Bhumika Karki.docx

Islington College,Nepal

Document Details

Submission ID
trn:oid:::3618:79905937

Submission Date
Jan 23, 2025, 8:04 AM GMT+5:45

Download Date
Jan 23, 2025, 8:05 AM GMT+5:45

File Name
Bhumika Karki.docx

File Size
35.9 KB

64 Pages

4,713 Words

29,174 Characters

19% Overall Similarity

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

Match Groups

- **74** Not Cited or Quoted 19%
  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

0% Internet sources
0% Publications
19% 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.

Match Groups

- **74** Not Cited or Quoted 19%
  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

0% Internet sources
0% Publications
19% Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1	Submitted works	islingtoncollege on 2025-01-02 — 5%
2	Submitted works	islingtoncollege on 2025-01-03 — 2%
3	Submitted works	islingtoncollege on 2025-01-03 — 1%
4	Submitted works	islingtoncollege on 2025-01-03 — 1%
5	Submitted works	Asia Pacific University College of Technology and Innovation (UCTI) on 2021-05-19 — <1%
6	Submitted works	Washington University of Science and Technology on 2024-01-18 — <1%
7	Submitted works	University of Hertfordshire on 2022-04-27 — <1%
8	Submitted works	islingtoncollege on 2025-01-02 — <1%
9	Submitted works	Australian Catholic University on 2024-08-30 — <1%
10	Submitted works	NCC Education on 2024-06-24 — <1%

11 Submitted works
Columbia College of Missouri on 2017-12-03 <1%

12 Submitted works
NCC Education on 2017-05-07 <1%

13 Submitted works
University of Bristol on 2007-06-07 <1%

14 Submitted works
Gedu College of Business Studies on 2024-04-15 <1%

15 Submitted works
RMIT University on 2023-05-16 <1%

16 Submitted works
Torrens Global Education Services Pty Ltd on 2024-08-27 <1%

17 Submitted works
Oxford Brookes University on 2023-05-05 <1%

18 Submitted works
Pathfinder Enterprises on 2019-05-13 <1%

19 Submitted works
University of Reading on 2023-05-10 <1%

20 Submitted works
Arab Open University on 2024-11-07 <1%

21 Submitted works
Sheffield Hallam University on 2023-04-17 <1%

22 Submitted works
Southern New Hampshire University - Continuing Education on 2024-10-07 <1%

23 Submitted works
Florida Institute of Technology on 2022-07-18 <1%

24 Submitted works
Southern New Hampshire University - Continuing Education on 2022-02-28 <1%

25 Submitted works

Columbia College of Missouri on 2017-10-08 <1%

26 Submitted works

Corinda State High School on 2012-08-30 <1%

27 Submitted works

Adtalem Global Education on 2024-11-04 <1%

28 Submitted works

Asia Pacific University College of Technology and Innovation (UCTI) on 2021-02-04 <1%

29 Submitted works

West Herts College on 2023-06-23 <1%

30 Submitted works

De Montfort University on 2011-02-17 <1%

31 Internet

journal.unismuh.ac.id <1%

32 Submitted works

Informatics Education Limited on 2009-11-08 <1%

Introduction

1.1. Introduction to the University

The founder of the Mid-west University, Ms. Mary proposes the E-Classroom platform to make it easier and more efficient for the university to manage their students, instructors, programs, and modules. It offers a centralized digital environment for academic operations, ensuring efficient resource distribution and stakeholder communication. Students can use the platform to access their academic information, including announcements, assessments, and modules, while teachers can manage their respective modules and distribute resources to students. Administrators can also oversee programs, assign teachers to modules, and use results to track the progress of students.

This system aims to establish a smooth and organized learning environment by reducing manual procedures and ensuring efficient data management. The teaching and learning process is improvised by features like resource sharing, assessment tracking, and automatic result generation. Additionally, announcements for modules also provide students with important academic information.

