

# 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 GUI 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.



PostgreSQL

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

You will use the Books database 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

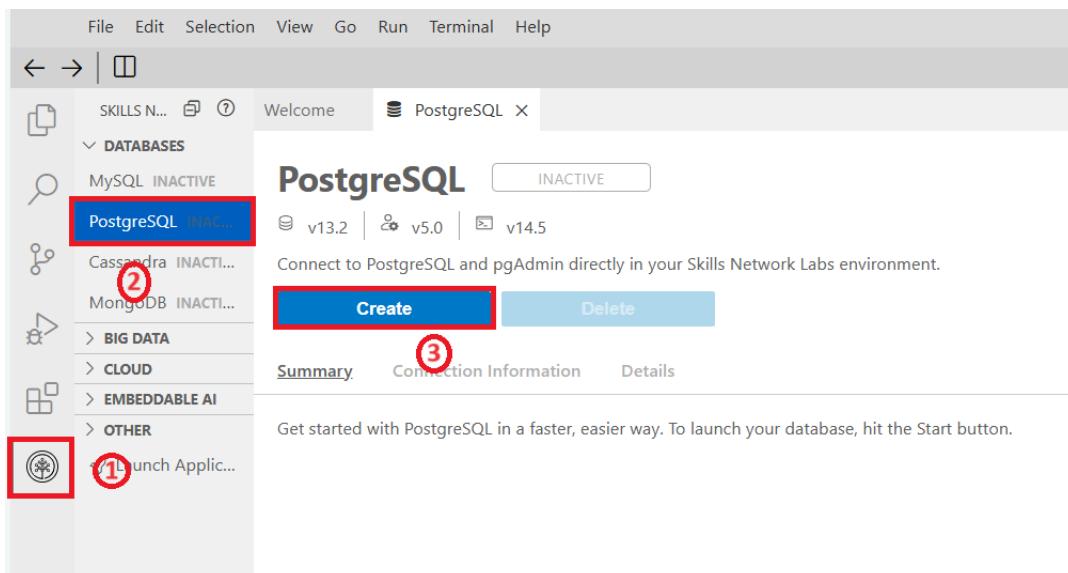
## Lab Structure

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 launch a PostgreSQL server instance on Cloud IDE and open the pgAdmin Graphical User Interface.

1. Click the Skills Network extension button on the left side of the window.
2. Open the **DATABASES** menu and click **PostgreSQL**.
3. Click **Create**. PostgreSQL may take a few moments to start.



4. Next, open the pgAdmin Graphical User Interface by clicking **pgAdmin** in the Cloud IDE interface.

Welcome PostgreSQL

**PostgreSQL** ACTIVE

v13.2 | v5.0 | v14.5

Connect to PostgreSQL and pgAdmin directly in your Skills Network Labs environment.

Create Delete

Summary Connection Information Details

Your database and pgAdmin server are now ready to use and available with the following login credentials. For more details on how to navigate PostgreSQL, please check out the Details section.

You can manage PostgreSQL via:

**pgAdmin** (highlighted with a red box)

Or to interact with the database in the terminal, select one of these options:

PostgreSQL CLI New Terminal

5. Once the pgAdmin GUI opens, click **Servers** tab on the left side of the page. You will be prompted to enter a password.

PostgreSQL pgAdmin

https://davidpastern-5050.theiadocker-6-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai

**pgAdmin**

Browser Connect to Server

Servers > pos

Please enter the password for the user 'postgres' to connect the server "postgres"

