IBM Developer
SKILLS NETWORK

Hands-on Lab : Create Tables and Load Data in PostgreSQL using pgAdmin

Estimated time needed: 20 minutes

In this lab, you will learn how to create tables and load data in the PostgreSQL database service using the pgAdmin graphical user interface (GUI) tool. The pgAdmin GUI provides an alternative to the command line for interacting with a PostgreSQL database using a graphical interface. This provides a number of key features for interacting with a PostgreSQL database in an easy to use format.

Software Used in this Lab

In this lab, you will use [PostgreSQL Database](#). PostgreSQL is a Relational Database Management System (RDBMS) designed to efficiently store, manipulate, and retrieve data.

To complete this lab you will utilize the PostgreSQL relational database service available as part of IBM Skills Network Labs (SN Labs) Cloud IDE. SN Labs is a virtual lab environment used in this course.

Database Used in this Lab

Books database has been used in this lab.

The following diagram shows the structure of the myauthors table from the Books database:

myauthors	
author_id	int
first_name	varchar(100)
middle_name	varchar(50)
last_name	varchar(100)

Objectives

After completing this lab, you will be able to use pgAdmin with PostgreSQL to:

- Create databases and tables in a PostgreSQL instance

- Load data into tables manually using the pgAdmin GUI

- Load data into tables from a text/script file

Lab Structure

In this lab, you will complete several tasks in which you will learn how to create tables and load data in the PostgreSQL database service using the pgAdmin graphical user interface (GUI) tool.

Task A: Create a database

First, to create a database on a PostgreSQL server instance, you'll first want to actually launch a PostgreSQL server instance on Cloud IDE and open up the pgAdmin Graphical User Interface.

