Lab 05- DDL

Submission:

Your submission will be a single WORD 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 all the following tables and their given constraints:

`LAB5_MOVIES` `(movieid:int, title:varchar(35), releaseYear:int, director:int,score:decimal(3,2))`

Column Name	Column DataType	PK	Not Null	Unique	FK	Default Value	Validation
movieid	Int	✓					
title	varchar(35)		✓				
releaseYear	Int		✓				
director	Int		✓				
score	decimal(3,2)						< 10 and > 3

LAB5_ACTORS (actorid:int, firstname:varchar(20), lastname:varchar(30))

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

LAB5_CASTINGS (movieid:int, actorid:int)

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

LAB5_DIRECTORS(directorid:int, firstname: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)		✓				

➔**Answer:**

```
SET AUTOCOMMIT ON;

CREATE TABLE lab5_movies
   (movieid int PRIMARY KEY,
   title varchar(35) NOT NULL,
   releaseYear int  NOT NULL,
   director int  NOT NULL,
   score decimal(3,2)
   CONSTRAINT score_chk CHECK (score >3 AND score <10 ) );


SET AUTOCOMMIT ON;
```

```
CREATE TABLE lab5_actors
        (actorid int PRIMARY KEY,
        firstName varchar(20) NOT NULL,
        lastname varchar(30)  NOT NULL);


SET AUTOCOMMIT ON;

CREATE TABLE lab5_castings
        (movieid int,
        actorid int,
        CONSTRAINT movieid_actorid_pk PRIMARY KEY (movieid,actorid),
        CONSTRAINT movieid_fk FOREIGN KEY (movieid) REFERENCES lab5_movies(movieid),
        CONSTRAINT actorid_fk FOREIGN KEY (actorid)  REFERENCES lab5_actors(actorid) );


SET AUTOCOMMIT ON;

CREATE TABLE lab5_directors
        (directorid int PRIMARY KEY,
        firstname varchar(20) NOT NULL,
        lastname varchar(30)  NOT NULL);
```

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

```
SET AUTOCOMMIT ON;
ALTER TABLE lab5_movies
ADD CONSTRAINT director_fk FOREIGN KEY (director) REFERENCES lab5_directors(directorid);
```

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

```
SET AUTOCOMMIT ON;

ALTER TABLE lab5_movies
ADD CONSTRAINT title_unique UNIQUE (title);
```

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

➔**Answer:**

```
SET AUTOCOMMIT ON;

INSERT ALL
INTO lab5_directors VALUES (1010,'Rob','Minkoff')
INTO lab5_directors VALUES (1020,'Bill','Condon')
INTO lab5_directors VALUES (1050,'Josh','Cooley')
INTO lab5_directors VALUES (2010,'Brad','Bird')
INTO lab5_directors VALUES (3020,'Lake','Bell')
INTO lab5_movies VALUES (100,'The Lion King',2019,3020,3.50)
INTO lab5_movies VALUES (200,'Beauty and the Beast',2017,1050,4.20)
INTO lab5_movies VALUES (300,'Toy Story 4',2019,1020,4.50)
INTO lab5_movies VALUES (400,'Mission Impossible',2018,2010,5.00)
INTO lab5_movies VALUES (500,'The Secret Life of Pets',2016,1010,3.90)
SELECT * FROM DUAL;
```

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

```
DROP TABLE lab5_castings;
DROP TABLE lab5_actors;
DROP TABLE lab5_movies;
DROP TABLE lab5_directors;
```

==> When removing tables, the order is important. Because of Referential Integrity, the tables with foreign keys must be removed first then the parent tables can be deleted

Part B (More DML):

6. Create a new empty table (that means the table will not have any data after creating) ***employeecopy*** the same as table ***retailemployees.*** Use a single statement to create the table and insert the data at the same time (Hint use a WHERE clause that is false like 1=2)
   ➔**Answer:**

```
SET AUTOCOMMIT ON;

CREATE TABLE employeecopy AS
(SELECT * FROM retailemployees
WHERE 1=2);
```

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

```
SET AUTOCOMMIT ON;

ALTER TABLE employeecopy
ADD username CHAR(50);
```

8. Re-insert all data from the ***retailemployees.*** table into your new table ***employeecopy*** using a single statement.
   ➔**Answer:**

```
SET AUTOCOMMIT ON;
```

```
INSERT INTO employeecopy
(employeenumber,lastname,firstname,extension,email,officecode,reportsto,jobtitle)
        (SELECT employeenumber,lastname,firstname,extension,email,officecode,reportsto,jobtitle
        FROM retailemployees );
```

9. In table ***employeecopy***, generate the email address for column ***username*** for each student by concatenating the employeeid and the string "@seneca.ca". For instance, the username of employee 123 will be "123@seneca.ca'.
   ➔**Answer:**

```
SET AUTOCOMMIT ON;

UPDATE employeecopy
SET username = (employeenumber || '@seneca.ca');
```

10. Delete all the employeecopy data and display the data in the table. Does employeecopy exist? If not how can you delete table ***employeecopy***.
    ➔**Answer:**

```
SET AUTOCOMMIT ON;

DELETE FROM employeecopy;
SELECT * FROM employeecopy;
```

==> The table employeecopy still exists but has no data. To delete employee table, I do the DROP command

```
DROP TABLE employeecopy;

SELECT * FROM employeecopy;
```