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 WORD file with the solutions provided for all the questions with SQL queries and screenshots.***

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.

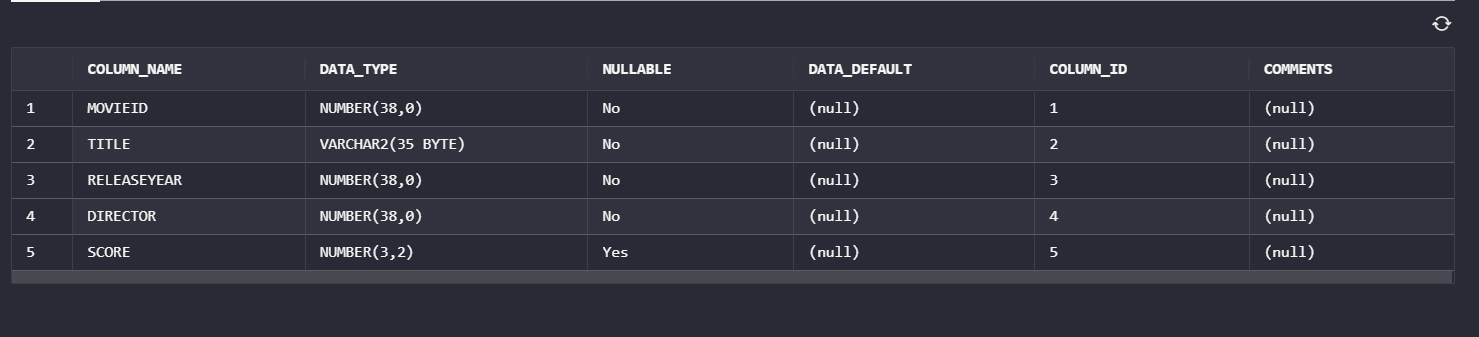
# Tasks:

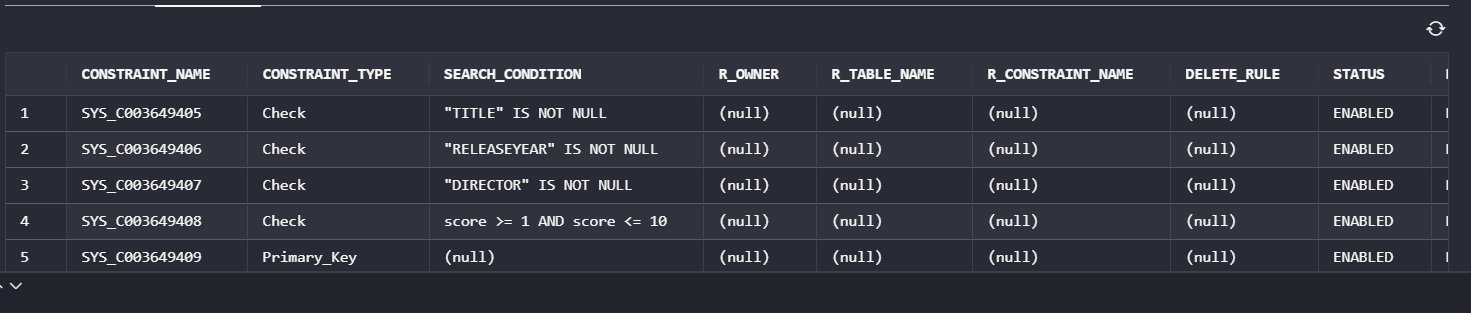
Add   
SET AUTOCOMMIT ON;   
under the comment header and execute it

1. Create the following table and their given constraints. Show the table query and the screenshot of columns tab and constraints tab after executing the table query.

**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 > 1 |

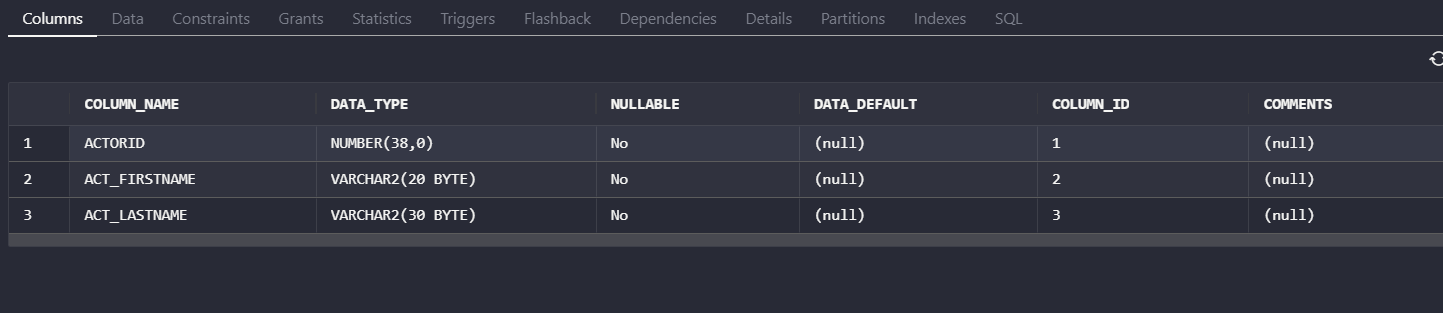


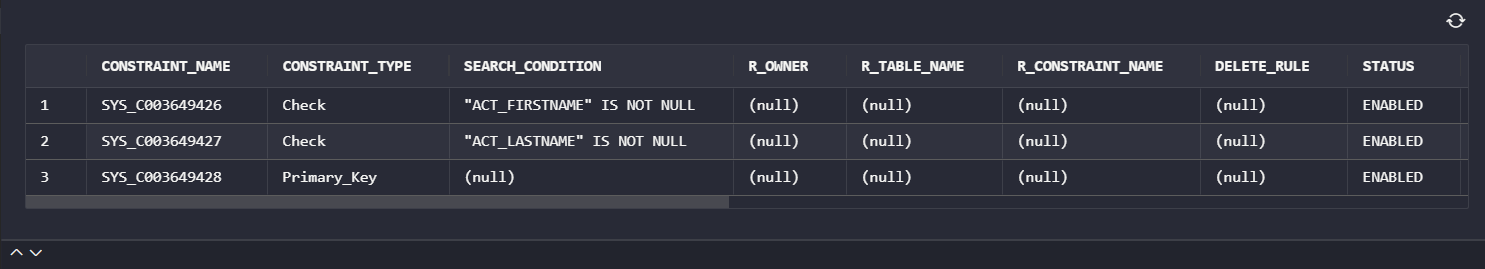


1. Create the following table and their given constraints. Show the table query and the screenshot of columns tab and constraints tab after executing the table query.

