***Lab 2***

1- ALTER TABLE Students

-> ADD COLUMN Gender ENUM('male', 'female');

A screenshot of a computer

Description automatically generated

2- mysql> ALTER TABLE Students ADD COLUMN BirthDate DATE;

A screenshot of a computer

Description automatically generated

3- mysql> alter table Students

-> drop column Name;

A screenshot of a computer

Description automatically generated

mysql> alter table Students

->add column FirstName VARCHAR(10);

mysql> alter table Students

-> modify column FirstName VARCHAR(10) after ID;

A screenshot of a computer

Description automatically generated

mysql> alter table Students add column LastName VARCHAR(10);

mysql> alter table Students modify column LastName VARCHAR(10) after FirstName;

A screenshot of a computer

Description automatically generated

4- mysql> alter table Student\_Phone add constraint foreign key(StudentID) references Students(ID) on delete cascade;

5- mysql> UPDATE Students set Gender = 'female' where ID = 4;

mysql> UPDATE Students set Gender = 'female' where ID = 5;

mysql> UPDATE Students set Gender = 'male' where ID IN (2,3);

A screenshot of a computer

Description automatically generated

mysql> update Students set BirthDate = '1997-03-03' where ID = 1;

mysql> update Students set BirthDate = '1991-10-01' where ID = 2;

mysql> update Students set BirthDate = '1988-12-13' where ID = 3;

mysql> update Students set BirthDate = '1970-05-07' where ID = 4;

mysql> update Students set BirthDate = '1994-06-04' where ID = 5;

A screenshot of a computer

Description automatically generated

mysql> update Students

-> set FirstName = (case when ID = 1 then 'Asmaa'

when ID = 2 then 'Mohamed'

when ID = 3 then 'Ahmed'

when ID = 4 then 'Maha'

when ID = 5 then 'Lora'

end);

A screenshot of a computer

Description automatically generated

mysql> update Students set LastName = (case when ID = 1 then 'Hamada'

when ID = 2 then 'Omar'

when ID = 3 then 'Ibrahim'

when ID = 4 then 'Mansour'

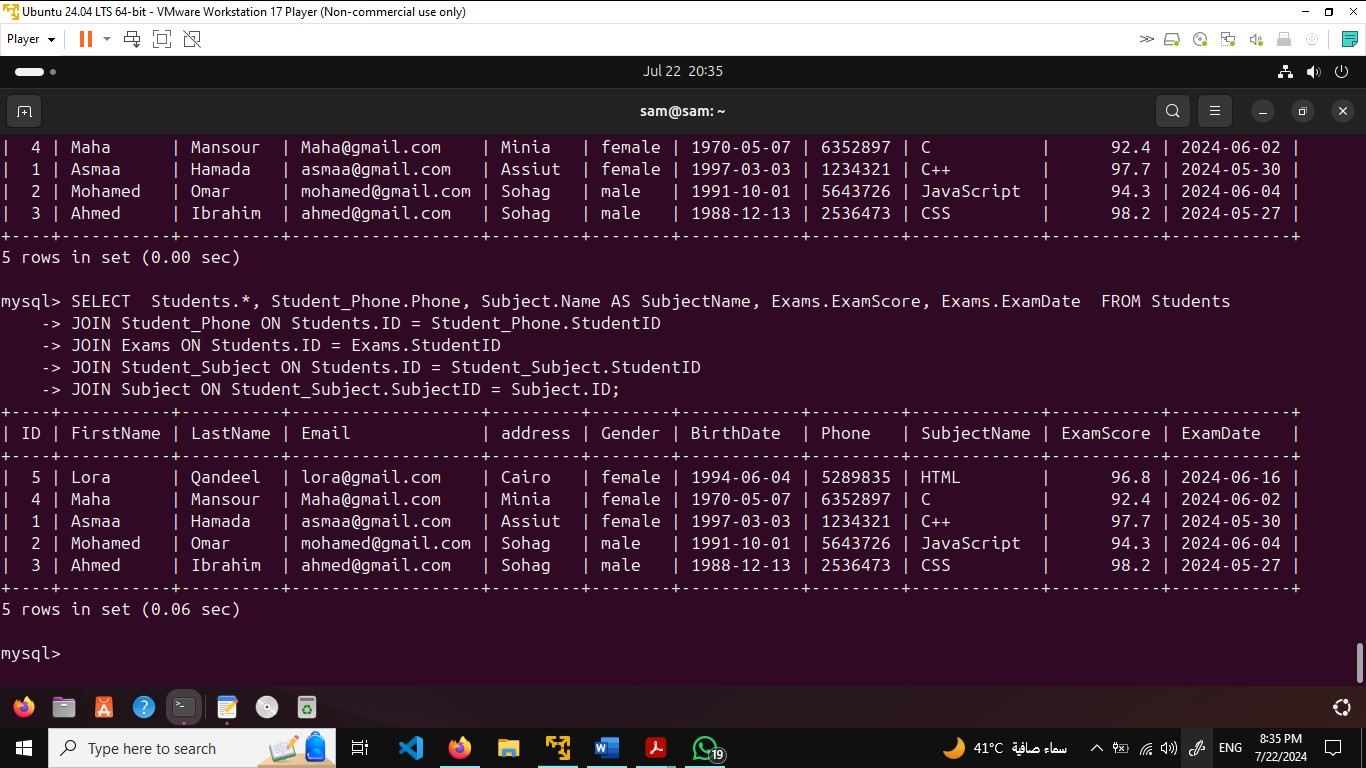
when ID = 5 then 'Qandeel'

