

**SCHOOL OF BUSINESS**

**COLLEGE OF BUSINESS AND ECONOMIC**

**DEPARTMENT OF BUSINESS INFORMATION TECHNOLOGY**

**LECTURER name: Dr BUGINGO Emmanuel**

**Module name: DATABASE**

**STUDENT name: HAZAJYABERA Samuel**

**REG name: 222003581**

**SECTION I**

**1.Describe all entities and their corresponding attributes that are in student management system.**

**1.Student entity: 2.Course entity:**

**Attributes: Attributes:**

**-Student ID -Course ID**

**-First name -Course name**

**-Last name -Course code**

**-Date of birth -Credit hours**

**-Contact information -Description**

**(Phone and email) -Department**

**-Gender -Semester/Year**

**-Address**

**3.Enrollment entity**:

**Attributes:**

**-Enrollment ID 4.Teacher entity:**

**-Student ID (Foreign key) Attributes:**

**-Course ID (Foreign key) -Teacher ID**

**-Enrollment Date -First name**

**-Last name**

**-Date of birth**

**-Contact Information (phone and email)**

**-Address**

**-Gender**

**-Qualifications**

**5.Attendance Entity:**

**-A**ttendanceID (Primary Key)

-EnrollmentID (Foreign Key)

-AttendanceDate

-Status (e.g., Present, Absent)

**6.Library Entity:**

**Attributes:**

**-L**ibraryID (Primary Key)

-BookTitle

-Author

**7.Class Entity :**

Attributes:

-ClassID (Primary Key)

-CourseID (Foreign Key)

-TeacherID (Foreign Key)

