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) |  | ✓ |  |  |  |  |

è**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);

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

è**Answer:**

SET AUTOCOMMIT ON;

ALTER TABLE lab5\_movies

ADD CONSTRAINT director\_fk FOREIGN KEY (director) REFERENCES lab5\_directors(directorid);

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

è**Answer:**

SET AUTOCOMMIT ON;

ALTER TABLE lab5\_movies

ADD CONSTRAINT title\_unique UNIQUE (title);

1. 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 |

è**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;

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

è**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):

1. 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)

è**Answer:**

SET AUTOCOMMIT ON;

CREATE TABLE employeecopy AS

(SELECT \* FROM retailemployees

WHERE 1=2);

1. 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.

è**Answer:**

SET AUTOCOMMIT ON;

ALTER TABLE employeecopy

ADD username CHAR(50);

1. Re-insert all data from the ***retailemployees.***  table into your new table ***employeecopy*** using a single statement.

è**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 );

1. 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’.

è**Answer:**

SET AUTOCOMMIT ON;

UPDATE employeecopy

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

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

è**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;