**ACTORS** (actorid:int, firstname:varchar(20), lastname:varchar(30))

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Column  Name | Column  DataType | PK | Not  Null | Unique | FK | Default  Value | Validation |
| actorid | Int | ✓ |  |  |  |  |  |
| Act\_firstName | varchar(20) |  | ✓ |  |  |  |  |
| Act\_lastName | Varchar(30) |  | ✓ |  |  |  |  |

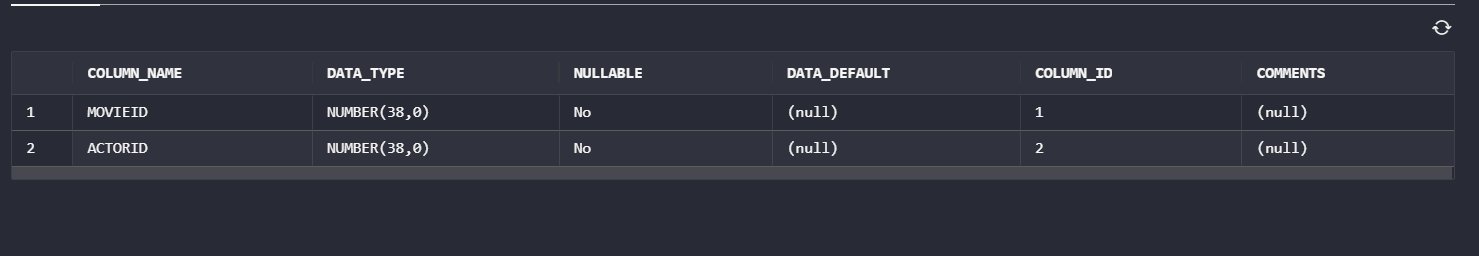


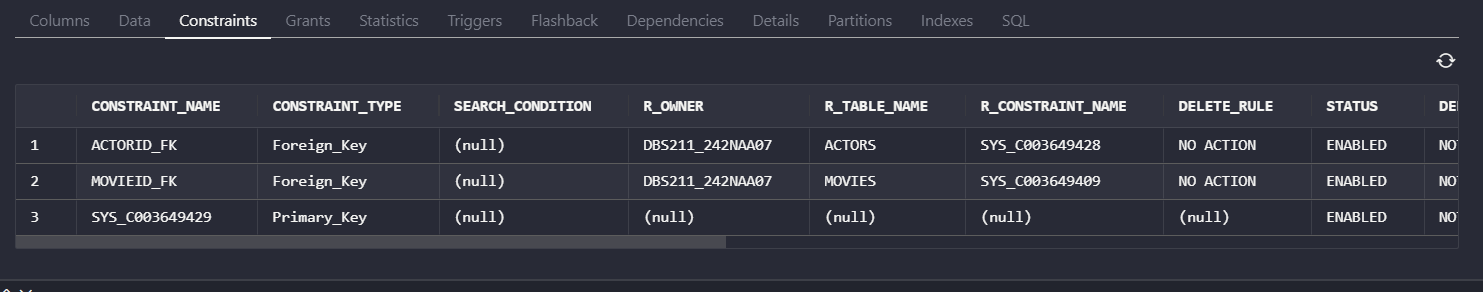


1. Create the following table and their given constraints. Show the table query and the screenshot of columns tab and constraints tab after executing the table query.

**CASTINGS** (movieid:int, actorid:int)

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

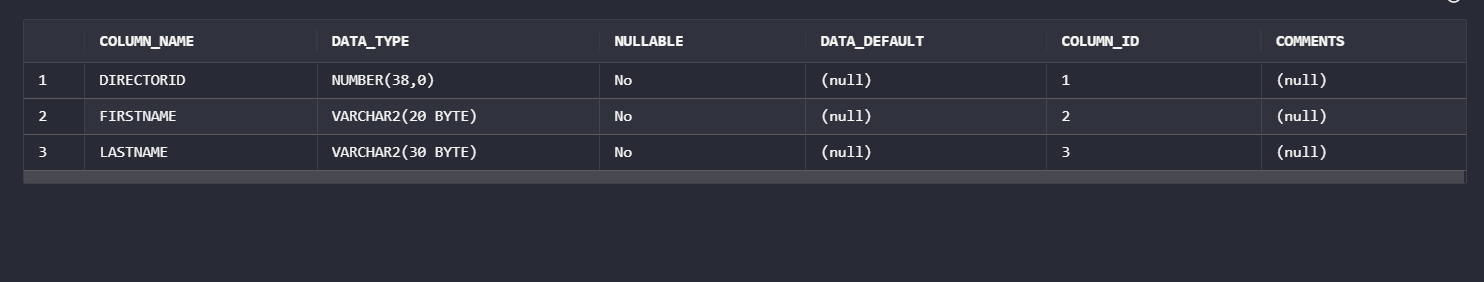


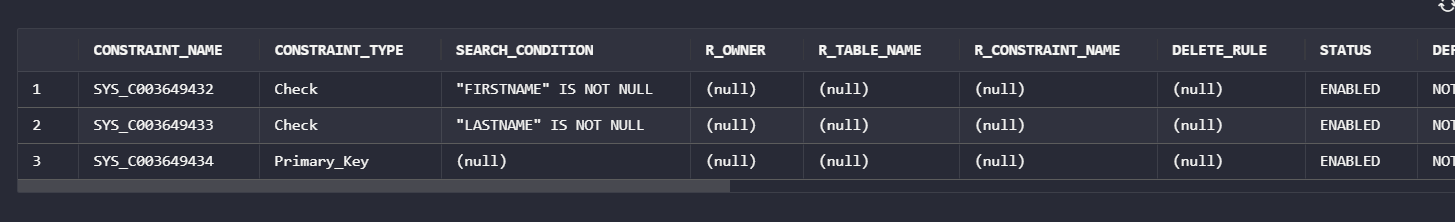


1. Create the following table and their given constraints. Show the table query and the screenshot of columns tab and constraints tab after executing the table query.

