# PostgreSQL Retake Exam - 7 August 2024

# StageMaster

## *Step into the spotlight with StageMaster - the definitive solution for managing all aspects of entertainment productions. As the director of your creative team, you'll harness the power of data to orchestrate seamless performances, coordinate talented actors, and categorize productions with precision. With StageMaster, the stage is set for unparalleled excellence in entertainment management.*

## Section 0: Database Overview

You are given an Entity/Relationship Diagram of the **StageMaster** System. This diagram illustrates the connections among different entities within **StageMaster**, offering a visual depiction of the database structure:

A screenshot of a computer

Description automatically generated

The **StageMaster** must contain information about **productions**, **countries**, **actors**, **categories**, and **productions additional info**.

Your task is to set up a database named stage\_master with the following **tables**:

* countries - contains information about the countries.
* categories - contains information about the categories.
* actors - contains information about the actors.
* productions – contains short information about the productions.
* productions\_info - contains detailed information about the productions.
* productions\_actors - a many-to-many mapping table between the productions and the actors.
* categories\_productions - a many-to-many mapping table between the categories and the productions**.**

## Section 1: Data Definition Language (DDL) - 30 Pts

Make sure you implement the **database** **tables** correctly.

**Important Note:** When working with dates, please adhere strictly to the specified data types in the model tables. For example, if a column is defined as type '**DATE**,' ensure you utilize the '**DATE**' data type. Similarly, if a column is designated as '**TIMESTAMP**,' use the '**TIMESTAMP**' data type. Failure to use the correct data type may result in your submission being rejected by the Judge system.

### Table Design

You have been tasked to create the tables in the database by following the specified models.

Submit only your **CREATE** statements for all tables to the Judge System.

#### countries

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| id | **Integer**,from **0** to **2,147,483,647** | **Primary Key**, Unique table identification, Auto-incremented |
| name | A **string** containing a maximum of **40 characters** | **NULL** is **NOT** permitted,  **UNIQUE** values |
| continent | A **string** containing a maximum of **40 characters** | **NULL** is **NOT** permitted |
| currency | A **string** containing a maximum of **5 characters** | **NULL** ispermitted |

#### categories

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| id | **Integer**,from **0** to **2,147,483,647** | **Primary Key**, Unique table identification, Auto-incremented |
| name | A **string** containing a maximum of **50 characters** | **NULL** is **NOT** permitted, **UNIQUE** values |

#### actors

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| id | **Integer**,from **0** to **2,147,483,647** | **Primary Key**, Unique table identification, Auto-incremented |
| first\_name | A **string** containing a maximum of **50 characters** | **NULL** is **NOT** permitted |
| last\_name | A **string** containing a maximum of **50 characters** | **NULL** is **NOT** permitted |
| birthdate | The birthdate (type **DATE**) of the actor | **NULL** is **NOT** permitted |
| height | **Integer**,from **0** to **2,147,483,647** (the actor's height in centimeters) | **NULL** is permitted |
| awards | **Integer**,from **0** to **2,147,483,647** | **NULL** is **NOT** permitted, **DEFAULT** **0**,The value must be **greater than** or **equal to 0** |
| country\_id | **Integer**,from **0** to **2,147,483,647** | Relationship with table countries. Cascade operations. **NULL** is **NOT** permitted |

#### productions\_info

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| id | **Integer**,from **0** to **2,147,483,647** | **Primary Key**, Unique table identification, Auto-incremented |
| rating | **DECIMAL**, up to **4 digits**, **2** of which are after the **decimal point** | **NULL** is **NOT** permitted |
| duration | **Integer**,from **1** to **2,147,483,647** (the duration in minutes) | **NULL** is **NOT** permitted. The value must be **greater than 0** |
| budget | **DECIMAL**, up to **10 digits**, **2** of which are after the **decimal point** | **NULL** is permitted |
| release\_date | The release **date** (type **DATE**) of the production | **NULL** is **NOT** permitted |
| has\_subtitles | Can be **TRUE** or **FALSE** | **NULL** is **NOT** permitted, **DEFAULT** **FALSE** |
| synopsis | An unlimited **text** | **NULL** is permitted |

#### productions

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| id | **Integer**,from **0** to **2,147,483,647** | **Primary Key**, Unique table identification, Auto-incremented |
| title | A **string** containing a maximum of **70 characters** | **NULL** is **NOT** permitted, **UNIQUE** values |
| country\_id | **Integer**,from **0** to **2,147,483,647** | Relationship with table countries. Cascade operations. **NULL** is **NOT** permitted |
| production\_info\_id | **Integer**,from **0** to **2,147,483,647** | Relationship with table productions \_info. Cascade operations. **NULL** is **NOT** permitted, **UNIQUE** values |

#### productions\_actors

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| production\_id | **Integer**,from **0** to **2,147,483,647** | Relationship with table productions. Cascade operations. **NULL** is **NOT** permitted |
| actor\_id | **Integer**,from **0** to **2,147,483,647** | Relationship with table actors. Cascade operations. **NULL** is **NOT** permitted |
| - | **-** | **Composite Primary Key** on both columns (**production\_id** and **actor\_id**) |

#### categories\_productions

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| category\_id | **Integer**,from **0** to **2,147,483,647** | Relationship with table categories. Cascade operations. **NULL** is **NOT** permitted |
| production\_id | **Integer**,from **0** to **2,147,483,647** | Relationship with table productions. Cascade operations. **NULL** is **NOT** permitted |
| - | **-** | **Composite Primary Key** on both columns (**category\_id** and **production\_id**) |

## Section 2: Data Manipulation Language (DML) - 10 Pts

Before starting, ensure to import the **'dataset.sql'** file. Successful insertion of data is contingent upon the proper creation of the database structure.