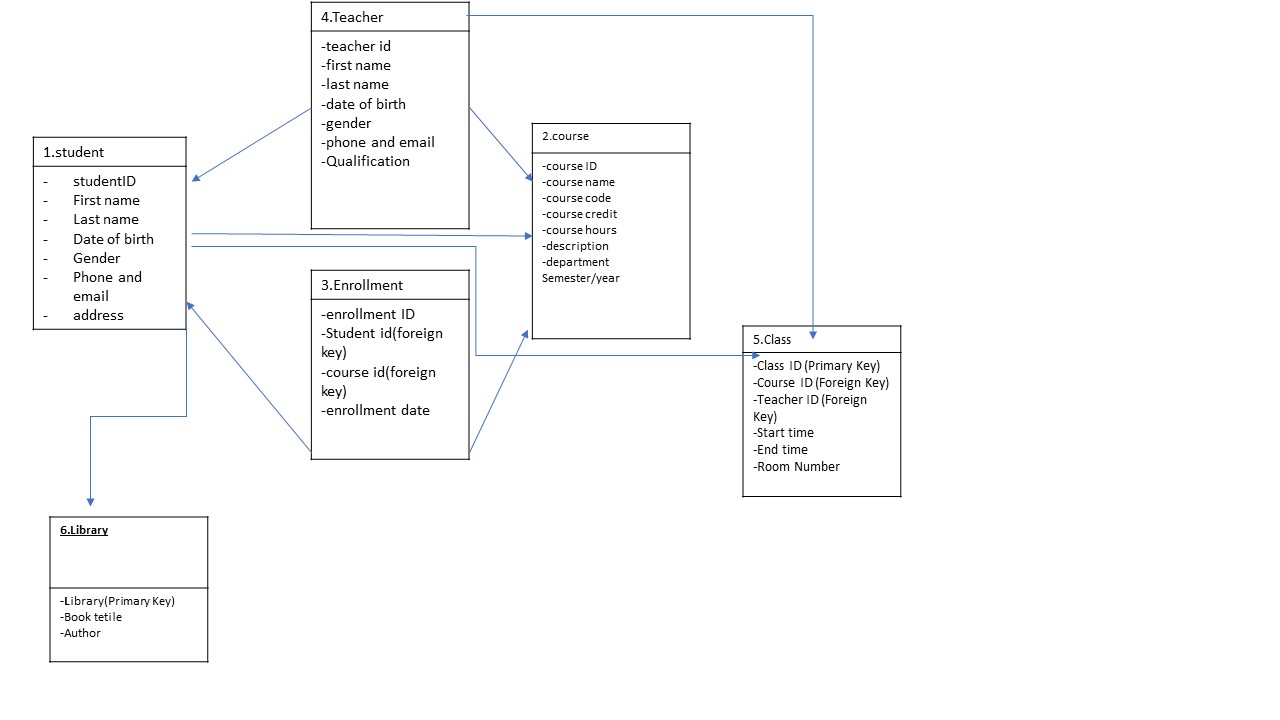
-StartTime

-EndTime

-RoomNumber

-Day Of Week

**Q2. Create an LDM of your entities**



**Q3. Create an ERD**



**Section II SQL**

**1. Create the database of your system**

Create the database of your system mysql -u root

show databases;

create database hazajyabera\_samuel\_222003581;

use hazajyabera\_samuel\_222003581;

**2. Write queries to create all the tables and relationships of your system**.

**Table of students**

Create table students(

Stu\_id int auto\_increment primary key,

Student\_name varchar(100) not null,

Email varchar(50) not null,

Telephone varchar(10) not null,

Date\_of\_birth date not null,

Nationality varchar(20) not null,

Gender char,

Course\_name varchar(50) ,

Enrollment\_date date,

Parents varchar(100) not null

);

**Table of courses**

**Create table courses(**

**Course\_id int auto\_increment primary key,**

**Course\_name varchar(100) not null,**

**Course\_code varchar(50) not null,**

**Credit int not null,**

**Department varchar(50) not null**

**);**

**Table of Enrollment**

Create table Enrollment(

Enrollment\_id int auto\_increment primary key,

Stu\_id int,

Course\_id int,

Enrollment\_Date date,

Foreign key(stu\_id) references students(stu\_id) ,

Foreign key(course\_id) references courses(course\_id)

);

**Table of Teachers**

**Create table teachers(**

**Teacher\_id int auto\_increment primary key,**

**Teacher\_name varchar(100) not null,**

**Contact varchar(40) not null,**

**Email varchar(50) not null,**

**Course\_name varchar(100),**

**Office\_hours int not null**

**);**

**Table of attendance**

**Create table attendance(**

**Attendance\_id int auto\_increment primary key,**

**Stu\_id int,**

**Class\_id int,**

**Attendance\_date date,**

**Status varchar(50) not null,**

**Foreign key(stu\_id) references students(stu\_id),**

**Foreign key(class\_id) references classes(class\_id)**

**);**

**Table of classes**

**Create table classes(**

**Class\_id int auto\_increment primary key,**

**Class\_level int not null,**

**Location varchar(20) not null,**

**Year int not null**

**);**

**Table of library**

**Create table librarys(**

**Book\_id int auto\_increment primary key,**

**Title varchar(50) not null,**

**Publication\_year date not null,**

**Author varchar(30) not null**

**);**

**Table of payments**

**Create table payments(**

**Payment\_id int auto\_increment primary key,**

**Stu\_id int,**

**Payment\_date date,**

**Payment\_amaunt int not null,**

**Payment\_method varchar(50) not null,**

**Status varchar(50) not null,**

**Foreign key(stu\_id) references students(stu\_id)**

**);**

**3. write queries to insert data into your tables.**

**Inserting sample data into the students table for Rwanda**

INSERT INTO students (Student\_name, Email, Telephone, Date\_of\_birth, Nationality, Gender, Course\_name, Enrollment\_date, Parents)

VALUES

('Emmanuel Nzeyimana', 'emmanuel@example.com', '250788888888', '2000-03-12', 'Rwanda', 'M', 'Computer Science', '2022-09-01', 'Grace Nzeyimana'),

('Marie Uwase', 'marie@example.com', '250737777777', '2001-06-25', 'Rwanda', 'F', 'Law', '2022-08-15', 'Jean Uwase'),

