**Homework 6**

**Due: 10/31/2024 by 11:59 PM**

In this assignment, you will create a database called *“movies”*, and then will create 4 tables under it. The names of the tables are as follows; “Actors”, “Directors”, “Movies” and “Movies Revenues”. After creating the database and the tables, you will create several variables under these tables. In the final stage, you will insert data into these tables. Below is the screenshot of the *“movies”* database and the 4 tables under it.

**A screenshot of a computer

Description automatically generated A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

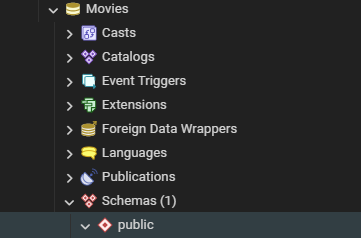
Description automatically generated A screenshot of a computer

Description automatically generated**

1. So, your first step is to go ahead and create this database called “movies”. As you can see, a sample screenshot from the output is given below. After creating the database, please attach your own output screenshot (just like I did below).

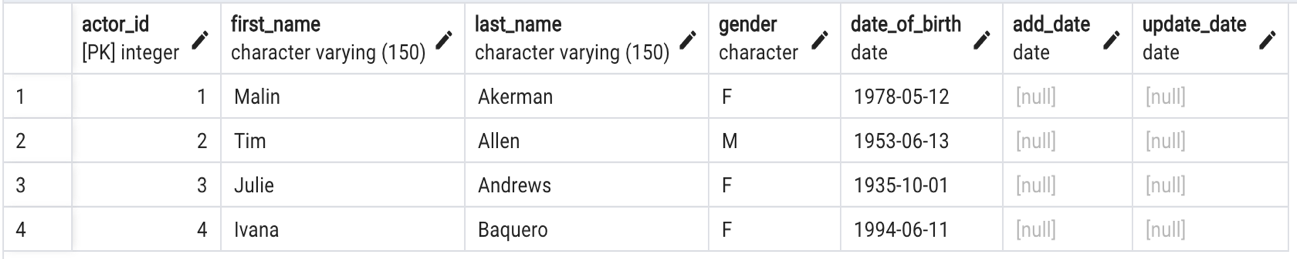
A picture containing graphical user interface

Description automatically generated



1. Write a SQL query to create the “**actors**” table as well as the related 7 variables. The features of these variables (that should be considered while creating these variables) are provided below. As can be seen, a sample screenshot is given (below) from the “actors” table. (Note that, I have provided couple of rows for you to see what this table will eventually look like. However, you should not insert any data values at this step. You should only create the table with the following variables and that is it.

* **actor\_id** has the “serial” type. Note that this column is the primary key of the table.
* **first\_ name** and the **last\_name** are strings that have the “limited varying character (150 characters)” type.
* Note that the **last\_name** cannot take “null” values (constraint)
* **gender** is string that have the “fixed length character (1character)” type.
* **Date\_of\_birth, add\_date** and the **update\_date** are date columns that have the “date” type.



CREATE TABLE actors (

actor\_id serial primary key,

first\_name varchar(150),

last\_name varchar(150) not null,

gender char(1),

date\_of\_birth date,

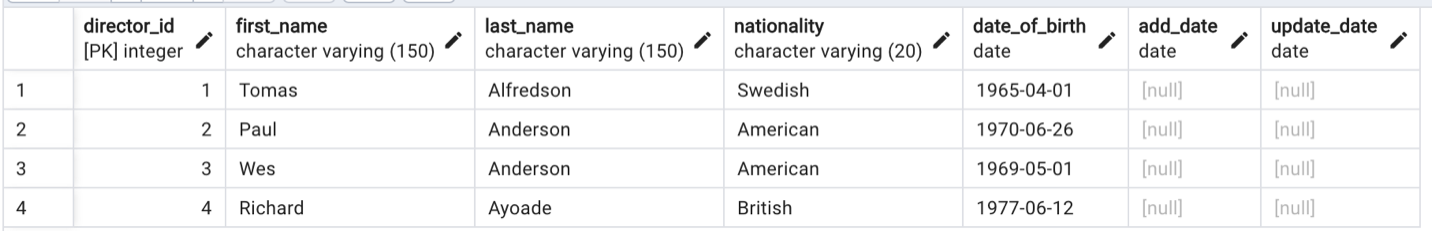
add\_date date,

update\_date date

);