1.2. Aims and Objectives

The ultimate aim of the organization is to develop and implement a digital platform, the E-Classroom Platform, which enables effective management of students, programs and modules, providing a structured and dynamic study environment. The following objectives will be performed in accordance with the question's requirements as follows:

Create and implement a functional database capable of managing the details of Students, their associated Programs, and the Modules included in each program.
Identify and track the relationships between Programs, Modules, and Students, ensuring seamless data integration for academic and administrative purposes.
Maintain a comprehensive record of all modules, including attributes such as module credits and the programs they belong to.
Provide a structure that enables the management of module assessments and resources in alignment with students' academic progress.
Design a system that supports querying and analysing data for informational and transactional purposes, such as retrieving student enrolment details and module associations.
Develop a framework that ensures scalability and supports many-to-many relationships, such as modules shared across programs.

1.3. Current Business Activities

The E-Classroom platform follows a set of core activities to control its functionality and ensure seamless operations. These activities include:

4 Students are enrolled in a certain program, and each program consists of several modules.

Students are assigned to a single program, and their data is maintained in the database.

Each program offers multiple modules that are specific to the program's curriculum.

Module performance is evaluated, and the results are calculated and stored in the database.

To guarantee accuracy and accessibility; program, student, and module information is updated on a regular basis.

1.4. Business Rules

To ensure the efficient management and operation of the E-Classroom platform for Mid-west University, it is critical to establish and adhere to specific business rules. These rules outline how the platform will function and are listed below:

Each Program is uniquely identified by a ProgramID and includes attributes such as program name and description. A Program consists of multiple modules, and each module can belong to many programs.

Each Student is uniquely identified by a StudentID and is linked with exactly one Program, whereas a program contains multiple students. Students have details such as name, email, phone, and address.

Each Module is uniquely identified by a ModuleID and includes attributes like module name, credits, and TeacherID (primary teacher). Modules can have associated Resources, Assessments, and Announcements.

Each Teacher is uniquely identified by a TeacherID and is assigned to one or more Modules, each module is assigned to one teacher. Teachers have attributes such as name and department.

Each Resource is uniquely identified by a ResourceID and is tied to a specific Module.

ResourceCompletion tracks if a student has completed a resource before being granted to the next one. Resources include attributes like title, and type to facilitate structured learning.

Each Assessment is uniquely identified by an AssessmentID and can be shared across multiple modules. Assessments include details such as title, deadline, weightage, TotalMarks. This relationship is managed through the ModuleAssessment bridge table.

Each Announcement is uniquely identified by an AnnouncementID and is associated with a specific Module and posted by a specific teacher. Announcements contain details like content and posting date.

Each Result is uniquely identified by a ResultID and is associated with a Student and an Assessment. Results track attributes like ObtainedMarks, and grade where grade is derived based on the percentage of ObtainedMarks out of TotalMarks.

A Module can have multiple Resources, and Announcements, but each of these entities belongs to only one Module. A module can have multiple Assessments, and assessments can also belong to modules across different programs.

Assumptions:

A student is enrolled in one program only. Students have details such as name, email, phone, DOB, and address.

Each program consists of multiple modules, and a module can belong to multiple programs (many-to-many relationship). Modules are mandatory for the students who are enrolled in the programs that they are linked to.

Resources are linked to a module, and it must be completed in a specific order.

Completion status and date of the resources is tracked by using a ResourceCompletion table. Resources include attributes like title, type to facilitate structured learning.

Each assessment is linked to a single module. However, modules can share assessments across different programs (many-to-many relationship via ModuleAssessment). Assessments include details such as title, deadline, and weightage.

The result of assessments is tracked per student and is uniquely identified by ResultID.

Each Teacher is assigned to one or more Modules, and each module is assigned to one primary teacher. Teacher is uniquely identified by a TeacherID and have attributes such as name and department.

Announcements are linked to a specific module and posted by the assigned teacher. Announcements are uniquely identified by AnnouncementID.

2. Initial ERD

An entity relationship diagram (ERD) is a visual representation of a relational database. ERDs can be used to visualize and understand an existing database or to develop a new one.

Understanding what an entity is and what an entity set is, is the first step in identifying ERDs. In this context, an entity is an object or a piece of data. The characteristics of these entities may be defined by their attributes. An ERD demonstrates the logical structure of databases by defining the entities, their properties, and the relationships

between them. In relational databases, entities are equivalent to tables (rows). The characteristics of that entity (columns) that you wish to record are called attributes. Lastly, relationships describe the interactions between the entities (Secoda, 2024).

