

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.



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

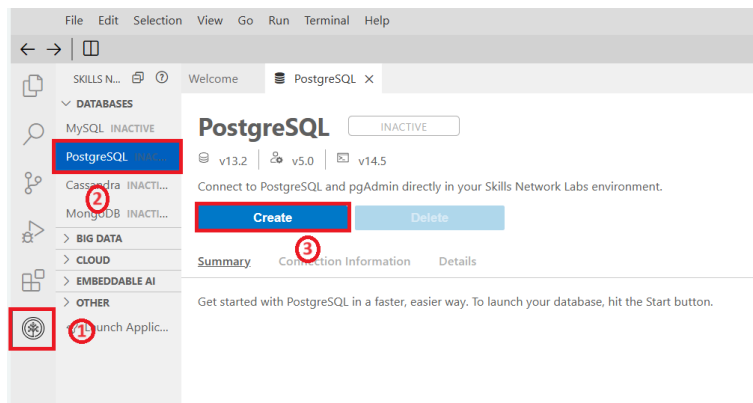
Lab Structure

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.

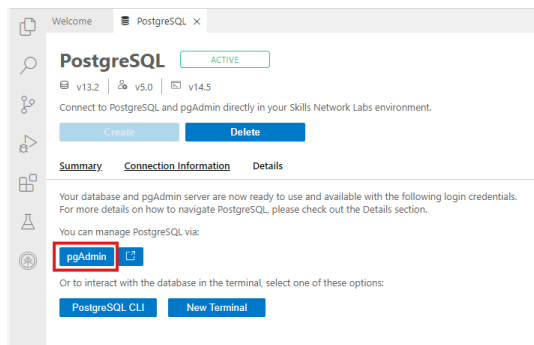
Task A: Create a database

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.

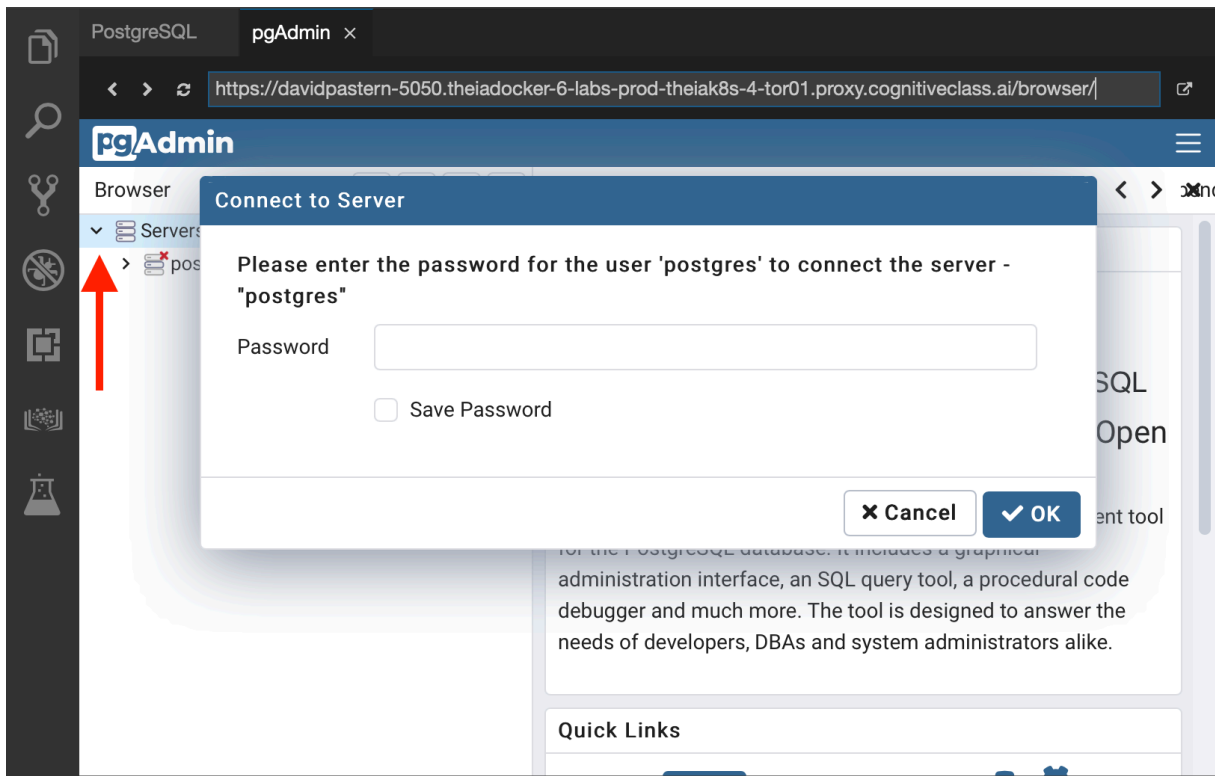
1. Click the Skills Network extension button on the left side of the window.
2. Open the DATABASES menu and click **PostgreSQL**.
3. Click **Create**. PostgreSQL may take a few moments to start.