1. Click on the Skills Network extension button on the left side of the window.

2. Open the "DATABASES" drop down menu and click on "PostgreSQL"

3. Click on the "Start" button. PostgreSQL may take a few moments to start.
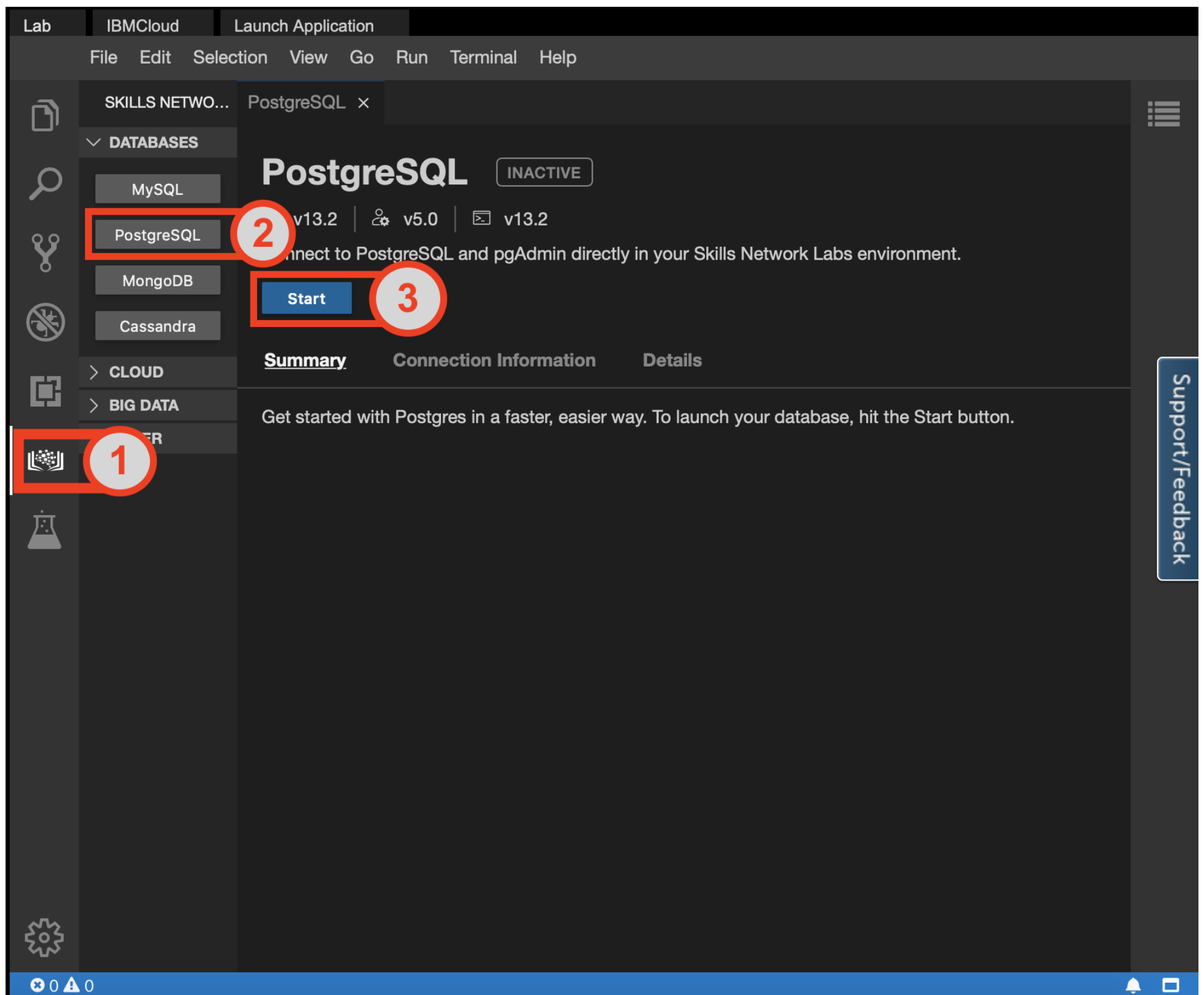
4. Next, open the pgAdmin Graphical User Interface by clicking the "pgAdmin" button in the Cloud IDE interface.
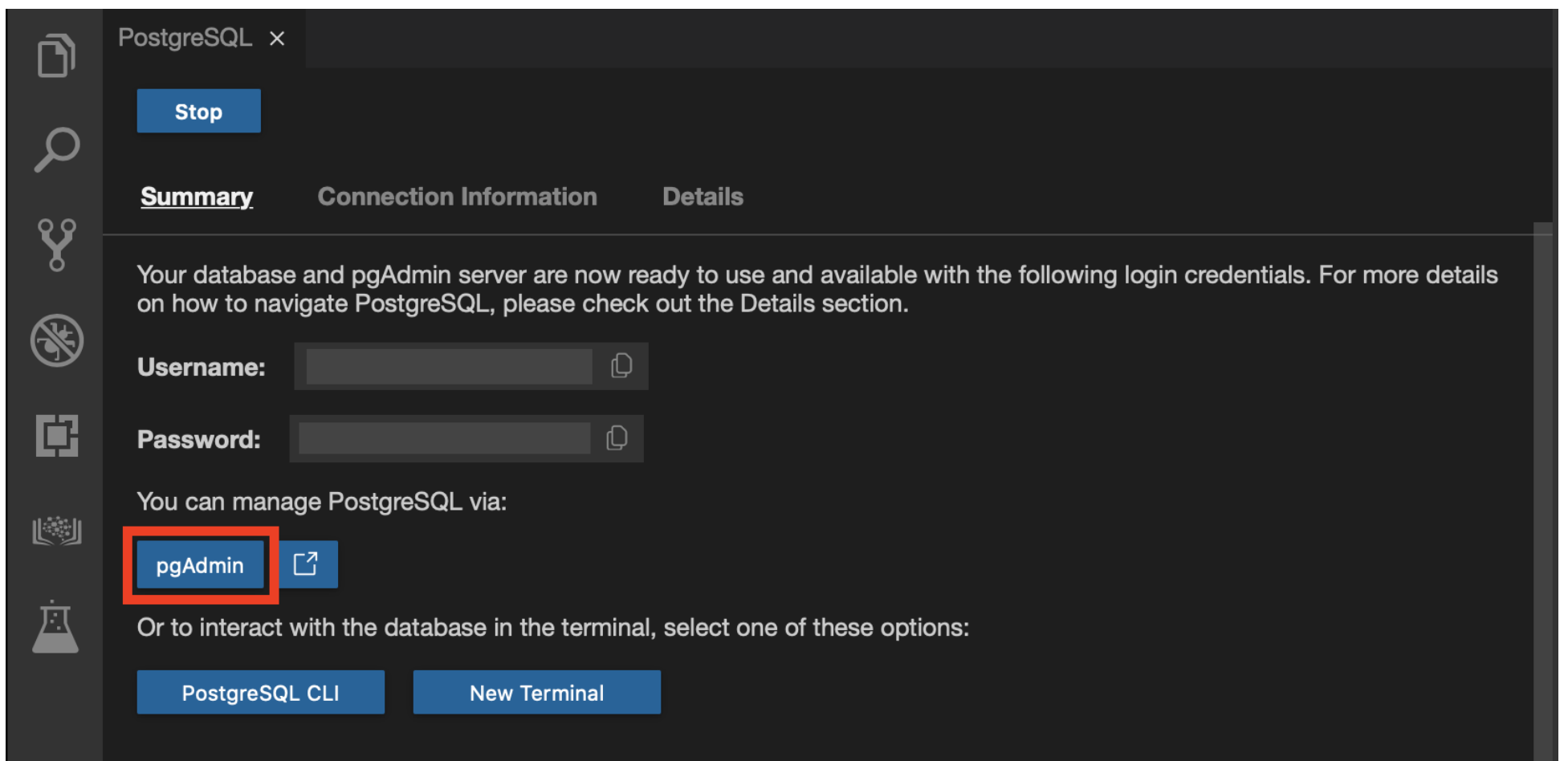
5. Once the pgAdmin GUI opens, click on the `Servers` tab on the left side of the page. You will be prompted to enter a password.
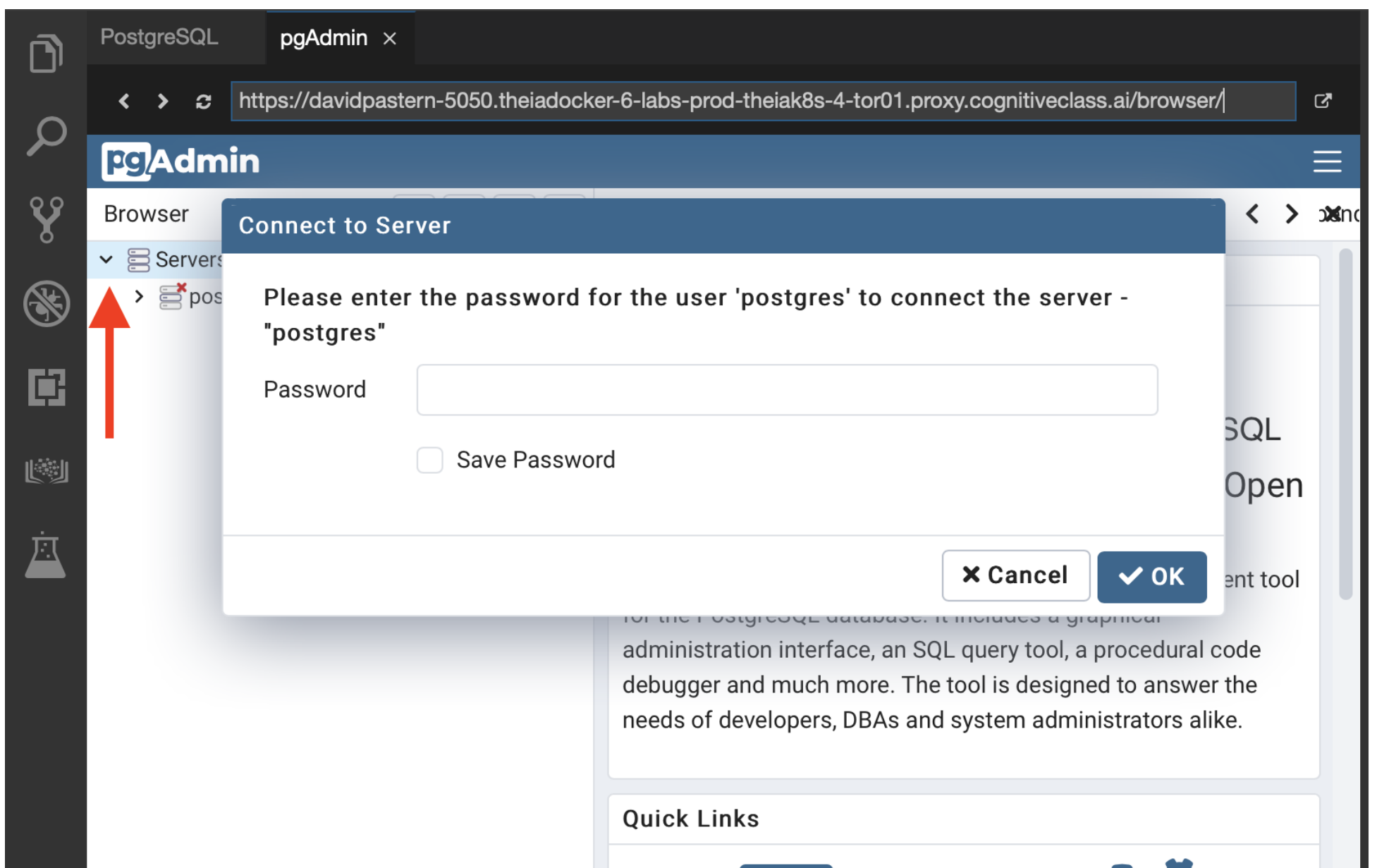
6. To retrieve your password, click on the "PostgreSQL" tab near the top of the interface.

