Table of Contents

List of Figures

List of Tables

1. Introduction

1.1 Database

Data are collection of raw facts and information are processed raw facts that have some meaning. (Stair & Reynolds, 2016)

Database is an organized set of structured information that can be easily stored, accessed and managed. Usually, Database management system software is used to control database for access, storing, updating, managing data and information. Database contains one or multiple tables with columns and rows. Distributed database, rational database, objective database are types of database available. (Oracle Corporation, 2021)

Example: - In store, it needs to store, manage, update and present data related to its employee, products, order details, total sold product etc it is done by using any form of database system. Electricity service provider also use database system for storing, manipulating, and querying data of its staffs and customers details.

1.2 Description of organization

Here Islington college is college of IT and business education. College offers multiple number of courses like computer networking and IT security, Multimedia, computing, Business for spring and autumn intake. Every Spring and Autumn Students enrols in a course and studies multiple numbers of modules throughout the course duration period. Each Course contains multiple numbers of modules. Each module is taught by one or multiple number of teachers. College has Administration block that enrol and registers students for courses. It has finance block where all the financial transaction of college takes place. College also provides learning environment in library and also rent books to students. IT administration block where they take care of all IT related works and provide services. College also has lecture hall and workshop room that provide quality education to students. Student are divided into groups according to courses they enrol. Further they are divided into section in each course. Here college has to keep track of every transactions in finance block, enrolments and registering of students in courses, borrowing, returning of books and registering of new books,

keeping records of employees etc. So, college has database system for storing and managing all information and data needed for college.

1.3 Description of project

Here, the project is about college course management. College has to collect, stores, update and manage detail information of offered courses, enrolments, students, teachers etc in college.

In this project A database named “collegecoursesmanagement” is created with only 5 main selective entities that has been chosen they are TEACHER, STUDENTS, COURSES, ENROLLMENTS, MODULES. In this database college can store, update, collect and manage data and information for students, courses, teachers, enrolments, course’s modules.

This helps college to keep record of information on available courses offered by college. The database system created for college course management can keep track of number of students enrolments in each course. It can store information of each course contents like modules that has to be study by students. It keeps track of information about the teachers which are going to teach course’s modules. It can track the information about each module belonging to which particular course.

College can get detail information and data on course and its modules. Students and Teacher information is stored in organized manner and can get detail information on all students and teachers. College can also track the total number of enrolments in course by students. College can easily manage and manipulate data and information. College can get required information easily and in less time. College can always make ensure that data and information remain update easily

2. Database Model

2.1 Business Rule Of College Course Management System

In Islington college course management system scenario,

-There is a constraint that each student must have only one associated enrolment i.e. Each student can only have each enrolment in college for a particular course, a single student cannot do multiple enrolments in college. Similarly, each enrolment details can only belong to each student in college i.e. Each student can have at most one associated enrolment in college.

-Similarly, each course can have multiple numbers of enrolments because many numbers of students can also enrol in same course. Each enrolment must have exactly one associate course i.e. each enrolment details belongs to each course. So, each enrolment details can have at most one associated course details.

- Each course can contain one or many numbers of modules that is learn by students. Similarly, each module can belong to one or many courses because some courses in Islington college have common modules like networking and computing courses have same information system module in their course content.

-Further, each module can have multiple teachers as team teaching is allowed in Islington college i.e. two or more teacher is allowed to teach each module and each teacher can teach multiple modules in college.

2.2 Entity Relation Model Diagram

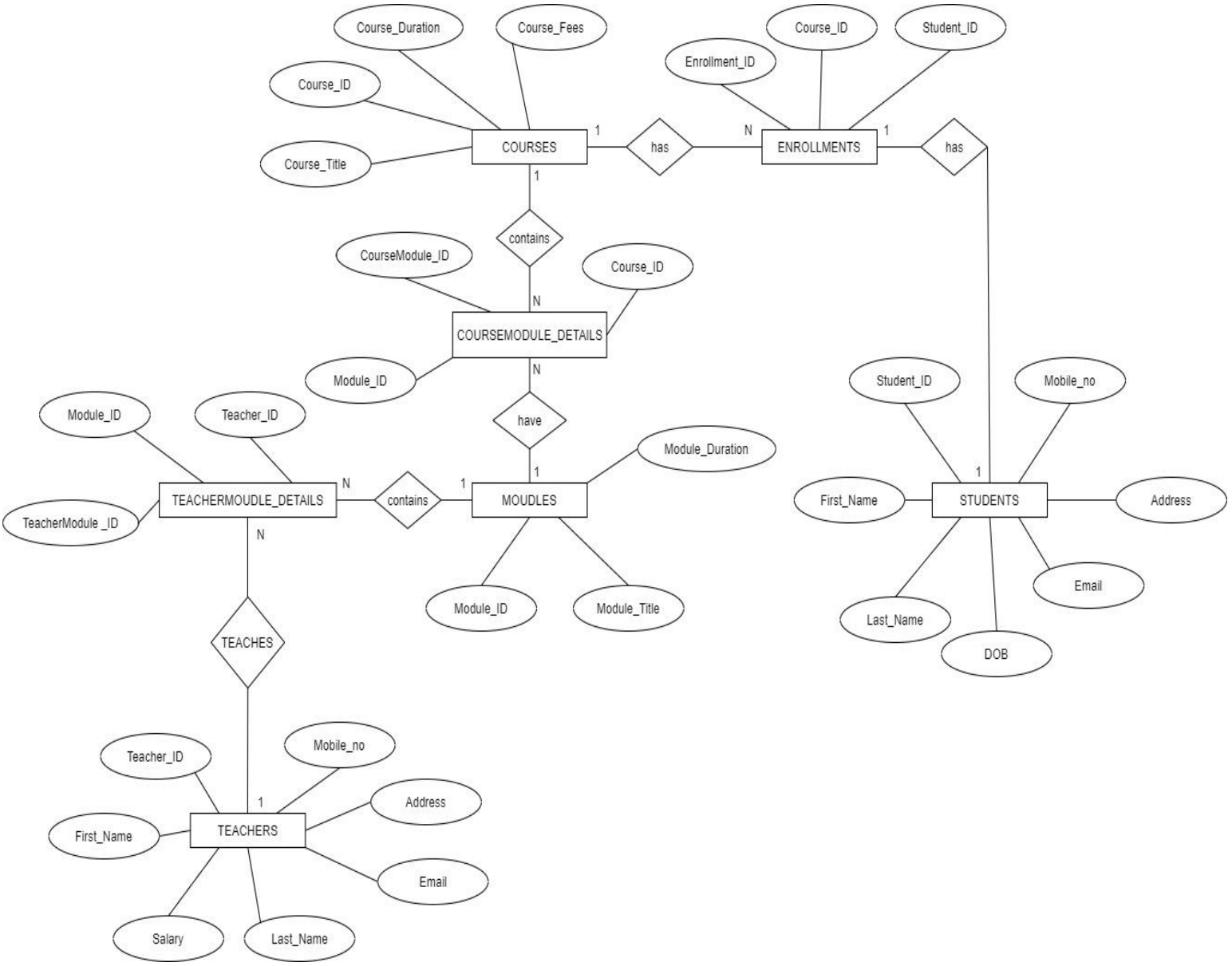

Figure 1: Entity Relation Model Diagram of CollegeCoursesManagement.

