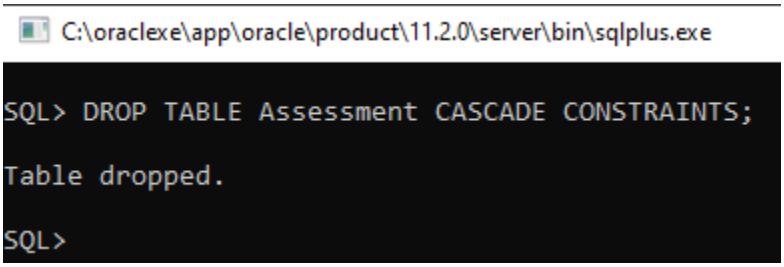

Figure 72:Dropping Assessment Table

9.4.7 Dropping Announcement Table

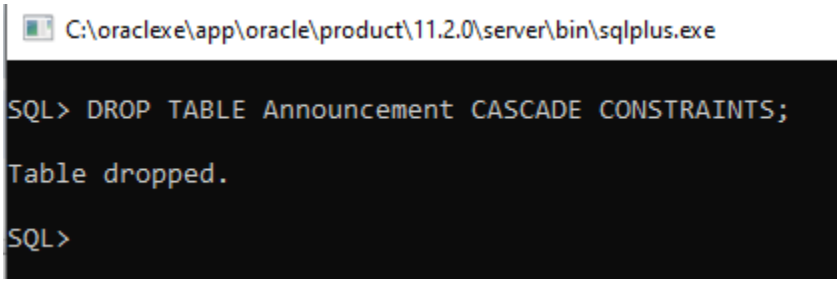

Figure 73:Dropping Announcement Table

9.4.8 Dropping Student_Module Table

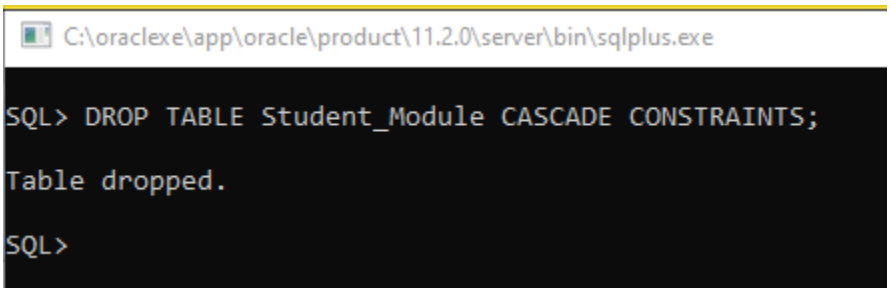

Figure 74:Dropping Student_Module Table

9.4.9 Dropping Teacher_Student_Module Table

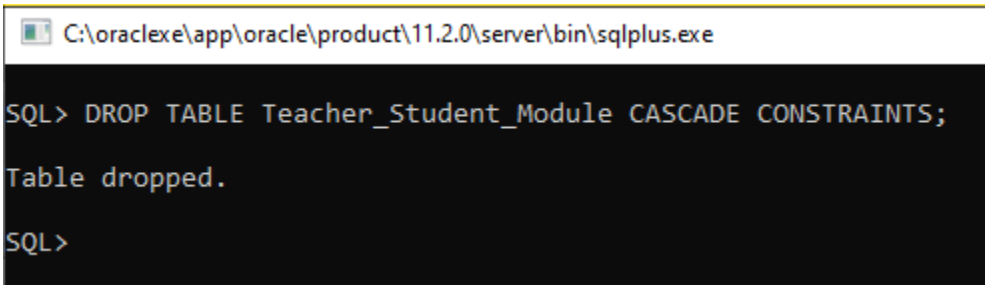

Figure 75:Dropping Teacher_Student_Module Table

9.4.10 Dropping Resources_Module_Student Table

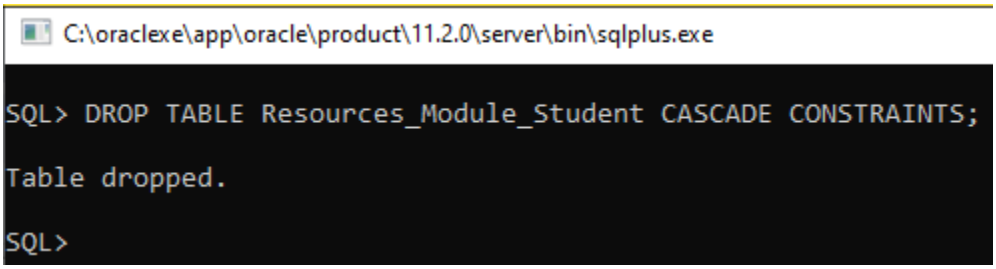

Figure 76:Dropping Resources_Module_Student Table

9.4.11 Dropping Assessment_Module_Student Table

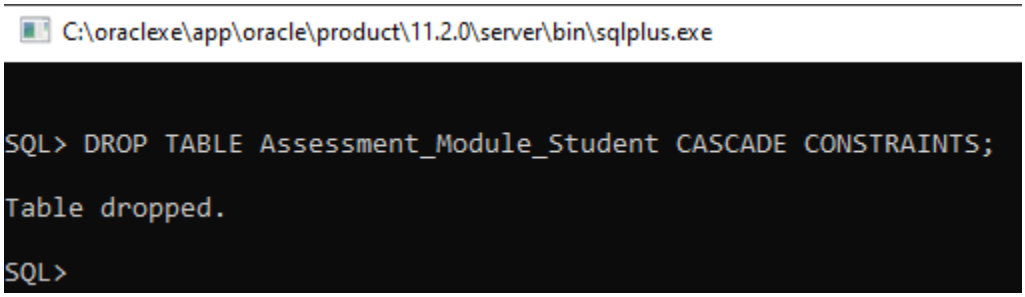

Figure 77:Dropping Assessment_Module_Student Table

9.4.12 Dropping Student_ Module_Teacher_Announcement Table

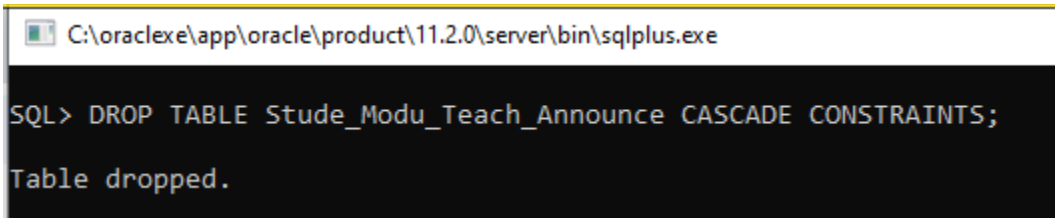

Figure 78:Dropping Student_ Module_Teacher_Announcement Table

10. Critical Evaluation

The module provided a structured approach to understanding database design and implementation, emphasizing practical applications like normalization, ER diagrams, and SQL commands. Through this module, the principles of database integrity, optimization, and scalability were thoroughly explored, equipping students with industry-relevant skills. The module enabled the creation of normalized databases, ensuring efficient data storage and retrieval which proved valuable for managing complex data interactions in the E-Classroom platform. We learned techniques for constructing queries to extract meaningful information from databases and gained foundational knowledge for developing applications such as learning management systems, e-commerce platforms and financial systems. The module also complemented other subjects such as software engineering, by providing backend solutions for managing data, and data structures, by building upon concepts like indexing and efficient storage. Additionally, the integration of database concepts into web development and cybersecurity highlighted the importance of secure, persistent data storage and access controls.

The coursework effectively covered all phases of database development, from identifying entities and relationships to implementing normalized tables with SQL. It emphasized real-life applicability by focusing on the design of an E-Classroom platform, bridging the gap between theoretical knowledge and practical scenarios. Hands-on activities, such as creating and populating tables, enhanced technical proficiency in SQL and provided valuable skill development. However, the coursework had certain limitations, such as a lack of exposure to advanced topics like database security, indexing, and performance optimization. It also focused primarily on Oracle Database, which limited opportunities to explore other widely used DBMS platforms like MySQL or PostgreSQL. Challenges included understanding and implementing normalization, particularly transitive and partial dependencies, debugging SQL errors during table creation and data insertion, and accurately mapping entities and relationships in the ERD. Despite these difficulties, iterative practice helped overcome these issues, resulting in a rewarding learning experience. Future improvements could include incorporating advanced topics such as query optimization, database indexing and NoSQL databases to provide a broader