Password

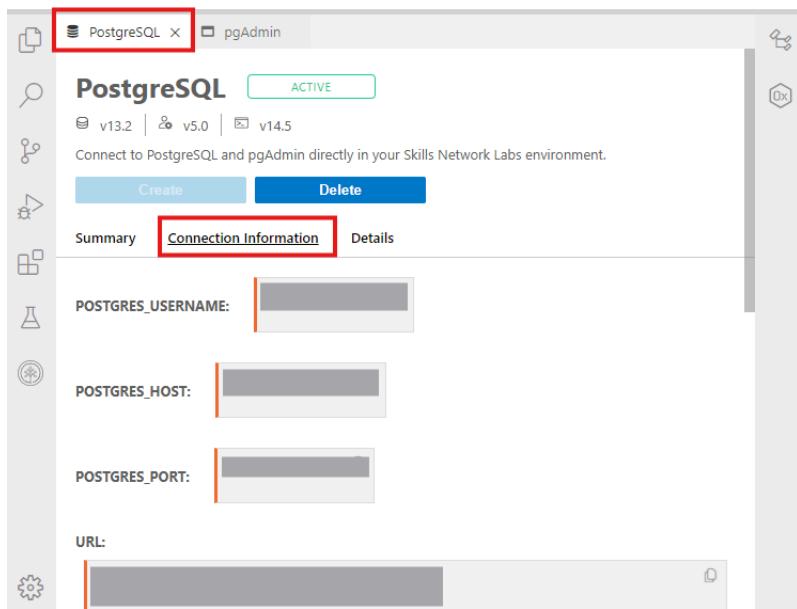
Save Password

**Cancel**

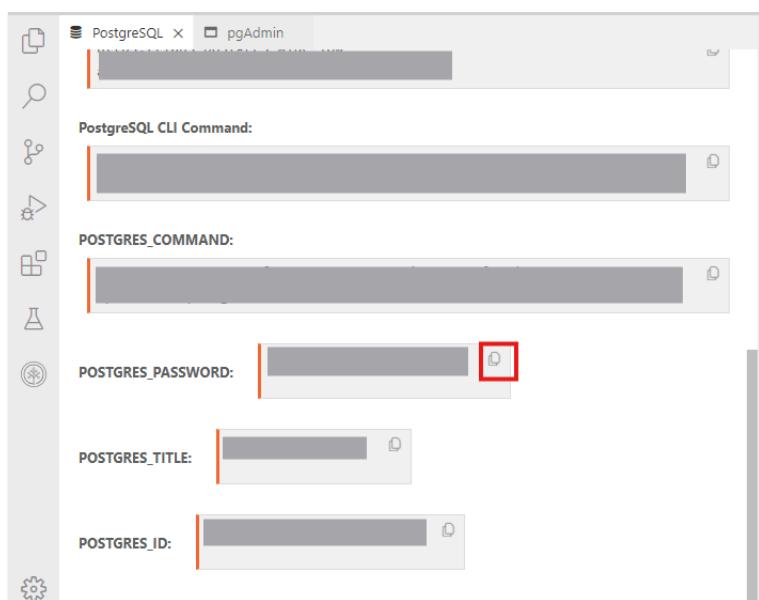
For the PostgreSQL database. It includes a graphical administration interface, an SQL query tool, a debugger and much more. The tool is designed for the needs of developers, DBAs and system administrators.

Quick Links

6. To retrieve your password, click **PostgreSQL** tab near the top of the interface and select **Connection Information** tab.



7. Scroll down and click the Copy icon on 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.

← → C ⌂ sandipsahajo-5050.theiadocker-27.proxy.cognitiveclass.ai/browser/

**pgAdmin** File ▾ Object ▾ Tools ▾ Help ▾

Browser     Dashboard Properties SQL Statistics Depen

>  Servers

Welcome

 pg**Admin**  
Management Tools for Post  
Feature rich | Maximises PostgreSQL  
pgAdmin is an Open Source administration and management tool designed to answer the needs of developers, DBAs

Quick Links

 Add New

Getting Started

 PostgreSQL Documentation

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.