This section entails executing various data manipulation tasks:

### Insert

You must **insert** records with specific data into the **actors'** table.

The new data will be based on **actors** with **IDs** between **10 and 20** (**inclusive**).

**Insert data** into the **actors'** table with the **following values**:

• **first\_name** - set it to the **first name** of the actor but **reversed**.

• **last\_name** - set it to the **last name** of the actor but **reversed**.

• **birthdate** - set it to the **birthdate** of the **actor** but **2 days earlier**.

• **height** - set it to the **height** of the **actor** plus **10**. Note that there should **not be NULL** height values for the new records. See the provided examples.

• **awards** - set them to the **country\_id**.

• **country\_id** - set it to the **id** of **Armenia** (Do **not** hard-code the id).

#### Example

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **id** | **first\_name** | **last\_name** | **birthdate** | **height** | **awards** | **country\_id** |
| … | … | … | … | … | … | … |
| 13 | Fayth | Mather | 2001-05-01 | 174 | 10 | 19 |
| 14 | Stephan | Lundberg | 1993-01-25 | 176 | 0 | 10 |
| 15 | King | Beadel | 1985-08-24 | [null] | 0 | 9 |
| … | … | … | … | … | … | … |
| 304 | htyaF | rehtaM | 2001-04-29 | 184 | 19 | 3 |
| 305 | nahpetS | grebdnuL | 1993-01-23 | 186 | 10 | 3 |
| 306 | gniK | ledaeB | 1985-08-22 | 10 | 9 | 3 |
| … | … | … | … | … | … | … |

### Update

**Increase** the **production's duration** for **productions** with **no synopsis and** the following **ids** (**productions\_info** table):

* If the **id** is **less than 15** - **increase** the **productions' duration** by **15 minutes**.
* If the **id** is **equal to or greater than 20** - **increase** the **productions' duration** by **20 minutes**.
* Otherwise, **do not change** it.

#### Example

|  |  |  |
| --- | --- | --- |
| **id** | **duration** | **synopsis** |
| … | … | … |
| 6 | 110 | A thrilling mystery that keeps you guessing until the very end. |
| 7 | 145 | [null] |
| 8 | 140 | A musical journey that celebrates the power of music and friendship. |
| … | … | … |
| 22 | 125 | A documentary exploring the cultural heritage of an ancient civilization. |
| 23 | 31 | [null] |
| … | … | … |

### Delete

As you may recall, during our initial work, data was inserted and updated. Now, there is a need to **remove** certain records from the database.

**Delete** all **countries** that don't have associated **actors** or **productions**.

#### Example

|  |  |
| --- | --- |
| **id** | **name** |
| … | … |
| 48 | United Kingdom |
| 49 | United States |
| 50 | Uruguay |

## Section 3: Querying - 40 Pts

**Important Note**: Now, we'll conduct some data extraction tasks. Please ensure that the database is cleared of any manipulations from the previous operations in the Data Manipulation Language (DML) section. **Insert** the provided **dataset** **again** to maintain consistency with the examples in this section.

### Countries

**Extract** from the database info about the **countries** in **South America** with **currencies** that **start** with the **letter 'P' or** the **letter 'U'**.

**Sort** the results by **currency** in **descending** order and then by **id ascending**.

#### Required Columns

* id
* name
* continent
* currency

#### Example

|  |  |  |  |
| --- | --- | --- | --- |
| **id** | **name** | **continent** | **currency** |
| 50 | Uruguay | South America | UYU |
| 15 | Ecuador | South America | USD |
| 35 | Paraguay | South America | PYG |
| 36 | Peru | South America | PEN |

### Productions by Release Year

**Extract** information about **productions** (**id**, **title**, **duration**, **budget**, and **release\_date**).

**Filter** **productions** that have been **released** in **2023** and **2024** year and have a **budget** **greater than** **1500000.00**.

Format the **release date** as **'MM-YY'** and **round** the **budget** value to the **first decimal place**.

**Order** the results by **budget** **ascending**, then by **duration** **descending**,and finally by **production id ascending**.

Display the **first 3** results.

#### Required Columns

* id
* title
* duration
* budget
* release\_date

#### Example

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **id** | **title** | **duration** | **budget** | **release\_date** |
| 22 | Tea For Two | 125 | 5500000.0 | 01-23 |
| 1 | Lumiere and Company | 120 | 5500000.0 | 01-23 |
| 10 | Sammy and Rosie | 125 | 6000000.0 | 03-23 |

### Casting

Some **actors** can apply for the **casting** of a **new production**. You must search for them and prepare their application data.

