

COLLEGE: Business and Economics

SCHOOL: Business

DEPARTMENT: BIT

LEVEL: 2

YEAR: 2

NAMES: MUSEMAKWELI Rossa Belyse Arlande

Reg nbr : 222010101

ASSIGNMENT OF DATABESE MANAGEMENT SYSTEM

ENTITY 1: Users

ATTRIBUTES

UserID

Username

Email

Password

ENTITY 2: Tests

ATTRIBUTES

TestID

UserID

TestDate

ENTITY 3: Questions

ATTRIBUTES

QuestionID

QuestionText

ENTITY 4: TestQuestions

ATTRIBUTES

TestQuestionID

TestID

QuestionID

ENTITY 5: Answers

ATTRIBUTES

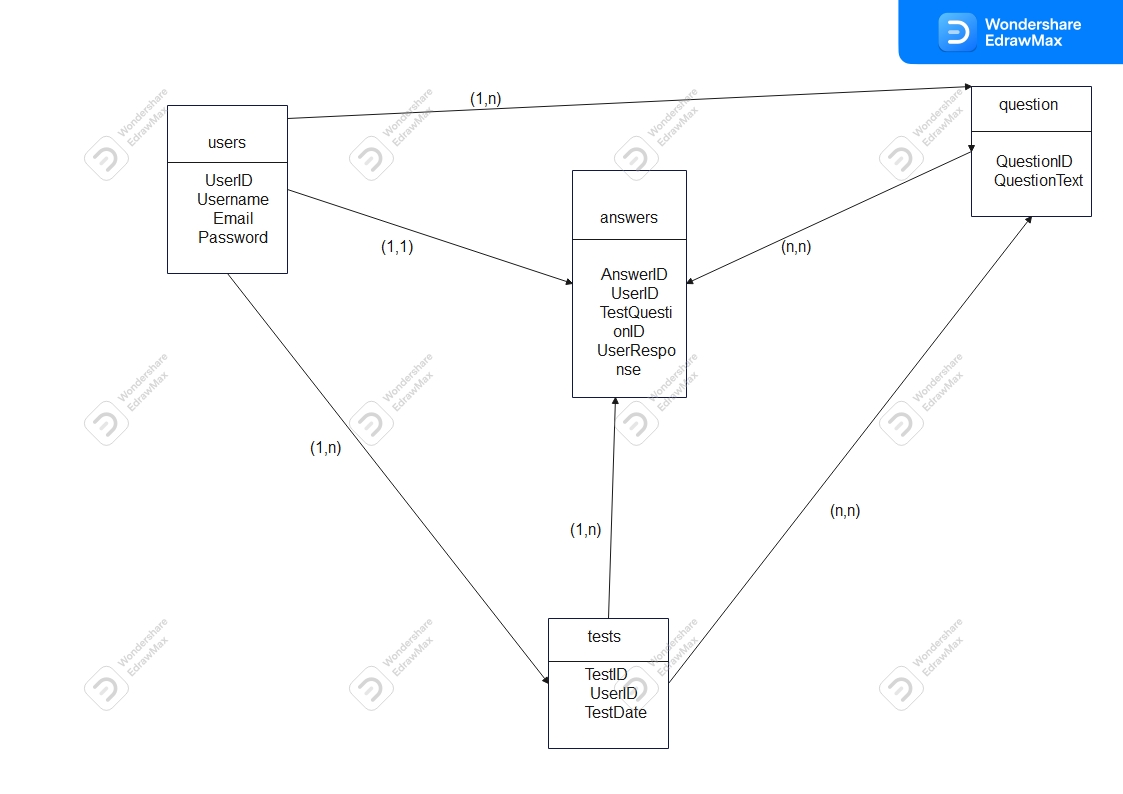
AnswerID

UserID

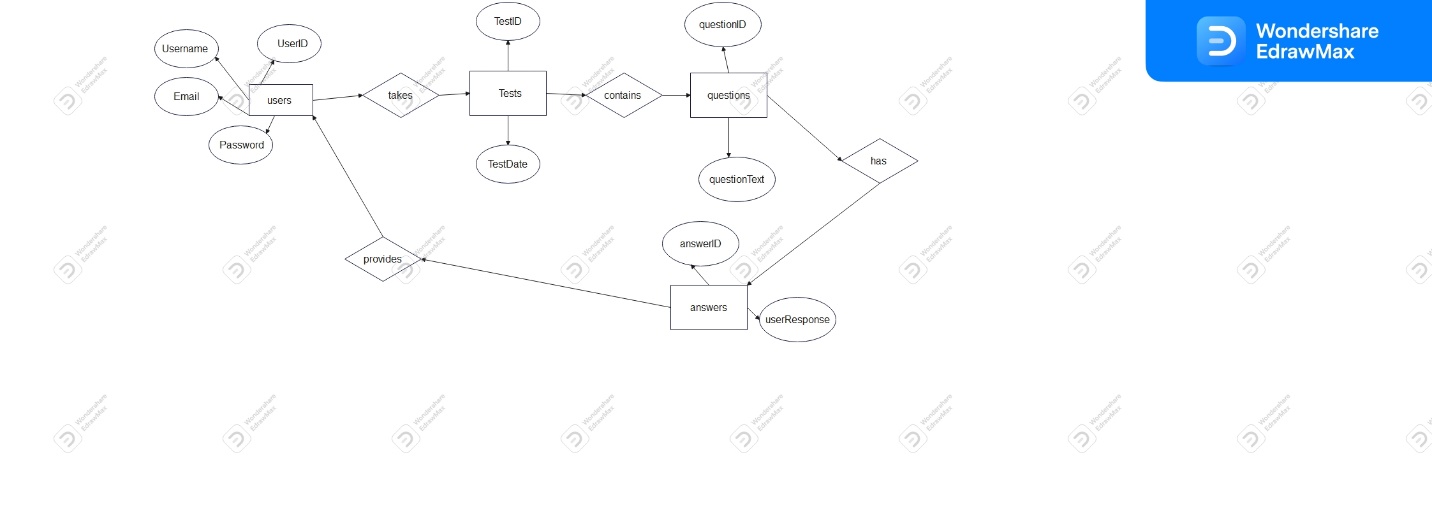
TestQuestionID

UserResponse

2.



3.



Section II SQL

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

CREATE DATABASE online\_pregnancy\_test\_system;

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

CREATE TABLE Users (

UserID INT PRIMARY KEY AUTO\_INCREMENT,

Username VARCHAR(255) NOT NULL,