understanding of modern database technologies. Additionally, offering tutorials or resources on alternative DBMS platforms and incorporating collaborative tasks would enhance peer learning and better simulate team-based projects.

11. CONCLUSION

The aim of this coursework was to design, develop and create a database for Ms. Mary's E-Classroom Platform. This project gave me a good understanding of how databases work in real life. Databases are very important in today's world because they help store and manage data in an organized way.

While working on this project, I faced many challenges. The first challenge was finding the right entities and attributes from the case study and business rules. At first, I included extra attributes that were not needed, which made normalization harder. After trying multiple times, I managed to figure out which attributes were necessary. Another big challenge was normalization, especially with second normal form (2NF) and third normal form (3NF). It was difficult to understand the difference between partial dependency and full dependency in tables with multiple keys. I practiced a lot and did research to solve these problems, but understanding transitive dependency in 3NF was still tricky.

The implementation part was also hard. I made many mistakes while creating tables and inserting data. These mistakes happened because I did not have much practice, but with time and effort, I finished the implementation. Writing queries was another problem, but I overcame it by reviewing workshop materials and doing extra research.

Database Query part was hard. I had made many mistakes while creating information query and transaction query. It was hard but at last I did my best so that I can finish this database query.

In the end, I was able to complete the coursework by designing and developing the database for the E-Classroom Platform. Although database design is not my favourite subject, I now understand how important it is in the modern world. Many organizations, such as schools, colleges and banks use databases to store and retrieve data. This coursework has taught me practical skills that I can use in the future. I think the knowledge I gained will help me in my final year project and in other tasks that require database work.

12. REFEREENCES

geeksforgeeks. (2024, 12 27). *Introduction of ER Model - GeeksforGeeks*. Retrieved from geeksforgeeks: https://www.geeksforgeeks.org/introduction-of-er-model/

GeeksforGeeks. (2025, 01 16). *GeeksforGeeks*. Retrieved from Introduction of ER Model - GeeksforGeeks: https://www.geeksforgeeks.org/introduction-of-er-model/

Learn, M. (2024, 11 23). *Microsoft Learn*. Retrieved from Database normalization description - Microsoft 365 Apps | Microsoft Learn: https://learn.microsoft.com/en-us/office/troubleshoot/access/database-normalization-description