Lab 04 - DDL

Objective:

The purpose of this lab is to introduce you to the DDL set of statements in SQL. By writing SQL to create tables, constraints, and views, you will have the tools needed to implement database designs that you will create later in the course. By finishing this lab, the student will be able to:

- create, modify, and drop tables based on design specifications provided,
- inserting new data into tables, update data in tables, and delete data from tables while considering referential integrity,
- enforce constraints on tables to ensure data integrity and consistency,
- create a table using the structure and data from an existing table,
- import data into a table from other tables.

Submission:

Your submission will be a single text-based .SQL file with the solutions provided.

Your submission needs to include a comment header block and be commented to include the question and the solutions. Make sure every SQL statement terminates with a semicolon.

Tasks:

Add
```
SET AUTOCOMMIT ON;
```
under the comment header and execute it

Consider the following table specifications:

Part A (DDL) :

1. Create table the following tables and their given constraints:

MOVIES `(movieid:int, title:varchar(35), year:int, director:int,score:decimal(3,2))`

Column Name	Column DataType	PK	Not Null	Unique	FK	Default Value	Validation
mid	Int	✓					
title	varchar(35)		✓				
releaseYear	Int		✓				
director	Int		✓				
score	decimal(3,2)						< 5 and > 0

ACTORS (actorid:int, name:varchar(20), lastname:varchar(30))

Column Name	Column DataType	PK	Not Null	Unique	FK	Default Value	Validation
aid	Int	✓					
firstName	varchar(20)		✓				
lastName	Varchar(30)		✓				

CASTINGS (movieid:int, actorid:int)

Column Name	Column DataType	PK	Not Null	Unique	FK	Default Value	Validation
movieid	Int	✓			✓ (movies)		
actorid	int	✓			✓ (actors)		

DIRECTORS(id:int, name:varchar(20), lastname:varchar(30))

Column Name	Column DataType	PK	Not Null	Unique	FK	Default Value	Validation
directorid	Int	✓					
firstname	varchar(20)		✓				
lastname	varchar(30)		✓				

2. Modify the ***movies*** table to create a foreign key constraint that refers to table ***directors***.

3. Modify the ***movies*** table to create a new constraint so the uniqueness of the movie title is guaranteed.

4. Write insert statements to add the following data to table ***directors*** and ***movies***.

Director

directorid	First name	Last name
1010	Rob	Minkoff
1020	Bill	Condon
1050	Josh	Cooley
2010	Brad	Bird
3020	Lake	Bell

Movies

id	title	year	director	score
100	The Lion King	2019	3020	3.50
200	Beauty and the Beast	2017	1050	4.20
300	Toy Story 4	2019	1020	4.50
400	Mission Impossible	2018	2010	5.00
500	The Secret Life of Pets	2016	1010	3.90

5. Write SQL statements to remove all above tables.
   Is the order of tables important when removing? Why?

Part B (More DML):

6. Create a new empty table ***employee2*** the same as table ***employees***. Use a single statement to create the table and insert the data at the same time.

7. Modify table ***employee2*** and add a new column ***username*** to this table. The value of this column is not required and does not have to be unique.

8. Delete all the data in the ***employee2*** table

9. Re-insert all data from the ***employees*** table into your new table ***employee2*** using a single statement.

10. In table ***employee2***, write a SQL statement to change the first name and the last name of employee with ID ***1002*** to your name.

11. In table ***employee2***, generate the email address for column ***username*** for each student by concatenating the first character of employee's first name and the employee's last name. For instance, the username of employee Peter Stone will be ***pstone***. NOTE: the username is in all lower case letters.

12. In table ***employee2***, remove all employees with office code 4.

13. Drop table ***employee2***.