Browser **1**

Servers (1) **2**

postgres **3**

Databases (1)

- postgres
  - Cast...
  - Catalog...
  - Event Triggers
  - Extensions
  - Foreign Data Wrappers
  - Languages
  - Publications
  - Schemas
  - Subscriptions
  - Login/Group Roles
  - Tablespaces

Create Database... **4**

Server sessions  
7

Tuples in  
1

### Create - Database

General Definition Security Parameters Advanced SQL

Database **Books**

Owner postgres

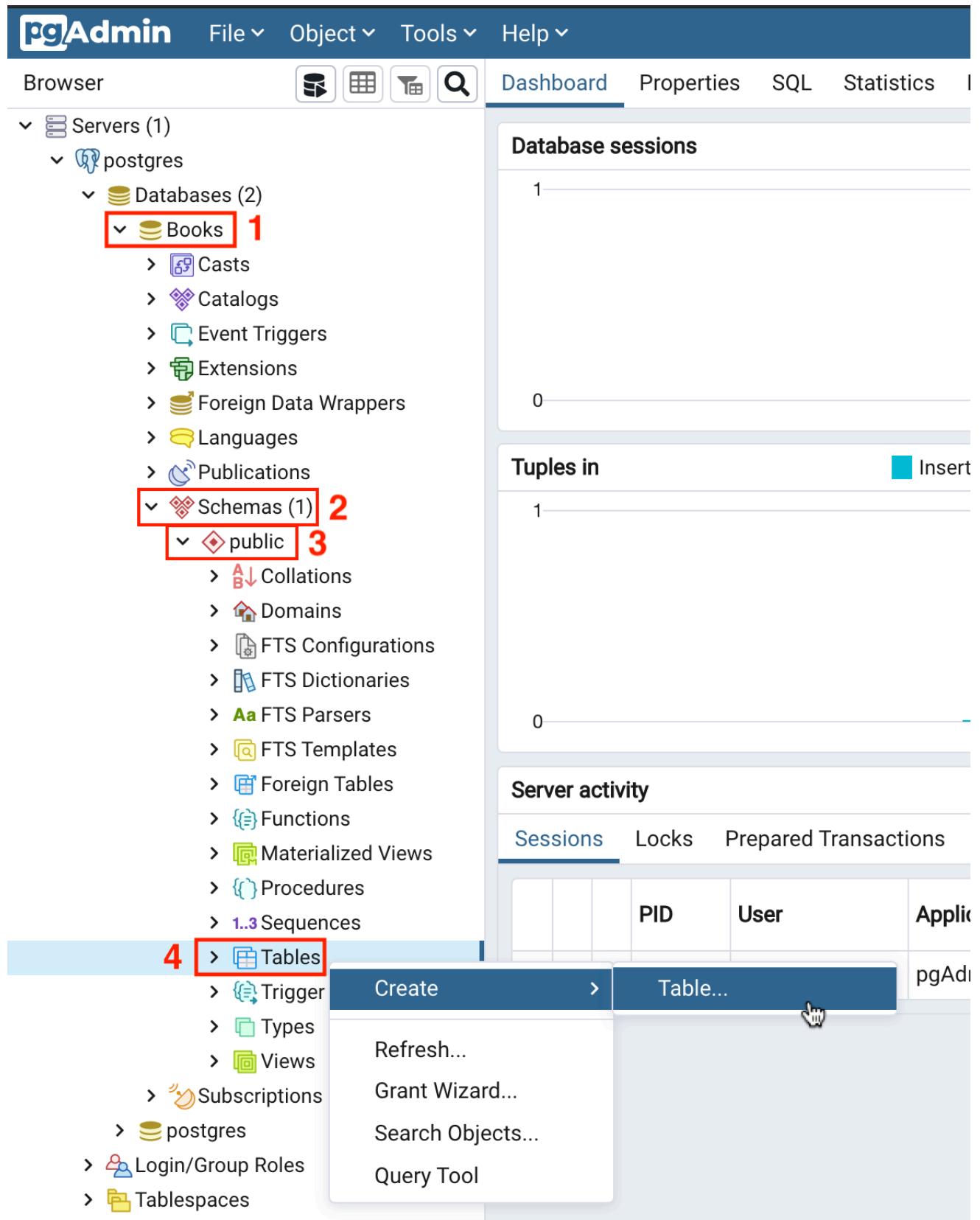
Comment

**i** **?** **Cancel** **Reset** **S**

## Task B: Create tables

Now that you have your PostgreSQL service active and have created the **Books** database using pgAdmin, let's create a few tables to populate the database and store the data that you 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