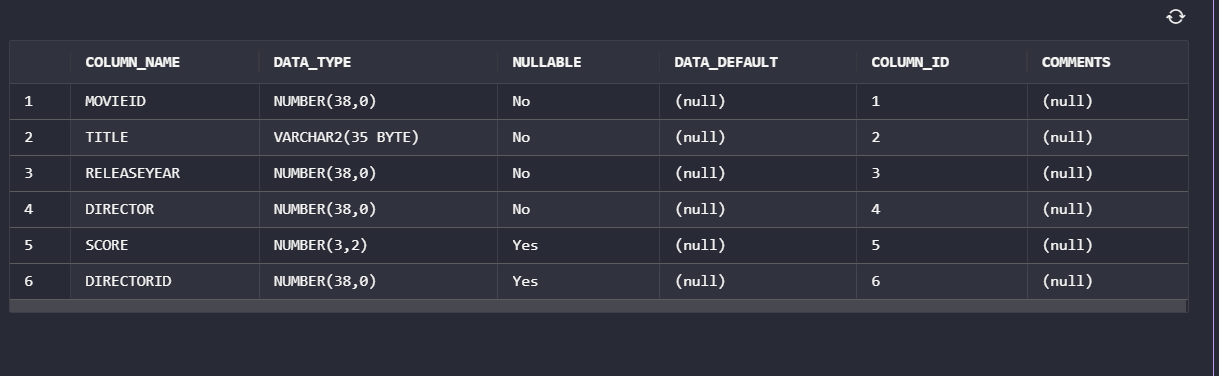
**DIRECTORS**(directorid:int, firstname:varchar(20), lastname:varchar(30))

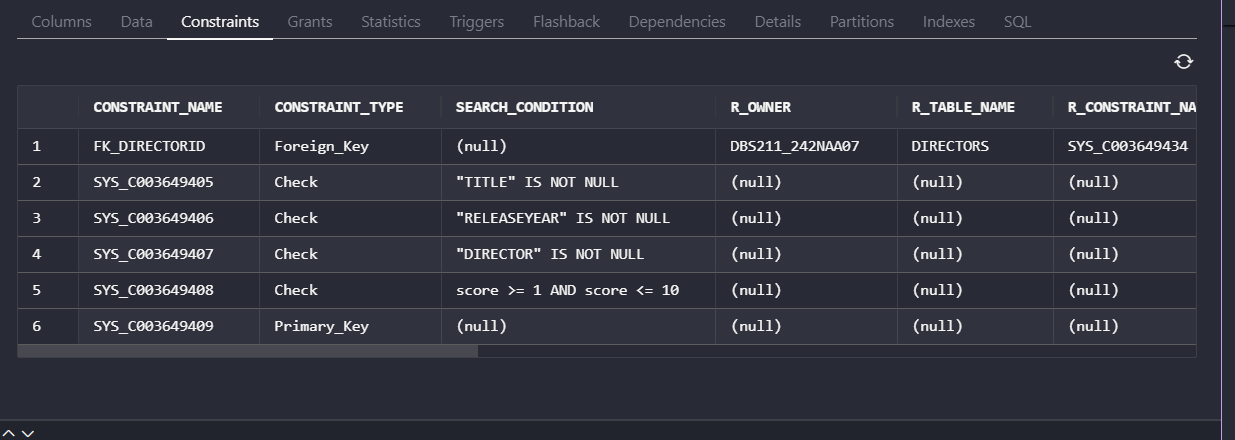
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Column  Name | Column  DataType | PK | Not  Null | Unique | FK | Default  Value | Validation |
| directorid | Int | ✓ |  |  |  |  |  |
| Dir\_firstname | varchar(20) |  | ✓ |  |  |  |  |
| Dir\_lastname | varchar(30) |  | ✓ |  |  |  |  |



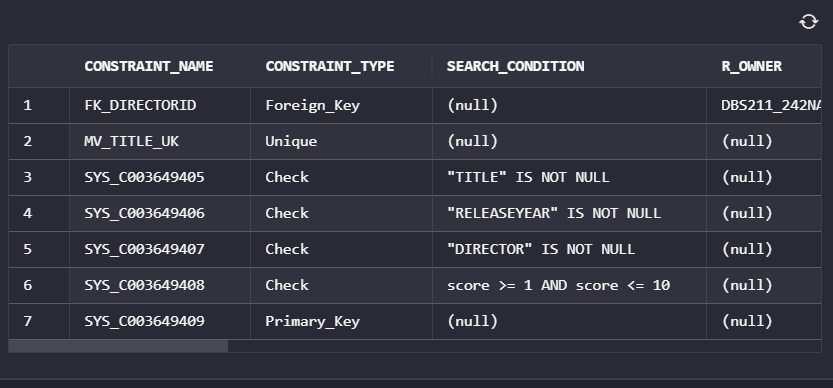


1. Modify the ***movies*** table to create a foreign key constraint that refers to table ***directors***. Show the modify table query and the screenshot of constraints tab after executing the modify table query.





1. Modify the ***movies*** table to create a new constraint so the uniqueness of the movie title is guaranteed. Show the modify table query and the screenshot of constraints tab after executing the modify table query.



1. Write insert statements to add the following data to table ***directors***. Show the INSERT query and after execution, show the results of all the table data by a SELECT query.