2.3 Relational Model Diagram

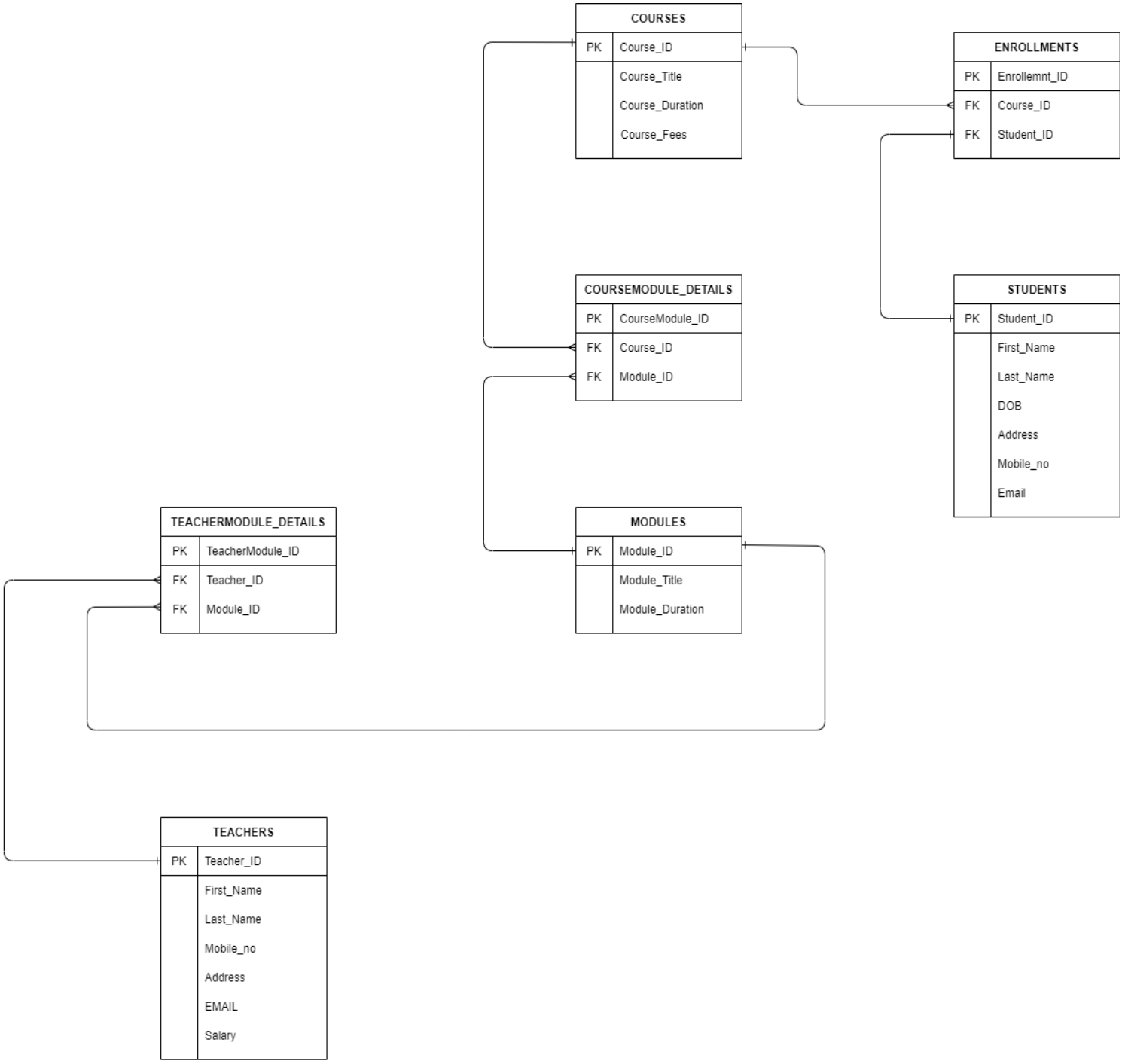

Figure 2: Relation Model Diagram Of CollegeCoursesManagement.

2.4 List Of Tables

Table name: - COURSES

This entity represents list of available courses offered by college. It helps to provide detail information on each course which helps students in enrolling courses.

Attributes

Course_ID: - ID of each course. It stores unique integer data for each record in table. It helps to identify each record for each course in table uniquely.

Course_Title: - Title of each Course. It stores variable character data for title or name of each course. It gives information on name of course.

Course_Duration: - Duration for each course completion. It stores variable character data for duration of course. It gives information of total years needed for completion of each course.

Course_Fees: - Total amount of Fees for each course. It stores integer data for fees of each course. It gives information of total payment need to enrol for each course.

CREATION OF COURSES TABLE

```
MariaDB [collegecoursesmanagement]> CREATE TABLE COURSES(
    -> Course_ID INT PRIMARY KEY NOT NULL UNIQUE AUTO_INCREMENT,
    -> Course_Title VARCHAR(35) NOT NULL,
    -> Course_Duration VARCHAR(20) DEFAULT "Not Available",
    -> Course_Fees INT NOT NULL);
Query OK, 0 rows affected (0.026 sec)

MariaDB [collegecoursesmanagement]> DESC COURSES;
+----------------+-------------+------+-----+---------------+----------------+
| Field          | Type        | Null | Key | Default       | Extra          |
+----------------+-------------+------+-----+---------------+----------------+
```

Figure 3: CREATE QUERY OF COURSES TABLE.

DESCRIPTION OF COURSES TABLE

```
MariaDB [collegecoursesmanagement]> DESC COURSES;
+-----------------+-------------+------+-----+---------------+----------------+
| Field           | Type        | Null | Key | Default       | Extra          |
+-----------------+-------------+------+-----+---------------+----------------+
| Course_ID       | int(11)     | NO   | PRI | NULL          | auto_increment |
| Course_Title    | varchar(35) | NO   |     | NULL          |                |
| Course_Duration | varchar(20) | YES  |     | Not Available |                |
| Course_Fees     | int(11)     | NO   |     | NULL          |                |
+-----------------+-------------+------+-----+---------------+----------------+
4 rows in set (0.028 sec)
```

Figure 4: DESC QUERY OF COURSES TABLE.

DATA INSERTION IN COURSES TABLE

```
MariaDB [collegecoursesmanagement]> INSERT INTO COURSES VALUES
    -> (1,"Networking And IT Security","3 YEARS",1400000),
    -> (2,"Computing","3 YEARS",1300000),
    -> (3,"Multimedia","3 YEARS",1450000),
    -> (4,"Business","4 YEARS",1250000),
    -> (5,"Arts","4 YEARS",1000000);
Query OK, 5 rows affected (0.005 sec)
Records: 5  Duplicates: 0  Warnings: 0

MariaDB [collegecoursesmanagement]> SELECT * FROM COURSES;
```

Figure 5: INSERT QUERY OF COURSES TABLE.

SELECTING ALL DATA FROM COURSES

```
MariaDB [collegecoursesmanagement]> SELECT * FROM COURSES;
+-----------+---------------------------+-----------------+-------------+
| Course_ID | Course_Title              | Course_Duration | Course_Fees |
+-----------+---------------------------+-----------------+-------------+
|         1 | Networking And IT Security | 3 YEARS         |     1400000 |
|         2 | Computing                 | 3 YEARS         |     1300000 |
|         3 | Multimedia                | 3 YEARS         |     1450000 |
|         4 | Business                  | 4 YEARS         |     1250000 |
|         5 | Arts                      | 4 YEARS         |     1000000 |
+-----------+---------------------------+-----------------+-------------+
5 rows in set (0.001 sec)
```

Figure 6: SELECT QUERY OF COURSES TABLE.

Table name: - STUDENTS

This entity represents students enrolled in college. It helps to provide detail information of each student like name, phone, address etc.

Attributes

Student_ID: - ID of each student. It stores unique integer data for each record in table. It helps to identify each record of each student in table uniquely.

First_Name: - First name of each student. It stores variable character data for first name of each student. It gives information on first name of each student.

Last_Name: - Last name of each student. It stores variable character data for last of each student. It gives information of last name of each student.

DOB: - Date of birth of each student. It stores date of birth of each student. It provides information date of birth of each student.

Mobile no: - Mobile contact number of each student. It stores integer data for mobile number of each student. It gives information of contact number for each student.

Address: - Address of each student. It stores variable character data for home location of each student. It is gives information on home location of each student.

Email: - Email of each student. It stores variable character data for electronic mail address of each student. It gives information of email address for contact of each student.

CREATING STUDENTS TABLE

```
MariaDB [collegecoursesmanagement]> CREATE TABLE STUDENTS(
    -> Student_ID INT PRIMARY KEY NOT NULL UNIQUE AUTO_INCREMENT,
    -> First_Name VARCHAR(25) NOT NULL,
    -> Last_Name VARCHAR(25) NOT NULL,
    -> Mobile_no INT NOT NULL UNIQUE,
    -> DOB DATE NOT NULL,
    -> Address VARCHAR(50) NOT NULL,
    -> Email VARCHAR(60) UNIQUE DEFAULT "Not Available");
Query OK, 0 rows affected (0.041 sec)
```

Figure 7: CREATE QUERY OF STUDENTS TABLES.

