BACHELOR OF INFORMATION TECHNOLOGY

Semester Project

USER MANUAL

College Academic System Database Design

Submitted by:

Muhammad Muzzamal - 110837

Imran Ali – 110855

Supervised by:

Mam Sehrish Khan

Department of Information Technology

Govt. Post Graduate College Civil Lines Sheikhupura

May 2025

Entity Relation Diagram

Table and Schema

Statement:

College Academic System with Results and Teaching Load:

"This system handles student enrollment, course registration, attendance, assessments, and grade processing. Professors have maximum teaching loads, and class timetables are auto- generated to avoid conflicts. Students have access to academic history, backlogs, and improvement options. Faculty members submit grades online and can mark attendance via an internal app. The system tracks results and generates department-wise performance analytics."

Entities:

- Department
- Courses
- Professor
- Enrollment
- Attendance
- Student
- Grade

Entities with attributes:

Department (dpt_id, dpt_name, dpt_performance)

Course (course_id, course_title, credit-hours)

Professor (prof_id, prof_name, maxTeachingLoad, prof-dpt)

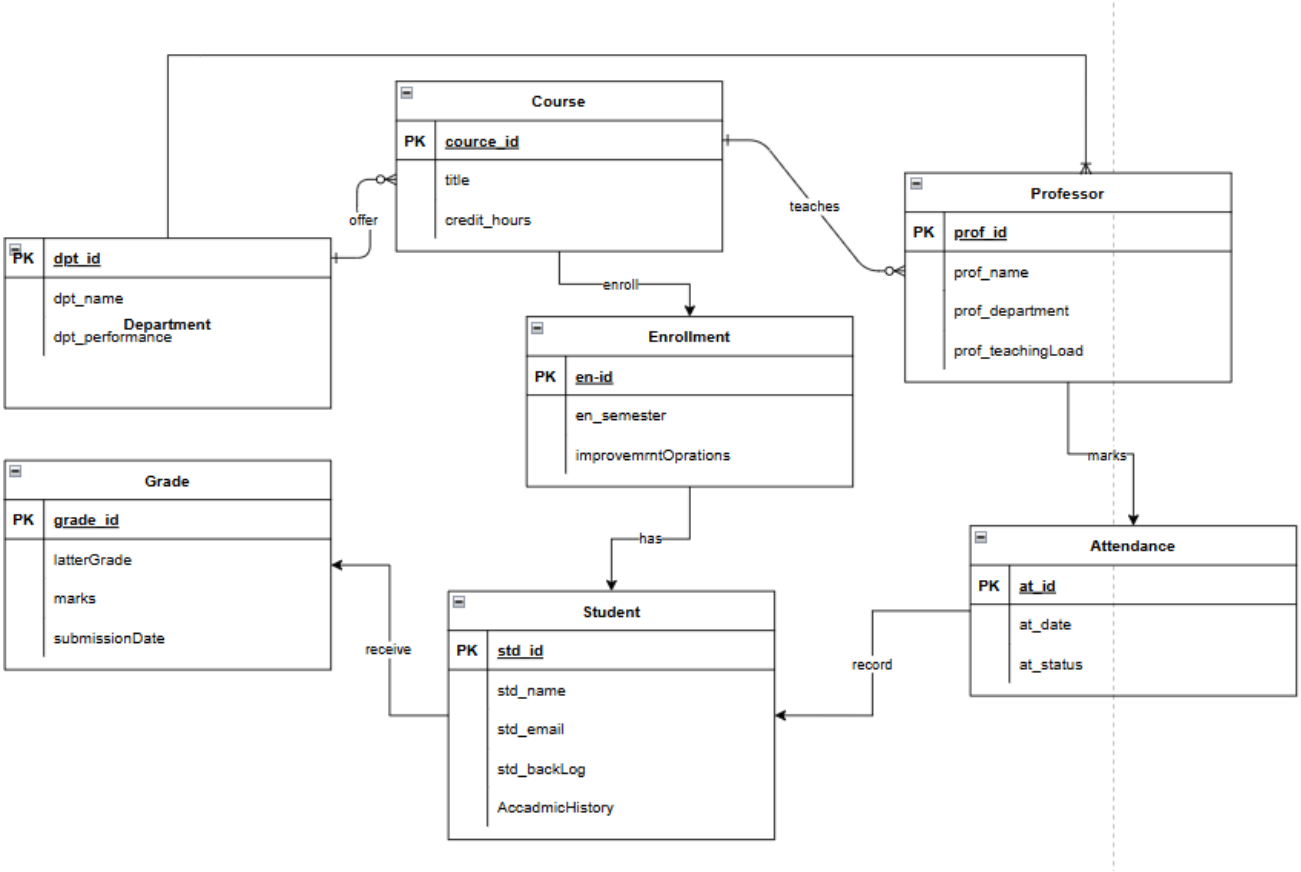
Enrollment (enroll_id, enroll-semester, improvementOprations)