Cancel

Reset



3. Switch to the 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
		<input type="text"/>	Select an item... ▾		<input checked="" type="checkbox"/> No	<input type="checkbox"/> No
		<input type="text"/>	Select an item... ▾		<input checked="" type="checkbox"/> No	<input type="checkbox"/> No
		<input type="text"/>	Select an item... ▾		<input checked="" type="checkbox"/> No	<input type="checkbox"/> No
		<input type="text"/>	Select an item... ▾		<input checked="" type="checkbox"/> No	<input type="checkbox"/> No



Cancel

Reset



- 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
	author_id	integer				
	first_name	character varying	100			
	middle_name	character varying	50			
	last_name	character varying	100			



Cancel

Reset



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

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 **myauthors** and go to **View/Edit Data > All Rows**.

pgAdmin   File ▾   Object ▾   Tools ▾   Help ▾

Browser   Dashboard   Properties   SQL   Statistics   Dependencies

Servers (1)   postres   Databases (2)   Books   Casts   Catalogs   Event Triggers   Extensions   Foreign Data Wrappers   Languages   Publications   Schemas (1)   public   Collations   Domains   FTS Configurations   FTS Dictionaries   FTS Parsers   FTS Templates   Foreign Tables   Functions   Materialized Views   Procedures   Sequences   Tables (1)   myauthors   Columns   Constraints (1)   Indexes   RLS Policies   Rules   Triggers

Type   Primary Key

Create   Refresh...   Count Rows   Delete/Drop   Drop Cascade   Reset Statistics   Import/Export...   Maintenance...   Scripts   Truncate   Backup...   Restore...   View/Edit Data   All Rows   Search Objects...   First 100 Rows   Last 100 Rows   Filtered Rows...   Properties...

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** icon. Proceed to Task D.

public.myauthors/Books/postgres@PostgreSQL

No limit ▾ E I S

**Query** Query History

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

Select "Add Row" to add values to the Table

Data Output Messages Notifications

Add row first\_name character varying (30) middle\_name character varying (20) last\_name character varying (30)

Alt Shift A

3. Enter the values into the table as shown below:

	author_id	first_name	middle_name	last_name
1	1	Merrit	[null]	Eric
2	2	Linda	[null]	Mul

Data Output Messages Notifications

author\_id [PK] integer first\_name character varying (30) middle\_name character varying (20) last\_name character varying (30)

1+ 2 Linda [null] Eric

2+ 1 Merritt [null] Mui

Double-click the cell to enter values into the table.

4. Save the values.

Data Output Messages Notifications Click here to save the values.

Save Data Changes F6

	author_id	first_name	middle_name	last_name
1+	2	Linda	[null]	Eric
2+	1	Merritt	[null]	Mui

## 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, the process becomes 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. You will import the remainder of the **myauthors** table data from a csv text file. Download the csv file below to your local computer:

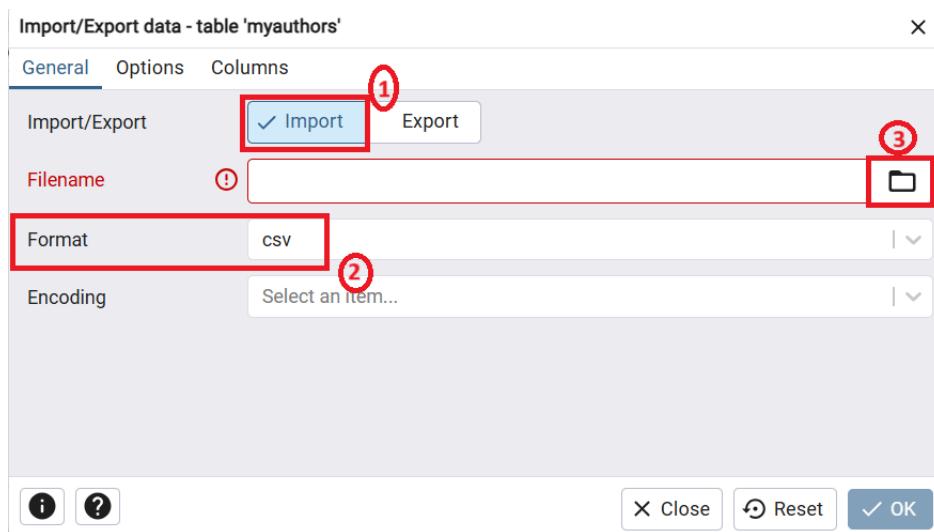
- o [myauthors.csv](#)

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

The screenshot shows the pgAdmin 4 interface. In the left sidebar, under the 'Servers' section, the 'postgres' database is selected. Within 'postgres', the 'Books' schema is expanded, and the 'Tables' node is highlighted with a red box labeled '1'. Under 'Tables', a specific table named 'maya' is selected and highlighted with a red box labeled '2'. A context menu is open over the 'maya' table, listing various actions: Create, Refresh..., Count Rows, Delete/Drop, Drop Cascade, Reset Statistics, Import/Export... (which is highlighted with a blue background and a cursor icon), Maintenance..., Scripts, Truncate, Backup..., Restore..., View/Edit Data, Search Objects..., Query Tool, and Properties... .

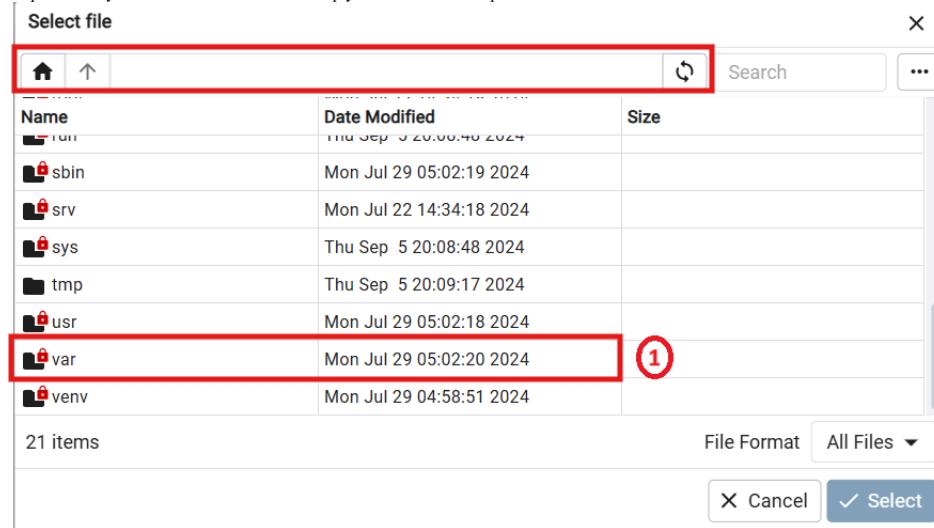
3. Follow the instructions below to import:

1. Make sure **Import/Export** is set to **Import**,
2. **Format = csv**.
3. Then click **Select file** icon by the **Filename** box.