('Fidele Niyonzima', 'fidele@example.com', '250799999999', '2000-11-10', 'Rwanda', 'M', 'Business Administration', '2022-07-20', 'Esther Niyonzima'),

('jean pierre Niyonzima', 'niyonzima@example.com', '250799999999', '2000-11-10', 'Zambia', 'M', 'Business Administration', '2022-07-25', 'Esther Niyomurinzi'),

('Grace Abijuru', 'grace@example.com', '2507999800999', '2001-11-10', 'Rwanda', 'M', 'Business Administration', '2023-07-20', 'Samuel Mbarimombazi'),

('Divine Mukamana', 'divine@example.com', '250712345678', '2002-04-03', 'Uganda', 'F', 'Engineering', '2022-06-30', 'Jean Mukamana');

**Inserting sample data into the courses table for Rwanda**

INSERT INTO courses (Course\_name, Course\_code, Credit, Department)

VALUES

('Computer Science', 'CS101', 3, 'Computer Science Department'),

('Law', 'LAW201', 4, 'Law Department'),

('Business function', 'ACC2137', 3, 'Business information technology'),

('Engineering', 'ENG101', 4, 'Engineering Department');

('Business Administration', 'BUS101', 3, 'Business Department'),

**Inserting sample enrollment data**

INSERT INTO Enrollment (Stu\_id, Course\_id, Enrollment\_Date)

VALUES

(1, 1, '2022-09-01'),

(2, 2, '2022-08-15'),

(3, 3, '2022-07-20'),

(4, 1, '2023-07-20'),

(5, 1, '2022-06-30'),

(6, 3, '2022-08-01');

**Inserting sample teacher data**

INSERT INTO teachers (Teacher\_name, Contact, Email, Course\_name, Office\_hours)

VALUES

('Laura Johnson', '1234567890', 'laura@example.com', 'Computer Science', 10),

('Michael Smith', '9876543210', 'michael@example.com', 'Engineering', 8),

('Grace Nyirabukeye', '5555555555', 'grace@example.com', 'Mathematics', 9),

('Emmanuel Bugingo', '25078543210', 'emmanuel@example.com', 'database technology', 11),

('John Doe', '6666666666', 'john.doe@example.com', 'Physics', 7);

**Inserting sample attendance data**

INSERT INTO attendance (Stu\_id, Class\_id, Attendance\_date, Status)

VALUES

(1, 1, '2023-08-01', 'Present'),

(2, 2, '2023-08-01', 'Absent'),

(3, 3, '2023-08-01', 'Present'),

(4, 1, '2023-08-01', 'Present'),

(5, 1, '2023-08-01', 'Present'),

(6, 2, '2023-08-01', 'Absent');

**Inserting sample class data**

INSERT INTO classes (Class\_level, Location, Year)

VALUES

(1, 'Room A101', 2023),

(2, 'Room B202', 2023),

(1, 'Room C103', 2023),

(2, 'Room D204', 2023),

(3, 'Room E305', 2023);

**Inserting sample library book data**

INSERT INTO librarys (Title, Publication\_year, Author)

VALUES

('Introduction to Programming', '2020-01-01', 'Jane Smith'),

('The Art of Mathematics', '2019-05-15', 'John Doe'),

('Business Strategies for Success', '2022-08-20', 'Emily Johnson'),

('Engineering Mechanics Handbook', '2021-03-10', 'Michael Anderson'),

('Literary Classics Collection', '2018-11-30', 'Sophia Williams');

**Inserting sample payment data**

INSERT INTO payments (Stu\_id, Payment\_date, Payment\_amaunt, Payment\_method, Status)

VALUES

(1, '2023-08-01', 1000, 'Credit Card', 'Paid'),

(2, '2023-08-02', 1500, 'Bank Transfer', 'Paid'),

(3, '2023-08-03', 1200, 'Cash', 'Paid'),

(4, '2023-08-04', 900, 'Credit Card', 'Paid'),

(5, '2023-08-05', 1800, 'Bank Transfer', 'Unpaid');

**4. Write queries to display all the information in your tables**.

SELECT \* FROM payments;