Attendance (att-id , att-date, attr-status)

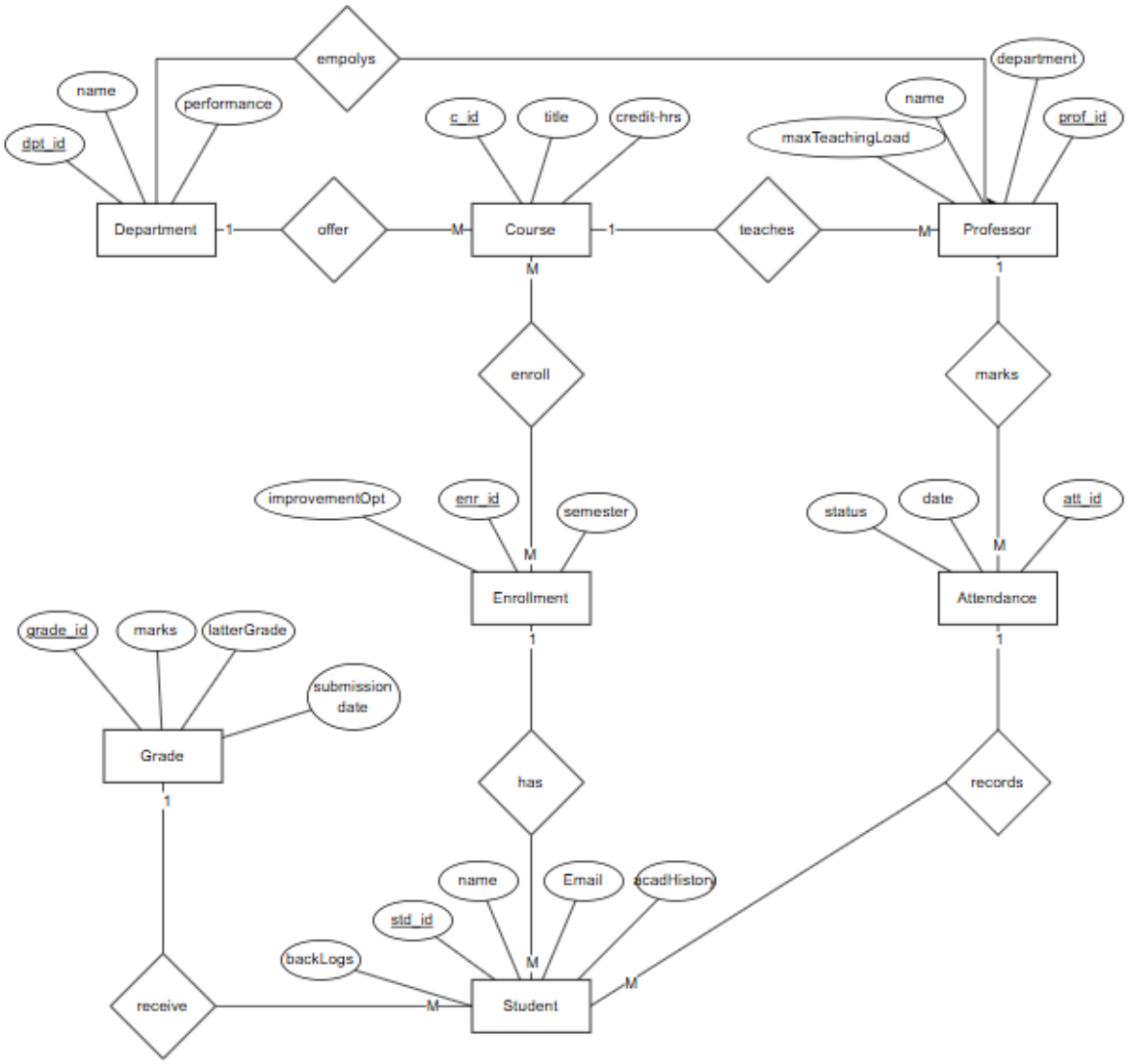
Student (std_id, std_name, std_academicHistory, std-Backlog, std_email)