7. Click on the Copy icon to the left of your password to copy the session password onto your clipboard.
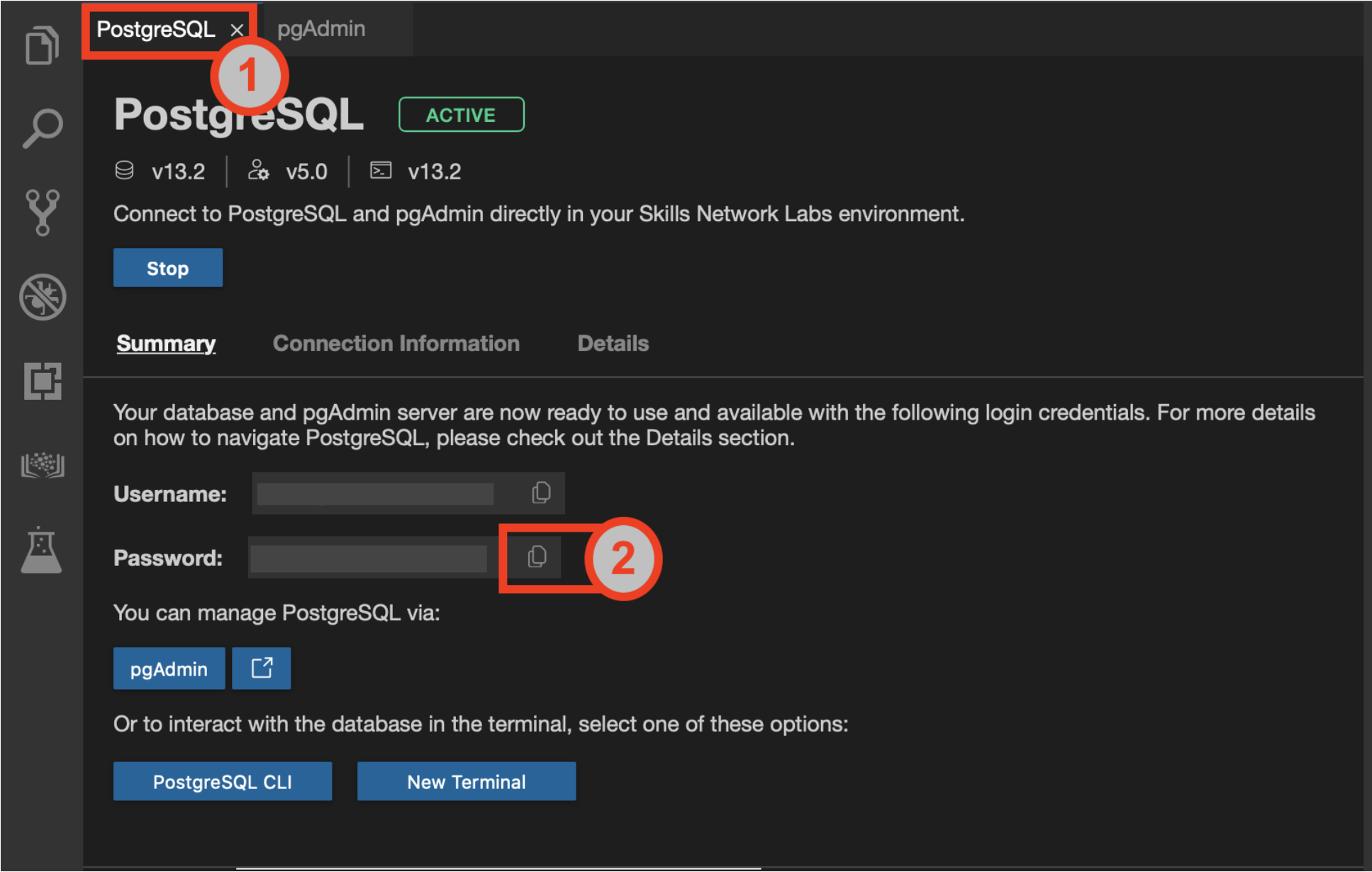
8. Navigate back to the "pgAdmin" tab and paste in your password, then click OK

