**create database grades;MySQL Labs**

**MySQL (Day1):**

|  |  |
| --- | --- |
|  | **Create a database called grades** |
|  | **create database grades;** |
|  | **Create the following tables in the grades database:**  ***courses***  ***course\_id*** *int pk*  *course\_name varchar(100) not null*  *credit\_hour int*  ***students\_courses***  ***course\_id*** *int*  ***student\_id*** *int*  *grade int*  *reg\_date date*  ***students***  ***student\_id*** *int pk*  *student\_name varchar (100) not null*  *email varchar (50)*  *tel varchar (20)* |
|  | **students table**  **create table students ( student\_id INT PRIMARY KEY,**  **-> student\_name VARCHAR(100) NOT NULL,**  **-> email VARCHAR(50),**  **-> tel VARCHAR(20));**    **Courses table**  **create table courses (course\_id INT PRIMARY KEY,**  **-> course\_name VARCHAR(100) NOT NULL,**  **-> credit\_hours INT);**    **Student\_courses table**  **CREATE TABLE student\_courses (**  **-> course\_id INT,**  **-> student\_id INT,**  **-> grade INT,**  **-> reg\_date DATE,**  **-> FOREIGN KEY (course\_id) REFERENCES courses(course\_id),**  **-> FOREIGN KEY (student\_id) REFERENCES students(student\_id),**  **-> PRIMARY KEY (course\_id, student\_id)**  **-> );** |
| **3** | **Modify the students table to allow for longer Student names (150 char)**  **Confirm your modification.** |
|  | **ALTER TABLE students**  **-> MODIFY student\_name VARCHAR(150);** |
| **4** | **Add constraint to force unique email for each student** |
|  | **ALTER TABLE students**  **-> MODIFY email VARCHAR(50) UNIQUE;** |
| **5** | **Get Time, Date, Current user, MySQL Version using prompt?** |
|  | **select NOW() as "time and date", VERSION() as "current version", USER() as "curretn user";** |
| **6** | **Add gender column for the students table. It holds two value (male or female)** |
|  | **ALTER TABLE students**  **-> ADD COLUMN gender enum('male', 'female');** |
| **7** | **Add birth\_date column for the students table.** |
|  | **ALTER TABLE students**  **-> ADD COLUMN birth\_date DATE;** |
| **8** | **Drop the student\_name column and replace it with first name and last name.** |
|  | **ALTER TABLE students**  **-> DROP COLUMN student\_name;**    **ALTER TABLE students**  **-> ADD COLUMN first\_name VARCHAR(100),**  **-> ADD COLUMN last\_name VARCHAR(100);** |
| **9** | **Insert your friend’s data into the table students.** |
|  | **INSERT INTO students**  **-> VALUES (1, 'youssefmagdy@ieee.org', '201211249403', 'male', '2002-12-02', 'Youssef', 'Magdy');** |
| **10** | **Create a new table (male\_students) based on students table and fill it with the data of male students** |
|  | **CREATE TABLE male\_students AS**  **-> SELECT \* FROM students**  **-> WHERE gender='male';** |

**Part II**

**Create another database “OS”**

**Use OS**

**Run Lab Script then answer the following**

**# SOURCE D:\lab\_script.txt**

|  |  |
| --- | --- |
| **1** | **Display all students’ information.** |
|  | **SELECT \* FROM students;** |
| **2** | **Display male students only.** |
|  | **SELECT \* FROM students WHERE gender=’male’;** |
| **3** | **Display the number of female students.** |
|  | **SELECT COUNT(gender) FROM students WHERE gender=’female’;** |
| **4** | **Display the students’ data for the students who are born before 1992-10-01.** |
|  | **SELECT \* FROM students WHERE birth\_date < ‘1992-10-01’;** |
| **5** | **Display the students’ data for the students who are born before 1991-10-01.** |
|  | **SELECT \* FROM students WHERE birth\_date < ‘1991-10-01’ AND gender = 'male';** |
| **6** | **Display course\_id and their grades sorted by grades.** |
|  | **SELECT course\_id, grade FROM students\_courses ORDER BY grade;** |
| **7** | **Display students’ names that begin with A.** |
|  | **SELECT CONCAT(first\_name, ' ', last\_name) AS "Student Name" FROM students**  **-> WHERE first\_name LIKE 'A%';** |
| **8** | **Display the gender, number of males and females.** |
|  | **SELECT gender, COUNT(gender) FROM students**  **-> GROUP BY (gender);** |
| **9** | **Display the repeated first names and their counts if higher than 2.** |
|  | **SELECT first\_name, COUNT(first\_name)**  **-> FROM students**  **-> GROUP BY first\_name**  **-> HAVING COUNT(first\_name) > 2;** |
| **10** | **Display the subject with highest grade** |
|  | **SELECT course\_id FROM students\_courses ORDER BY grade DESC LIMIT 1;** |