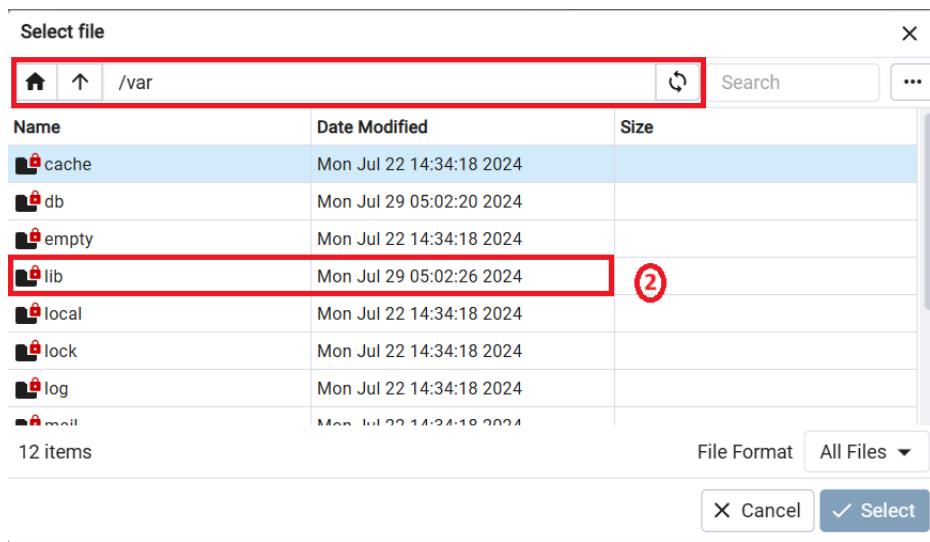


4. Steps to Upload File.

- Step 1: Initially make sure the folder details empty and select the var option from the list as shown in the screenshot below. Select var folder



- Step 2: Select lib folder.



- Step 3: Select pgadmin folder. Here you could notice the folders are locked except the pgadmin folder.

Select file

Name	Date Modified	Size
misc	Mon Jul 22 14:34:18 2024	
pgadmin	Fri Sep 6 01:00:10 2024	
postfix	Thu Sep 5 20:09:12 2024	
sudo	Mon Jul 29 05:02:20 2024	