SELECT \* FROM librarys;

SELECT \* FROM classes;

SELECT \* FROM attendance;

SELECT \* FROM teachers;

SELECT \* FROM Enrollment;

SELECT \* FROM courses;

SELECT \* FROM students;

**5. Write a query to update information in any of the two tables of your system.**

UPDATE courses

SET Course\_name = 'New Course Name'

WHERE Course\_id = 1; -- Replace 1 with the course ID you want to update

UPDATE students

SET Email = 'new\_email@example.com'

WHERE Stu\_id = 1; -- Replace 1 with the student ID you want to update

**Section III**

**1. Create a view to insert data into your tables**.

create view students\_enrolled\_in\_2023 AS

select\* from students

where enrollment\_date BETWEEN '2023-01-01' AND '2023-12-31';

CREATE VIEW teachers\_with\_date\_inserted AS

SELECT \*,

(CURRENT\_DATE - '2023-08-01') AS days\_since\_enrollment

FROM teachers;

CREATE VIEW librarys\_with\_publication\_date AS

SELECT \*

FROM librarys

WHERE publication\_year BETWEEN '2022-01-01' AND '2023-12-31';

CREATE VIEW payments\_with\_date AS

SELECT \*

FROM payments

WHERE payment\_date BETWEEN '2023-01-01' AND '2023-12-31';

CREATE VIEW attendance\_with\_date AS

SELECT \*

FROM attendance

WHERE attendance\_date BETWEEN '2023-01-01' AND '2023-12-31';

CREATE VIEW enrollment\_with\_date AS

SELECT \*

FROM enrollment

WHERE enrollment\_date BETWEEN '2023-01-01' AND '2023-12-31';

CREATE VIEW courses\_with\_credit AS

SELECT \*

FROM courses

WHERE Credit BETWEEN 1 AND 15;

CREATE VIEW classes\_with\_date AS

SELECT \*

FROM classes

WHERE Year BETWEEN 2021 AND 2025;

**2. Create a view to display all the information in your tables.**

CREATE VIEW combined\_data AS

SELECT s.Stu\_id,

s.Student\_name,

s.Email AS Student\_Email,

s.Telephone AS Student\_Telephone,

s.Date\_of\_birth AS Student\_DOB,

s.Nationality AS Student\_Nationality,

s.Gender AS Student\_Gender,

s.Course\_name AS Student\_Course,

s.Enrollment\_date AS Student\_Enrollment\_Date,

s.Parents AS Student\_Parents,

c.Course\_id,

c.Course\_name AS Course\_Name,

c.Course\_code AS Course\_Code,

c.Credit AS Course\_Credit,

c.Department AS Course\_Department,

e.Enrollment\_id,

e.Enrollment\_Date AS Enrollment\_Date,

t.Teacher\_name,

t.Contact AS Teacher\_Contact,

t.Email AS Teacher\_Email,

t.Course\_name AS Teacher\_Course,

t.Office\_hours AS Teacher\_Office\_Hours,

a.Attendance\_id,

a.Attendance\_date AS Attendance\_Date,

a.Status AS Attendance\_Status,

cl.Class\_id,

cl.Class\_level AS Class\_Level,

cl.Location AS Class\_Location,

cl.Year AS Class\_Year,

l.Book\_id,

l.Title AS Book\_Title,

l.Publication\_year AS Book\_Publication\_Year,

l.Author AS Book\_Author,

p.Payment\_id,

p.Payment\_date AS Payment\_Date,

p.Payment\_amaunt AS Payment\_Amount,

p.Payment\_method AS Payment\_Method,

p.Status AS Payment\_Status

FROM students s

LEFT JOIN Enrollment e ON s.Stu\_id = e.Stu\_id

LEFT JOIN courses c ON e.Course\_id = c.Course\_id

LEFT JOIN teachers t ON c.Course\_name = t.Course\_name

LEFT JOIN attendance a ON s.Stu\_id = a.Stu\_id

LEFT JOIN classes cl ON a.Class\_id = cl.Class\_id

LEFT JOIN librarys l ON l.Book\_id = l.Book\_id

LEFT JOIN payments p ON s.Stu\_id = p.Stu\_id;