1. Write a SQL query to create the “**directors**” table as well as the related 7 variables. The features of these variables (that should be considered while creating these variables) are provided below. As can be seen, a sample screenshot is given (below) from the “directors” table. (Note that, I have provided couple of rows for you to see what this table will eventually look like. However, you should not insert any data values at this step.

* **director\_id** has the “serial” type. Note that this column is the primary key of the table.
* **first\_name** and the **last\_name** are strings that have the “limited varying character (150 characters)” type.
* **nationality** is string that has the “limited varying character (20 characters)” type.
* **Date\_of\_birth, add\_date** and the **update\_date** are date columns that have the “date” type.



CREATE TABLE directors (

director\_id serial primary key,

first\_name varchar(150),

last\_name varchar(150),

nationality varchar(20),

date\_of\_birth date,

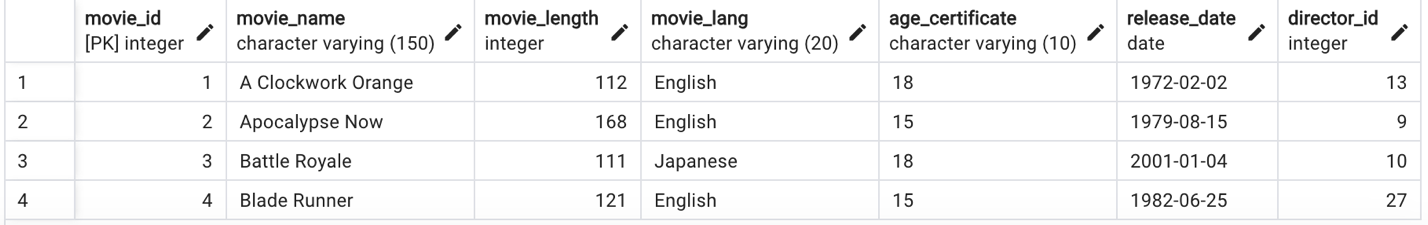
add\_date date,

update\_date date

);

1. Write a SQL query to create the “**movies**” table as well as the related 7 variables. The features of these variables (that should be considered while creating these variables) are provided below. As can be seen, a sample screenshot is given (below) from the “movies” table. (Note that, I have provided couple of rows for you to see what this table will eventually look like. However, you should not insert any data values at this step.

* **movie\_id** has the “serial” type. Note that this column is the primary key of the table.
* **movie\_name** is string that has the “limited varying character (150 characters)” type.
* Note that the **movie\_name** cannot take “null” values (constraint)
* **Movie\_length** has the “integer” type.
* **movie\_lang** is string that has the “limited varying character (20 characters)” type.
* **age\_certificate** is string that has the “limited varying character (10 characters)” type.
* **release\_date** is a date column that have the “date” type.
* **director\_id** has the “integer” type.
* **director\_id** is a foreign key that should be referenced to the director\_id from the directors table.



CREATE TABLE movies (

movie\_id serial primary key,

movie\_name varchar(150) not null,

movie\_length int,

movie\_lang varchar(20),

age\_certificate varchar(10),

release\_date date,

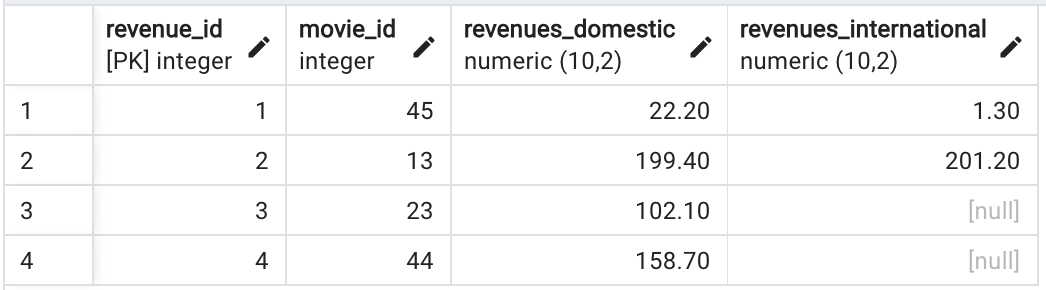
director\_id int,

foreign key(director\_id) references directors(director\_id)