Grade (grade-id, latterGrade, grade_marks, submissiondate)

Crow Foot Model Diagram



Chen Model


empolys
department
name
performance
dpt_id
c_id
title
credit-hrs
name
maxTeachingLoad
prof_id
Department
1
offer
M
Course
1
teaches
M
Professor
M
1
enroll
marks
improvementOpt
enr_id
semester
status
date
att_id
M
M
Enrollment
Attendance
grade_id
marks
latterGrade
submission date
1
1
Grade
has
records
1
name
Email
acadHistory
std_id
backLogs
M
M
receive
M
Student

Normalization:

1st Normal Form

Rule: *"Table format, No Repeating Groups, PRIMARY KEY Identification"*

Department (dpt_id, dpt_name, dpt_performance)

Course (course_id, course_title, credit_hours)

Professor (prof_id, prof_name, maxTeachingLoad, dpt_id)

Enrollment (enroll_id, enroll_semester, improvementOperations)

Attendance (att_id, att_date, att_status)

Student (std_id, std_name, std_email)

Grade (grade_id, letterGrade, grade_marks, submission_date)

2nd Normal Form:

Rule: "*1NF and No Partial Dependencies"*

Department (dpt_id, dpt_name, dpt_performance)

Course (course_id, course_title, credit_hours)

Professor (prof_id, prof_name, maxTeachingLoad, dpt_id)

Enrollment (enroll_id, enroll_semester, improvementOperations)

Attendance (att_id, att_date, att_status)

Student (std_id, std_name, std_email)

Grade (grade_id, letterGrade, grade_marks, submission_date)

3rd Normal Form:

Rule: *"2NF and No Transitive Dependencies."*

Department (dpt_id, dpt_name, dpt_performance)

Course (course_id, course_title, credit_hours)

Professor `(prof_id, prof_name, maxTeachingLoad, dpt_id)`

Enrollment `(enroll_id, enroll_semester, improvementOperations)`

Attendance `(att_id, att_date, att_status)`

Student `(std_id, std_name, std_email)`

Grade `(grade_id, grade_marks, submission_date)`

LetterGrade `(letterGrade, min_marks, max_marks)`

Structured Query Language (SQL)

Common SQL data types

DATA TYPE	FORMAT	COMMENTS
Numeric	NUMBER (L, D) or NUMERIC (L, D)	The declaration NUMBER (7,2) or NUMERIC (7,2) indicates that numbers will be stored with two decimal places and may be up to seven digits long, including the sign and the decimal place (For example, 12.32 or −134.99).
	INTEGER	May be abbreviated as INT. Integers are (whole) counting numbers, so they cannot be used if you want to store numbers that require decimal places.
	SMALLINT	Like INTEGER but limited to integer values up to six digits. If your integer values are relatively small, use SMALLINT instead of INT.
	DECIMAL (L, D)	Like the NUMBER specification, but the storage length is a minimum specification. that is, greater lengths are acceptable, but smaller ones are not. DECIMAL (9,2), DECIMAL (9), and DECIMAL are all acceptable.
Character	CHAR(L)	Fixed-length character data for up to 255 characters. If you store strings that are not as long as the CHAR parameter value, the remaining spaces are left unused. therefore, if you specify CHAR (25), strings such as Smith and Katzenjammer are each stored as 25 characters. However, a U.S. area code is always three digits long, so CHAR (3) would be appropriate if you wanted to store such codes.
	VARCHAR(L) or SVARCHAR2(L)	Variable-length character data. the designation VARCHAR2(25) or SVARCHAR (25) will let you store characters up to 25 characters long. However, unlike CHAR, VARCHAR will not leave unused spaces. oracle automatically converts VARCHAR to VARCHAR2.
Date	DATE	Stores dates in the Julian date format.

SQL data types and keys

➢ Keys:

- **PK (Primary Key):**
  - Uniquely identifies each row in a table.
  - Cannot be NULL.
  - Each table typically has one primary key.
- **FK (Foreign Key):**
  - A field (or set of fields) that refers to the primary key in another table.
  - Ensures referential integrity between related tables.