2.1. List of Created Objects

As a result of the preceding steps, significant measures were taken to define the framework for the database of the E-Classroom platform. The created objects include all the major entities and their associated attributes, which are listed below:

Three distinct objects are chosen as major entities based on their intrinsic value. Each of them is interconnected to define the database design and fulfill the requirements of the E-Classroom platform. They are briefly explained as follows:

Program: The program serves as the foundation for grouping related modules and students. Each program is uniquely identified by its ProgramID and includes additional details like the program name and description. Programs establish the academic structure for students.

Student: Students are the primary users of the platform. Each student is uniquely identified by their StudentID and is associated with a single program. Additional attributes include the student's name, email, phone, and address.

Module: Modules are essential academic units that are part of one or more programs. Each module has features like its name and credits and is uniquely identified by its ModuleID. Students, programs, assessments, and resources are all connected by modules, which provide an integrated academic framework.

2.2. Identification of Entities and Attributes

Student

Table 1: Defining the Data Types of the Student

Table

S. No.

1 Attributes

Data Type

Size

Constraints

StudentID

CHARACTER

25

PRIMARY KEY, NOT NULL, UNIQUE

StudentName

CHARACTER

50

NOT NULL

StudentEmail

CHARACTER

50

26 NOT NULL, UNIQUE

StudentPhone

CHARACTER

15

NOT NULL, UNIQUE

StudentDOB

DATE

NOT NULL

StudentAddress

CHARACTER

100

NOT NULL

Program

Table 2: Defining the Data Types of the Program

Table

S. No.

Attributes	Data Type	Size	Constraints
ProgramID	CHARACTER	25	PRIMARY KEY, NOT NULL, UNIQUE
ProgramName	CHARACTER	50	NOT NULL, UNIQUE
ProgramDescription	CHARACTER	100	NOT NULL

Module

Table 3: Defining the Data Types of the Module Table

S. No.

1 Attributes

Data Type

Size

Constraints

ModuleID

CHARACTER

25

PRIMARY KEY, NOT NULL, UNIQUE

ModuleName

CHARACTER

50

NOT NULL, UNIQUE

Credits

NUMBER

3

NOT NULL

TeacherID

18 CHARACTER

18

25

NOT NULL, UNIQUE

AssessmentID

CHARACTER

25

NOT NULL, UNIQUE

ResourceID

CHARACTER

25

NOT NULL, UNIQUE

AnnouncementID

CHARACTER

25

NOT NULL, UNIQUE

ResultID

CHARACTER

25

NOT NULL, UNIQUE

2.3. Representation of Primary and Foreign Keys

Based on the definition and meaning associated with each of the keys, the following constraints were used to determine their base primary and foreign keys. Each of the primary keys assigned to each of the entities is thus presented below:

ProgramID: As numerous program details may be similar, a unique key in the form of an ID enables them to be properly identified.

ModuleID: To keep track of each module offered under various programs, each module must be assigned a unique key for subsequent inquiry and inspection.

StudentID: As numerous student details (e.g., name) may be similar, a unique key in the form of an ID enables them to be properly identified.

ResultID: To keep track of each student's performance in different modules, a unique identifier is assigned to ensure proper tracking and analysis.

AssessmentID: As assessments within a module may overlap in terms of deadlines and titles, a unique key ensures accurate tracking for grading and scheduling.

TeacherID: A unique identifier used to distinguish each teacher, ensuring accurate module assignments and avoiding confusion even when teachers have similar names or departments.

ResourceID: To keep track of all resources linked to modules, each resource must be assigned a unique identifier for easy access and management.

AnnouncementID: As modules can have multiple announcements, a unique identifier is assigned to track announcements related to updates and information sharing.

The relationship between the given entities can be thus described along with the cardinality that they present as follows:

15 A program includes multiple modules, and a module can be part of multiple programs. This relationship establishes a many-to-many relationship between programs and modules.

A program has multiple students enrolled, but a student is enrolled in only one program. This relationship establishes a one-to-many relationship between programs and students.

13 2.4. Entity Relationship Diagram

This ERD represents the structure of an E-Classroom system for Mid-west University, which focuses on the relationships between students, programs, and modules. Each

student can be enrolled in one program only. Programs are made up of several modules, and modules might be a part of more than one program. The system enables effective management of academic data by tracking important attributes for students, programs, and modules.