9. You will then be able to access the pgAdmin GUI tool.
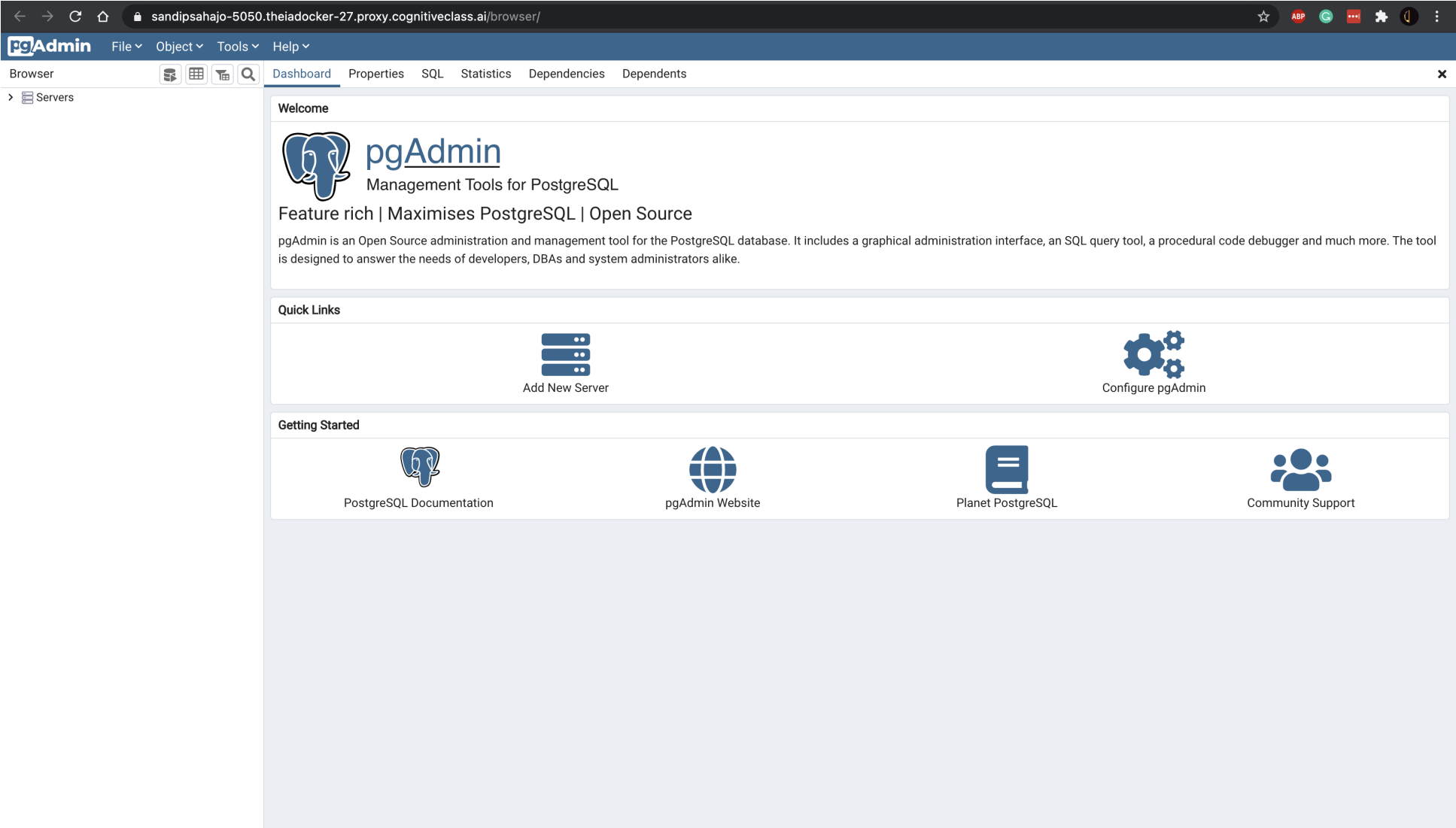
10. In the tree-view, expand **Servers** > **postgres** > **Databases**. If prompted, enter your PostgreSQL service session password. Right-click on **Databases** and go to **Create > Database**. In the **Database** box, type **Books** as the name for your new database, and then click **Save**. Proceed to Task B.

pgAdmin   File ˅   Object ˅   Tools ˅   Help ˅

Browser   **1**   🔁 🧮 📇 🔍   Dashboard   Properties   SQL   Statistics

˅ 🗄 Servers (1)   **2**   Server sessions

  ˅ 🐘 postgres   7

    ˅ 🗄 Databases (1)   **3**   4
                                                             3
      ˅ 🗄 postgres   ┌─ Create ──────────────── › ─┬─ Database... ──────┐   2
        › 🔩 Casts   │   │   🖱   1
        › 🔶 Catal   │ Refresh...   │   0
        › 🔲 Event Triggers
        › 🗃 Extensions   Tuples in   ▮ Inse
        › 🗄 Foreign Data Wrappers
        › 💬 Languages   1
        › 📡 Publications
        › 🔷 Schemas
        › 🌙 Subscriptions
      › 👥 Login/Group Roles
      › 📁 Tablespaces

🗄 **Create - Database**   ✕

General   Definition   Security   Parameters   Advanced   SQL

Database   Books

Owner   👤 postgres   ˅

Comment

                                                    ✖ Cancel   ♺ Reset   💾 Save
    ℹ   ?   🖱

Task B: Create tables

Now that you have your PostgreSQL service active and have created the **Books** database using pgAdmin, let's go ahead and create a few tables to populate the database and store the data that we wish to eventually upload into it.

1. In the tree-view, expand **Books** > **Schemas** > **public**. Right-click on **Tables** and go to **Create** > **Table**.

2. On the **General** tab, in the **Name** box, type **myauthors** as name of the table. Don't click Save, proceed to the next step.

▦ Create - Table   ✕

| General | Columns | Advanced | Constraints | Partitions | Parameters | Security | SQL |

Name

> myauthors

Owner

> 👤 postgres ▾

Schema

> ◈ public   ✕ ▾

Tablespace

> Select an item... ▾

Partitioned table?