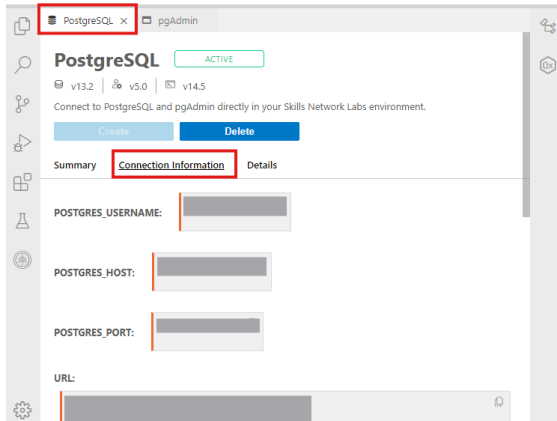
4. Next, open the pgAdmin Graphical User Interface by clicking **pgAdmin** in the Cloud IDE interface.



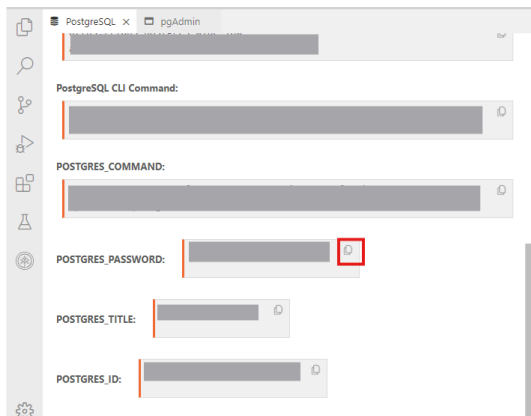
5. Once the pgAdmin GUI opens, click **Servers** tab on the left side of the page. You will be prompted to enter a password.



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.


← → ↻ 🏠 sandipsahajo-5050.theiadocker-27.proxy.cognitiveclass.ai/browser/

pgAdmin File ▾ Object ▾ Tools ▾ Help ▾

Browser Servers

Dashboard Properties SQL Statistics Dependencies Dependents

Welcome




pgAdmin

Management Tools for PostgreSQL

Feature rich | Maximises PostgreSQL | Open Source


pgAdmin is an Open Source administration and management tool for the PostgreSQL database. It is designed to answer the needs of developers, DBAs and system administrators alike.

Quick Links




Add New Server

Getting Started



PostgreSQL Documentation



pgAdmin Website

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 Servers (1) postgres Databases (1)

1 2 3

postgres

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

Create > Database...

Refresh...

Server sessions

7
4
3
2
1
0

Tuples in

1

Create - Database

General

Definition

Security

Parameters

Advanced

SQL

Database

Books

Owner

postgres

Comment

i

?