DESCRIPTION OF STUDENTS TABLE

```
MariaDB [collegecoursesmanagement]> DESC STUDENTS;
+------------+-------------+------+-----+---------------+----------------+
| Field      | Type        | Null | Key | Default       | Extra          |
+------------+-------------+------+-----+---------------+----------------+
| Student_ID | int(11)     | NO   | PRI | NULL          | auto_increment |
| First_Name | varchar(25) | NO   |     | NULL          |                |
| Last_Name  | varchar(25) | NO   |     | NULL          |                |
| Mobile_no  | int(11)     | NO   | UNI | NULL          |                |
| DOB        | date        | NO   |     | NULL          |                |
| Address    | varchar(50) | NO   |     | NULL          |                |
| Email      | varchar(60) | YES  | UNI | Not Available |                |
+------------+-------------+------+-----+---------------+----------------+
7 rows in set (0.029 sec)
```

Figure 8: DESC QUERY OF STUDENTS TABLE.

DATA INSERTION IN STUDENTS TABLE

```
MariaDB [collegecoursesmanagement]> INSERT INTO STUDENTS VALUES
    -> (1,"Raju","Tamang",9833249,"1993-09-12","Kathmandu","raju@gmail.com"),
    -> (2,"Pritam","Shrestha",9833041,"1991-05-14","Pokhara","pritam@gmail.com"),
    -> (3,"Hemant","Rana",9801098,"1996-09-23","Bhaktapur","hemant@gmail.com"),
    -> (4,"Gaurav","Shrestha",98010222,"1993-04-13","Kannpur","gaurav@gmail.com"),
    -> (5,"Jiwan","Limbu",98993344,"1991-05-23","Damak","jiwan@gmail.com"),
    -> (6,"Ram","Rai",9833189,"1992-09-12","Itahari","ram@gmail.com"),
    -> (7,"Romeo","Rai",98011772,"1994-08-12","Lalitpur","romeo@gmail.com"),
    -> (8,"Jiwan","Rai",98011755,"1995-08-02","Biratnagar","jiwanrai@gmail.com"),
    -> (9,"Sita","Gurung",98012300,"1995-06-05","Dharan","sita@gmail.com"),
    -> (10,"Resham","Gurung",9800566,"1992-01-12","Lalitpur","resham@gmail.com"),
    -> (11,"Rosy","Tamang",9882233,"1991-01-12","Itahari","rosy@gmail.com"),
    -> (12,"Sonam","Tamang",98112233,"1994-06-17","Dharan","sonam@gmail.com");
Query OK, 12 rows affected (0.011 sec)
Records: 12  Duplicates: 0  Warnings: 0
```

Figure 9: INSERT QUERY IN STUDENTS TABLE.

SELECTING ALL DATA FROM STUDENTS TABLE

```
MariaDB [collegecoursesmanagement]> SELECT * FROM STUDENTS;
+------------+------------+-----------+-----------+------------+------------+--------------------+
| Student_ID | First_Name | Last_Name | Mobile_no | DOB        | Address    | Email              |
+------------+------------+-----------+-----------+------------+------------+--------------------+
|          1 | Raju       | Tamang    |   9833249 | 1993-09-12 | Kathmandu  | raju@gmail.com     |
|          2 | Pritam     | Shrestha  |   9833041 | 1991-05-14 | Pokhara    | pritam@gmail.com   |
|          3 | Hemant     | Rana      |   9801098 | 1996-09-23 | Bhaktapur  | hemant@gmail.com   |
|          4 | Gaurav     | Shrestha  |  98010222 | 1993-04-13 | Kannpur    | gaurav@gmail.com   |
|          5 | Jiwan      | Limbu     |  98993344 | 1991-05-23 | Damak      | jiwan@gmail.com    |
|          6 | Ram        | Rai       |   9833189 | 1992-09-12 | Itahari    | ram@gmail.com      |
|          7 | Romeo      | Rai       |  98011772 | 1994-08-12 | Lalitpur   | romeo@gmail.com    |
|          8 | Jiwan      | Rai       |  98011755 | 1995-08-02 | Biratnagar | jiwanrai@gmail.com |
|          9 | Sita       | Gurung    |  98012300 | 1995-06-05 | Dharan     | sita@gmail.com     |
|         10 | Resham     | Gurung    |   9800566 | 1992-01-12 | Lalitpur   | resham@gmail.com   |
|         11 | Rosy       | Tamang    |   9882233 | 1991-01-12 | Itahari    | rosy@gmail.com     |
|         12 | Sonam      | Tamang    |  98112233 | 1994-06-17 | Dharan     | sonam@gmail.com    |
+------------+------------+-----------+-----------+------------+------------+--------------------+
12 rows in set (0.003 sec)
```

Figure 10: SELECT QUERY OF STUDENTS TABLE.

Table name: - TEACHERS

This entity represents teachers teaching in college. It provides detail information of each teacher like name, phone, address etc.

Attributes

Teacher_ID: - ID of each teacher. It stores unique integer data for each record in TEACHERS table. It helps to identify each record of each teacher in table uniquely.

First_Name: - First Name of each Teacher. It stores variable character data for first name of each teacher. It gives information on first name of each teacher.

Last_Name: - Last Name of each Teacher. It stores variable character data for last of each teacher. It provides information of last name of each teacher.

Mobile_no: - Moblie contact number of each Teacher. It stores integer data for mobile number of each teacher. It gives information of contact number for each teacher.

Address: - Address of each Teacher. It stores variable character data for home location of each teacher. It is gives information on home location of each teacher.

Email: - Email address of each teacher. It stores variable character data for electronic mail of each teacher. It gives information of email address for contact of each teacher.

Salary: -Total amount of salary of each teacher. It stores integer data for salary of teacher. It provides information on payment receive by teacher for teaching.

CREATION OF TEACHERS TABLE

```
MariaDB [collegecoursesmanagement]> CREATE TABLE TEACHERS(
    -> Teacher_ID INT PRIMARY KEY NOT NULL UNIQUE AUTO_INCREMENT,
    -> First_Name VARCHAR(25) NOT NULL,
    -> Last_Name VARCHAR(25) NOT NULL,
    -> Mobile_no INT NOT NULL UNIQUE,
    -> Address VARCHAR(50) NOT NULL,
    -> Email VARCHAR(60) UNIQUE DEFAULT "Not Available",
    -> Salary INT NOT NULL);
Query OK, 0 rows affected (0.032 sec)
```

Figure 11: CREATE QUERY OF TEACHERS TABLE.

DESCRIPTION OF TEACHERS TABLE

```
MariaDB [collegecoursesmanagement]> DESC TEACHERS;
+------------+-------------+------+-----+---------------+----------------+
| Field      | Type        | Null | Key | Default       | Extra          |
+------------+-------------+------+-----+---------------+----------------+
| Teacher_ID | int(11)     | NO   | PRI | NULL          | auto_increment |
| First_Name | varchar(25) | NO   |     | NULL          |                |
| Last_Name  | varchar(25) | NO   |     | NULL          |                |
| Mobile_no  | int(11)     | NO   | UNI | NULL          |                |
| Address    | varchar(50) | NO   |     | NULL          |                |
| Email      | varchar(60) | YES  | UNI | Not Available |                |
| Salary     | int(11)     | NO   |     | NULL          |                |
+------------+-------------+------+-----+---------------+----------------+
7 rows in set (0.037 sec)
```

Figure 12: DESC QUERY OF TACHERS TABLE.

DATA INSERTION IN TACHERS TABLE

```
MariaDB [collegecoursesmanagement]> INSERT INTO TEACHERS VALUES
    -> (1,"Janaki","Chaudary",98083450,"Pokhara","janaki@gmail.com",60000),
    -> (2,"Sofiya","Gajurel",980947457,"Kathmandu","sofiya@gmail.com",50000),
    -> (3,"Anuj","Shilpakar",9850367,"Kathmandu","anuj@gmail.com",45000),
    -> (4,"Pasang","Tamang",9001123,"Dharan","pasang@gmail.com",55000),
    -> (5,"Kriti","Rai",9779901,"Itahari","kriti@gmail.com",65000),
    -> (6,"Aadit","shrestha",9221990,"Bhaktapur","aadit@gmail.com",67000),
    -> (7,"Aafsa","Roy",998190450,"Kannpur","aafsa@gmail.com",59000),
    -> (8,"Bishwas","Limbu",980834803,"Dharan","bishwas@gmail.com",43000),
    -> (9,"Bisu","Rai",980834812,"Itahari","bisu@gmail.com",47000),
    -> (10,"Rajiv","Tamang",9091234,"Pokhara","rajiv@gmail.com",42000),
    -> (11,"Sangam","Chhetri",9109160,"Itahari","sangam@gmail.com",44000),
    -> (12,"Hari","Gurung",9903450,"Kathmandu","hari@gmail.com",50000);
Query OK, 12 rows affected (0.005 sec)
Records: 12  Duplicates: 0  Warnings: 0
```

Figure 13: INSERT QUERY OF TACHERS TABLE.