> No

Comment

ℹ   ?   ✖ Cancel   ⟳ Reset   🖫 Save

3. Switch to tab **Columns** and click the **Add new row** button four times to add **4** column placeholders. Don't click Save, proceed to the next step.

▦ Create - Table   ✕

| General | **Columns** | Advanced | Constraints | Partitions | Parameters | Security | SQL |

Inherited from table(s)   Select to inherit from...

Columns   **+**

		Name ▲	Data type	Length/Precision	Scale	Not NULL?	Primary key?
✎	🗑		Select an item... ▼			No	No
✎	🗑		Select an item... ▼			No	No
✎	🗑		Select an item... ▼			No	No
✎	🗑		Select an item... ▼			No	No

ℹ   ?   ✕ Cancel   ⟳ Reset   🖫 Save

4. Enter the **myauthors** table definition structure information as shown in the image below in the highlighted boxes. Then click **Save**. Proceed to Task C.

▦ Create - Table   ✕

| General | Columns | Advanced | Constraints | Partitions | Parameters | Security | SQL |

Inherited from table(s)   Select to inherit from...

Columns   ✚

		Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?
✎	🗑	author_id	integer ▾			Yes	Yes
✎	🗑	first_name	character varying ▾	100		No	No
✎	🗑	middle_name	character varying ▾	50		No	No
✎	🗑	last_name	character varying ▾	100		No	No

ℹ   ❓   ✕ Cancel   ⟳ Reset   💾 Save

Task C: Load data into tables manually using the pgAdmin GUI

Great! You now have a database and have created tables within it. With the pgAdmin GUI, you can insert values into the tables manually. This is useful if you have a few new entries you wish to add to the database. Let's see how to do it.

1. In the tree-view, expand **Tables**. Right-click on **myauthors** and go to **View/Edit Data > All Rows**.
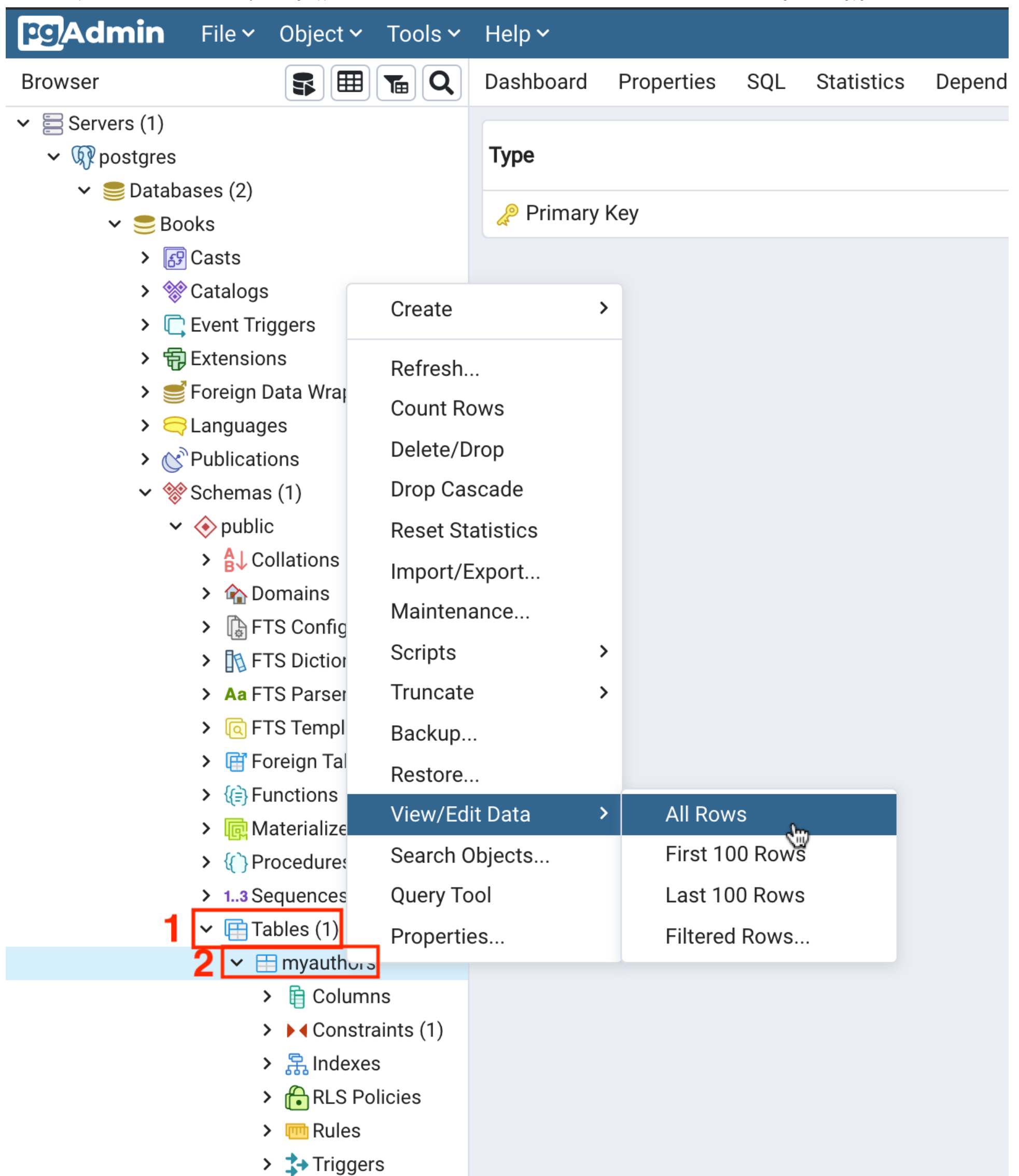
2. You will insert 2 rows of data into the **myauthors** table. In the lower **Data Output** pane, enter **myauthors** table data information for 2 rows as shown in the highlighted boxes in the image below. Then click the **Save Data Changes** button. Proceed to Task D.

| Dashboard | Properties | SQL | Statistics | Dependencies | Dependents | ⊞ public.myauthors/Books |

Save Data Changes icon

public.myauthors/B

Query Editor   Query History

```
1  SELECT * FROM pub
2  ORDER BY author_id
```

Data Output   Explain   Messages   Notifications

	author_id [PK] integer ✎	first_name character varying (100) ✎	middle_name character varying (50) ✎	last_name character varying (100) ✎
1	1	Merrit	[null]	Eric
2	2	Linda	[null]	Mul

Task D: Load data into tables using a text/script file

In the previous task, you entered some data entries into a table manually with pgAdmin. While this method can be useful for small additions, if you wish to upload large amounts of data at once, that process becomes far too tedious. An alternative is to load data into tables from a text or script file containing the data you wish to enter. Let's take a look at how to do this.