);

1. Write a SQL query to create a table called “**movies\_revenues**” table as well as the related 4 variables. The features of these variables (that should be considered while creating these variables) are provided below. As can be seen, a sample screenshot is given (below) from the “**movies\_revenues**” table. (Note that, I have provided couple of rows for you to see what this table will eventually look like. However, you should not insert any data values at this step.

* **revenue\_id** has the “serial” type. Note that this column is the primary key of the table.
* **movie\_id** has the “integer” type
* **movie\_id** is a foreign key that references to the movie\_id from the movies table.
* **Revenues\_domestic** and the **revenues\_international** are numeric columns that have the “numeric” type that have precision of “10”, and the scale of “2”



CREATE TABLE movies\_revenue (

revenue\_id serial primary key,

movie\_id int,

revenue\_domestic numeric(10,2),

revenue\_international numeric(10,2),

foreign key(movie\_id) references movies (movie\_id)

);

**Now, we will go ahead and insert data into our tables that we have created in the previous steps.**

**At this point, please go ahead and download the sql files/scripts from the “Assignment 6 Data.zip“ folder under module *Week 8*. You will find four(4) sql files such as “movies.sql”, “directors.sql”, “actors.sql”, and “movies\_revenues.sql” .**

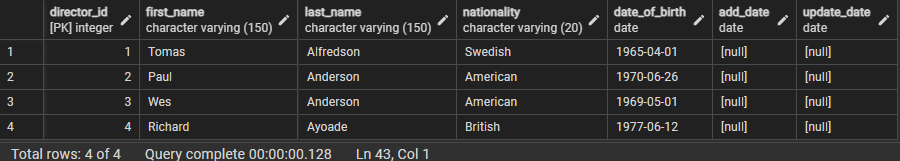
**Make sure you follow the sequence of inserting data in the order below as we are adding data to tables with foreign\_key constraints.**

1. Go ahead and open the “directors.sql” file in your PgAdmin environment and insert the values given in that file into “Directors” table. Then go ahead and make sure that your data is inserted by running;

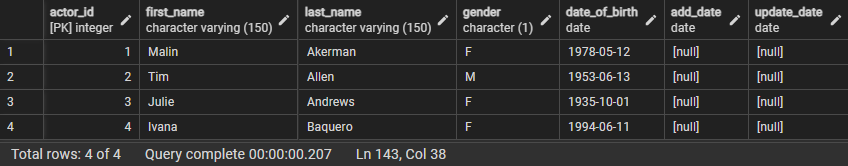
***SELECT \* FROM directors***

***LIMIT 4;***

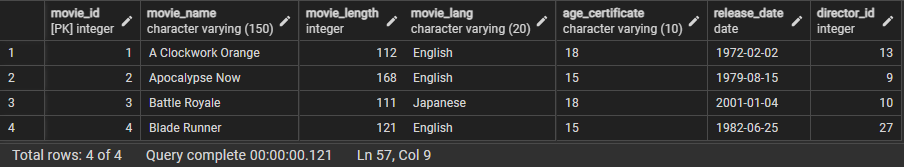
and provide the screenshot of your output below.



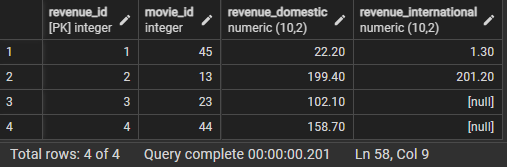
1. Do the same thing for “actors” table.