Cancel

Reset





Save

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**.

pgAdmin File ▾ Object ▾ Tools ▾ Help ▾

Browser     **Dashboard** Properties SQL Statistics

▼ Servers (1)

- ▼ postgres
 - ▼ Databases (2)
 - ▼ **Books** ¹
 - > Casts
 - > Catalogs
 - > Event Triggers
 - > Extensions
 - > Foreign Data Wrappers
 - > Languages
 - > Publications
 - ▼ Schemas (1) ²
 - ▼ **public** ³
 - > Collations
 - > Domains
 - > FTS Configurations
 - > FTS Dictionaries
 - > FTS Parsers
 - > FTS Templates
 - > Foreign Tables
 - > Functions
 - > Materialized Views
 - > Procedures
 - > Sequences
 - > **Tables** ⁴
 - Create > **Table...**
 - Refresh...
 - Grant Wizard...
 - Search Objects...
 - Query Tool
 - > Trigger
 - > Types
 - > Views
 - > Subscriptions
 - > postgres
 - > Login/Group Roles
 - > Tablespaces

Database sessions

1

0

Tuples in ■ Insert

1

0

Server activity

Sessions Locks Prepared Transactions

	PID	User	Applic
			pgAdi

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 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 key?
		<div></div>	<div>Select an item...</div>		<div>No</div>	<div>No</div>
		<div></div>	<div>Select an item...</div>		<div>No</div>	<div>No</div>
		<div></div>	<div>Select an item...</div>		<div>No</div>	<div>No</div>
		<div></div>	<div>Select an item...</div>		<div>No</div>	<div>No</div>

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?
		<div>author_id</div>	<div>integer</div>		<div>Yes</div>	<div>Yes</div>
		<div>first_name</div>	<div>character varying</div>	<div>100</div>	<div>No</div>	<div>No</div>
		<div>middle_name</div>	<div>character varying</div>	<div>50</div>	<div>No</div>	<div>No</div>
		<div>last_name</div>	<div>character varying</div>	<div>100</div>	<div>No</div>	<div>No</div>

i

?