Write a query that returns the **full name**, **email**, and **awards** of all **actors** **not participating** in any productions.

Construct their **email** using the **first letter** of their **first name** (**lower case**),the **last two letters** of their **last name** followed by the **number** of **characters** of the **last name**, and the domain "**@sm-cast.com**".

Order by **awards** in **descending** order then by **id ascending**.

#### Required Columns

* full\_name (first\_name + " " + last\_name)
* email (a resulting string, according to the description above)
* awards

#### Example

|  |  |  |
| --- | --- | --- |
| **full\_name** | email | awards |
| Dorolisa Sabate | dte6@sm-cast.com | 30 |
| Noe Diemer | ner6@sm-cast.com | 30 |
| Ichabod Elman | ian5@sm-cast.com | 30 |
| Bride Steagall | bll8@sm-cast.com | 30 |
| Hallie Swaffield | hld9@sm-cast.com | 29 |
| Cora Feedham | cam7@sm-cast.com | 29 |
| … | … | … |

### Nominees

The international festival is coming soon. You need to help the jury nominate the **countries** suitable to host the event.

Extract from the database the **name** of a **country**, the **number** of **productions** associated with it, and the **average production budget** per country.

Display **zeros** when there are **no related** data for average **budgets**.

Filter **only** those **countries** that have **at least one** production.

Order the results by **number of productions** **descending** and then by **country name ascending**.

#### Required Columns

* name as country\_name
* productions\_count
* avg\_budget

#### Example

|  |  |  |
| --- | --- | --- |
| country\_name | productions\_count | avg\_budget |
| Argentina | 2 | 4000000.000000000000 |
| Armenia | 2 | 1250000.000000000000 |
| Cuba | 2 | 22700000.000000000000 |
| … | … | … |
| Germany | 1 | 0 |
| … | … | … |
| United States | 1 | 150000.000000000000 |

### Classify by Rating

Extract **production information** from the database, including the **title**, **rating**, **subtitles**, and the **count** of **participating actors** for each **production**.

**Classify** the **rating** as follows:

* If the rating is **equal** to or **less** than **3.50**, display "**poor**"
* If the rating is **above** **3.50** and **less** than or **equal** to **8.00**,display "**good**"
* If the rating is **above** **8.00**, display "**excellent**"

If the production has **subtitles**, display "**Bulgarian**", otherwise "**N/A**".

**Order** the results by **rating ascending** (classified value), then by the **count** of **participating actors descending**, and finally by **title ascending**.

#### Required Columns

* title
* rating
* subtitles
* actors\_count

#### Example

|  |  |  |  |
| --- | --- | --- | --- |
| **title** | rating | subtitles | actors\_count |
| Ask the Dust | excellent | N/A | 5 |
| American Me | excellent | Bulgarian | 4 |
| … | … | … | … |
| Lumiere and Company | good | Bulgarian | 5 |
| Vikings | good | Bulgarian | 5 |
| … | … | … | … |
| Rolling Thunder | poor | N/A | 4 |

## Section 4: Programmability - 20 Pts

Now it's time to showcase your database skills with some dynamic scripting. Get ready to write a series of functions and procedures to demonstrate your versatility.

### Productions Count by Category

Create a **user-defined function** named **udf\_category\_productions\_count(category\_name VARCHAR(50))** that **receives** a **category** **name** and **returns** the **total number** of associated **productions** in a **string** format**:**

* "**Found {count} productions.**"

Submit **only** your **user-defined function** to the Judge system.

#### Example

|  |
| --- |
| **Test Query** |
| **SELECT** udf\_category\_productions\_count('Nonexistent') **AS** message\_text; |
| **Result** |
| **message\_text** |
| Found 0 productions. |
|  |
| **Test Query** |
| SELECT udf\_category\_productions\_count('History') AS message\_text; |
| Result |
| message\_text |
| Found 3 productions. |

### Awarded Production

A production has been awarded. Your task is to find the related actors and give them their awards.

Create a stored procedure **udp\_awarded\_production** with the following parameters:

* production\_title(VARCHAR(70))

The procedure **udp\_awarded\_production(production\_title VARCHAR(70))** receives a **production title** as input and **modifies** the associated **actors'** **number** of **awards** by **increasing** them by **1** if the **title exists**.

Submit **only** your **stored procedure** to the Judge system.

#### Example

|  |
| --- |
| **Test Query** |
| CALL udp\_awarded\_production('Tea For Two'); |

|  |  |  |
| --- | --- | --- |
| **Initial State of Associated Actors** | | |
| **first\_name** | **last\_name** | **awards** |
| Brandon | Eykelhof | 0 |
| Flem | Loomis | 19 |
| Jared | Di Batista | 0 |
| Wolfgang | Vowdon | 7 |

|  |  |  |
| --- | --- | --- |
| **Resulting State** | | |
| **first\_name** | **last\_name** | **awards** |
| Brandon | Eykelhof | 1 |
| Flem | Loomis | 20 |
| Jared | Di Batista | 1 |
| Wolfgang | Vowdon | 8 |