3. Normalization

The process of organizing the provided data in a database by reducing the redundancy in a relation is known as normalization. Here, we remove the anomalies—update, insertion, and deletion. The single table is divided into smaller tables by normalization, and relationships are used to connect them. We can reduce the amount of redundancy in the database table by using the various normal forms (Rajput, 2024).

3.1. Un-Normalized Form (UNF)

The rules to be followed while the initial process of UNF is listed below:

All the relevant entities and relationships are identified in a single table.

Curly braces {} are used to denote the repetition of a group of letters.

The main key for each of the entities is identified and presented.

Scenario of UNF:

Student (StudentID, StudentName, StudentEmail, StudentDOB, StudentAddress, StudentPhone, ProgramID, ProgramName, ProgramDescription, {ModuleID, ModuleName, Credits, TeacherID, TeacherName, Department, {AssessmentID, Title,

Deadline, Weightage, TotalMarks, ResultID, ObtainedMarks, Grade}, {ResourceID, ResourceName, ResourceType, CompletionOrder, CompletionStatus, CompletionDate}, {AnnouncementID, AnnouncementDetails, PostedDate}})

3.2. First Normalized Form (1NF)

The following requirements must be followed in 1NF:

It is crucial that all of the values in a column originate from the same domain; single-valued attributes and columns should only be used when absolutely necessary.

13 The primary key of the base table is sent to the inner tables as composite keys; the repeating group should be separated from the main base table.

Scenario of 1NF:

3 Student (StudentID, StudentName, StudentEmail, StudentDOB, StudentAddress, StudentPhone, ProgramID, ProgramName, ProgramDescription)

Module (ModuleID, ModuleName, Credits, TeacherID, TeacherName, Department)

Assessment (AssessmentID, Title, Deadline, Weightage, TotalMarks, ModuleID*)

Result (ResultID, ObtainedMarks, Grade, StudentID*, AssessmentID*)

Resource (ResourceID, ResourceName, ResourceType, CompletionOrder, ModuleID*)

ResourceCompletion (StudentID*, ResourceID*, CompletionStatus, CompletionDate)

Announcement (AnnouncementID, AnnouncementDetails, PostedDate, ModuleID*, TeacherID*)

Functional Dependencies in 1NF:

StudentID StudentName, StudentEmail, StudentDOB, StudentAddress, StudentPhone, ProgramID, ProgramName, ProgramDescription

ModuleID ModuleName, Credits, TeacherID, TeacherName, Department

AssessmentID Title, Deadline, Weightage, TotalMarks, ModuleID

ResultID StudentID, AssessmentID, ObtainedMarks, Grade

ResourceID ResourceName, ResourceType, CompletionOrder, ModuleID

StudentID, ResourceID CompletionStatus, CompletionDate

AnnouncementID AnnouncementDetails, PostedDate, ModuleID, TeacherID

3.3. Second Normalized Form (2NF)

These instructions must be followed in order to convert to 2NF:

If an entity has a partial dependency, it can be eliminated by looking through all of its composite keys or unique identifiers.

Partial dependency analysis across the tables is not necessary when representing a single unique identifier.

Scenario of 2NF:

Checking for Partial Dependency

Student (StudentID, StudentName, StudentEmail, StudentDOB, StudentAddress, StudentPhone, ProgramID, ProgramName, ProgramDescription)

The attributes ProgramName, and ProgramDescription are depended on ProgramID and not on StudentID which shows that they are partially dependent. So, Program is separated into its own table.

StudentID StudentName, StudentEmail, StudentDOB, StudentAddress, StudentPhone, ProgramID

ProgramID ProgramName, ProgramDescription

Module (ModuleID, ModuleName, Credits, TeacherID, TeacherName, Department)

ProgramID and ModuleID form a composite key in the ProgramModule table, which resolves the many-to-many relationship between programs and modules. Also, the attributes ModuleName, Credits, and TeacherID are fully dependent on ModuleID and not on ProgramID; so, the partial dependency is removed accordingly.

ProgramID ProgramName, ProgramDescription

ModuleID ModuleName, Credits, TeacherID, TeacherName, Department

This becomes necessary to evaluate characteristics to see how many are functionally dependent on unique identifiers and how many are not because of the presence of composite keys or unique identifiers in this scenario.