➢ Data Types:

- **CHAR(n):**
  - Fixed-length character string.
  - Always stores exactly n characters (padded with spaces if shorter).
- **VARCHAR(n):**
  - Variable-length character string.
  - Stores up to n characters.
  - In Oracle, VARCHAR is automatically treated as VARCHAR2.
- **NUMBER (p, s): (Oracle)**
  - Numeric type with p total digits and s digits after the decimal point.
  - Example: NUMBER (9,2) allows 7 digits before the decimal and 2 after.
- **NUMERIC (p, s):**
  - Same idea as NUMBER, but used in systems like PostgreSQL or SQL Server.
  - Ensures precision for exact values.
- **INT / INTEGER:**
  - Stores whole numbers.
  - In Oracle, it's converted to NUMBER.

- **SMALLINT:**
  - Like INT but for smaller range of values (uses less storage).
  - Also converted to NUMBER in Oracle.

- **DATE:**
  - Stores date values.
  - Common formats:
    - DD-MON-YYYY (e.g., 02-MAY-2025)
    - MM/DD/YYYY (e.g., 05/02/2025)

USEFUL SQL COMMANDS

- `SHOW DATABASES;`

- `USE database_name;`

- `SHOW TABLES;`

- `create table table_name(attribute_id_1 type(domain) primary key, attribute_2 type(domain),…,n);`

- `describe table_name;`

- `select * from table_name;`

- `alter table table_name ADD constraint fk_table_name foreign key (attribute) references ref_table(ref_attribute);`

- `insert into table_name value('val.1', 'val.2', 'val.3',….,n );`

- `alter table table_name ADD attribute_name type(domain);`

- `update table_name set attribute='value' WHERE attribute_PK='target-value';`

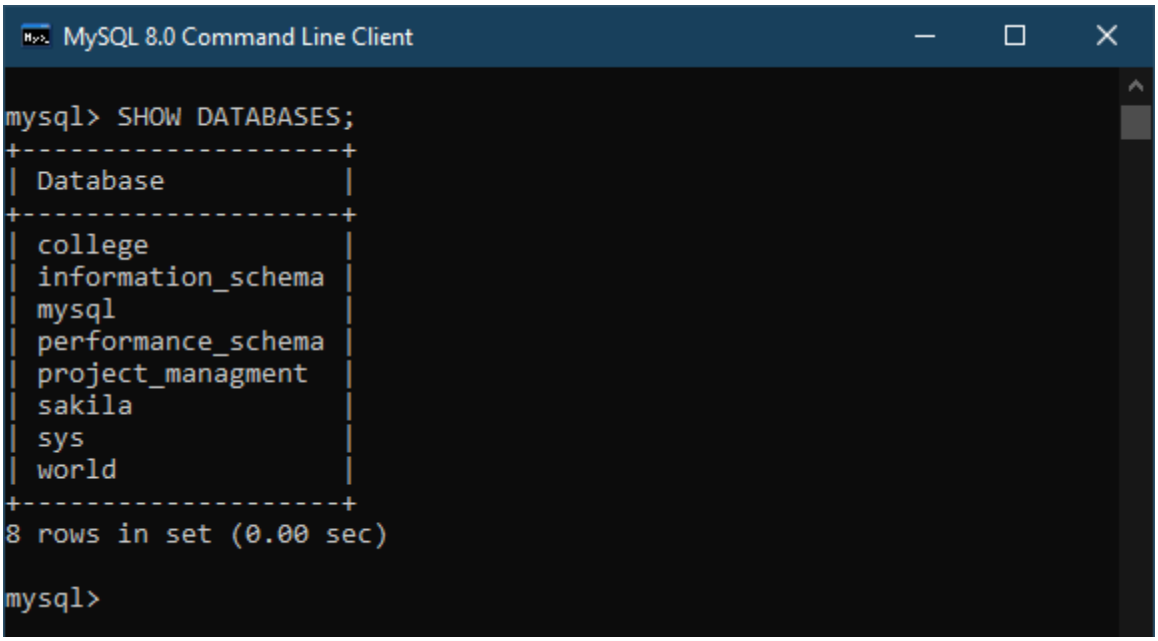
Database Using Command Line Client

Procedure to create Database in SQL command Line:

Create database;

1. Open the SQL Command Line Client.
2. Enter the password.

3. Enter the command to show the databases. → `SHOW DATABASES;`



4. If database is already existing then use command to manipulate in target database.
   - `USE database_name;`
   - Database name : college
   - Example : `USE college_acadmic_system;`

```
MySQL 8.0 Command Line Client
mysql> USE college_acadmic_system;
Database changed
mysql>
```

5. if we want to delete database use this command.
   - `DROP DATABASE database_name;`
   - Example database name is college
   - use command like this: `DROP DATABASE college;`

```
MySQL 8.0 Command Line Client
mysql> drop database college;
Query OK, 3 rows affected (0.06 sec)

mysql>
```

6. If we want to create new database use this command.
   - `CREATE DATABASE database_name;`
   - Example database name is college_acadmedic_system
   - use command like this: `USE DATABASE college_acadmedic_system;`

```
MySQL 8.0 Command Line Client

mysql> CREATE DATABASE college_academic_system;
Query OK, 1 row affected (0.01 sec)

mysql>
```

7. check the either database has created or not
   - `SHOW DATABASES;`

```
MySQL 8.0 Command Line Client

mysql> SHOW DATABASES;
+--------------------------+
| Database                 |
+--------------------------+
| college                  |
| college_academic_system  |
| information_schema       |
| mysql                    |
| performance_schema       |
| project_managment        |
| sakila                   |
| sys                      |
| world                    |
+--------------------------+
9 rows in set (0.00 sec)

mysql>
```

Our database has created.

Add tables:

8. Use command to target the database for manipulation.
   - Command: `USE database_name;`
   - Database name: college_academic_system.
   - `USE college_academic_system;`

```
MySQL 8.0 Command Line Client
mysql> USE college_academic_system;
Database changed
mysql>
```

9. Create table using command :
   Command : `CREATE TABLE department(dpt_id INT(10) PRIMARY KEY, dpt_name VARCHAR(25), dpt_performance INT(10) );`

```
MySQL 8.0 Command Line Client
mysql> CREATE TABLE department(
    -> dpt_id INT(10) PRIMARY KEY,
    -> dpt_name VARCHAR(25),
    -> dpt_performance INT(10)
    -> );
Query OK, 0 rows affected, 2 warnings (0.02 sec)

mysql>
```

10. Check the description of the table:
    Command: `desc depertment;`

```
MySQL 8.0 Command Line Client
mysql> desc department;
+-----------------+-------------+------+-----+---------+-------+
| Field           | Type        | Null | Key | Default | Extra |
+-----------------+-------------+------+-----+---------+-------+
| dpt_id          | int         | NO   | PRI | NULL    |       |
| dpt_name        | varchar(25) | YES  |     | NULL    |       |
| dpt_performance | int         | YES  |     | NULL    |       |
+-----------------+-------------+------+-----+---------+-------+
3 rows in set (0.01 sec)

mysql>
```

Create other tables using same commands:
Course table :

```
MySQL 8.0 Command Line Client
mysql> CREATE TABLE course(
    -> course_id INT(10) PRIMARY KEY,
    -> course_title VARCHAR(25),
    -> credit_hours INT(3)
    -> );
Query OK, 0 rows affected, 2 warnings (0.02 sec)

mysql>
```

Professor table:

```
MySQL 8.0 Command Line Client
mysql> CREATE TABLE professor(
    -> prof_id INT(10) PRIMARY KEY,
    -> prof_name VARCHAR(25),
    -> max_teaching_load INT(10)
    -> );
Query OK, 0 rows affected, 2 warnings (0.03 sec)

mysql>
```

Enrollment Table:

```
MySQL 8.0 Command Line Client
mysql> CREATE TABLE enrollment(
    -> enroll_id INT(10) PRIMARY KEY,
    -> enroll_semester INT(5),
    -> improvement_operations INT(5)
    -> );
Query OK, 0 rows affected, 3 warnings (0.02 sec)

mysql>
```

Attendance Table:

```
MySQL 8.0 Command Line Client
mysql> CREATE TABLE attendance(
    -> att_id INT(10) PRIMARY KEY,
    -> att_date DATE,
    -> att_status VARCHAR(10)
    -> );
Query OK, 0 rows affected, 1 warning (0.03 sec)

mysql>
```