1. Finally, you will import the remainder of the **myauthors** table data from a csv text file. Download the csv file below to your local computer:

   ○ myauthors.csv

2. In the tree-view, right-click on **myauthors** and go to **Import/Export**.

pgAdmin   File ⌄   Object ⌄   Tools ⌄   Help ⌄

Browser   ⊞ ⊞ ⊞ Q   Dashboard   Pr

⌄ ⊟ Servers (1)
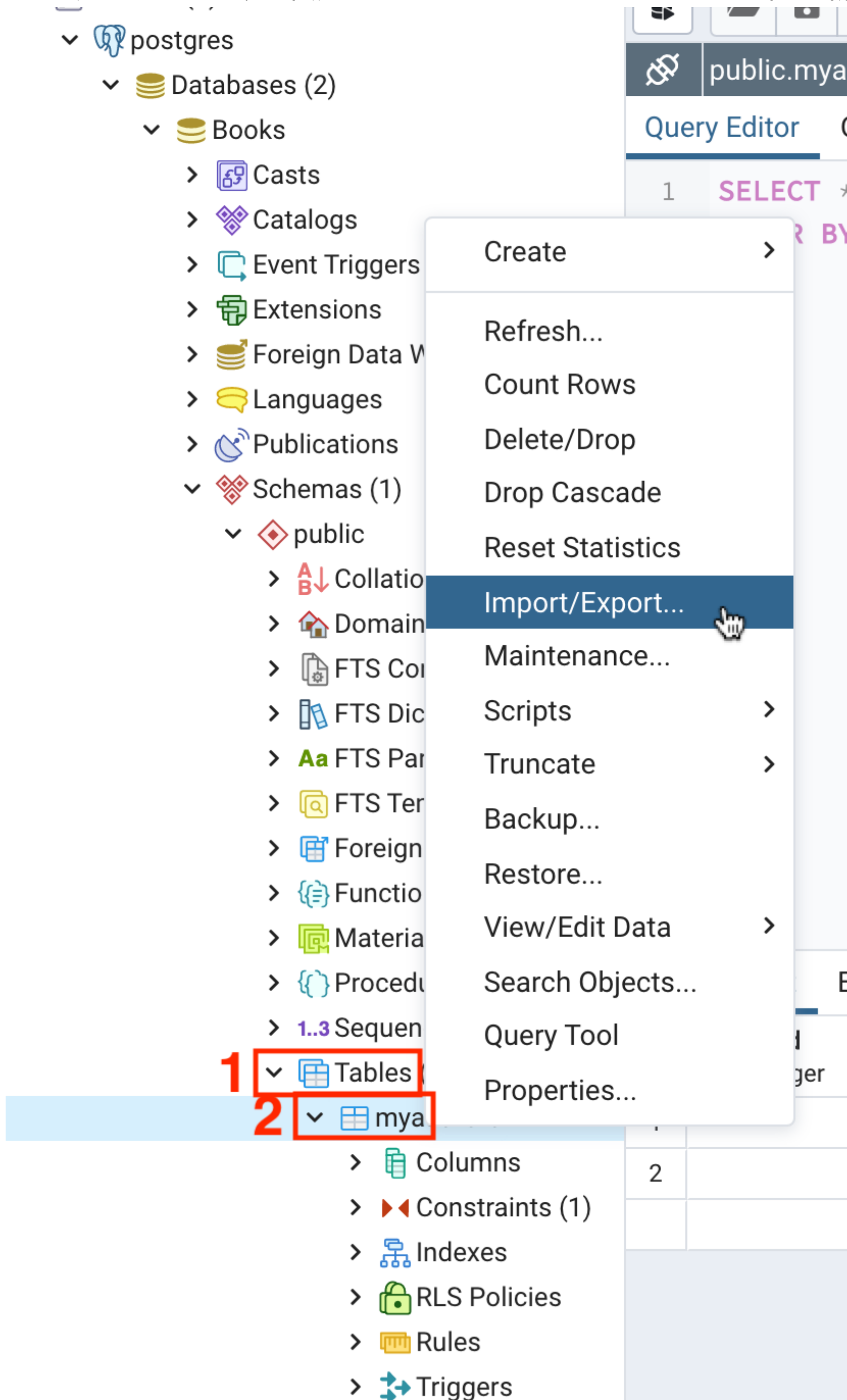
3. Follow the instructions below to import:

- Make sure Import/Export is set to **Import**, Format = **csv** and Header = **Yes**. Then click on the **Select file** button by the Filename box.

Import/Export data - table 'myauthors'

Options   Columns

Import/Export   Import   **1**

File Info   **4**

Filename ⚠   •••

Format   csv   **2**

Encoding   Select an item...

Miscellaneous

OID   No

Header   Yes   **3**

Delimiter   Select from list...

Specifies the character that separates columns within each row (line) of the file. The default is a tab character in text format, a comma in CSV format. This must be a single one-byte character. This option is not allowed when using binary format.

✗ Cancel   ✓ OK

○ Click the **Upload File** button.

Select file

⌂   ↥   /var/lib/pgadmin/   ⟳   ✎   ⬆   📂⁺   ▦   ☰

Name	Size	Modifie Upload File	
📁 sessions	4.0 kB	Mon Mar 22 02:15:08 2021	
📁 storage	4.0 kB	Mon Mar 22 02:11:24 2021	

Show hidden files and folders? ☐   Format   csv

✗ Cancel   📄 Select

○ Double-click on the drop files area and load the **myauthors.csv** you downloaded earlier from your local computer storage.