SELECTION OF ALL DATA IN TEACHERS TABLE

```
MariaDB [collegecoursesmanagement]> SELECT * FROM TEACHERS;
+------------+------------+-----------+-----------+-----------+--------------------+--------+
| Teacher_ID | First_Name | Last_Name | Mobile_no | Address   | Email              | Salary |
+------------+------------+-----------+-----------+-----------+--------------------+--------+
|          1 | Janaki     | Chaudary  |  98083450 | Pokhara   | janaki@gmail.com   |  60000 |
|          2 | Sofiya     | Gajurel   | 980947457 | Kathmandu | sofiya@gmail.com   |  50000 |
|          3 | Anuj       | Shilpakar |   9850367 | Kathmandu | anuj@gmail.com     |  45000 |
|          4 | Pasang     | Tamang    |   9001123 | Dharan    | pasang@gmail.com   |  55000 |
|          5 | Kriti      | Rai       |   9779901 | Itahari   | kriti@gmail.com    |  65000 |
|          6 | Aadit      | shrestha  |   9221990 | Bhaktapur | aadit@gmail.com    |  67000 |
|          7 | Aafsa      | Roy       | 998190450 | Kannpur   | aafsa@gmail.com    |  59000 |
|          8 | Bishwas    | Limbu     | 980834803 | Dharan    | bishwas@gmail.com  |  43000 |
|          9 | Bisu       | Rai       | 980834812 | Itahari   | bisu@gmail.com     |  47000 |
|         10 | Rajiv      | Tamang    |   9091234 | Pokhara   | rajiv@gmail.com    |  42000 |
|         11 | Sangam     | Chhetri   |   9109160 | Itahari   | sangam@gmail.com   |  44000 |
|         12 | Hari       | Gurung    |   9903450 | Kathmandu | hari@gmail.com     |  50000 |
+------------+------------+-----------+-----------+-----------+--------------------+--------+
12 rows in set (0.000 sec)
```

Figure 14: SELECT QUERY OF TEACHERS TABLE.

Table Name: - MOUDLES

This entity represents list of modules for each course. It gives details information of each module belonging in each course.

Attributes

Module_ID: - ID of each module. It stores unique integer data for each record in MODULES table. It helps to identify each record of each module in table uniquely.

Module_Title: - Title of each module. It stores variable character data for title or name of each module. It gives information on name of module.

Module_Duration: - Duration of each course completion. It stores variable character data for duration to complete each module. It gives information of total weeks need to finish each module.

CREATION OF MODULES TABLE

```
MariaDB [collegecoursesmanagement]> CREATE TABLE COURSES(
    -> Course_ID INT PRIMARY KEY NOT NULL UNIQUE AUTO_INCREMENT,
    -> Course_Title VARCHAR(35) NOT NULL,
    -> Course_Duration VARCHAR(20) DEFAULT "Not Available",
    -> Course_Fees INT NOT NULL);
Query OK, 0 rows affected (0.026 sec)

MariaDB [collegecoursesmanagement]> DESC COURSES;
+-----------------+-------------+------+-----+---------------+----------------+
| Field           | Type        | Null | Key | Default       | Extra          |
+-----------------+-------------+------+-----+---------------+----------------+
```

Figure 15: CREATE QUERY OF MODULES TABLE.

DESCRIPTION OF MODULES TABLE

```
MariaDB [collegecoursesmanagement]> DESC COURSES;
+-----------------+-------------+------+-----+---------------+----------------+
| Field           | Type        | Null | Key | Default       | Extra          |
+-----------------+-------------+------+-----+---------------+----------------+
| Course_ID       | int(11)     | NO   | PRI | NULL          | auto_increment |
| Course_Title    | varchar(35) | NO   |     | NULL          |                |
| Course_Duration | varchar(20) | YES  |     | Not Available |                |
| Course_Fees     | int(11)     | NO   |     | NULL          |                |
+-----------------+-------------+------+-----+---------------+----------------+
4 rows in set (0.028 sec)
```

Figure 16: DESC QUERY OF MODULES TABLE.

DATA INSERTION IN MODULES TABLE

```
MariaDB [collegecoursesmanagement]> INSERT INTO MODULES VALUES
    -> (1,"Information System","12 Week"),
    -> (2,"Programing","28 Week"),
    -> (3,"communication Engineering","28 Week"),
    -> (4,"Fundamental of Computing","12 Week"),
    -> (5,"Logic and Problem Solving","12 Week"),
    -> (6,"Digital Desgin and Image Making","28 Week"),
    -> (7,"Computer Hardware and Software","12 Week"),
    -> (8,"Introducton to IT","12 Week"),
    -> (9,"Psychology","12 Week"),
    -> (10,"Sociology","12 Week"),
    -> (11,"Foundational Mathematics","28 Week");
Query OK, 11 rows affected (0.044 sec)
Records: 11  Duplicates: 0  Warnings: 0
```

Figure 17: INSERT QUERY OF MODULES TABLE.

SELECTION OF ALL DATA IN MODULES TABLE

```
MariaDB [collegecoursesmanagement]> SELECT * FROM MODULES;
+-----------+---------------------------------+-----------------+
| Module_ID | Module_Title                    | Module_Duration |
+-----------+---------------------------------+-----------------+
|         1 | Information System              | 12 Week         |
|         2 | Programing                      | 28 Week         |
|         3 | communication Engineering       | 28 Week         |
|         4 | Fundamental of Computing        | 12 Week         |
|         5 | Logic and Problem Solving       | 12 Week         |
|         6 | Digital Desgin and Image Making | 28 Week         |
|         7 | Computer Hardware and Software  | 12 Week         |
|         8 | Introducton to IT               | 12 Week         |
|         9 | Psychology                      | 12 Week         |
|        10 | Sociology                       | 12 Week         |
|        11 | Foundational Mathematics        | 28 Week         |
+-----------+---------------------------------+-----------------+
11 rows in set (0.010 sec)
```

Figure 18: SELECT QUERY OF MODULES TABLE.

Table Name: - ENROLLMENTS

This entity shows enrolments of each student in each course in college. It provides detail information of numbers of enrolment of each student in each course.

Attributes

Enrollment_ID: - ID of each enrolment. It stores unique integer data for each record in table. It helps to identify each record of each enrolment in table uniquely.

Course_ID: - Course_ID is ID of each course which references from “Course_ID” attribute of COURSES table.

Student_ID: - Student_ID is ID of each student which references from “Student_ID” attribute of STUDENTS table.

CREATION OF ENROLLMENTS TABLE

```
MariaDB [collegecoursesmanagement]> CREATE TABLE ENROLLMENTS(
    -> Enrollment_ID  INT PRIMARY KEY NOT NULL UNIQUE AUTO_INCREMENT,
    -> Course_ID INT NOT NULL,
    -> Student_ID INT NOT NULL,
    -> FOREIGN KEY(Course_ID) REFERENCES COURSES(Course_ID),
    -> FOREIGN KEY(Student_ID) REFERENCES STUDENTS(Student_ID));
Query OK, 0 rows affected (0.031 sec)
```

Figure 19: CREATE QUERY OF ENROLLMENTS TABLE.

DESCRIPTION OF ENROLLMENTS TABLE

```
MariaDB [collegecoursesmanagement]> DESC ENROLLMENTS;
+---------------+---------+------+-----+---------+----------------+
| Field         | Type    | Null | Key | Default | Extra          |
+---------------+---------+------+-----+---------+----------------+
| Enrollment_ID | int(11) | NO   | PRI | NULL    | auto_increment |
| Course_ID     | int(11) | NO   | MUL | NULL    |                |
| Student_ID    | int(11) | NO   | MUL | NULL    |                |
+---------------+---------+------+-----+---------+----------------+
3 rows in set (0.026 sec)
```

Figure 20: DESC QUERY OF ENROLLMENTS TABLE

DATA INSERTION IN ENROLLMENTS TABLE

```
MariaDB [collegecoursesmanagement]> INSERT INTO ENROLLMENTS VALUES
    -> (1,1,1),
    -> (2,1,2),
    -> (3,1,3),
    -> (4,2,4),
    -> (5,2,5),
    -> (6,3,6),
    -> (7,3,7),
    -> (8,3,8),
    -> (9,4,9),
    -> (10,4,10),
    -> (11,5,11),
    -> (12,5,12);
Query OK, 12 rows affected (0.010 sec)
Records: 12  Duplicates: 0  Warnings: 0
```

Figure 21: INSERT QUERY OF ENROLLMENTS TABLE.