**Directors**

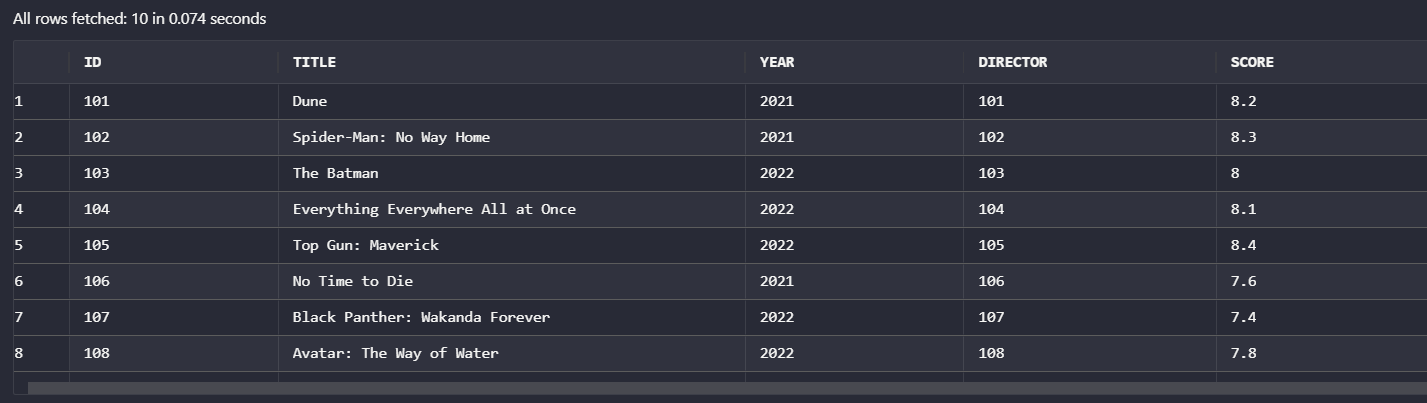
|  |  |  |
| --- | --- | --- |
| **directorid** | **Dir\_Firstname** | **Dir\_Lastname** |
| 101 | Denis | Villeneuve |
| 102 | Jon | Watts |
| 103 | Matt | Reeves |
| 104 | Daniel Scheinert | Daniel Kwan |
| 105 | Joseph | Kosinski |
| 106 | Cary Joji | Fukunaga |
| 107 | Ryan | Coogler |
| 108 | James | Cameron |
| 109 | Adam | McKay |
| 110 | Wes | Anderson |



1. Write insert statements to add the following data to table **Movies**. Show the INSERT query and after execution, show the results of all the table data by a SELECT query.

**Movies**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Id** | **Title** | **Year** | **Director** | **Score** |
| 101 | Dune | 2021 | 101 | 8.2 |
| 102 | Spider-Man: No Way Home | 2021 | 102 | 8.3 |
| 103 | The Batman | 2022 | 103 | 8 |
| 104 | Everything Everywhere All at Once | 2022 | 104 | 8.1 |
| 105 | Top Gun: Maverick | 2022 | 105 | 8.4 |
| 106 | No Time to Die | 2021 | 106 | 7.6 |
| 107 | Black Panther: Wakanda Forever | 2022 | 107 | 7.4 |
| 108 | Avatar: The Way of Water | 2022 | 108 | 7.8 |
| 109 | Don't Look Up | 2021 | 109 | 7.2 |
| 110 | The French Dispatch | 2021 | 110 | 7.5 |

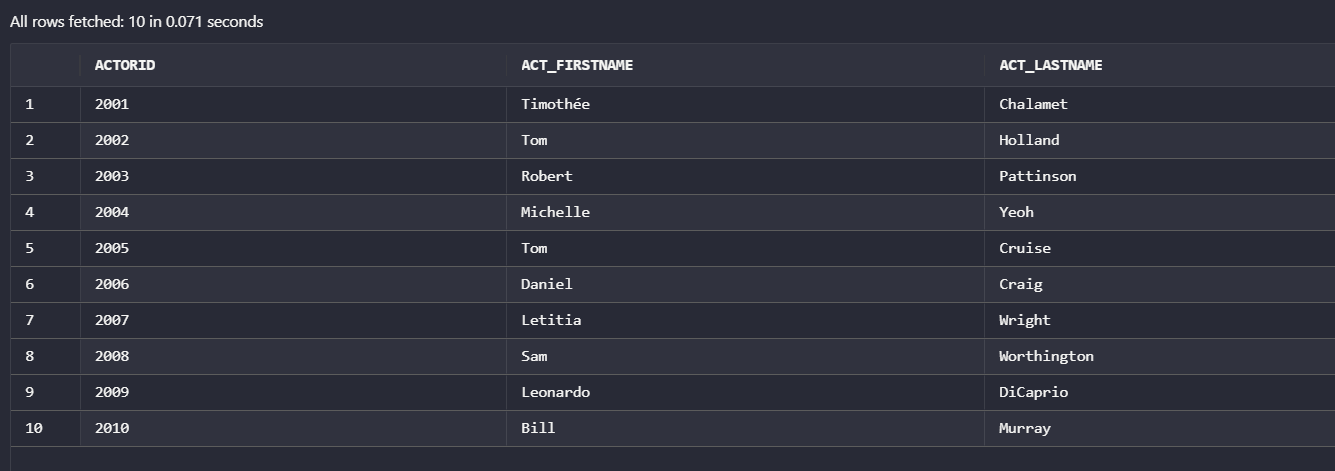


1. Write insert statements to add the following data to table ***actors.*** Show the INSERT query and after execution, show the results of all the table data by a SELECT query.