ProgramID, ModuleID

ProgramModule (ProgramID*, ModuleID*)

Similarly, a StudentModule table is introduced to resolve the many-to-many relationship between students and modules.

StudentID, ModuleID

StudentModule (StudentID*, ModuleID*)

19 Modules can share assessments between different programs which leads to a many-to-many relationship between Module and Assessment. A single module can have multiple assessments and, the same assessment can be shared across modules in different programs.

To resolve this, a bridge table called ModuleAssessment is introduced.

ModuleID, AssessmentID

ModuleAssessment (ModuleID*, AssessmentID*)

4 Final Tables after 2NF:

Student (StudentID, StudentName, StudentEmail, StudentDOB, StudentAddress, StudentPhone, ProgramID*)

Program (ProgramID, ProgramName, ProgramDescription)

Module (ModuleID, ModuleName, Credits, TeacherID, TeacherName, Department)

ProgramModule (ProgramID*, ModuleID*)

StudentModule (StudentID*, ModuleID*)

4 Assessment (AssessmentID, Title, Deadline, Weightage, TotalMarks)

ModuleAssessment (ModuleID*, AssessmentID*)

Result (ResultID, ObtainedMarks, Grade, StudentID*, AssessmentID*)

Resource (ResourceID, ResourceName, ResourceType, Duration, CompletionOrder, ModuleID*)

ResourceCompletion (StudentID*, ResourceID*, CompletionStatus, CompletionDate)

Announcement (AnnouncementID, AnnouncementDetails, PostedDate, ModuleID*, TeacherID*)

3.4. Third Normalized Form (3NF)

31 The following guidelines must be followed in order to complete 3NF:

Every table's transitive dependency must be eliminated.

Every non-key attribute in a relation that has more than one must be checked for.

Later separation and transitive dependency are necessary.

Before 3NF:

Module (ModuleID, ModuleName, Credits, TeacherID, TeacherName, Department)

Since the module, referred through its ID provides the non-key attributes such as TeacherName and Department based on TeacherID, the scenario must be examined for transitive dependency.

Scenario of 3NF:

ModuleID TeacherID TeacherName, Department

This indicates a transitive dependency because TeacherName and Department depend on TeacherID, which in turn depends on ModuleID. To eliminate this transitive dependency, teacher details must be separated into a distinct Teacher table, leaving only TeacherID in the Module table.

2 Functional Dependencies in 3NF:

StudentID StudentName, StudentEmail, StudentDOB, StudentAddress, StudentPhone, ProgramID

ProgramID ProgramName, ProgramDescription

ModuleID ModuleName, Credits, TeacherID

TeacherID TeacherName, Department

AssessmentID Title, Deadline, Weightage, TotalMarks, ModuleID

ResultID StudentID, AssessmentID, ObtainedMarks, Grade

ResultID, StudentID, AssessmentID ObtainedMarks

ResourceID ResourceName, ResourceType, CompletionOrder, ModuleID

StudentID, ResourceID CompletionStatus, CompletionDate

AnnouncementID AnnouncementDetails, PostedDate, ModuleID, TeacherID

Therefore, the final tables after normalization are as follows:

Student (StudentID, StudentName, StudentEmail, StudentDOB, StudentAddress, StudentPhone, ProgramID*)

Program (ProgramID, ProgramName, ProgramDescription)

Module (ModuleID, ModuleName, Credits, TeacherID*)

Teacher (TeacherID, TeacherName, Department)

ProgramModule (ProgramID*, ModuleID*)

StudentModule (StudentID*, ModuleID*)

Assessment (AssessmentID, Title, Deadline, Weightage, TotalMarks)

ModuleAssessment (ModuleID*, AssessmentID*)

Result (ResultID, ObtainedMarks, Grade, StudentID*, AssessmentID*)

Resource (ResourceID, ResourceName, ResourceType, Duration, CompletionOrder, ModuleID*)

ResourceCompletion (StudentID*, ResourceID*, CompletionStatus, CompletionDate)

Announcement (AnnouncementID, AnnouncementDetails, PostedDate, ModuleID*, TeacherID*)

4. Final ERD

Following the normalization process, all the tabulated forms are now grouped into a single chart, each of which is assigned a primary key and a foreign key in order to link or join two distinct tables that share common properties. The most critical visualization of all the procedures is the development of a bridge entity, which is responsible for reducing data redundancy and anomalies that might contaminate the database and make it appear messy. Thus, the Program-Module and Student-Module illustrating the Many to Many relationships are further broken down by the relationship shown by the specified bridge entity: ProgramModule and StudentModule.