Select file	

🏠 ⬆ /var/lib/pgadmin/   🔁 ✏ ⬆ 📂⁺ ⦂⦂⦂ ☰

✖

Double click on this space

Drop files here to upload. The file size limit (per file) is 50 mb.

Show hidden files and folders? ☐   Format | csv ▾ |

✖ Cancel   📄 Select
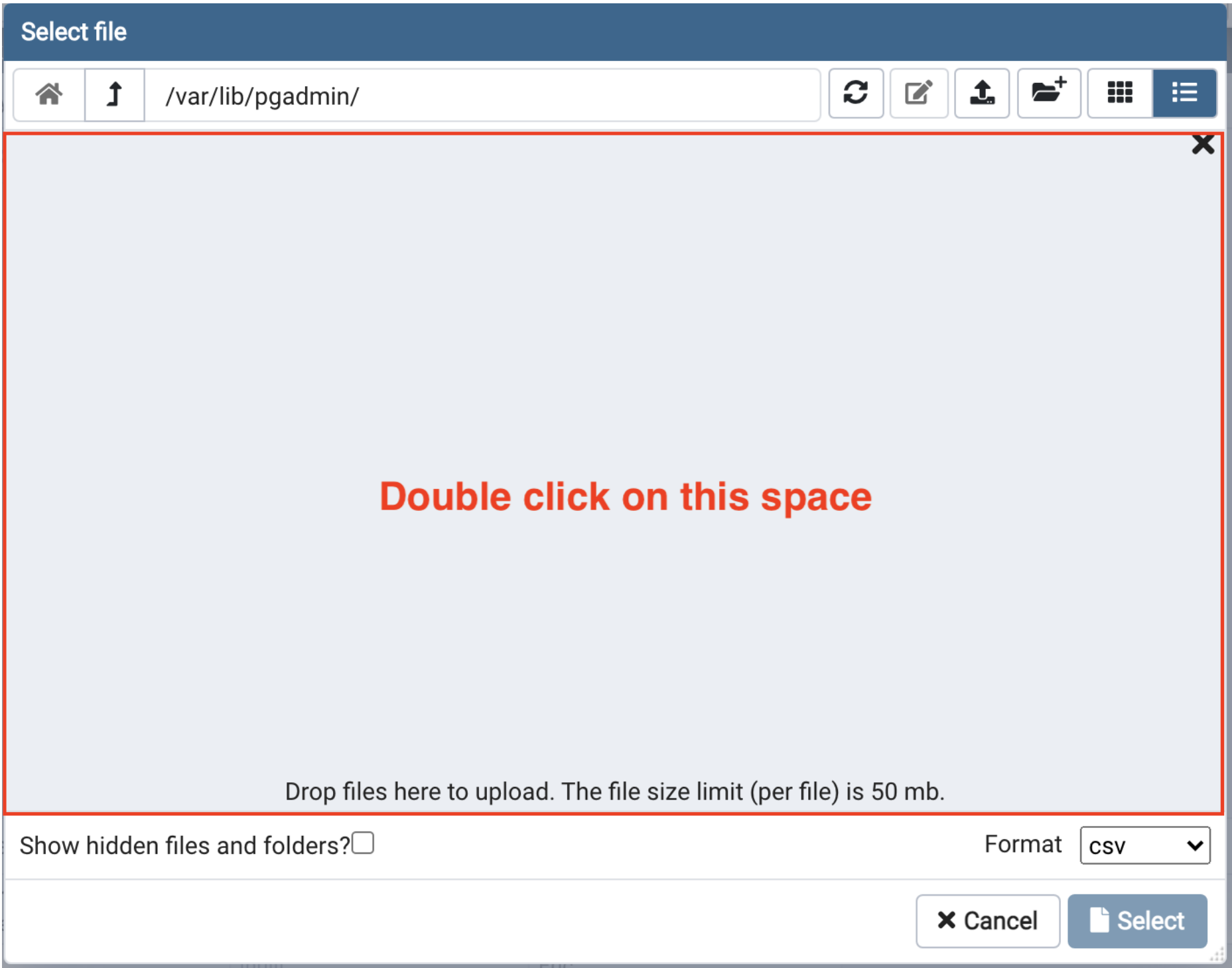
○ When the upload is complete, close the drop files area clicking the **X** button.
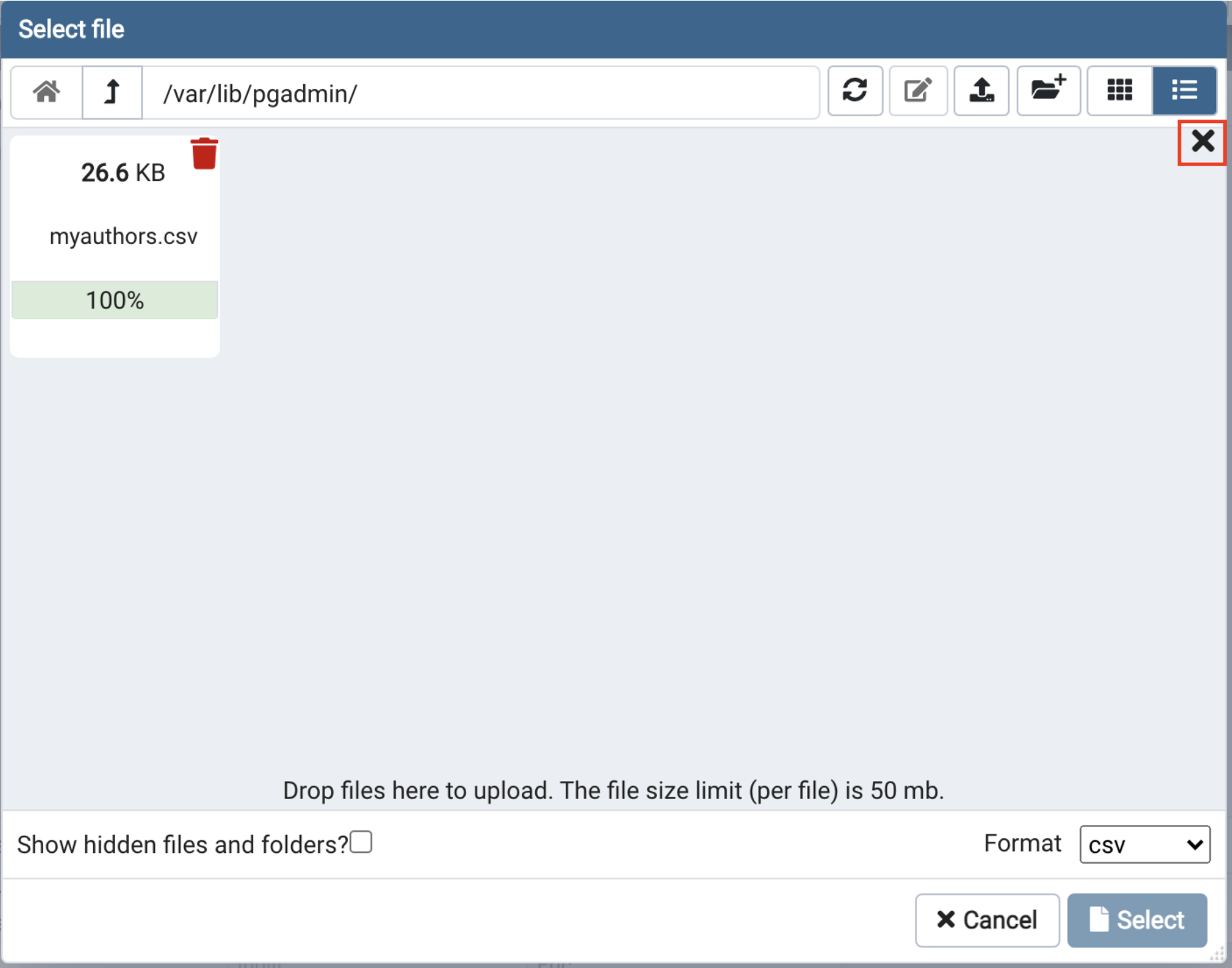
- Select the uploaded **myauthors.csv** file from the list and click the **Select** button.