SELECTION OF ALL DATA OF ENROLLMENTS TABLE

```
MariaDB [collegecoursesmanagement]> SELECT * FROM ENROLLMENTS;
+---------------+-----------+------------+
| Enrollment_ID | Course_ID | Student_ID |
+---------------+-----------+------------+
|             1 |         1 |          1 |
|             2 |         1 |          2 |
|             3 |         1 |          3 |
|             4 |         2 |          4 |
|             5 |         2 |          5 |
|             6 |         3 |          6 |
|             7 |         3 |          7 |
|             8 |         3 |          8 |
|             9 |         4 |          9 |
|            10 |         4 |         10 |
|            11 |         5 |         11 |
|            12 |         5 |         12 |
+---------------+-----------+------------+
12 rows in set (0.001 sec)
```

Figure 22: SELECT QUERY OF ENROLLMENTS TABLE.

Table Name: - COURSEMODULE_DETAILS

This is also a bridge entity between COURSES and MOUDLES, it joins two different tables. Because of many to many relationships between COURSES and MODULES tables. Its gives detail information about “each module belonging to each course”.

Attributes

CourseModule_ID: - ID of each record in COURSEMODULE_DETAILS table. It stores unique integer data for each record in “COURSEMODULE_DETAILS” table. It helps to identify each record in “COURSEMODULE_DETAILS” Table uniquely.

Course_ID: - Course_ID is ID of each course which references from “Course_ID” attribute of COURSES table.

Module_ID: - Module_ID is ID of each module which references from “Module_ID” attribute of MODULES table.

CREATION OF COURSEMODULE_DETAILS TABLE

```
MariaDB [collegecoursesmanagement]> CREATE TABLE COURSEMODULE_DETAILS(
    -> CourseModule_ID INT PRIMARY KEY NOT NULL UNIQUE AUTO_INCREMENT,
    -> Course_ID INT NOT NULL,
    -> Module_ID INT NOT NULL,
    -> FOREIGN KEY(Course_ID) REFERENCES COURSES(Course_ID),
    -> FOREIGN KEY(Module_ID) REFERENCES MODULES(Module_ID));
Query OK, 0 rows affected (0.030 sec)

MariaDB [collegecoursesmanagement]> DESC COURSEMODULE_DETAILS;
```

Figure 23: CREATE QUERY OF COURSEMODULE_DETAILS.

DESCRIPTION IN COURSEMODULE_DETAILS TABLE

```
MariaDB [collegecoursesmanagement]> DESC COURSEMODULE_DETAILS;
+-----------------+---------+------+-----+---------+----------------+
| Field           | Type    | Null | Key | Default | Extra          |
+-----------------+---------+------+-----+---------+----------------+
| CourseModule_ID | int(11) | NO   | PRI | NULL    | auto_increment |
| Course_ID       | int(11) | NO   | MUL | NULL    |                |
| Module_ID       | int(11) | NO   | MUL | NULL    |                |
+-----------------+---------+------+-----+---------+----------------+
3 rows in set (0.032 sec)
```

Figure 24: DESC QUERY OF COURSEMODULE_DETAILS TABLE.

INSERTION OF DATA IN COURSEMODULE_DETAILS TABLE

```
MariaDB [collegecoursesmanagement]> INSERT INTO COURSEMODULE_DETAILS VALUES
    -> (1,1,1),
    -> (2,1,2),
    -> (3,1,3),
    -> (4,1,4),
    -> (5,2,1),
    -> (6,2,2),
    -> (7,2,4),
    -> (8,2,5),
    -> (9,2,7),
    -> (10,3,6),
    -> (11,3,2),
    -> (12,3,4),
    -> (13,4,8),
    -> (14,4,11),
    -> (15,5,11),
    -> (16,5,9),
    -> (17,5,10);
Query OK, 17 rows affected (0.012 sec)
Records: 17  Duplicates: 0  Warnings: 0
```

Figure 25: INSERT QUERY IN COURSEMODULE_DETAILS TABLE.

SELECTION OF ALL DATA IN COURSEMODULE_DETAILS TABLE

```
MariaDB [collegecoursesmanagement]> SELECT * FROM COURSEMODULE_DETAILS;
+-----------------+-----------+-----------+
| CourseModule_ID | Course_ID | Module_ID |
+-----------------+-----------+-----------+
|               1 |         1 |         1 |
|               2 |         1 |         2 |
|               3 |         1 |         3 |
|               4 |         1 |         4 |
|               5 |         2 |         1 |
|               6 |         2 |         2 |
|               7 |         2 |         4 |
|               8 |         2 |         5 |
|               9 |         2 |         7 |
|              10 |         3 |         6 |
|              11 |         3 |         2 |
|              12 |         3 |         4 |
|              13 |         4 |         8 |
|              14 |         4 |        11 |
|              15 |         5 |        11 |
|              16 |         5 |         9 |
|              17 |         5 |        10 |
+-----------------+-----------+-----------+
17 rows in set (0.000 sec)
```

Figure 26: SELECT QUERY OF COURSEMODULE_DETAILS TABLE.

Table Name: - TEACHERMOUDLE_DETAILS

As it is a bridge entity between TEACHERS and MOUDLES, it joins two different table. Because of many to many relationships between TEACHERS and MODULES tables. It helps to give detail information about “a module taught by a teacher”.

Attributes

TeacherModule _ID: - ID of each record in TEACHERMODULE_DETAILS table. It stores unique integer data for each record in “TEACHERMODULE_DETAILS” Table. It helps to identify each record in “TEACHERMODULE_DETAILS” Table uniquely.

Teacher_ID: - Teacher_ID is Id of each teacher which references from “Teacher_ID” attribute of TEACHERS table.

Module_ID: - Module_ID is ID of each module which references from “Module_ID” attribute of MODULES table.

CREATION OF TEACHERMOUDLE_DETAILS TABLE

```
MariaDB [collegecoursesmanagement]> CREATE TABLE TEACHERMODULE_DETAILS(
    -> TeacherModule_ID INT PRIMARY KEY NOT NULL UNIQUE AUTO_INCREMENT,
    -> Teacher_ID INT NOT NULL,
    -> Module_ID INT NOT NULL,
    -> FOREIGN KEY(Teacher_ID) REFERENCES TEACHERS(Teacher_ID),
    -> FOREIGN KEY(Module_ID) REFERENCES MODULES(Module_ID));
Query OK, 0 rows affected (0.028 sec)
```

Figure 27: CREATE QUERY OF TEACHERMOUDLE_DETAILS TABLE.

DESCRIPTION OF TEACHCERMODULE_DETAILS TABLE

```
MariaDB [collegecoursesmanagement]> DESC TEACHERMODULE_DETAILS;
+------------------+---------+------+-----+---------+----------------+
| Field            | Type    | Null | Key | Default | Extra          |
+------------------+---------+------+-----+---------+----------------+
| TeacherModule_ID | int(11) | NO   | PRI | NULL    | auto_increment |
| Teacher_ID       | int(11) | NO   | MUL | NULL    |                |
| Module_ID        | int(11) | NO   | MUL | NULL    |                |
+------------------+---------+------+-----+---------+----------------+
3 rows in set (0.024 sec)
```

Figure 28: DESC QUERY OF TACHERMODULE_DETAILS TABLE.

DATA INSERTION IN TACHERMODULE_DETAILS TABLE

```
MariaDB [collegecoursesmanagement]> INSERT INTO TEACHERMODULE_DETAILS VALUES
    -> (1,1,1),
    -> (2,1,4),
    -> (3,2,3),
    -> (4,3,3),
    -> (5,4,2),
    -> (6,5,2),
    -> (7,6,1),
    -> (8,7,5),
    -> (9,8,6),
    -> (10,9,7),
    -> (11,7,8),
    -> (12,10,9),
    -> (13,11,10),
    -> (14,12,11);
Query OK, 14 rows affected (0.007 sec)
Records: 14  Duplicates: 0  Warnings: 0
```

Figure 29: INSERT QUERY OF TACHERMODULE_DETAILS TABLE.