Similarly, the relationship between modules and teachers follows a one-to-many structure, where each module is assigned to a specific teacher through a foreign key (TeacherID). Transitive dependencies are removed by separating teacher details, such as TeacherName and Department, into a distinct Teacher table, ensuring that module records only store the TeacherID. Additionally, the ModuleAssessment table is introduced to resolve the many-to-many relationship between module and assessment, which allows a single assessment to be shared across multiple modules in different programs. Each of the entities is connected to the others, providing the mandatory relationships on each side to demonstrate how the database values for each of the entities were optimized.

Furthermore, a ResourceCompletion table is established to track the progress of students on an individual resources within a module. This table makes sure that a resource must be completed before the access is given to the next one which also supports structured learning. The StudentModule table ensures that students are

properly associated with the modules they are enrolled in, which resolves the many-to-many relationships and makes it possible to connect students to resources, assessments, and results for specific modules.

The relationship between Module and Assessment is further improvised through the ModuleAssessment bridge table, ensuring the flexibility in linking assessments to multiple modules. The relationship between Assessment and Result is made in such a way that Result depends on both AssessmentID and StudentID to accurately track student's performance without redundancy.

8 A Program consists of multiple modules, and each module can belong to many

6 programs. This establishes a many-to-many relationship between programs and modules through the ProgramModule bridge table.

Modality: A program must have at least one module, a module cannot exist without being linked to a program.

2 Modality: A student must be enrolled in one program; a program can exist without students.

Modality: A student must be enrolled in modules which is a part of their program; a module can exist without students.

22 A teacher can be assigned to multiple modules, but each module is taught by only one teacher. This establishes a one-to-many relationship between teachers and modules.

Modality: Each module must have a teacher; a teacher can exist without being assigned to a module.

A module can have multiple assessments, but each assessment belongs to only one module and the assessment is shared across modules in various programs. This establishes a many-to-many relationship between modules and assessments.
Modality: Each assessment must belong to a module; a module must have at least one assessment.

A student can have multiple results, but each result belongs to one student and is associated with one assessment. This establishes a many-to-many relationship between Student and Assessment through the Result table.
Modality: Each result must belong to an assessment and a student; assessment or student can exist without a result.

6 A module can generate multiple announcements, but each announcement is tied to only one module. This establishes a one-to-many relationship between modules and announcements.

2 Modality: Announcements cannot exist without being linked to a module; a module may have no announcements.

14 A module can have multiple resources, but each resource is linked to only one module. This establishes a one-to-many relationship between modules and resources.

Modality: A resource cannot exist without being linked to a module; a module may have no resources.

17 A student can complete multiple resources and a resource can be completed by multiple students. This establishes a many-to-many relationship between Student and Resources through the ResourceCompletion table. ResourceCompletion tracks if a student has completed a resource before being granted to the next one.

Modality: A student may not have interacted with a resource yet, and a resource may noy have been accessed by any students.

5. Data Dictionary

5.1. Data Dictionary for Student Table

S. No.	Attributes	Data Type	Size	Constraints
	StudentID	CHARACTER	25	PRIMARY KEY, NOT NULL, UNIQUE
	StudentName	CHARACTER	50	NOT NULL
	StudentEmail	CHARACTER		

50

NOT NULL, UNIQUE

StudentPhone

CHARACTER

15

NOT NULL, UNIQUE

2 StudentDOB

DATE

NOT NULL

StudentAddress

CHARACTER

100

NOT NULL

ProgramID

CHARACTER

25

NOT NULL

5.2. Data Dictionary for Program Table

S. No.	Attributes	Data Type	Size	Constraints
1	ProgramID	CHARACTER	25	PRIMARY KEY, NOT NULL, UNIQUE
	ProgramName	CHARACTER	50	NOT NULL
	ProgramDescription	CHARACTER	100	NOT NULL

5.3. Data Dictionary for Module Table

S. No.	Attributes	Data Type	Size	Constraints
	ModuleID	CHARACTER	25	PRIMARY KEY, NOT NULL, UNIQUE
	ModuleName	CHARACTER	50	NOT NULL
	Credits	NUMBER	3	NOT NULL