**3. Create a view to update information in any of the two tables of your system**

CREATE VIEW update\_student\_info AS

SELECT \*

FROM students

WHERE student\_name = 'Emmanuel Nzeyimana';

**Example:** **update\_student\_info**

UPDATE update\_student\_info

SET email = 'emmanuel.nzeyimana@example.com',

telephone = '250788888889'

WHERE student\_name = 'Emmanuel Nzeyimana';

CREATE VIEW update\_course\_info AS

SELECT \*

FROM courses

WHERE course\_name = 'Computer Science';

**Example:** **update\_course\_info**

UPDATE update\_course\_info

SET credit = 4;

UPDATE update\_course\_info

SET department = 'Computer Science Department'

**4. Create a view to delete data in any two of your tables according to any simple**

**condition of your choice.**

CREATE VIEW delete\_teachers\_with\_no\_students AS

SELECT \*

FROM teachers

WHERE teacher\_id NOT IN (

SELECT teacher\_id

FROM enrollment

);

**Example:** DELETE FROM delete\_teachers\_with\_no\_students;

**5. In your database, create one view of your choice that considers sub-query**

CREATE VIEW students\_enrolled\_in\_credit\_courses AS

SELECT \*

FROM students

WHERE stu\_id IN (

SELECT stu\_id

FROM enrollment

WHERE course\_id IN (

SELECT course\_id

FROM courses

WHERE credit = 4

)

);

**Section IV**

**1. Create a stored procedure to insert data into your tables**.

DELIMITER //

CREATE PROCEDURE InsertStudents(

IN studentName VARCHAR(100),

IN email VARCHAR(50),

IN telephone VARCHAR(10),

IN dateOfBirth DATE,

IN nationality VARCHAR(20),

IN gender CHAR,

IN courseName VARCHAR(50),

IN enrollmentDate DATE,

IN parents VARCHAR(100)

)

BEGIN

INSERT INTO students (Student\_name, Email, Telephone, Date\_of\_birth, Nationality, Gender, Course\_name, Enrollment\_date, Parents)

VALUES (studentName, email, telephone, dateOfBirth, nationality, gender, courseName, enrollmentDate, parents);

END //

DELIMITER ;

CALL InsertStudent('John Doe', 'john@example.com', '1234567890', '2000-01-15', 'Rwanda', 'M', 'Computer Science', '2022-08-15', 'Jane Doe');

CALL InsertCourse('Computer Science', 'CS101', 3, 'Computer Science Department');

CALL InsertCourse('Computer Science', 'CS101', 3, 'Computer Science Department');

**Insert into Teachers**

DELIMITER //

CREATE PROCEDURE InsertTeacher(

IN teacherName VARCHAR(100),

IN contact VARCHAR(40),

IN email VARCHAR(50),

IN courseName VARCHAR(100),

IN officeHours INT

)

BEGIN

INSERT INTO teachers (Teacher\_name, Contact, Email, Course\_name, Office\_hours)

VALUES (teacherName, contact, email, courseName, officeHours);

END //

DELIMITER ;

CALL InsertTeacher('Laura Johnson', '1234567890', 'laura@example.com', 'Computer Science', 10);

**Insert into InsertEnrollment**

DELIMITER //

CREATE PROCEDURE InsertEnrollment(

IN studentID INT,

IN courseID INT,

IN enrollmentDate DATE

)

BEGIN

INSERT INTO Enrollment (Stu\_id, Course\_id, Enrollment\_Date)

VALUES (studentID, courseID, enrollmentDate);

END //

DELIMITER ;

CALL InsertEnrollment(1, 1, '2022-09-01');

CALL InsertLibraryBook('Introduction to Programming', '2020-01-01', 'Jane Smith');

DELIMITER //

CREATE PROCEDURE InsertLibraryBook(

IN title VARCHAR(50),

IN publicationYear DATE,

IN author VARCHAR(30)