SELECTION OF ALL DATA IN TEACHERMOUDLE_DETAILS TABLE

```
MariaDB [collegecoursesmanagement]> SELECT * FROM TEACHERMODULE_DETAILS;
+------------------+------------+-----------+
| TeacherModule_ID | Teacher_ID | Module_ID |
+------------------+------------+-----------+
|                1 |          1 |         1 |
|                2 |          1 |         4 |
|                3 |          2 |         3 |
|                4 |          3 |         3 |
|                5 |          4 |         2 |
|                6 |          5 |         2 |
|                7 |          6 |         1 |
|                8 |          7 |         5 |
|                9 |          8 |         6 |
|               10 |          9 |         7 |
|               11 |          7 |         8 |
|               12 |         10 |         9 |
|               13 |         11 |        10 |
|               14 |         12 |        11 |
+------------------+------------+-----------+
14 rows in set (0.001 sec)
```

Figure 30: SELECT QUERY OF TEACHERMOUDLE_DETAILS TABLE.

3. Data Dictionary Of Entities

COURSES

Entity Name	Entity Description	Column Name	Column Description	Data type	Length	Primary Key	Foreign Key	Null able	unique	Notes
COURSES	This entity represents list of available courses offered by college. It helps to provide detail information on each course which helps students in enrolling courses.	Course_ID	ID of each course, identifies each course uniquely.	INT		TRUE	FALSE	FALSE	TRUE	Atuo incremented
		Course_Title	Name of each course.	VARCHAR	35	FALSE	FALSE	FALSE	FALSE	
		Course_Duration	Duration of each course to complete.	VARCHAR	20	FALSE	FALSE	TRUE	FALSE	Default "Not Available"
		Course_Fees	Amount of fees to be paid for each course enrolled.	INT		FALSE	FALSE	FALSE	FALSE	

Table 1: Data Dictionary Of COURSES.

STUDENTS

Entity Name	Entity Description	Column Name	Column Description	Data type	Length	Primary Key	Foreign Key	Null able	unique	Notes
STUDENTS	This entity represents students enrolled in college. It helps to provide detail information of students like name, phone, address etc	Student_ID	ID of each student, identifies each student uniquely.	INT		TRUE	FALSE	FALSE	TRUE	Atuo incremented
		First_Name	First Name of each student.	VARCHAR	25	FALSE	FALSE	FALSE	FALSE	
		Last_Name	Last Name of each student	VARCHAR	25	FALSE	FALSE	FALSE	FALSE	
		Mobile_no	Contact mobile number of each student.	INT		FALSE	FALSE	FALSE	TRUE	
		DOB	Date of birth of each student.	DATE		FALSE	FALSE	FALSE	FALSE	
		Address	Home location address of each student.	VARCHAR	50	FALSE	FALSE	FALSE	FALSE	
		Email	Email address of each student.	VARCHAR	60	FALSE	FALSE	TRUE	TRUE	Default "Not Available"

Table 2: Data Dictionary Of STUDENTS.

TEACHERS

Entity Name	Entity Description	Column Name	Column Description	Data type	Length	Primary Key	Foreign Key	Null able	unique	Notes
TEACHERS	This entity represents teachers teaching in college. It provides detail information of teachers details like name, phone, address etc.	Teacher_ID	ID of each student, identifies each student uniquely.	INT		TRUE	FALSE	FALSE	TRUE	Atuo incremented
		First_Name	First Name of each student.	VARCHAR	25	FALSE	FALSE	FALSE	FALSE	
		Last_Name	Last Name of each student	VARCHAR	25	FALSE	FALSE	FALSE	FALSE	
		Mobile_no	Contact mobile number of each student.	INT		FALSE	FALSE	FALSE	TRUE	
		Address	Home location address of each student.	VARCHAR	50	FALSE	FALSE	FALSE	FALSE	
		Email	Email address of each student.	VARCHAR	60	FALSE	FALSE	FALSE	TRUE	Default "Not Available"
		Salary	Amount of salary received by each Teacher.	INT		FALSE	FALSE	FALSE	FALSE	

Table 3: Data Dictionary Of TEACHERS.

MODULES

Entity Name	Entity Description	Column Name	Column Description	Data type	Length	Primary Key	Foreign Key	Null able	unique	Notes
MODULES	This entity represents list of modules for each course. It gives details information of each module belonging in each course.	Module_ID	ID of each module, identifies each module uniquely.	INT		TRUE	FALSE	FALSE	TRUE	Atuo incremented.
		Module_Title	Name of each module.	VARCHAR	35	FALSE	FALSE	FALSE	FALSE	
		Module_Duration	Duration for each module to complete.	VARCHAR	20	FALSE	FALSE	TRUE	FALSE	Default "Not Available"

Table 4: Data Dictionary Of MODULES.

ENROLLMENTS

Entity Name	Entity Description	Column Name	Column Description	Data type	Length	Primary Key	Foreign Key	Null able	unique	Notes
ENROLLMENTS	This entity shows enrolments of each student in each course in college. It provides detail information of numbers of enrolment of each student in each course.	Enrollment_ID	ID of each enrolment record, identifies each record in ENROLLMENTS table uniquely.	INT		TRUE	FALSE	FALSE	TRUE	Atuo incremented.
		Course_ID	Course_ID is a Id of each course and also foreign key which references from "COURSE_ID" attribute of COURSES table.	INT		FALSE	TRUE	FALSE	FALSE	References from "Course_ID" attribute of COURSES table.
		Student_ID	Student_ID is Id each student and also a foreign key which references from "Student_ID" attribute of STUDENTS	INT		FALSE	TRUE	FALSE	FALSE	References from "Student_ID" attribute of STUDENTS table.

Table 5: Data Dictionary Of ENROLLMENTS.

COURSEMODEL_DETAILS

Entity Name	Entity Description	Column Name	Column Description	Data type	Length	Primary Key	Foreign Key	Null able	unique	Notes
COURSEMODEL_DETAILS	This is also a bridge entity between COURSES and MOUDLES, it joins two different entities. It is fromed because of many to many realtionship between COURSES and MODULES entities. Its gives detail information about "each module belonging to each course".	CourseModule_ID	ID of each record in COURSEMODUEL_DETAILS table., identifies each record in COURSEMODUEL_DETAILS table uniquely..	INT		TRUE	FALSE	FALSE	TRUE	Atuo incremented .
		Course_ID	Course_ID is a Id of each course and also foreign key which references from "COURSE_ID" attribute of COURSES table.	INT		FALSE	TRUE	FALSE	FALSE	References from "Course_ID" attribute of COURSES table.
		Module_ID	Module_ID is a id of each module and also a foreign key which references from "Module_ID" attribute of MODULES table.	INT		FALSE	TRUE	FALSE	FALSE	References from "Module_ID" attribute of MODULES table.

Table 6: Data Dictionary Of COUREMODEL_DETAILS.

TEACHERMODULE_DETAILS

Entity Name	Entity Description	Column Name	Column Description	Data type	Length	Primary Key	Foreign Key	Null able	unique	Notes
TEACHERMODULE_DETAILS	As it is a bridge entity between TEACHERS and MOUDLES, it joins two different entities. It is fromed because of many to many realtionship between TEACHERS and MODULES entities . It helps to give detail information about "a module taught by a teacher".	TeacherModule_ID	ID of each record in TEACHERMODUEL_DETAILS table, identifies each record in TEACHERMODUEL_DETAILS table uniquely.	INT		TRUE	FALSE	FALSE	TRUE	Atuo incremented.
		Teacher_ID	Teacher_ID is a Id of each teacher and also a foreign key which references from "Teacher_ID" attribute of TEACHERS table.	INT		FALSE	TRUE	FALSE	FALSE	References from "Course_ID" attribute of COURSES table.
		Module_ID	Module_ID is a id of each module and also a foreign key which references from "Module_ID" attribute of MODULES table.	INT		FALSE	TRUE	FALSE	FALSE	References from "Module_ID" attribute of MODULES table.

Table 7: Data Dictionary Of TEACHERMODULE_DETAILS.