**Actors**

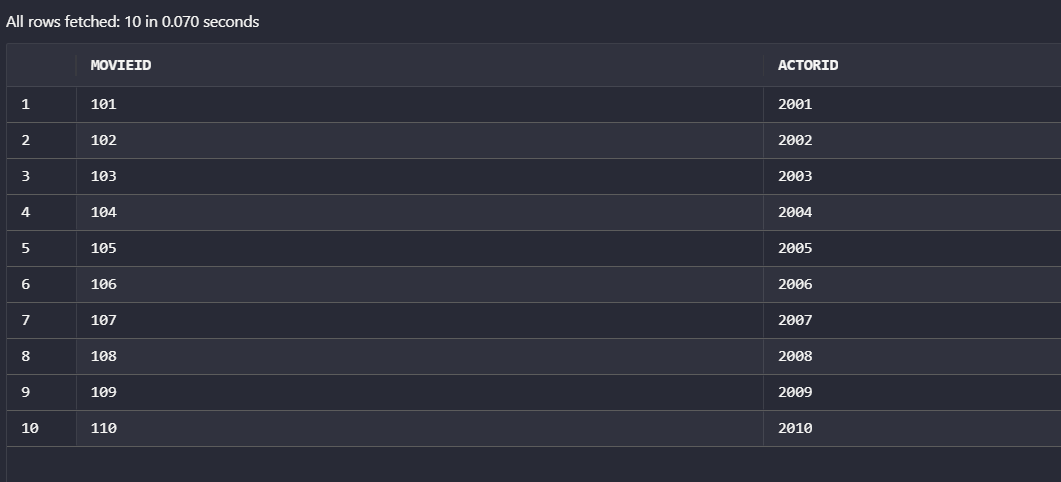
|  |  |  |
| --- | --- | --- |
| **actorid** | **Act\_firstName** | **Act\_lastName** |
| 2001 | Timothée | Chalamet |
| 2002 | Tom | Holland |
| 2003 | Robert | Pattinson |
| 2004 | Michelle | Yeoh |
| 2005 | Tom | Cruise |
| 2006 | Daniel | Craig |
| 2007 | Letitia | Wright |
| 2008 | Sam | Worthington |
| 2009 | Leonardo | DiCaprio |
| 2010 | Bill | Murray |

1. Write insert statements to add the following data to table **castings.** Show the INSERT query and after execution, show the results of all the table data by a SELECT query.

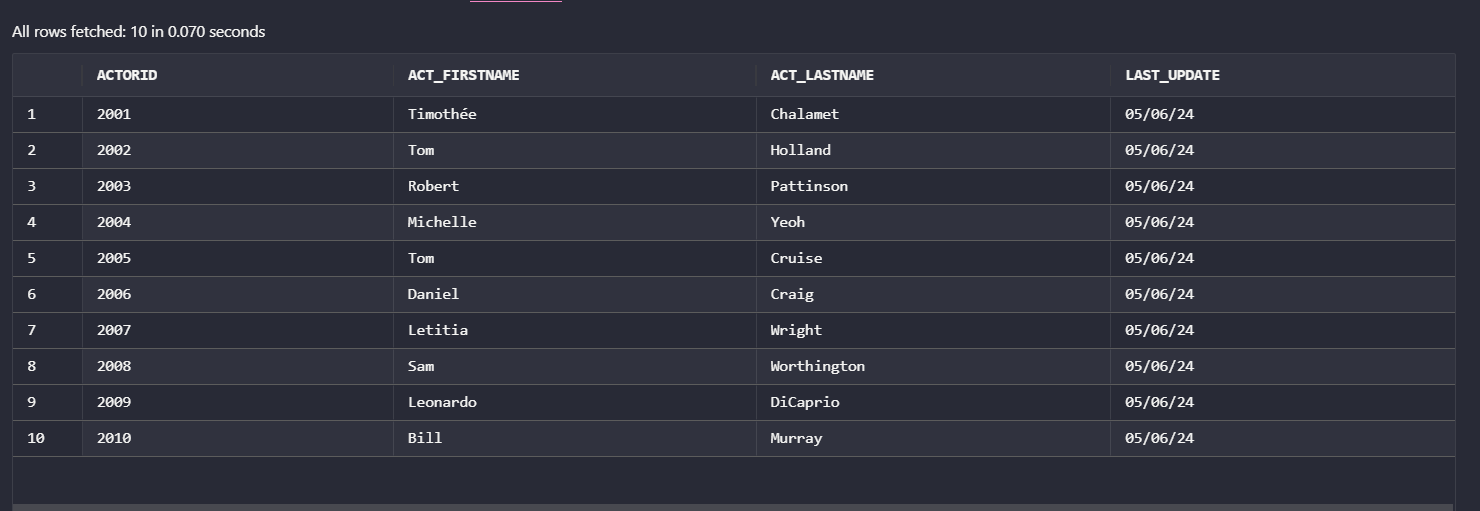


**Castings**

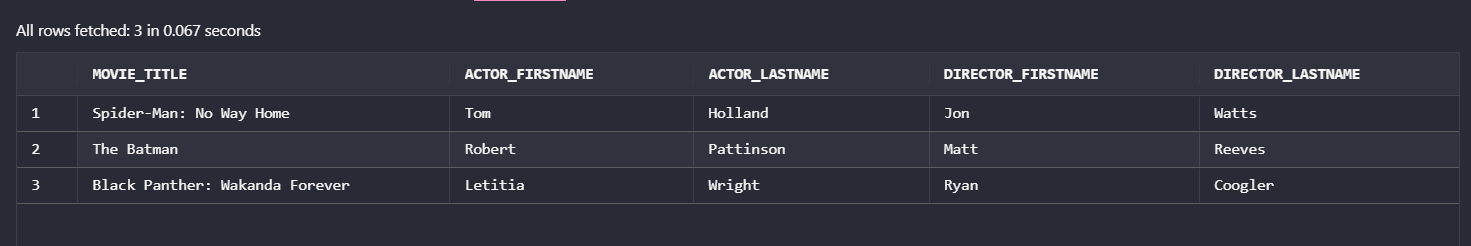
|  |  |
| --- | --- |
| **movieid** | **actorid** |
| 101 | 2001 |
| 102 | 2002 |
| 103 | 2003 |
| 104 | 2004 |
| 105 | 2005 |
| 106 | 2006 |
| 107 | 2007 |
| 108 | 2008 |
| 109 | 2009 |
| 110 | 2010 |



