**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)* |
|  | ***USE grades;***  ***CREATE TABLE students (***  ***Student\_id INT PRIMARY KEY AUTO\_INCREMENT,***  ***Student\_name VARCHAR(100) NOT NULL,***  ***Email VARCHAR(50),***  ***Tel VARCHAR(20)***  ***);***  ***CREATE TABLE courses (***  ***Course\_id INT PRIMARY KEY AUTO\_INCREMENT,***  ***Course\_name VARCHAR(100) NOT NULL,***  ***Credit\_hour INT***  ***);***  ***CREATE TABLE students\_courses (***  ***Course\_id INT,***  ***Student\_id INT,***  ***Grade INT,***  ***Reg\_date DATE,***  ***CONSTRAINT `FK\_COURSE\_ID` FOREIGN KEY(`course\_id`) REFERENCES `courses`(`course\_id`) ON DELETE CASCADE,***  ***CONSTRAINT `FK\_STUDENT\_ID` FOREIGN KEY(`student\_id`) REFERENCES `students`(`student\_id`) ON DELETE CASCADE,***  ***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 COLUMN Student\_name VARCHAR(150);*** |
| **4** | **Add constraint to force unique email for each student** |
|  | ***ALTER TABLE students ADD UNIQUE(email);*** |
| **5** | **Get Time, Date, Current user, MySQL Version using prompt?** |
|  | ***SELECT VERSION(), USER(), CURRENT\_DATE(), CURRENT\_TIME();*** |
| **6** | **Add gender column for the students table. It holds two value (male or female)** |
|  | ***ALTER TABLE students ADD gender ENUM('male', 'female');*** |
| **7** | **Add birth\_date column for the students table.** |
|  | ***ALTER TABLE students ADD birth\_date DATE;*** |
| **8** | **Drop the student\_name column and replace it with first name and last name.** |
|  | ***ALTER TABLE students DROP student\_name;***  ***ALTER TABLE students ADD first\_name VARCHAR(50);***  ***ALTER TABLE students ADD last\_name VARCHAR(50);*** |
| **9** | **Insert your friend’s data into the table students.** |
|  | ***INSERT INTO students (first\_name, last\_name, gender, email) VALUES ('ahmed', 'magdy', 'male', 'test@app.com');*** |
| **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 “OS42”**

**Use OS42**

**Run Lab Script then answer the following**

|  |  |
| --- | --- |
| **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(student\_id) AS number\_of\_females 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 male students who are born before 1991-10-01.** |
|  | ***SELECT \* FROM students WHERE birth\_date < '1991-10-01';*** |
| **6** | **Display course\_id and their grades sorted by grades.** |
|  | ***SELECT course\_id, grade FROM students\_courses ORDER BY grade DESC;*** |
| **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 COUNT(student\_id) AS count, gender FROM students GROUP BY gender;*** |
| **9** | **Display the repeated first names and their counts if higher than 2.** |
|  | ***SELECT COUNT(first\_name) AS count, first\_name FROM students GROUP BY first\_name HAVING COUNT(first\_name) > 2;*** |
| **10** | **Display the subject with highest grade** |
|  | ***SELECT \* FROM courses, (SELECT course\_id, MAX(grade) FROM students\_courses) q WHERE courses.course\_id = q.course\_id;*** |