4. Queries

4.1 Query No 1

Query No.	Query 1
Query	SELECT * FROM STUDENTS WHERE YEAR(DOB) BETWEEN “1992” AND “1996”;
Keyword Used	SELECT, FROM, WHERE, BETWEEN, AND
Purpose/Result	Shows all records of student having date of birth (DOB) between 1992 and 1996.

Table 8: Query No 1.

```
MariaDB [collegecoursesmanagement]> SELECT * FROM STUDENTS WHERE YEAR(DOB) BETWEEN "1992" AND "1996";
+------------+------------+-----------+-----------+------------+------------+--------------------+
| Student_ID | First_Name | Last_Name | Mobile_no | DOB        | Address    | Email              |
+------------+------------+-----------+-----------+------------+------------+--------------------+
|          1 | Raju       | Tamang    |   9833249 | 1993-09-12 | Kathmandu  | raju@gmail.com     |
|          3 | Hemant     | Rana      |   9801098 | 1996-09-23 | Bhaktapur  | hemant@gmail.com   |
|          4 | Gaurav     | Shrestha  |  98010222 | 1993-04-13 | Kannpur    | gaurav@gmail.com   |
|          6 | Ram        | Rai       |   9833189 | 1992-09-12 | Itahari    | ram@gmail.com      |
|          7 | Romeo      | Rai       |  98011772 | 1994-08-12 | Lalitpur   | romeo@gmail.com    |
|          8 | Jiwan      | Rai       |  98011755 | 1995-08-02 | Biratnagar | jiwanrai@gmail.com |
|          9 | Sita       | Gurung    |  98012300 | 1995-06-05 | Dharan     | sita@gmail.com     |
|         10 | Resham     | Gurung    |   9800566 | 1992-01-12 | Lalitpur   | resham@gmail.com   |
|         12 | Sonam      | Tamang    |  98112233 | 1994-06-17 | Dharan     | sonam@gmail.com    |
+------------+------------+-----------+-----------+------------+------------+--------------------+
9 rows in set (0.016 sec)
```

Figure 31: Query No 1.

4.2 Query No 2

Query No.	Query 2
Query	SELECT * FROM TEACHERS WHERE Teacher_ID IN (1,3,6,10);
Keyword Used	SELECT, FROM, WHERE, IN
Purpose/Result	Shows all records of Teacher having ID no 1,3,6,10.

Table 9: Query No 2.

```
MariaDB [collegecoursesmanagement]> SELECT * FROM TEACHERS WHERE Teacher_ID IN (1,3,6,10);
+------------+------------+-----------+-----------+-----------+------------------+--------+
| Teacher_ID | First_Name | Last_Name | Mobile_no | Address   | Email            | Salary |
+------------+------------+-----------+-----------+-----------+------------------+--------+
|          1 | Janaki     | Chaudary  |  98083450 | Pokhara   | janaki@gmail.com |  60000 |
|          3 | Anuj       | Shilpakar |   9850367 | Kathmandu | anuj@gmail.com   |  45000 |
|          6 | Aadit      | shrestha  |   9221990 | Bhaktapur | aadit@gmail.com  |  67000 |
|         10 | Rajiv      | Tamang    |   9091234 | Pokhara   | rajiv@gmail.com  |  42000 |
+------------+------------+-----------+-----------+-----------+------------------+--------+
4 rows in set (0.014 sec)
```

Figure 32: Query No 2.

4.3 Query No 3

Query No.	Query 3
Query	SELECT * FROM STUDENTS WHERE First_Name LIKE (“R%”) AND Last_Name LIKE (“%a%”);
Keyword Used	SELECT, FROM, WHERE, LIKE, AND
Purpose/Result	Shows all records of student having First name starting with letter “R” and Lastname having middle letter “a”.

Table 10: Query No 3

```
MariaDB [collegecoursesmanagement]> SELECT * FROM STUDENTS WHERE First_Name LIKE ("R%") AND Last_Name LIKE ("%a%");
+------------+------------+-----------+-----------+------------+-----------+-----------------+
| Student_ID | First_Name | Last_Name | Mobile_no | DOB        | Address   | Email           |
+------------+------------+-----------+-----------+------------+-----------+-----------------+
|          1 | Raju       | Tamang    |   9833249 | 1993-09-12 | Kathmandu | raju@gmail.com  |
|          6 | Ram        | Rai       |   9833189 | 1992-09-12 | Itahari   | ram@gmail.com   |
|          7 | Romeo      | Rai       |  98011772 | 1994-08-12 | Lalitpur  | romeo@gmail.com |
|         11 | Rosy       | Tamang    |   9882233 | 1991-01-12 | Itahari   | rosy@gmail.com  |
+------------+------------+-----------+-----------+------------+-----------+-----------------+
4 rows in set (0.005 sec)
```

Figure 33: Query No 3.

4.4 Query No 4

Query No.	Query 4
Query	SELECT First_Name,Last_Name,Mobile_no,Address FROM STUDENTS ORDER BY Address;
Keyword Used	SELECT, FROM, ORDER BY
Purpose/Result	Shows first name, last name, mobile number, address of students sorted according to their address.

Table 11: Query No 4.

```
MariaDB [collegecoursesmanagement]> SELECT First_Name,Last_Name,Mobile_no,Address FROM STUDENTS ORDER BY Address;
+------------+-----------+-----------+------------+
| First_Name | Last_Name | Mobile_no | Address    |
+------------+-----------+-----------+------------+
| Hemant     | Rana      |   9801098 | Bhaktapur  |
| Jiwan      | Rai       |  98011755 | Biratnagar |
| Jiwan      | Limbu     |  98993344 | Damak      |
| Sonam      | Tamang    |  98112233 | Dharan     |
| Sita       | Gurung    |  98012300 | Dharan     |
| Rosy       | Tamang    |   9882233 | Itahari    |
| Ram        | Rai       |   9833189 | Itahari    |
| Gaurav     | Shrestha  |  98010222 | Kannpur    |
| Raju       | Tamang    |   9833249 | Kathmandu  |
| Resham     | Gurung    |   9800566 | Lalitpur   |
| Romeo      | Rai       |  98011772 | Lalitpur   |
| Pritam     | Shrestha  |   9833041 | Pokhara    |
+------------+-----------+-----------+------------+
12 rows in set (0.004 sec)
```

Figure 34: Query No 4.

4.5 Query No 5

Query No.	Query 5
Query	SELECT DISTINCT(Address) FROM STUDENTS;
Keyword Used	SELECT, DISTINCT, FROM
Purpose/Result	Shows all unique addresses of students.

Table 12: Query No 5

```
MariaDB [collegecoursesmanagement]> SELECT DISTINCT(Address) FROM STUDENTS;
+------------+
| Address    |
+------------+
| Kathmandu  |
| Pokhara    |
| Bhaktapur  |
| Kannpur    |
| Damak      |
| Itahari    |
| Lalitpur   |
| Biratnagar |
| Dharan     |
+------------+
9 rows in set (0.006 sec)
```

Figure 35: Query No 5.

4.6 Query No 6

Query No.	Query 6
Query	SELECT * FROM COURSES ORDER BY Course_Fees DESC LIMIT 2;
Keyword Used	SELECT, ORDER BY, DESC, LIMIT, FROM
Purpose/Result	Shows 0nly 2 records from COURSES Table sorted according to their Course_fees in descending order. It also List out two expensive courses in table.

Table 13: Query No 6.

```
MariaDB [collegecoursesmanagement]> SELECT * FROM COURSES ORDER BY Course_Fees DESC LIMIT 2;
+-----------+--------------------------+-----------------+-------------+
| Course_ID | Course_Title             | Course_Duration | Course_Fees |
+-----------+--------------------------+-----------------+-------------+
|         3 | Multimedia               | 3 YEARS         |     1450000 |
|         1 | Networking And IT Security | 3 YEARS         |     1400000 |
+-----------+--------------------------+-----------------+-------------+
2 rows in set (0.005 sec)
```

Figure 36: Query No 6.

4.7 Query No 7

Query No.	Query 7
Query	SELECT COUNT(*) AS TOTAL_ENROLLMENTS FROM ENROLLMENTS;
Keyword Used	SELECT, COUNT, FROM,AS
Purpose/Result	Shows Total number of enrolments of students in courses.