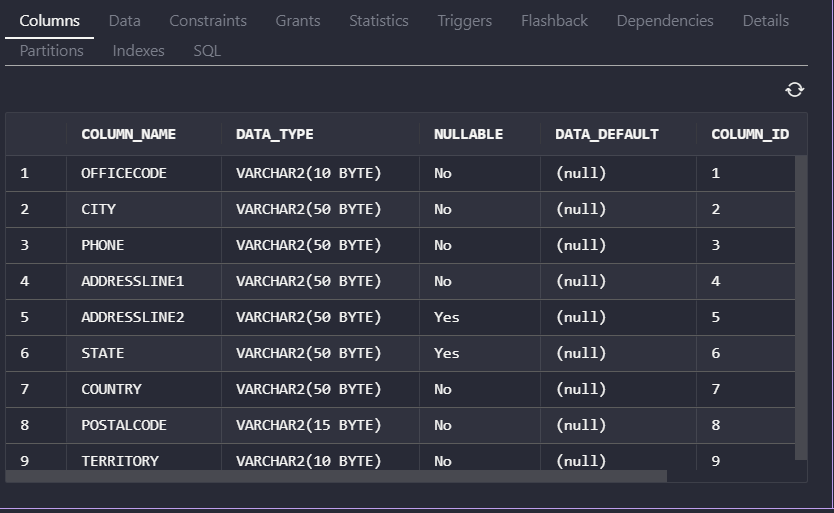
1. Add a new column last\_update date to Actors table. Show the query and column tab screenshot screenshot Add today’s date as last\_update date for all the actors. Show the results of update.

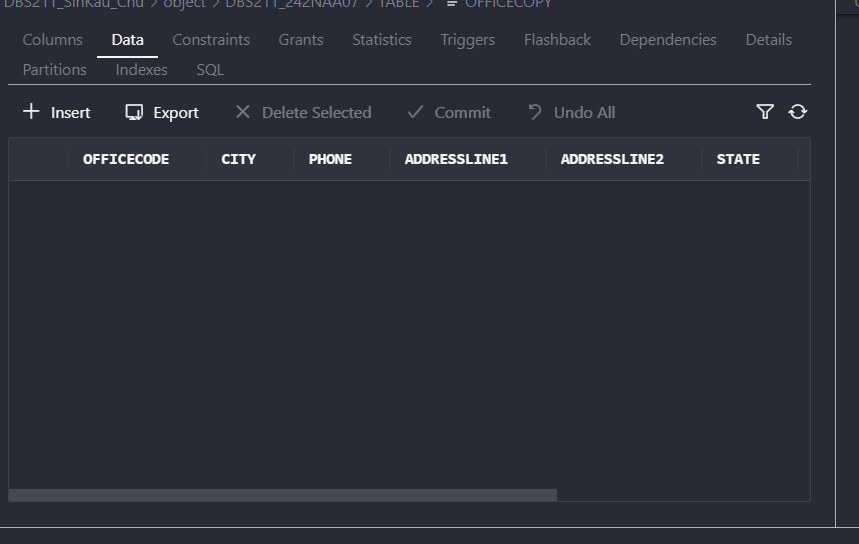


1. Using the syntax SELECT and INSERT combine. Create a new table called Mov\_Dir\_Actor with the columns movie Title, Actor’s firstname and lastname, Directors firstname and lastname for the movieid 2,3 and 7 and actor id 2002,2003 and 2007 and director id 102,103,107. Show the query and the screenshot of results.

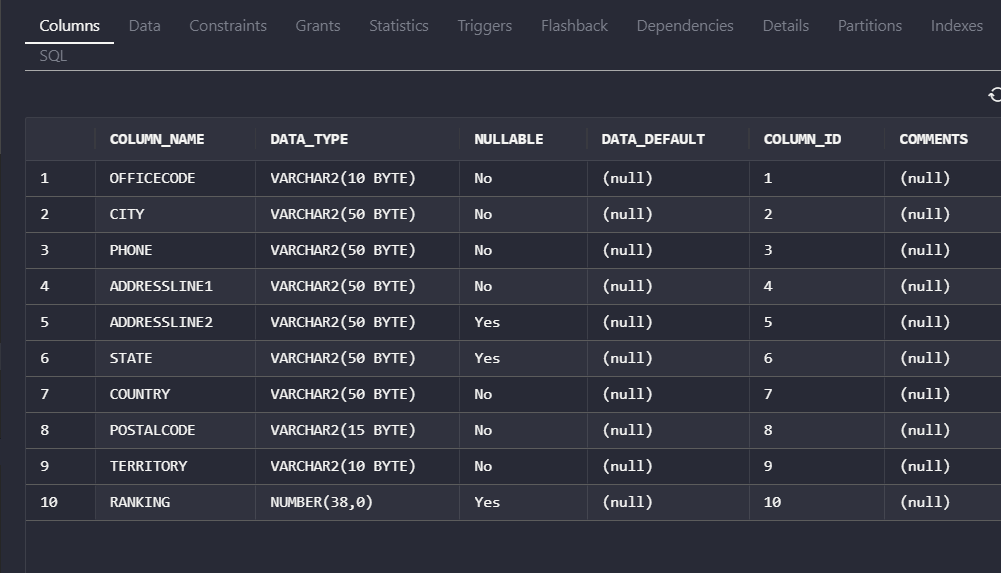


1. Create a new empty table (that means the table will not have any data after creating) ***officecopy*** the same as table ***retailoffices.***  Use a single statement to create the table and insert the data at the same time (Hint use a WHERE clause that is false like 1=2). Don’t create the table from scratch.