)

BEGIN

INSERT INTO library (Title, Publication\_year, Author)

VALUES (title, publicationYear, author);

END //

DELIMITER ;

CALL InsertAttendance(1, 1, '2023-08-01', 'Present');

DELIMITER //

CREATE PROCEDURE InsertAttendance(

IN studentID INT,

IN classID INT,

IN attendanceDate DATE,

IN status VARCHAR(50)

)

BEGIN

INSERT INTO attendance (Stu\_id, Class\_id, Attendance\_date, Status)

VALUES (studentID, classID, attendanceDate, status);

END //

DELIMITER ;

CALL InsertPayment(1, '2023-08-01', 1000, 'Credit Card', 'Paid');

DELIMITER //

CREATE PROCEDURE InsertPayment(

IN studentID INT,

IN paymentDate DATE,

IN paymentAmount INT,

IN paymentMethod VARCHAR(50),

IN status VARCHAR(50)

)

BEGIN

INSERT INTO payments (Stu\_id, Payment\_date, Payment\_amaunt, Payment\_method, Status)

VALUES (studentID, paymentDate, paymentAmount, paymentMethod, status);

END //

DELIMITER ;

CALL InsertClass(1, 'Room A101', 2023);

DELIMITER //

CREATE PROCEDURE InsertClass(

IN classLevel INT,

IN location VARCHAR(20),

IN year INT

)

BEGIN

INSERT INTO classes (Class\_level, Location, Year)

VALUES (classLevel, location, year);

END //

DELIMITER ;

**2. Create a stored procedure to display all the information in your tables.**

DELIMITER //

CREATE PROCEDURE DisplayAllTablesData()

BEGIN

SELECT \* FROM students;

SELECT \* FROM courses;

SELECT \* FROM Enrollment;

SELECT \* FROM teachers;

SELECT \* FROM attendance;

SELECT \* FROM classes;

SELECT \* FROM library;

SELECT \* FROM payments;

END //

DELIMITER ;

**3. Create a stored procedure to update information in any of the two tables of your**

**system.**

DELIMITER //

CREATE PROCEDURE UpdateStudentOrCourse(

IN tableName VARCHAR(50),

IN recordID INT,

IN columnName VARCHAR(50),

IN newValue VARCHAR(100)

)

BEGIN

DECLARE tableNameLowercase VARCHAR(50);

SET tableNameLowercase = LOWER(tableName);

SET @updateQuery = CONCAT('UPDATE ', tableNameLowercase, ' SET ', columnName, ' = ?', ' WHERE ', tableNameLowercase, '\_id = ?');

PREPARE stmt FROM @updateQuery;

EXECUTE stmt USING newValue, recordID;

DEALLOCATE PREPARE stmt;

END //

DELIMITER ;

CALL UpdateStudentOrCourse('student\_id', 1, 'Email', 'uwiduhaye@gmail.com');

DELIMITER //

CREATE PROCEDURE DeleteStudentOrTeacher(

IN tableName VARCHAR(50),

IN columnName VARCHAR(50),

IN conditionValue VARCHAR(100)

)

BEGIN

DECLARE tableNameLowercase VARCHAR(50);

SET tableNameLowercase = LOWER(tableName);

SET @deleteQuery = CONCAT('DELETE FROM ', tableNameLowercase, ' WHERE ', columnName, ' = ?');

PREPARE stmt FROM @deleteQuery;

EXECUTE stmt USING conditionValue;

DEALLOCATE PREPARE stmt;

END //

DELIMITER ;

CALL DeleteStudentOrTeacher('teachers', 'Course\_name', 'Computer Science');

CALL DeleteStudentOrTeacher('students', 'Nationality', 'Rwanda');

**4. Create a stored procedure to delete data in any two of your tables according to any**

**simple condition of your choice.**

DELIMITER //

CREATE PROCEDURE DeleteStudentOrTeacher(

IN tableName VARCHAR(50),

IN columnName VARCHAR(50),

IN conditionValue VARCHAR(100)

)