end);

A screenshot of a computer

Description automatically generated

7-



8-

A screenshot of a computer

Description automatically generated

9-

A screenshot of a computer

Description automatically generated

10-

A screenshot of a computer

Description automatically generated

11-

A screenshot of a computer

Description automatically generated

12-

A screenshot of a computer

Description automatically generated

13-

A screenshot of a computer

Description automatically generated

14-

A screenshot of a computer

Description automatically generated

15-

A screenshot of a computer

Description automatically generated

16- mysql> SELECT FirstName, COUNT(\*) FROM Students GROUP BY FirstName HAVING COUNT(\*) > 2;

17-

A screenshot of a computer

Description automatically generated

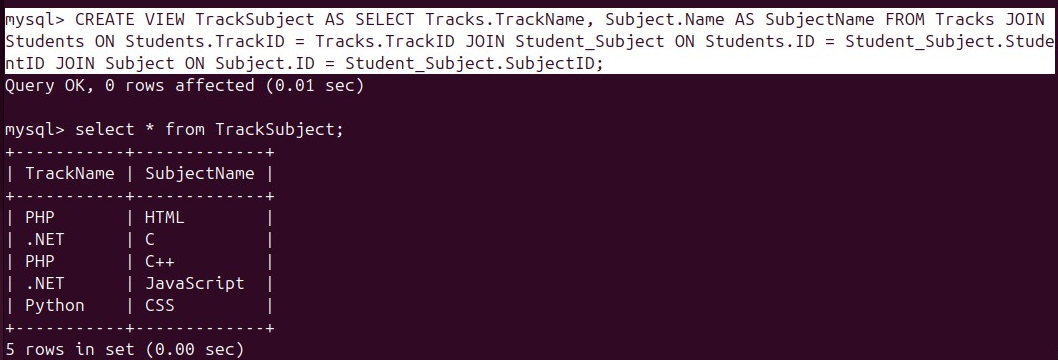
mysql> CREATE VIEW StudentTracks AS SELECT Students.FirstName, Tracks.TrackName FROM Students JOIN Tracks ON Students.TrackID = Tracks.