Email VARCHAR(255) NOT NULL,

Password VARCHAR(255) NOT NULL

);

CREATE TABLE Tests (

TestID INT PRIMARY KEY AUTO\_INCREMENT,

UserID INT NOT NULL,

TestDate DATE NOT NULL,

Result ENUM('Positive', 'Negative', 'Pending') NOT NULL,

FOREIGN KEY (UserID) REFERENCES Users(UserID)

);

CREATE TABLE Questions (

QuestionID INT PRIMARY KEY AUTO\_INCREMENT,

QuestionText TEXT NOT NULL

);

CREATE TABLE TestQuestions (

TestQuestionID INT PRIMARY KEY AUTO\_INCREMENT,

TestID INT NOT NULL,

QuestionID INT NOT NULL,

FOREIGN KEY (TestID) REFERENCES Tests(TestID),

FOREIGN KEY (QuestionID) REFERENCES Questions(QuestionID)

);

CREATE TABLE Answers (

AnswerID INT PRIMARY KEY AUTO\_INCREMENT,

UserID INT NOT NULL,

TestQuestionID INT NOT NULL,

UserResponse ENUM('Yes', 'No') NOT NULL,

FOREIGN KEY (UserID) REFERENCES Users(UserID),

FOREIGN KEY (TestQuestionID) REFERENCES TestQuestions(TestQuestionID)