BEGIN

DECLARE tableNameLowercase VARCHAR(50);

SET tableNameLowercase = LOWER(tableName);

SET @deleteQuery = CONCAT('DELETE FROM ', tableNameLowercase, ' WHERE ', columnName, ' = ?');

PREPARE stmt FROM @deleteQuery;

EXECUTE stmt USING conditionValue;

DEALLOCATE PREPARE stmt;

END //

DELIMITER ;

**5. In your database, stored the procedure view of your choice that considers sub**

**query.**

DELIMITER //

CREATE PROCEDURE GetStudentsInCourse(

IN courseName VARCHAR(100)

)

BEGIN

SELECT Student\_name, Email

FROM students

WHERE Stu\_id IN (

SELECT Stu\_id

FROM Enrollment

WHERE Course\_id = (SELECT Course\_id FROM courses WHERE Course\_name = courseName)

);

END //

DELIMITER ;

CALL GetStudentsInCourse('Computer Science');

**Section V**

**1.Create after inserting triggers for any two tables of your choice.**

DELIMITER //

CREATE TRIGGER AfterInsertlibrary

AFTER INSERT ON librarys

FOR EACH ROW

BEGIN

INSERT INTO librarys\_audit (book\_id, action, action\_date)

VALUES (NEW.book\_id, 'INSERT', NOW());

END //

DELIMITER ;

DELIMITER //

CREATE TRIGGER AfterInsertattendance

AFTER INSERT ON attendance

FOR EACH ROW

BEGIN

INSERT INTO attendance\_audit (attendance\_id, action, action\_date)

VALUES (NEW.attendance\_id, 'INSERT', NOW());

END //

DELIMITER ;

2. **Create after-update triggers for any two tables of your choice**

DELIMITER //

CREATE TRIGGER AfterUpdateteachers

AFTER UPDATE ON teachers

FOR EACH ROW

BEGIN

INSERT INTO teachers\_audit (teacher\_id, action, action\_date)

VALUES (NEW.teacher\_id, 'UPDATE', NOW());

END //

DELIMITER ;

DELIMITER //

CREATE TRIGGER AfterUpdatecourses

AFTER UPDATE ON courses

FOR EACH ROW

BEGIN

INSERT INTO courses\_audit (course\_id, action, action\_date)

VALUES (NEW.course\_id, 'UPDATE', NOW());

END //

DELIMITER ;

1. **Create after deleting triggers for any two tables of your choice**

DELIMITER //

CREATE TRIGGER AfterDeleteClasses

AFTER DELETE ON classes

FOR EACH ROW

BEGIN

INSERT INTO class\_audit (class\_id, action, action\_date)

VALUES (OLD.class\_id, 'DELETE', NOW());

END //

DELIMITER ;

DELIMITER //

CREATE TRIGGER AfterDeletestudents

AFTER DELETE ON students

FOR EACH ROW

BEGIN

INSERT INTO students\_audit (Stu\_id, action, action\_date)

VALUES (OLD.Stu\_id, 'DELETE', NOW());

END //

DELIMITER ;

**Section VI**

1. **Create a user with your name as username and your student number as password**

**and grant all privileges to the created user.**

mysql -u root -p

CREATE USER '<hazajyaberasamuel>'@'localhost' IDENTIFIED BY '222003581';

GRANT ALL PRIVILEGES ON \*.\* TO ' hazajyaberasamuel '@'localhost' WITH GRANT OPTION;

**2. Create a user with your "names\_semi" as username and your student number as**

**password and give him insert, update, and delete privileges to the created user.**

mysql -u root -p

CREATE USER 'samuel'@'localhost' IDENTIFIED BY '222003581';

GRANT INSERT,UPDATE,DELETE ON\*.\*TO'samuel'@'localhost';

FLUSH PRIVILEGES;

EXIT;

**3. Revoke insert privileges to the last user you created.**

mysql -u root -p

REVOKE INSERT ON \*.\* FROM 'samuel'@'localhost';

FLUSH PRIVILEGES;

EXIT;