TrackID;

A screenshot of a computer program

Description automatically generated

18-



19-

mysql> CREATE VIEW StudentSubjects AS SELECT Students.FirstName, Subject.Name AS SubjectName FROM Students JOIN Student\_Subject ON Students.ID = Student\_Subject.StudentID JOIN Subject ON Student\_Subject.SubjectID = Subject.ID;

A screenshot of a computer program

Description automatically generated

20-

A screenshot of a computer

Description automatically generated

21-

A screenshot of a computer program

Description automatically generated

***Lab3***

1-

mysql> START TRANSACTION;

mysql>

mysql> INSERT INTO Students (FirstName, LastName, Email, address)

-> VALUES ('Omar', 'Ahmed', 'omar@gmail.com', 'Alexandria');

mysql> DECLARE @NewStudent INT;

mysql> SET @NewStudent = LAST\_INSERT\_ID();

mysql> INSERT INTO Student\_Phone (StudentID, Phone)

-> VALUES (@NewStudent, '7783784');

mysql> INSERT INTO Exams (StudentID, SubjectID, ExamScore)

-> VALUES (@NewStudent, @NewStudent, 85),

-> (@NewStudent, @NewStudent, 90),

-> (@NewStudent, @NewStudent, 78);

mysql>

mysql> COMMIT;

2-

A computer screen with white text

Description automatically generated

3-

A screen shot of a computer

Description automatically generated

4-

A screenshot of a computer

Description automatically generated

5-

A computer screen shot of a number

Description automatically generated

6-

mysql> INSERT INTO Exams (StudentID, SUbjectID, ExamScore, ExamDate) VALUES (6,6, 85, NOW());

A screenshot of a computer program

Description automatically generated

7-

A computer screen shot of white text

Description automatically generated

8-

DELIMITER //

CREATE FUNCTION MultiplyNumbers (a FLOAT, b FLOAT)

RETURNS FLOAT

DETERMINISTIC

BEGIN

RETURN a \* b;

END //

DELIMITER ;

A screenshot of a computer

Description automatically generated

9-

DELIMITER //

CREATE FUNCTION GeExamScore(StudentID INT, exam\_id INT)

RETURNS INT

DETERMINISTIC

BEGIN DECLARE exam\_score INT;

SELECT ExamScore

INTO exam\_score FROM exams

WHERE StudentID = StudentID AND SubjectID = exam\_id;

RETURN exam\_score;

END

DELIMITER ;

10-

def count\_failed\_students(exam\_scores, exam\_id):

failed\_count = 0 for student\_scores in exam\_scores.values():

if exam\_id in student\_scores and student\_scores[exam\_id] < 50:

failed\_count += 1 return failed\_count

11- def average\_max\_grades(subject\_max\_scores, subject\_idif subject\_id not in subject\_max\_scores: return f"Subject with ID {subject\_id} not found." max\_score = subject\_max\_scores[subject\_id] return max\_score # Example usage subject\_max\_scores = { "subject\_1": 100, "subject\_2": 100 } subject\_id = "subject\_1" average\_score = average\_max\_grades(subject\_max\_scores, subject\_id) print(f"Average max grades for subject {subject\_id}: {average\_score}")

12-

CREATE TABLE Deleted\_Students ( Name VARCHAR(100), Email VARCHAR(100), Address VARCHAR(255), id INT PRIMARY KEY );

13-

CREATE TRIGGER after\_student\_delete AFTER DELETE ON Students FOR EACH ROW BEGIN INSERT INTO Deleted\_Students (Name, Email, Address, id) VALUES (OLD.Name, OLD.Email, OLD.Address, OLD.id); END;

14-

CREATE TABLE Backup\_Students AS

SELECT \* FROM Students WHERE 0;

CREATE TRIGGER after\_student\_insert

AFTER INSERT ON Students FOR EACH ROW BEGIN INSERT INTO Backup\_Students (Name, Email, Address, id) VALUES (NEW.Name, NEW.Email, NEW.Address, NEW.id); END;

16-

sudo mysqldump --databases GradingDatabase > GradingDatabase.sql

17-

sudo mysqldump --databases GradingDatabase Students > students\_table.sql

18-

sudo mysqldump --databases Grading\_Backup < grading\_database.sql