Table 14: Query No 7.

```
MariaDB [collegecoursesmanagement]> SELECT COUNT(*) AS TOTAL_ENROLLMENTS FROM ENROLLMENTS;
+-------------------+
| TOTAL_ENROLLMENTS |
+-------------------+
|                12 |
+-------------------+
1 row in set (0.007 sec)
```

Figure 37: Query No 7.

4.8 Query No 8

Query No.	Query 8
Query	SELECT Course_ID,COUNT(Student_ID) AS TOTAL_STUDENTS FROM ENROLLMENTS GROUP BY Course_ID;
Keyword Used	SELECT, COUNT, GROUP BY, FROM, AS
Purpose/Result	Shows the List of courses and total number of students enrolled in each course.

Table 15: Query No 8.

```
MariaDB [collegecoursesmanagement]> SELECT Course_ID,COUNT(Student_ID) AS TOTAL_STUDENTS FROM ENROLLMENTS GROUP BY Course_ID;
+-----------+----------------+
| Course_ID | TOTAL_STUDENTS |
+-----------+----------------+
|         1 |              3 |
|         2 |              2 |
|         3 |              3 |
|         4 |              2 |
|         5 |              2 |
+-----------+----------------+
5 rows in set (0.010 sec)
```

Figure 38: Query No 8.

4.9 Query No 9

Query No.	Query 9
Query	SELECT Course_ID,COUNT(Student_ID) AS TOTAL_STUDENTS FROM ENROLLMENTS GROUP BY Course_ID HAVING TOTAL_STUDENTS >2;
Keyword Used	SELECT, COUNT, GROUP BY, HAVING, FROM, AS
Purpose/Result	Shows only the List of courses and total number of students enrolled in each course having total students more than 2.

Table 16: Query No 9

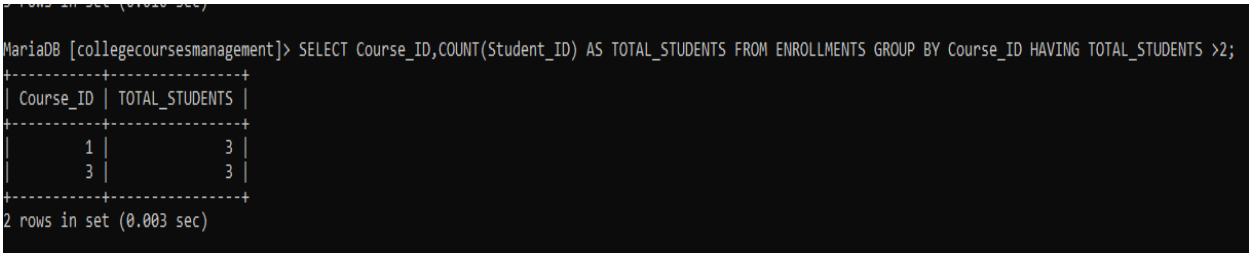

Figure 39: Query No 9.

4.10 Query No 10

Query No.	Query 10
Query	SELECT TEACHERMODULE_DETAILS.TeacherModule_ID, TEACHERS.Teacher_ID, TEACHERS.First_Name, TEACHERS.Last_Name, MODULES.Module_ID, MODULES.Module_Title FROM MODULES JOIN TEACHERMODULE_DETAILS ON MODULES.Module_ID=TEACHERMODULE_DETAILS.Module_ID JOIN TEACHERS ON TEACHERMODULE_DETAILS.Teacher_ID=TEACHERS.Teacher_ID ORDER BY TEACHERMODULE_DETAILS.TeacherModule_ID;
Keyword Used	SELECT, FROM, JOIN, ON, ORDER BY
Purpose/Result	Joining three tables i.e., MODULES, TEACHERMODULES_DETAILS and TEACHERS table but only showing specific column i.e., TeacherModule_ID, Teacher_ID, Teacher's first and last name, Module_ID, Module Title sorted according to TeacherModule_ID.

Table 17: Query No 10.

```
Database changed
MariaDB [collegecoursesmanagement]> SELECT TEACHERMODULE_DETAILS.TeacherModule_ID,
    -> TEACHERS.Teacher_ID,
    -> TEACHERS.First_Name,
    -> TEACHERS.Last_Name,
    -> MODULES.Module_ID,
    -> MODULES.Module_Title FROM MODULES JOIN TEACHERMODULE_DETAILS ON
    ->  MODULES.Module_ID=TEACHERMODULE_DETAILS.Module_ID
    ->  JOIN TEACHERS ON
    ->  TEACHERMODULE_DETAILS.Teacher_ID=TEACHERS.Teacher_ID
    ->   ORDER BY TEACHERMODULE_DETAILS.TeacherModule_ID;
+------------------+------------+------------+-----------+-----------+---------------------------------+
| TeacherModule_ID | Teacher_ID | First_Name | Last_Name | Module_ID | Module_Title                    |
+------------------+------------+------------+-----------+-----------+---------------------------------+
|                1 |          1 | Janaki     | Chaudary  |         1 | Information System              |
|                2 |          1 | Janaki     | Chaudary  |         4 | Fundamental of Computing        |
|                3 |          2 | Sofiya     | Gajurel   |         3 | communication Engineering       |
|                4 |          3 | Anuj       | Shilpakar |         3 | communication Engineering       |
|                5 |          4 | Pasang     | Tamang    |         2 | Programing                      |
|                6 |          5 | Kriti      | Rai       |         2 | Programing                      |
|                7 |          6 | Aadit      | shrestha  |         1 | Information System              |
|                8 |          7 | Aafsa      | Roy       |         5 | Logic and Problem Solving       |
|                9 |          8 | Bishwas    | Limbu     |         6 | Digital Desgin and Image Making |
|               10 |          9 | Bisu       | Rai       |         7 | Computer Hardware and Software  |
|               11 |          7 | Aafsa      | Roy       |         8 | Introducton to IT               |
|               12 |         10 | Rajiv      | Tamang    |         9 | Psychology                      |
|               13 |         11 | Sangam     | Chhetri   |        10 | Sociology                       |
|               14 |         12 | Hari       | Gurung    |        11 | Foundational Mathematics        |
+------------------+------------+------------+-----------+-----------+---------------------------------+
14 rows in set (0.028 sec)

MariaDB [collegecoursesmanagement]>
```

Figure 40: Query No 10.

5. Conclusion

Data are a set of raw facts that hardly gives any meaning and Information are raw facts that are processed and give some useful meaning. Data and information are vital in formation of database. Database can refer to a collection of tables having rows and columns where data are inserted, updated and managed for a certain purpose. Every big or small organization uses database system to store and manage data and information. Here “collegecoursesmanagement” named database is created for course management in college. Having a database in an organization, it can store very large amount of data and information. It can store data and information in organized manner and can be easily access anytime in less amount of time. An organization can be always be updated through updating database as updating process is very easy.

While developing project for database system, multiple researches were done. In a database, there are mainly two types of table parent and child. A parent table need to be created first before child table. In Relation diagram, there cannot exist many to many relationships between two entities and why bridge entity is need. In database, there is certain difference between primary key and unique key. Use of reserved word as a database name or table and column name gives sql syntax error. In database, why a table is called child table and parent table.

Similarly, after long research on above topics there were some findings. In database always a parent table is created first because a parent table always holds a primary key value only and that has to be referred in child table via foreign key. In relation diagram, many to many relationships cannot exit because two entities or tables cannot be child table of each other, thus third entity known as bridge entity created to resolve a many-to-many relationships of entities in the relational model. Primary key and Unique key have mainly two differences i.e. primary key can be used as foreign key in other table and primary key value cannot be empty or have “not null” values. In MYSQL, keywords cannot be used as table name, column name or database name as it gives syntax error because keywords are reserved words used in MYSQL which

specific meanings or functions. In database, parent table is any tables that store only primary key and Child table is any table that references the parent table with a foreign key (Curry, 2018).

References

Curry, C., 2018. *parent-child-tables.* [Online]
Available at: https://www.calebcurry.com/parent-child-tables/
[Accessed 26 april 2021].
Oracle Corporation, 2021. *what-is-database.* [Online]
Available at: https://www.oracle.com/database/what-is-database/
[Accessed 13 april 2021].
Stair, R. M. & Reynolds, G. .., 2016. In: *Principles Of Information Systems 13th Edition.* 3rd ed. Boston, MA 02210: Cengage Learning, pp. 5-6.