);

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

INSERT INTO Users (userID, username, email, password)

VALUES

(1, 'ruth\_kizere', 'ruth@gmail.com', 'ruth123'),

-> (2, 'jane\_smith', 'jane@gmail.com', 'secret456'),

-> (3, 'bob\_johnson', 'bob@gmail.com', 'bob789'),

-> (4, 'alice\_jones', 'alice@gmail.com', 'securepass321');

INSERT INTO Tests ( TestID, UserID, TestDate, Results )

(101, 1,’2022-12-01’, ‘positive’),

(102, 2,’2023-06\_08, ‘negative’),

(103,3,’2021-07-11, ‘negative’),

(104, 4, ‘2023-09-10', ‘Positive’),

> INSERT INTO Questions ( QuestionID , QuestionText)

-> VALUES (201,'Have you missed your period?'),

-> (202,'How many days late is your period?'),

-> (203,'Have you experienced any unusual fatigue?'),

-> (204,'Have you noticed any changes in your breast tenderness or size?'),

-> (205,'Have you experienced any unusual cravings or changes in appetite?'),

-> (206,'Have you had any mood swings or emotional changes recently?'),

-> (207,'Have you experienced any nausea or morning sickness?'),

-> (208,'Have you taken a home pregnancy test? If so, what were the results?'),

-> (209,'Have you noticed any changes in your urination frequency or habits?'),

-> (210,'Have you experienced any abdominal cramping or discomfort?'),

-> (211,'Do you have a history of irregular periods or menstrual cycle changes?'),

-> (212,'Have you recently engaged in unprotected sexual intercourse?'),

-> (213,'Are you currently taking any medications that might affect your menstrual cycle or pregnancy?'),

-> (214, 'Have you experienced any changes in your body temperature or basal body temperature?'),

-> (215,'Have you noticed any changes in your sense of smell or taste?');

INSERT INTO TestQuestions (TestQuestionID, TestID, QuestionID)

-> VALUES (301, 101, 201),

-> (302, 102,202),

-> (303, 103, 203),

-> (304, 104, 205),

-> (305, 101, 206),

-> (306, 102, 207),

-> (307, 103,208),

-> (308, 104, 209),

-> (309, 101, 210),

-> (310,103, 214);

INSERT INTO Answers (AnswerID, UserID, TestQuestionID, UserResponse)

VALUES (401 , 1, 301, 'Yes'),

-> (402, 2,302, 'No'),

-> (403, 3, 303, 'Yes'),

-> (404, 4, 304, 'No');

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

Select \* from users

Select \* from tests

Select \* from questions

Select \* from testquestions

Select \* from results

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

Update Tests

-> set testDate ='2023-09-01,'

-> where TestID = 101;

Update questions

-> set questiontext = 'what was your last date of your periods?'

-> where QuestionID = 202;

Section III

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

CREATE VIEW users\_view AS

SELECT \* FROM users;

CREATE VIEW tests\_view AS

SELECT \* FROM tests;

CREATE VIEW questions\_view AS

SELECT \* FROM questions;

CREATE VIEW testquestions\_view AS

SELECT \* FROM testquestions;

CREATE VIEW answers\_view AS

SELECT \* FROM answers;

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

SELECT \* FROM users\_view;

SELECT \* FROM tests\_view;

SELECT \* FROM questions\_view;

SELECT \* FROM testquestions\_view;

SELECT \* FROM answers\_view;

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

TABLE 1:

UPDATE users

SET username = 'kendra\_becci'

TABLE2:

WHERE userID = 2;

UPDATE answers

SET userResponse= 'yes’

WHERE answerID = 402;

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

Table1:

DELETE FROM questions

WHERE questionID =215;

TABLE2:

DELETE FROM testquestions

WHERE testquestionID =305;

