Lab 05 – SQL (DDL)

# **Objectives:**

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.***

**DBS211\_L05\_LastName.sql**

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

Example Submission

|  |
| --- |
| -- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  -- Name: Your Name  -- ID: #########  -- Date: The current date  -- Purpose: Lab 05 DBS211  -- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  -- Q1 SOLUTION --  SELECT \* FROM TABLE;  -- Q2 SOLUTION –  SELECT \* FROM TABLE; |

Locate, select, and submit the file to the Lab 05 link.

## **Setup**

# Create a new worksheet in SQL developer and add an appropriate comment header that includes your name, student id, the date and the purpose of the file (i.e. DBS211 – Lab 05).

## **Style Guide**

Your SQL should be written using the standard coding style:

* all keywords are to be upper case,
* all user-defined names are to be lower case, (example: table and field names)
* there should be a carriage return before each major part of the SQL statements (i.e. before SELECT, FROM, WHERE and ORDER BY)

See the following sample:

**SELECT** columns

**FROM** tables

**WHERE** conditions

**ORDER** **BY** column1, column2;

**Marking Scheme**

|  |  |  |  |
| --- | --- | --- | --- |
| **Question** | **Points** | **Question** | **Points** |
| **1** | 4 |  |  |
| **2** | 1 |  |  |
| **3** | 1 |  |  |
| **4** | 1 |  |  |
| **5** | 1 |  |  |
| **6** | 1 |  |  |
| **7** | 1 |  |  |

Total: 10

**Grade Policy**

* Solutions with errors do not get any marks. (They get zero.)
  + Execute your *.sql* file using the “Run Script” button to make sure there is no errors in your file.
* If your result in a question does not match the sample output results, you do not get any marks.
* You do not receive marks for the missing or incomplete solutions.
* The name of the tables you will create in this lab have to match exactly with the names given this lab document.

# **Tasks:**

Add

SET AUTOCOMMIT ON;   
under the comment header and execute it.

For each question write the question number:

**-- Q1**

**Solution**

If you do not know the answer to a question, write the following statement for that question:

**SELECT ‘No Solution’**

**FROM dual;**

Consider the following table specifications:

## **Part A (DDL)**

1. Create the following tables and their given constraints: (4 Marks)

**L5\_MOVIES**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Column  Name | Column  DataType | PK | Not  Null | Unique | FK | Default  Value | Validation |
| m\_id | int | ✓ |  |  |  |  |  |
| title | varchar(35) |  | ✓ |  |  |  |  |
| release\_year | int |  | ✓ |  |  |  |  |
| director | int |  | ✓ |  | (L5\_DIRECTORS) |  |  |
| score | decimal(3,2) |  |  |  |  |  | < 5 and > 0 |

**L5\_ACTORS**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Column  Name | Column  DataType | PK | Not  Null | Unique | FK | Default  Value | Validation |
| a\_id | int | ✓ |  |  |  |  |  |
| first\_name | varchar(20) |  | ✓ |  |  |  |  |
| last\_name | Varchar(30) |  | ✓ |  |  |  |  |

**L5\_CASTINGS**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Column  Name | Column  DataType | PK | Not  Null | Unique | FK | Default  Value | Validation |
| movie\_id | int | ✓ |  |  |  |  |  |
| actor\_id | int | ✓ |  |  |  |  |  |

**L5\_DIRECTORS**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Column  Name | Column  DataType | PK | Not  Null | Unique | FK | Default  Value | Validation |
| director\_id | int | ✓ |  |  |  |  |  |
| first\_name | varchar(20) |  | ✓ |  |  |  |  |
| last\_name | varchar(30) |  | ✓ |  |  |  |  |

1. Modify the ***L5\_CASTINGS*** table to create two foreign key constraints that refer to tables ***L5\_ACTORS*** *and* ***L5\_MOVIES***.
2. Modify the ***L5\_DIRECTORS*** table to add a new column named *username* to store the director’s username. This is a column of type character with maximum length of 60.
3. Modify the ***L5\_DIRECTORS*** table to create a new constraint so the uniqueness of the username column is guaranteed.
4. Write a single *INSERET* statement to add the following data to the table ***L5\_DIRECTORS.***

**L5\_DIRECTORS**

|  |  |  |
| --- | --- | --- |
| director\_id | first\_name | last\_name |
| 1010 | Rob | Minkoff |
| 1020 | Bill | Condon |
| 1050 | Josh | Cooley |
| 2010 | Brad | Bird |
| 3020 | Lake | Bell |

1. In table ***L5\_Director***, generate the username for column ***username*** for each director by concatenating the first character of the first name and the last name. For instance, the username of a director Peter Stone will be ***pstone***. NOTE: the username is in all lower case letters.
2. Write SQL statements to remove all above tables.   
   Is the order of tables important when removing? Why?

After completing the lab, in SQL Developer, select all statements and click on “Run Statement”. Make sure there is no error in your statements.

Good luck.