- 4 items
- File Format All Files ▾
- Cancel  Select

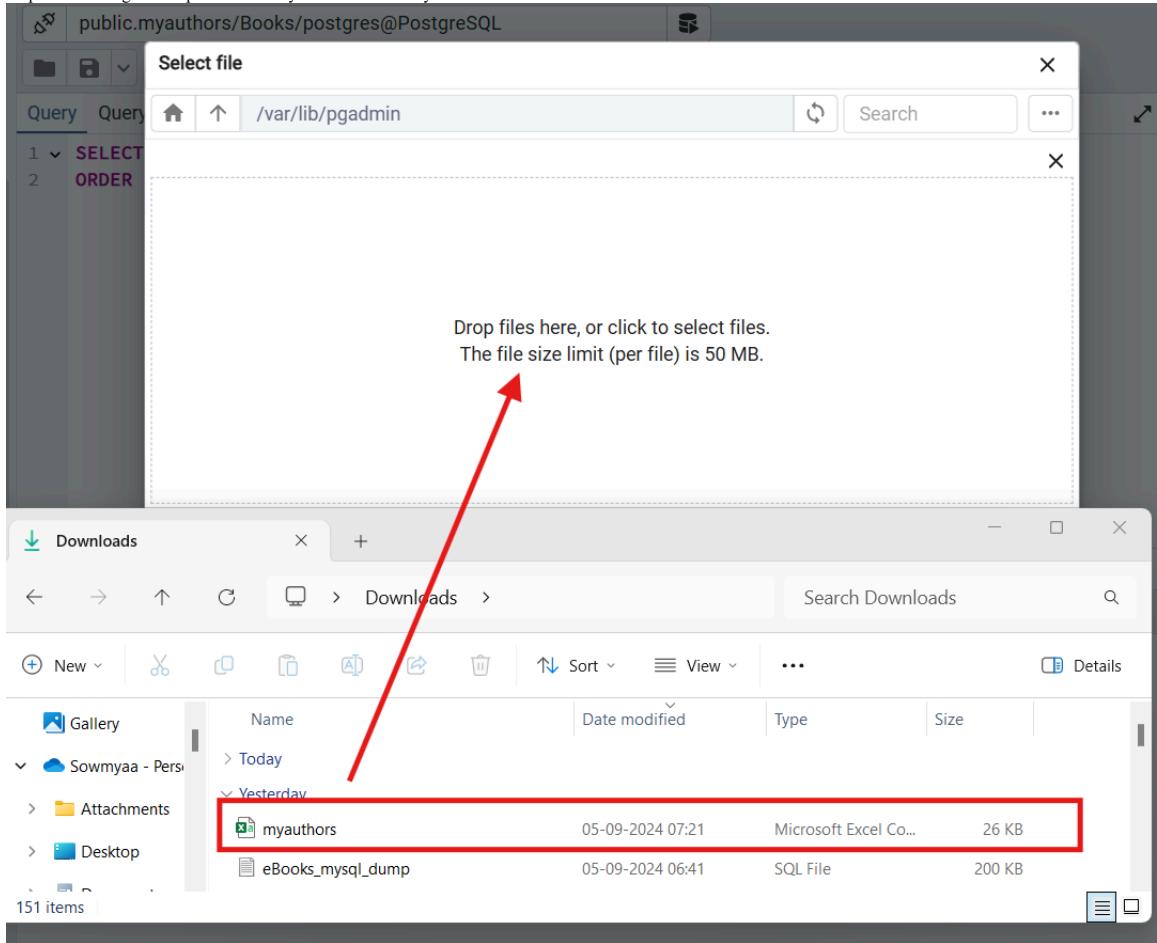
- Step 4: Now select upload as mentioned here.

Select file

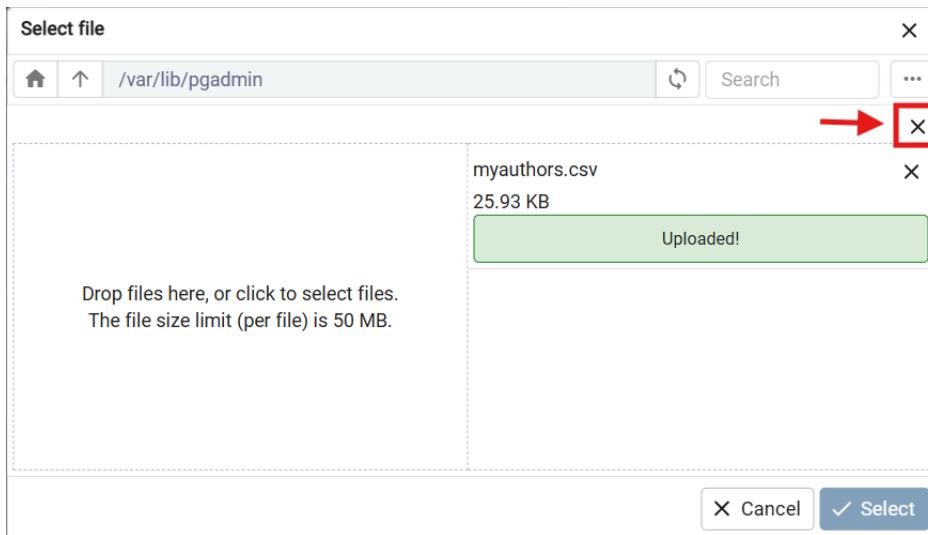
Name	Date Modified	Size	
azurecredentialcache	Thu Sep 5 20:08:53 2024		
pgadmin4.db	Fri Sep 6 01:04:34 2024	164.0 kB	<input type="button"/> Upload <input checked="" type="checkbox"/> List View <input type="button"/> Grid View <input type="button"/> Show Hidden Files
sessions	Thu Sep 5 23:43:26 2024		
storage	Thu Sep 5 20:08:53 2024		

- 4 items
- File Format All Files ▾
- Cancel  Select

- Step 5: Now Drag and drop the file from your downloads on your local machine.