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

CREATE VIEW pregnancy\_test\_summary AS

SELECT

(SELECT COUNT(\*) FROM users) AS total\_users,

(SELECT COUNT(\*) FROM questions) AS total\_questions;

Section IV

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

Table 1

DELIMITER //

CREATE PROCEDURE InsertUser(

IN p\_userID INT,

IN p\_username VARCHAR(255),

IN p\_email VARCHAR(255),

IN p\_password VARCHAR(255)

)

BEGIN

INSERT INTO users (userID, username, email, password)

VALUES (p\_userID, p\_username, p\_email, p\_password);

END //

DELIMITER ;

**Table 2**

DELIMITER //

CREATE PROCEDURE InsertQuestion(

IN p\_questionID INT,

IN p\_questionText VARCHAR(255)

)

BEGIN

INSERT INTO questions (questionID, questionText)

VALUES (p\_questionID, p\_questionText);

END //

DELIMITER ;

**Table 3**

DELIMITER //

CREATE PROCEDURE InsertTest(

IN p\_TestID INT,

IN p\_UserID INT,

IN p\_TestDate DATE,

IN p\_Result VARCHAR(255)

)

BEGIN

INSERT INTO tests (TestID, UserID, TestDate, Result)

VALUES (p\_TestID, p\_UserID, p\_TestDate, p\_Result);

END //

DELIMITER ;

**Table 4**

DELIMITER //

CREATE PROCEDURE InsertTestQuestion(

IN p\_TestQuestionID INT,

IN p\_TestID INT,

IN p\_QuestionID INT

)

BEGIN

INSERT INTO testquestions (TestQuestionID, TestID, QuestionID)

VALUES (p\_TestQuestionID, p\_TestID, p\_QuestionID);

END //

DELIMITER ;

**Table 5**

DELIMITER //

CREATE PROCEDURE InsertAnswer(

IN p\_AnswerID INT,

IN p\_UserID INT,

IN p\_TestQuestionID INT,

IN p\_UserResponse VARCHAR(255)

)

BEGIN

INSERT INTO answers (AnswerID, UserID, TestQuestionID, UserResponse)

VALUES (p\_AnswerID, p\_UserID, p\_TestQuestionID, p\_UserResponse);

END //

DELIMITER ;

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

**Table users**

DELIMITER //

CREATE PROCEDURE DisplayUsers()

BEGIN

SELECT \* FROM users;

END //

DELIMITER ;

**Table tests**

DELIMITER //

CREATE PROCEDURE DisplayTests()

BEGIN

SELECT \* FROM tests;

END //

DELIMITER ;

**Table questions**

DELIMITER //

CREATE PROCEDURE GetAllQuestions()

BEGIN

SELECT \* FROM questions;

END //

DELIMITER ;

**Table testquestions**

DELIMITER //

CREATE PROCEDURE DisplayTestQuestions()

BEGIN

SELECT \* FROM testquestions;

END //

DELIMITER **;**

**Table answers**

DELIMITER //

CREATE PROCEDURE GetAllAnswers()

BEGIN

SELECT \* FROM answers;

END //

DELIMITER ;

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

Table 1:

DELIMITER //

CREATE PROCEDURE UpdateUserInfo(

IN userId INT,

IN newUsername VARCHAR(255),

IN newEmail VARCHAR(255)

)

BEGIN

UPDATE users

SET username = newUsername, email = newEmail

WHERE id = userId;

END //

DELIMITER ;

Table 2

DELIMITER //

CREATE PROCEDURE UpdateTestInfo(

IN testId INT,

IN newTestName VARCHAR(255),

IN newTestDescription TEXT

)

BEGIN

UPDATE tests

SET test\_name = newTestName, test\_description = newTestDescription

WHERE test\_id = testId;

END //

DELIMITER ;

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

Table 1

DELIMITER //

CREATE PROCEDURE DeleteQuestion(

IN questionId INT

)

BEGIN

DELETE FROM questions WHERE id = questionId;