1. Do the same thing for “movies” table.



1. Do the same thing for “movies\_revenues” table.



1. By using inner join, retrieve movie\_id, movie\_name, movie\_lang, release\_date, revenues\_domestic and revenues\_international. Save the output of this question as “movies\_total” table.

***Use movies\_total table for the questions 11, 12, 13 and 14.***

CREATE TABLE movies\_total AS

SELECT movie\_id, movie\_name, movie\_lang,release\_date, revenue\_domestic, revenue\_international

from movies

inner join movies\_revenue

using(movie\_id);

1. Find the total domestic revenue amount for movies with “English” language.

select sum(revenue\_domestic) as total\_revenue\_domestic

from movies\_total

where movie\_lang = 'English';

1. Replace the NULL values in the revenues\_domestic column with **30**.

update movies\_total

set revenue\_domestic = 30

where revenue\_domestic is null;

1. Add a NOT NULL constraint for revenues\_domestic column.

alter table movies\_total

alter column revenue\_domestic set not null;

1. Add a DEFAULT value of **50 for** revenues\_international column and show that default value is working by inserting a record to the movies\_total table.

alter table movies\_total

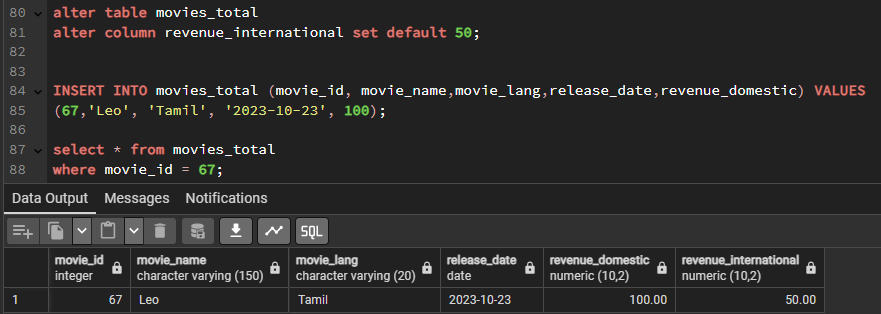
alter column revenue\_international set default 50;

INSERT INTO movies\_total (movie\_id, movie\_name,movie\_lang,release\_date,revenue\_domestic) VALUES

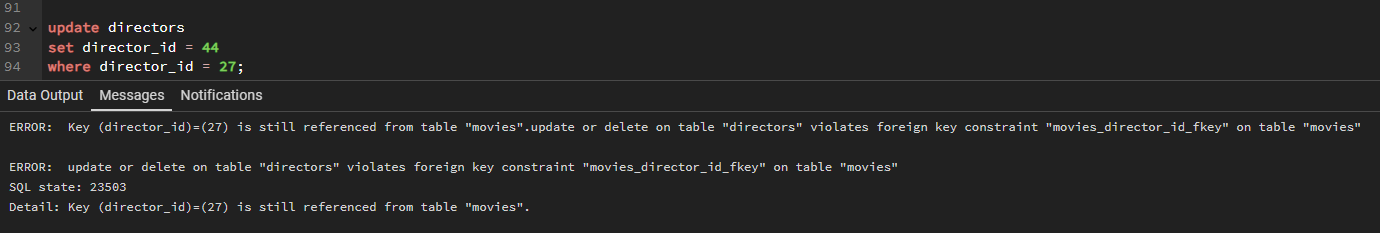
(67,'Leo', 'Tamil', '2023-10-23', 100);

select \* from movies\_total

where movie\_id = 67;



1. Attempt to update the **director\_id** from 27 to 44 in the **directors** table. Describe the type of output you receive upon trying to update the **director\_id** in the **directors** table. What should have been done to avoid the error encountered? (Note: You don’t have to recreate the **directors** table. Just explain the solution that allows us to update the **director\_id** in the **directors** table.)



We got this error because director\_id in directors table is referenced as a foreign key in movies table.

This update violates foreign key constraint on movies table.

To solve this issue we have to define the foreign key with ‘On update cascade’ constraint. This constraint will automatically update the values in the foreign key of the child table when the primary key of the parent table is updated.