Student Table:

```
MySQL 8.0 Command Line Client
mysql> CREATE TABLE student(
    -> std_id INT(10) PRIMARY KEY,
    -> std_name VARCHAR(25),
    -> std_email VARCHAR(50)
    -> );
Query OK, 0 rows affected, 1 warning (0.02 sec)

mysql>
```

Grade Table:

```
MySQL 8.0 Command Line Client
mysql> CREATE TABLE grade(
    -> grade_id INT(10) PRIMARY KEY,
    -> grade_mark INT(10),
    -> submission_date date
    -> );
Query OK, 0 rows affected, 2 warnings (0.02 sec)

mysql>
```

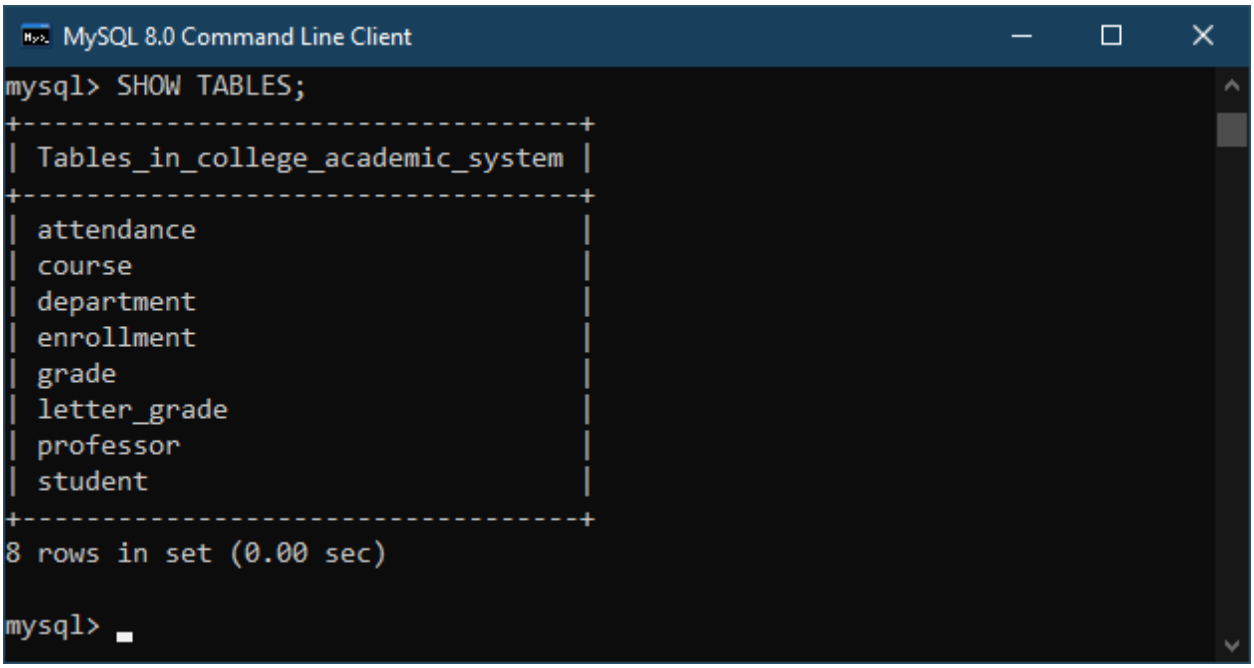
Latter Grade Table:

```
MySQL 8.0 Command Line Client
mysql> CREATE TABLE letter_grade(
    -> latter_grade INT(5) PRIMARY KEY,
    -> min_marks INT(10),
    -> max_marks INT(10)
    -> );
Query OK, 0 rows affected, 3 warnings (0.03 sec)

mysql>
```

SHOW TABLES:

Show table using the following command:
Command : `SHOW TABLES;`
It shows the all the created tables of database.