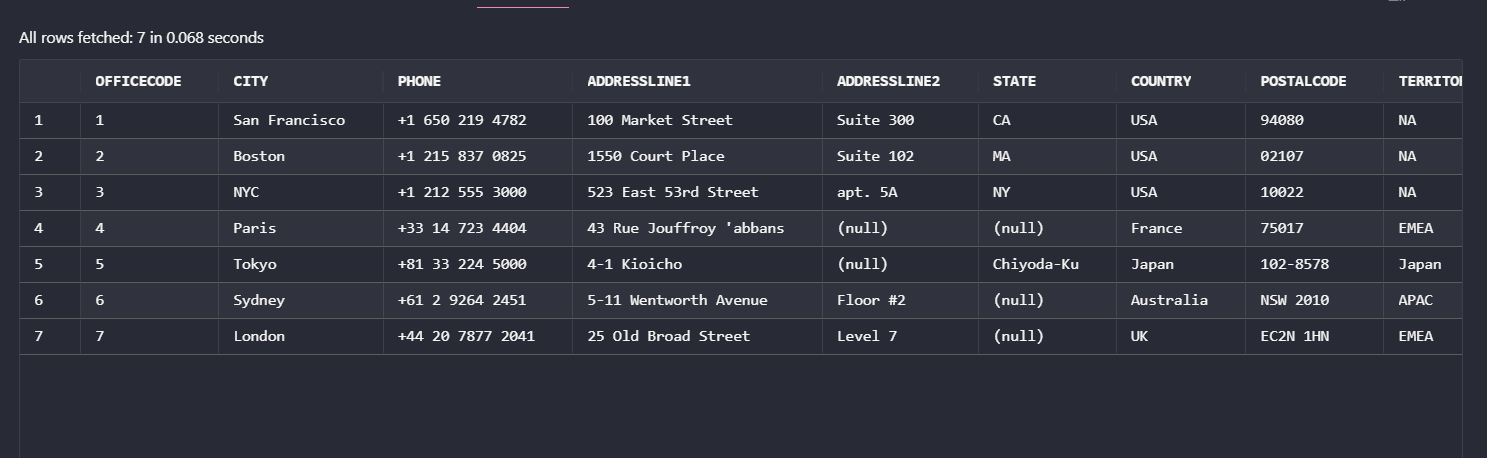
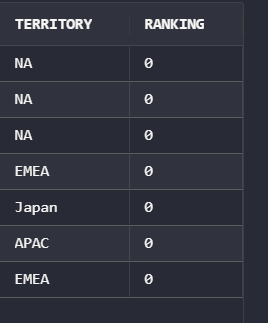




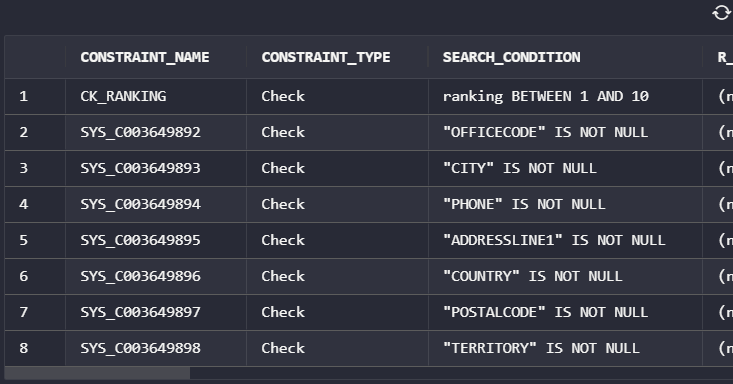
1. Modify table ***officecopy*** and add a new column ***ranking*** to this table. The value of this column is not required and does not have to be unique. Show the SQL query and the column tab screenshot



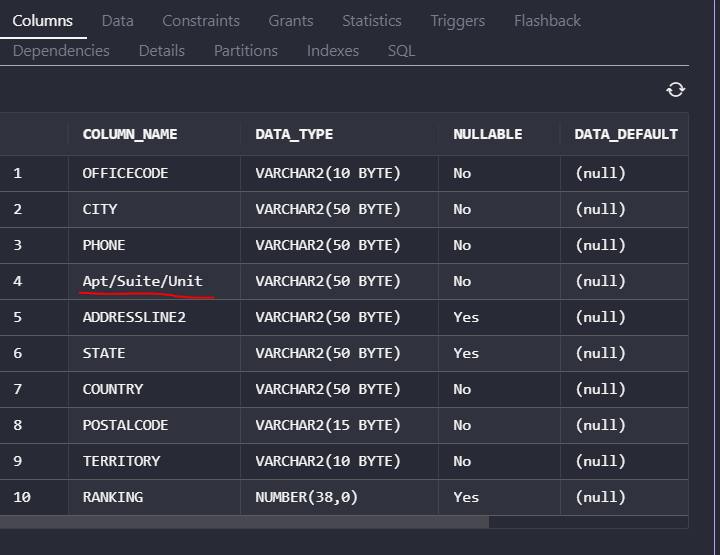
1. Re-insert all data from the ***retailoffices.***  table into your new table ***officecopy*** using a single statement. Hint using INSERT-SELECT statement.

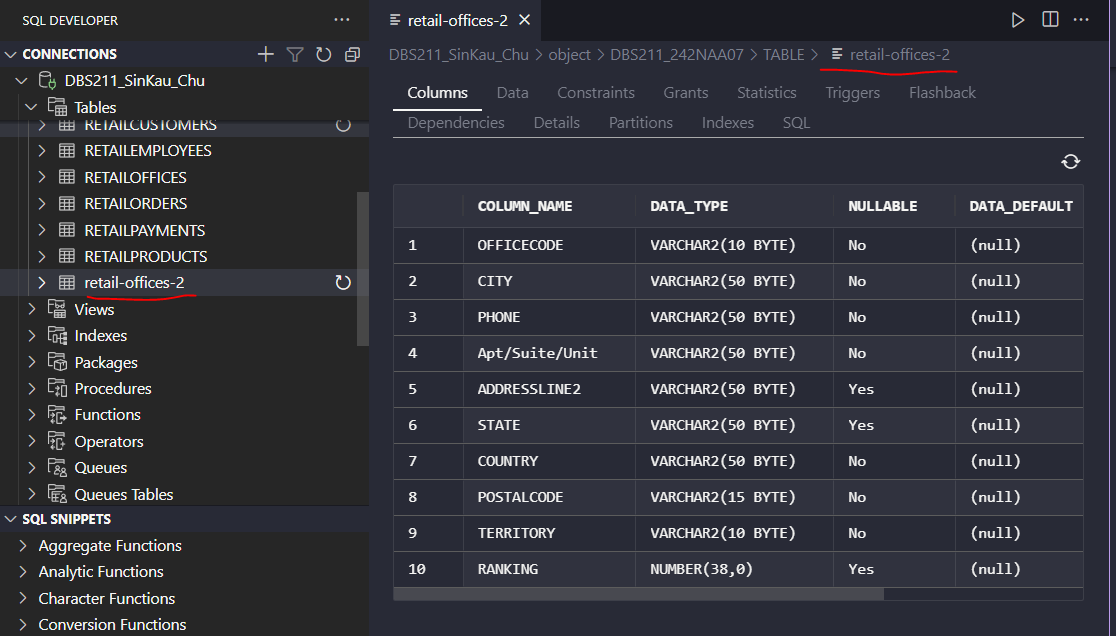
1. Add a validation constraint for ranking column in officecopy table as 1 to 10. Write the query and show the constraints tab after execution of the query.