END //

DELIMITER ;

DELIMITER //

Table 2

CREATE PROCEDURE DeleteTestQuestion(

IN testQuestionId INT

)

BEGIN

DELETE FROM testquestions WHERE testquestion\_id = testQuestionId;

END //

DELIMITER ;

DELIMITER //

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

CREATE PROCEDURE GetPregnancyTestResultsForUser(

IN userId INT

)

BEGIN

SELECT

u.username AS user\_name,

ptr.test\_date AS test\_date,

ptr.result AS pregnancy\_result

FROM users u

JOIN pregnancy\_test\_results ptr ON u.id = ptr.user\_id

WHERE u.id = userId

ORDER BY ptr.test\_date DESC;

END //

DELIMITER ;

Section V

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

Table 1

DELIMITER //

CREATE TRIGGER AfterTestInsert

AFTER INSERT

ON tests FOR EACH ROW

BEGIN

INSERT INTO audit\_log (test\_id, action, timestamp)

VALUES (test\_id, 'Test Inserted', NOW());

END //

DELIMITER ;

Table 2

DELIMITER //

CREATE TRIGGER AfterUserInsert

AFTER INSERT

ON users FOR EACH ROW

BEGIN

INSERT INTO audit\_log (user\_id, action, timestamp)

VALUES (user\_id, 'User Inserted', NOW());

END //

DELIMITER ;

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

DELIMITER //

CREATE TRIGGER AfterQuestionUpdate

AFTER UPDATE

ON questions FOR EACH ROW

BEGIN

IF NEW.questionText <> OLD.questionText THEN

INSERT INTO audit\_log (questionID, action, old\_questionText, new\_questionText, timestamp)

VALUES (NEW.questionID, 'Question Updated', OLD.questionText, NEW.questionText, NOW());

END IF;

END //

DELIMITER ;

Table 2

DELIMITER //

CREATE TRIGGER AfterAnswerUpdate

AFTER UPDATE

ON answers FOR EACH ROW

BEGIN

IF NEW.UserResponse <> OLD.UserResponse THEN

INSERT INTO audit\_log (AnswerID, UserID, TestQuestionID, action, old\_UserResponse, new\_UserResponse, timestamp)

VALUES (NEW.AnswerID, NEW.UserID, NEW.TestQuestionID, 'Answer Updated', OLD.UserResponse, NEW.UserResponse, NOW());

END IF;

END //

DELIMITER ;

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

Table 1

DELIMITER //

CREATE TRIGGER AfterTestQuestionDelete

AFTER DELETE

ON testquestions FOR EACH ROW

BEGIN

INSERT INTO audit\_log (TestQuestionID, TestID, QuestionID, action, timestamp)

VALUES (OLD.TestQuestionID, OLD.TestID, OLD.QuestionID, 'Test Question Deleted', NOW());

END //

DELIMITER ;

Table 2

DELIMITER //

CREATE TRIGGER AfterQuestionDelete

AFTER DELETE

ON questions FOR EACH ROW

BEGIN

INSERT INTO audit\_log (QuestionID, QuestionText, action, timestamp)

VALUES (OLD.QuestionID, OLD.QuestionText, 'Question Deleted', 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**.

CREATE USER 'musemakweli\_rossa\_belyse\_arlande'@'localhost' IDENTIFIED BY '222010101';

GRANT ALL PRIVILEGES ON \*. \* TO 'musemakweli\_rossa\_belyse\_arlande'@'localhost';

**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.**

CREATE USER 'musemakweli\_rossa\_belyse\_arlande\_semi'@'localhost' IDENTIFIED BY '222010101';

GRANT INSERT, UPDATE, DELETE ON your\_database\_name.\* TO 'musemakweli\_rossa\_belyse\_arlande\_semi'@'localhost';

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

REVOKE INSERT ON your\_database\_name.\* FROM 'musemakweli\_rossa\_belyse\_arlande\_semi'@'localhost';