- Step 6: Finally, the upload is successful. When the upload is complete, close the drop files area by clicking X.



- Select the uploaded **myauthors.csv** file from the list and click **Select**.

Select file

		/var/lib/pgadmin/myauthors.csv					
Name	Size	Modified					
myauthors.csv	26.0 kB	Mon Mar 22 08:19:2					
sessions	4.0 kB	Mon Mar 22 02:15:0					
storage	4.0 kB	Mon Mar 22 02:11:2					

Show hidden files and folders?

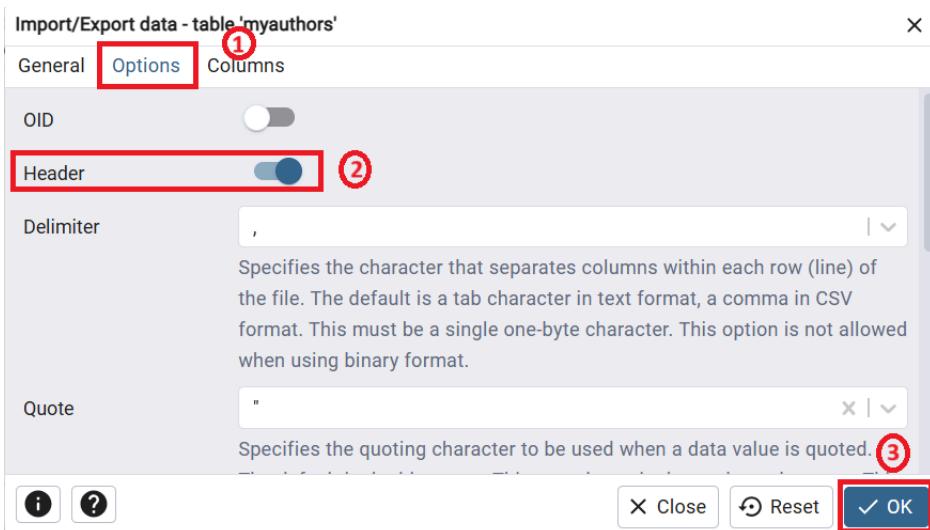
Format

- Ensure the file has selected.

Import/Export data - table 'myauthors'

General	Options	Columns
Import/Export	<input checked="" type="button" value="Import"/>	<input type="button" value="Export"/>
Filename	<input type="text" value="/var/lib/pgadmin/myauthors.csv"/>	
Format	csv	
Encoding	Select an item...	

- Under **Options** enable **Header** and Click OK and notification of import success will appear.



Dashboard × Properties × SQL × Statistics × Dependencies × Dependents × Processes × public.myauthors/Books/postgres@PostgreSQL

public.myauthors/Books/postgres@PostgreSQL

No limit

Query History

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

Scratch Pad ×

Data Output Messages Notifications

	author_id	first_name	middle_name	last_name
1	2	Linda	[null]	Murphy
2	1	Merritt	[null]	Erik

**Process completed**  
Copying table data 'public.myauthors' on database 'Books' and server 'PostgreSQL (low-mechanic:5432)'  
[View Processes](#)

**Process started**  
Copying table data 'public.myauthors' on database 'Books' and server 'PostgreSQL (low-mechanic:5432)'  
[View Processes](#)

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



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 have learned how to create databases and tables in a PostgreSQL instance, load data into tables manually using the pgAdmin GUI, and load data into tables from a text/script file.

## Author

- [Sandip Saha Joy](#)

## Other Contributors

- [David Pasternak](#)



# Skills Network

© IBM Corporation. All rights reserved.