Select file

/var/lib/pgadmin/myauthors.csv

Name	Size	Modified
📄 myauthors.csv	26.0 kB	Mon Mar 22 08:19:26 2021
📁 sessions	4.0 kB	Mon Mar 22 02:15:08 2021
📁 storage	4.0 kB	Mon Mar 22 02:11:24 2021

Show hidden files and folders? ☐   Format csv

✕ Cancel   📄 Select

- Click **OK** and notification of import success should appear.

Import/Export data - table 'myauthors'

Options　　**Columns**

Import/Export　　**Import**

File Info

Filename　　　/var/lib/pgadmin/myauthors.csv　　　···

Format　　　　csv　　　　　　　　　　　　　　　　▼

Encoding　　　Select an item...　　　　　　　　　　▼

Miscellaneous

OID　　　　　　No

Header　　　　Yes

Delimiter　　　Select from list...　　　　　　　　　▼

Specifies the character that separates columns within each row (line) of the file. The default is a tab character in text format, a comma in CSV format. This must be a single one-byte character. This option is not allowed when using binary format.

✕ Cancel　　　✔ OK

Import - Copying table data　　　　✕

Copying table data 'public.myauthors' on database 'Books' and server (postgres:5432)

Mon Mar 22 2021 02:26:40 GMT-0600 (Mountain Daylight Time)

🕐 0.02 seconds　　　　ℹ More details...　　　❌ Stop Process

✔　　　　　　　Successfully completed.

4. Repeat Task C Step 1 to check that the newly imported data rows appear along with your previously inserted 2 rows.

Dashboard	Properties	SQL	Statistics	Dependencies	Dependents	⊞ public.myauthors/Books/postgres@postgres

public.myauthors/Books/postgres@postgres

Query Editor   Query History

```
1   SELECT * FROM public.myauthors
2   ORDER BY author_id ASC
```

Data Output   Explain   Messages   Notifications

	author_id ✎ [PK] integer	first_name ✎ character varying (100)	middle_name ✎ character varying (50)	last_name ✎ character varying (100)	
1	1	Merrit	[null]	Eric	
2	2	Linda	[null]	Mul	
3	3	Alecos	[null]	Papadatos	
4	4	Paul	C.van	Oorschot	
5	5	David	[null]	Cronin	
6	6	Richard	[null]	Blum	
7	7	Yuval	Noah	Harari	
8	8	Paul	[null]	Albitz	
9	9	David	[null]	Beazley	
10	10	John	Paul	Shen	
11	11	Andrew	[null]	Miller	
12	12	Melanie	[null]	Swan	
13	13	Neal	[null]	Ford	
14	14	Nir	[null]	Shavit	
15	15	Tim	[null]	Kindberg	
16	16	Mike	[null]	McQuaid	
17	17	Brian	P.	Hogan	
18	18	Jean-Philippe	[null]	Aumasson	
19	19	Lance	[null]	Fortnow	
20	20	Richard	C.	Jeffrey	
21	21	William	L.	Simon	
22	22	Magnus	Lie	Hetland	
23	23	Mike	[null]	McShaffry	
24	24	Norman	[null]	Matloff	
25	25	John	E.	Hopcroft	
26	26	S.	[null]	Sudarshan	

As you can see, the data contained in the **csv** file was successfully uploaded into the table and you did not have to manually input hundreds of entries.

Conclusion

Congratulations! You have completed this lab, and you are ready for the next topic.

Author

- Sandip Saha Joy

Other Contributors

- David Pasternak

Changelog

Date	Version	Changed by	Change Description
2021-03-15	1.0	Sandip Saha Joy	Created initial version
2021-10-18	1.1	David Pasternak	Updated lab instructions