TeacherID

CHARACTER

11 25

FOREIGN KEY, NOT NULL, UNIQUE

12 5.4. Data Dictionary for ProgramModule Table

S. No.

1 Attributes

Data Type

Size

Constraints

Composite Constraint

ProgramID

CHARACTER

25

FOREIGN KEY, NOT NULL, UNIQUE

PRIMARY KEY

ModuleID

CHARACTER

25

FOREIGN KEY, NOT NULL, UNIQUE

5.5. Data Dictionary for StudentModule Table

S. No.

Attributes

Data Type

Size

Constraints

Composite Constraint

StudentID

CHARACTER

25

FOREIGN KEY, NOT NULL, UNIQUE

PRIMARY KEY

ModuleID

CHARACTER

25

FOREIGN KEY, NOT NULL, UNIQUE

5.6. Data Dictionary for Teacher Table

S. No.

Attributes

Data Type

Size

Constraints

TeacherID

CHARACTER

25

PRIMARY KEY, NOT NULL, UNIQUE

TeacherName

CHARACTER

50

NOT NULL

Department

CHARACTER

50

NOT NULL

5.7. Data Dictionary for Assessment Table

S. No.

Attributes

Data Type

Size

Constraints

AssessmentID

CHARACTER

25

PRIMARY KEY, NOT NULL, UNIQUE

AssessmentTitle

CHARACTER

50

NOT NULL

Deadline

DATE

NOT NULL

Weightage

NUMBER

(5, 2)

NOT NULL

TotalMarks

NUMBER

3

ModuleID

CHARACTER

25

FOREIGN KEY, NOT NULL, UNIQUE

5.8. Data Dictionary for ModuleAssessment Table

S. No.

Attributes

Data Type

Size

Constraints

Composite Constraint

StudentID

CHARACTER

25

FOREIGN KEY, NOT NULL, UNIQUE

PRIMARY KEY

ModuleID

CHARACTER

25

FOREIGN KEY, NOT NULL, UNIQUE

5.9. Data Dictionary for Result Table

S. No.

Attributes

Data Type

Size

Constraints

Composite Constraint

ResultID

CHARACTER

25

PRIMARY KEY, NOT NULL, UNIQUE

ObtainedMarks

4 NUMBER

(5, 2)

NOT NULL

Grade

NUMBER

5

NOT NULL

AssessmentID

CHARACTER

25

FOREIGN KEY, NOT NULL, UNIQUE

PRIMARY KEY

StudentID

CHARACTER

12 25

FOREIGN KEY, NOT NULL, UNIQUE

5.10. Data Dictionary for Announcement Table

S. No.	Attributes	Data Type	Size	Constraints
	AnnouncementID	CHARACTER		

25

PRIMARY KEY, NOT NULL, UNIQUE

AnnouncementDetails

CHARACTER

100

NOT NULL

PostedDate

DATE

NOT NULL

ModuleID

CHARACTER

25

FOREIGN KEY, NOT NULL, UNIQUE

TeacherID

CHARACTER

25

FOREIGN KEY, NOT NULL, UNIQUE

5.11. Data Dictionary of Resources Table

S. No.	Attributes	Data Type	Size	Constraints
1	ResourceID	CHARACTER	25	PRIMARY KEY, NOT NULL, UNIQUE
	ResourceName	CHARACTER	50	NOT NULL
	ResourceType	CHARACTER	50	NOT NULL

CompletionOrder

NUMBER

3

ModuleID

CHARACTER

25

FOREIGN KEY, NOT NULL, UNIQUE

5.12. Data Dictionary for ResourceCompletion Table

S. No.

Attributes

Data Type

Size

Constraints

Composite Constraint

StudentID

CHARACTER

25

FOREIGN KEY, NOT NULL, UNIQUE

PRIMARY KEY

ResourceID

CHARACTER

50

FOREIGN KEY, NOT NULL, UNIQUE

CompletionStatus

CHARACTER

15

NOT NULL

CompletionDate

DATE

NOT NULL

6. Data Implementation

Based on the obtained normalized database and its tables, it is now necessary to start working on the implementation of the data and populating each of the finalized tables. The system is connected with the SQL Command Line followed by a user creation and its implementation to grant privileges so that it can access every resource to the user. Thus, the different steps of creation, description, insertion and selection is carried out to the database for its implementation, each of which are defined below.