DELETE TABLE:

Suppose we have created the table mistakenly we can delete it using following command.
Command : `DROP TABLE student_name;`

Suppose a table name student_name is created mistakenly.

```
MySQL 8.0 Command Line Client
mysql> DROP TABLE student_name;
Query OK, 0 rows affected (0.01 sec)

mysql>
| department                      |
| enrollment                      |
| grade                           |
| letter_grade                    |
| professor                       |
| student                         |
| student_name                    |
+---------------------------------+
9 rows in set (0.00 sec)

mysql>
```

Now check table list:

Our target table has been deleted.

```
MySQL 8.0 Command Line Client
mysql> SHOW TABLES;
+---------------------------------+
| Tables_in_college_academic_system |
+---------------------------------+
| attendance                      |
| course                          |
| department                      |
| enrollment                      |
| grade                           |
| letter_grade                    |
| professor                       |
| student                         |
+---------------------------------+
8 rows in set (0.00 sec)

mysql>
```

ADD FOREIGN KEY:

First of all we add the attribute in the child table as an alias and then reference to the parent PK.

We want to add the FK in the in the student table from course table here student table is child and course table is parent table. First of all we add the attribute as child in the student table and then reference it.

Use the following command to add new attribute to the table.

Command: `ALTER TABLE student ADD c_id INT(10);`

Check the student table:

```
MySQL 8.0 Command Line Client
mysql> desc student;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| std_id    | int         | NO   | PRI | NULL    |       |
| std_name  | varchar(25) | YES  |     | NULL    |       |
| std_email | varchar(50) | YES  |     | NULL    |       |
| c_id      | int         | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
4 rows in set (0.00 sec)

mysql>
```

Now make it as FK using following commands.

Command : `ALTER TABLE student ADD CONSTRAINT fk_student FOREIGN KEY (c_id) REFERENCES course(course_id);`

```
MySQL 8.0 Command Line Client
mysql> ALTER TABLE student ADD CONSTRAINT fk_student FOREIGN KEY (c_id) REF
ERENCES course(course_id);
Query OK, 0 rows affected (0.08 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql>
```

Now check the description of the student table.

Using Command: `desc student;`

You can notice the key section of c_id attribute.

```
MySQL 8.0 Command Line Client
mysql> desc student;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| std_id    | int         | NO   | PRI | NULL    |       |
| std_name  | varchar(25) | YES  |     | NULL    |       |
| std_email | varchar(50) | YES  |     | NULL    |       |
| c_id      | int         | YES  | MUL | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
4 rows in set (0.00 sec)

mysql>
```

ENTER THE VALUES IN TABLES:

Command to enter the values in table is:

`INSERT INTO professor  VALUES( ‘1’, ‘Muzzamal’ , ‘12’);`

```
MySQL 8.0 Command Line Client
mysql> INSERT INTO professor VALUES(
    -> '1',
    -> 'Muzzamal',
    -> '12'
    -> );
Query OK, 1 row affected (0.01 sec)

mysql>
```

Add more than 1 at a time :

```
MySQL 8.0 Command Line Client
mysql>  INSERT INTO professor(prof_id, prof_name, max_teaching_load)
    -> Values
    -> ('2', 'Imran Ali', '13'),
    -> ('3', 'third', '14');
Query OK, 2 rows affected (0.00 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql>
```

Now check the professor table again :

Using command: `SELECT * FROM professor;`

It print all the values in the table.

```
MySQL 8.0 Command Line Client
mysql> SELECT * FROM professor;
+---------+-----------+-------------------+
| prof_id | prof_name | max_teaching_load |
+---------+-----------+-------------------+
|       1 | Muzzamal  |                12 |
|       2 | Imran Ali |                13 |
|       3 | third     |                14 |
+---------+-----------+-------------------+
3 rows in set (0.00 sec)

mysql>
```

If we want to fint the value of specific row we target it using PK

Using command: `SELECT * FROM professor WHERE prof_id = 3;`

```
MySQL 8.0 Command Line Client
mysql> SELECT * FROM professor WHERE prof_id = 3;
+---------+-----------+-------------------+
| prof_id | prof_name | max_teaching_load |
+---------+-----------+-------------------+
|       3 | third     |                14 |
+---------+-----------+-------------------+
1 row in set (0.00 sec)

mysql>
```

If we update the value of specific attribute we can it using following command.

Command: `UPDATE professor set ATTRIBUTE='Chand' WHERE prof_id='3';`

```
MySQL 8.0 Command Line Client

mysql> UPDATE professor SET prof_name='Chand' WHERE prof_id='3';
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql>
```

Check the table's value:

Command : `SELECT * FROM professor;`

You can see the value has updated.