1. Rename the Addressline2 in officecopy table as Apt/Suite/Unit. Show the query and columns tab.

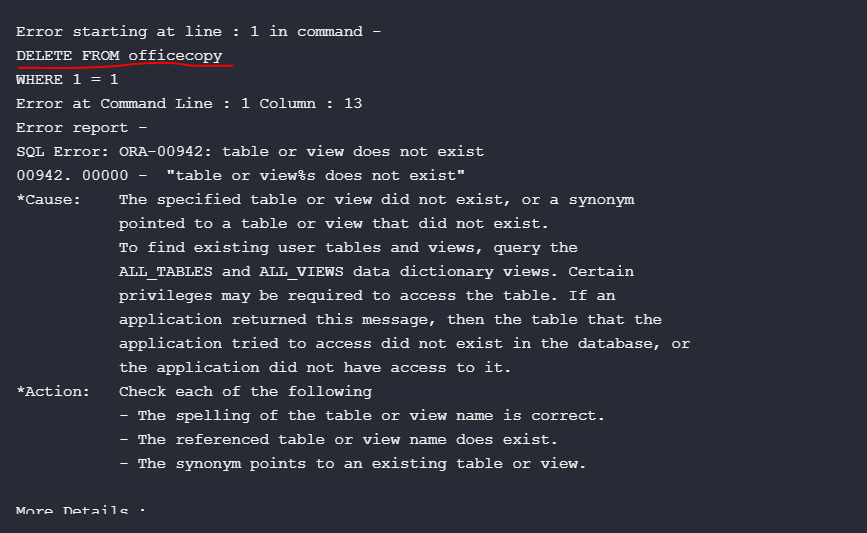


1. Rename the table officecopy as retail-offices-2. Show the query and columns tab.



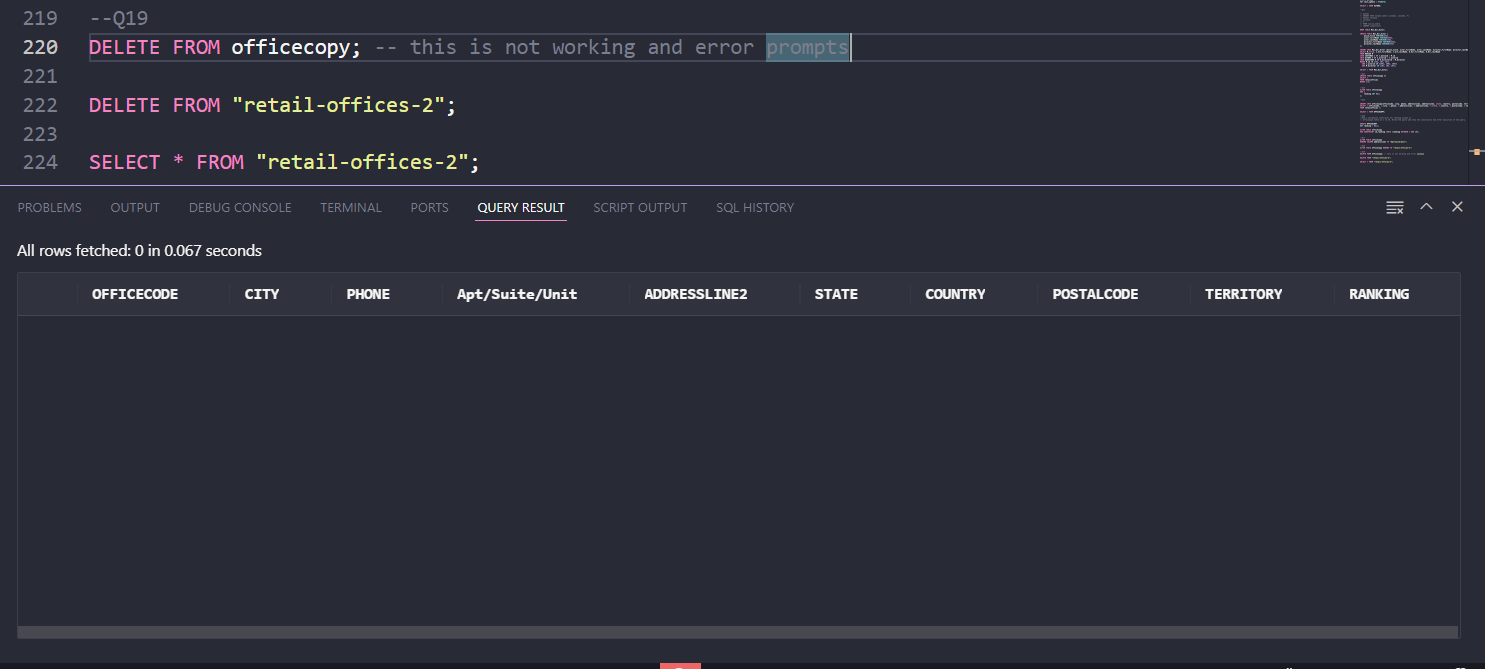
1. Delete all the officecopy data and display the data in the table. Does officecopy exist? If not how can you delete table ***officecopy***.

The officecopy is not existing and could not delete the row. It prompts an error.



Instead we have to use the sql statement

DELETE FROM "retail-offices-2";



1. Drop all the tables created from questions 1 to 5. Write the query and show the screenshot of tables menu on the left menu to show if the tables still exist.

