**Student Name: Weight: 1.25%**

**Student ID:** **Marks:** **/34**

# Lab: SQL Commands and Database Sequences and Views

## Equipment and Materials

For this lab, you will need:

* A Windows computer with a minimum of 16 GB RAM and 250 GB of free disk space, capable of nested virtualization
* Access to ORACLE SQL\*PLUS
* The More Movies database

**Note:** Execute the **Create\_MM.sql** script file, located in the *Course Resources* section of Brightspace.

* A CPRG 307 user account (see instructions below)

## Instructions

#### Part A: Complete the Pre-Lab Tasks

1. Attend the lectures related to the lab activities.
2. Complete the out-of-class learning activities, as indicated by your instructor.
3. Download and review the More Movies database.
4. Create a CPRG 307 user account.
   1. Execute the **Create\_CPRG307\_Users.sql** script located in the *Course Resources* section of Brightspace.
   2. Ensure that you log into the database as the user SYSTEM.

This script creates two new users with specific privileges you will be using in the labs for this course.

**Notes:**

* You don’t need to understand the code in this script.
* If you see errors when executing the script, ask your instructor for guidance.
* If you need to remove the users, execute **Drop\_CPRG307\_Users.sql** script in the *Course Resources* section of Brightspace.
* This step does not need to be presented in your discussion board post.

1. Create the More Movies database tables in your database.
   1. Log into the database using SQL\*Plus as user **CPRG307** with the password: **password**.
   2. Execute the script **Create\_MM.sql** (e.g., **@c:\cprg307\Create\_MM.sql**). Do not copy and paste.

**Notes:**

* For this lab, it is recommended that you use SQL\*Plus, rather than SQL Developer, to remind you how to execute a file.
* You can use SQL Developer in this course if you wish.
* This step does not need to be presented in your discussion board post.

1. See Brightspace for the lab due date.

#### Part B: Complete the Lab Tasks (create a script that includes the solutions)

1. Display the structure of the MM\_MEMBER table. (2 marks)
2. Add yourself as a member. (2 marks)

**Hint:** Only populate the first three columns.

1. Modify your membership by adding a made-up credit card number. **Do not** use your real-life credit card number. (2 marks)

**Hint:** There is a check constraint on this column.

1. Remove your membership. (2 marks)
2. Save your data changes. (2 marks)
3. Display the title of each movie, the rental ID and the last names of all members who have rented those movies. (2 marks)
   1. Sort the result set by the rental ID.
   2. Ensure that no other information appears.
   3. Use three tables for this query: MM\_MEMBER, MM\_MOVIE and MM\_RENTAL.

**Restriction:** Solve using JOIN…ON as your join method.

1. Display the title of each movie, the rental ID, and the last names of all members who have rented those movies. (2 marks)
   1. No other information should appear.
   2. Use three tables for this query: MM\_MEMBER, MM\_MOVIE and MM\_RENTAL.

**Restriction:** Solve using the traditional join method, where join is in the WHERE clause.

1. Create a new table called MY\_TABLE that is made up of three columns: MY\_NUMBER, MY\_DATE and MY\_STRING, and that have data types: NUMBER, DATE and VARCHAR2(5), respectively. (2 marks)
2. Create a new sequence called **seq\_movie\_id**. Have the sequence start at 20 and increment by 2. (2 marks)
3. Display the sequence information (at least the last number and increment by) from the data dictionary’s **user\_sequences** view.

**Note:** Your output should only show this one sequence.

1. Use a query to display the next sequence number on the screen. (2 marks)
2. Change the sequence created in Step 9 to increment by 5 instead of 2. (2 marks)
3. Add your favorite movie to the MM\_MOVIE table using the sequence created in Step 9 for the movie\_id. (2 marks)

**Notes:**

* You can create values for the other columns (all columns must be given a value).
* MM\_MOVIE has a foreign key, which means any value placed in this column must already exist as primary key value in the table being referenced.
* MM\_MOVIE has a check constraint.

1. Create a view named **VW\_MOVIE\_RENTAL** using the query from either Step 6 or Step 7. (2 marks)
2. Use a query to display the data accessed by the **VW\_MOVIE\_RENTAL** view. (2 marks)
3. Make the **VW\_MOVIE\_RENTAL** view **read only**. (2 marks)
4. Using the **VW\_MOVIE\_RENTAL** view in Step 14, **change** the last name of the member who rented the movie with the ID of 2 to **Tangier 1**. (2 marks)
   1. Why does this UPDATE cause an error?
5. Submit your completed script to the forum and topic in the Brightspace Discussion board indicated by your instructor by the due date.

**Note:** Submit your code in the body of the discussion board post rather than as an attachment.

#### Part C: Complete the Post-Lab Tasks

1. Compare your posted solution to the solution posted by your instructor.
2. Talk with your instructor if you are unsure why there are differences between the solutions.