Cancel

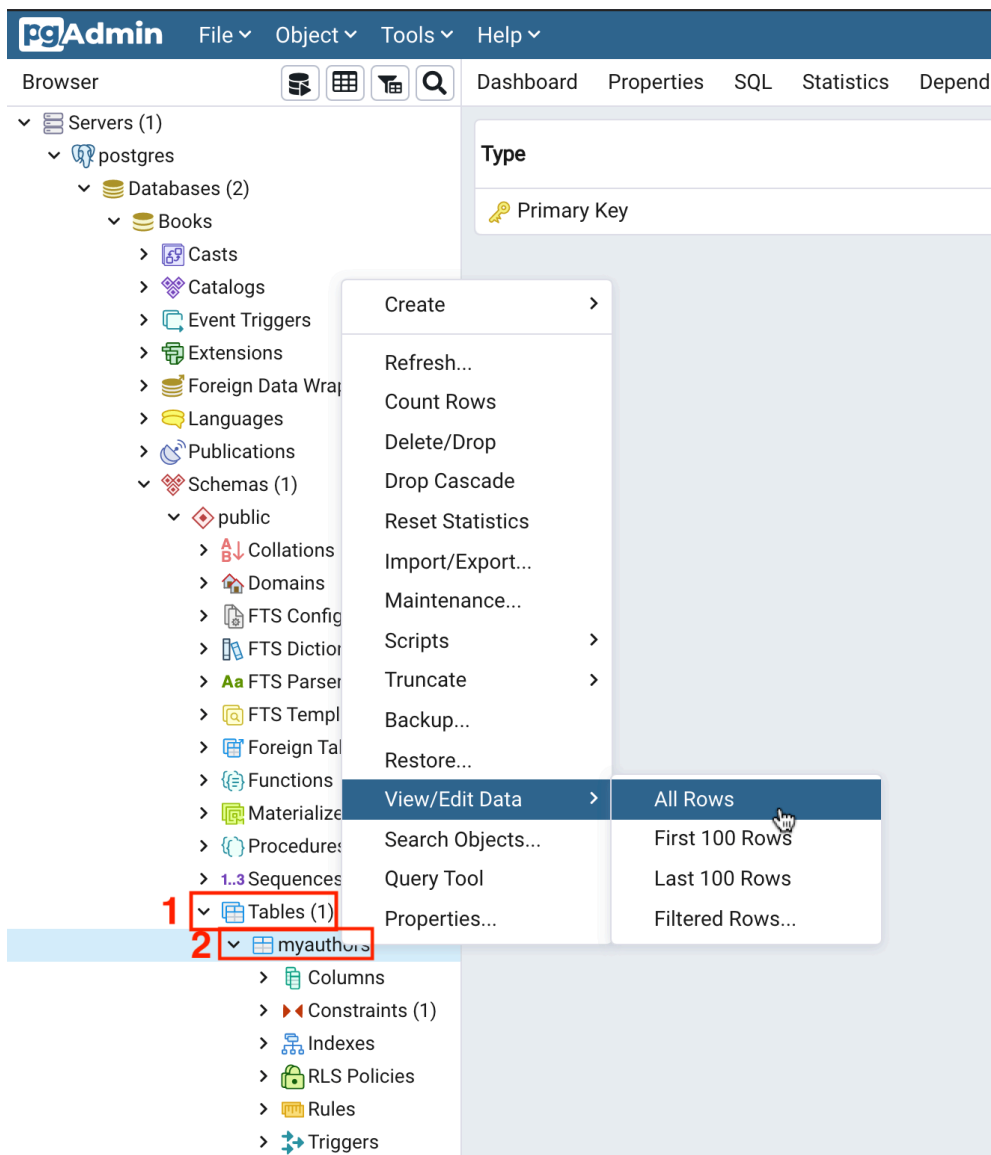
Reset

Save

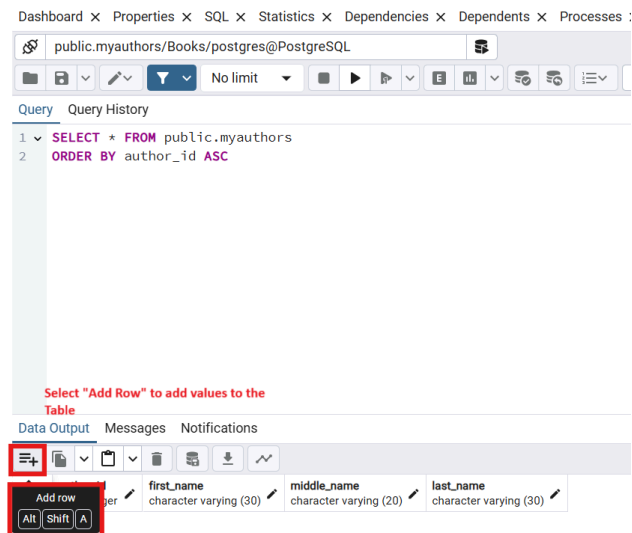
Task C: Load data into tables manually using the pgAdmin GUI

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**.

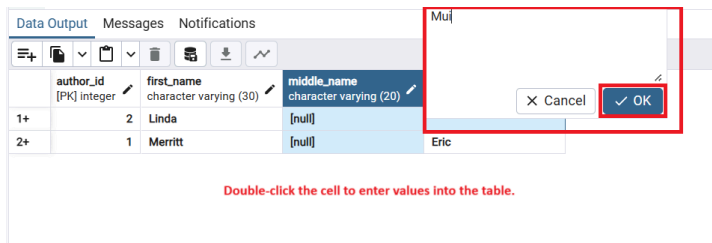


2. You will insert 2 rows of data into the **myauthors** table. In the lower **Data Output** pane, enter **myauthors** table data information for 2 rows as shown in the highlighted boxes in the image below. Then click the **Save Data Changes** icon. Proceed to Task D.