Connect system

```
CREATE USER Bhumika_Karki Identified by 23047584;
GRANT CONNECT, RESOURCE to Bhumika_Karki;
```

CONNECT Bhumika_Karki/23047584;

7 6.1. Implementation of Program Table

6.2. Implementation of Teacher Table

6.3. Implementation of Module Table

6.4. Implementation of ProgramModule Table

6.5. Implementation of Student Table

6.6. Implementation of StudentModule Table

6.7. Implementation of Assessment Table

6.8. Implementation of ModuleAssessment Table

6.9. Implementation of Result Table

Figure 51: Description of the Result Table

6.10. Implementation of Announcement Table

6.11. Implementation of Resources Table

6.12. Implementation of ResourceCompletion Table

7. Database Querying

Once all the implementation process is carried out and values are inserted inside the tables, it is necessary to query the system to find out the accuracy of the data entered

inside the tables to provide informational as well as transactional answers to different queries.

7.1. Information Query

7.1.1. Information Query: 1

7.1.2. Information Query: 2

7.1.3. Information Query: 3

7.1.4. Information Query: 4

7.1.5. Information Query: 5

7.2. Transaction Query

7.2.1. Transaction Query: 1

7.2.2. Transaction Query: 2

7.2.3. Transaction Query: 3

7.2.4. Transaction Query: 4

7.2.5. Transaction Query: 5

8. Critical Evaluation

8.1. Critical Evaluation of module, it’s usage and relation with other subjects

The course gave an in-depth knowledge of database design, development, and implementation with an emphasis on normalized database systems and structured relationships which ensure data integrity. It highlighted the theoretical and practical aspects of database systems, including normalization, ERD creation, and SQL query optimization, all of which are essential for any modern data-driven system.
Relational concepts like many-to-many relationships, bridge entities, and resolving transitive dependencies helped me understand how databases relate to other subjects like software engineering and system analysis. For instance, by reducing redundancy

and increasing efficiency, ideas like normalization have a direct connection to software design principles. Because databases often serve as the foundation of applications, working with SQL and Oracle databases also links to programming ideas, especially backend development.

I was able to gain a deeper understanding of the relationship between data and user requirements through this module's actual use in developing entities like StudentModule and ModuleAssessment. My problem-solving abilities were enhanced by the systematic approach to resolving real-world challenges, such as introducing resource completion tracking into place and making sure assessments could be shared across several programs through modules. It also highlighted the significance of accuracy and modularity in database systems.

8.2. Critical Assessment of coursework

The coursework required analyzing and creating a comprehensive database system for the E-Classroom platform. Creating links between entities, dealing with ambiguities in data attributes, and implementing normalization up to the third normal form (3NF) were some of the issues I faced during the project. The development of bridge tables like StudentModule and ModuleAssessment was particularly educative, which resolved many-to-many relationships and minimized redundancy.

Ensuring the accuracy of the queries was one of the main hurdles, particularly when handling complex relationships like identifying students who obtained above-average grades in a module or those who missed assessments. Writing queries was not the only difficulty; debugging errors, which often came from improperly established relationships or missing keys were also equally difficult. The understanding of database theory was further strengthened when the issue of ambiguous links—such as the initial absence of direct relationships between students and modules—required redesigning the structure and adding bridge entities.

Furthermore, I was able to identify errors in my initial approach by implementing theoretical ideas like cardinality and modality into practice. For instance, creating foreign keys and composite keys in tables like Result demonstrated how minor mistakes in schema design could lead to huge complications during query execution. Finding out how normalization and relationships directly affect query performance and data integrity was important.

I also learnt the importance of collaboration and iterative development from the module. My tutor's regular feedback was beneficial in enhancing the coursework. Some of the issues, such as organizing the ResourceCompletion table to track resource completion by students or creating the Result table to manage derived attributes like grades were resolved with continuous effort.

To conclude, this course gave me an opportunity to put theoretical ideas into practice, which increased my confidence in database design and query optimization. Even though it was challenging, I was able to create a database system that involved both academic and practical aspects, so I got to have a valuable experience. Without any

doubt, this project will provide a strong foundation for future database development and associated projects.