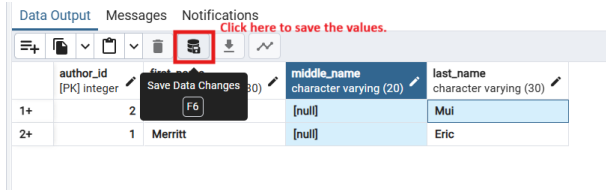


3. Enter the values into the table as shown below:

	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



4. Save the values.



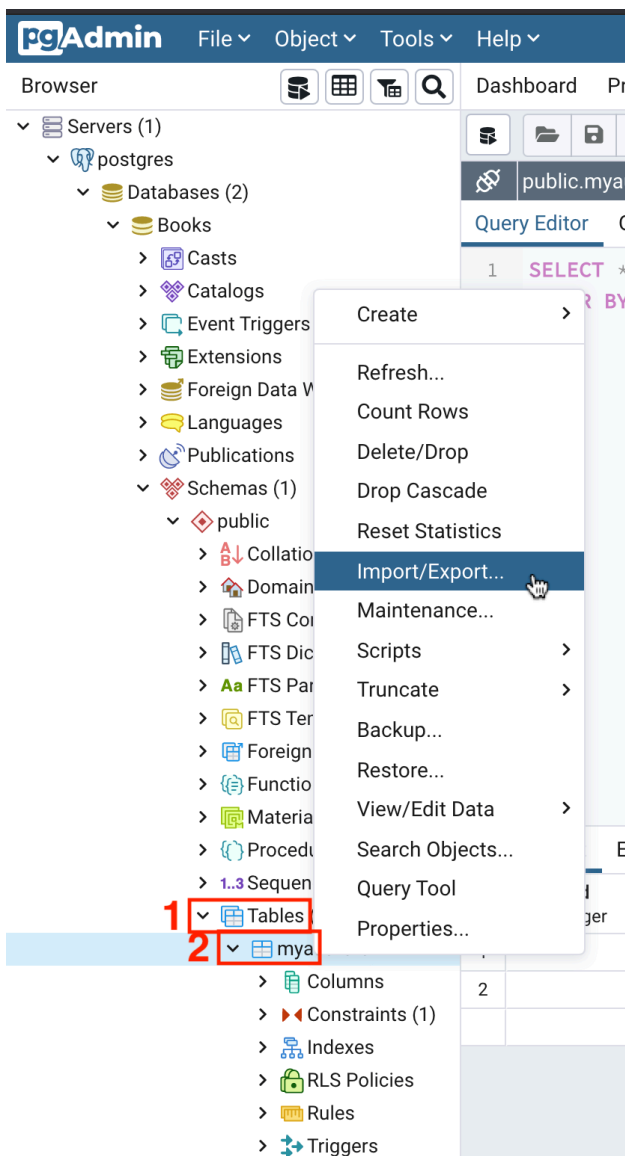
Task D: Load data into tables using a text/script file

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:

- [myauthors.csv](#)

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



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.

Import/Export data - table 'myauthors'

General
Options
Columns

Import/Export

Import

Export

Filename

Format

csv

Encoding

Select an item...

Close

Reset

OK

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

Select file

Home

Up

Search

Name	Date Modified	Size
bin	Thu Sep 5 20:08:53 2024	
sbin	Mon Jul 29 05:02:19 2024	
srv	Mon Jul 22 14:34:18 2024	
sys	Thu Sep 5 20:08:48 2024	
tmp	Thu Sep 5 20:09:17 2024	
usr	Mon Jul 29 05:02:18 2024	
var	Mon Jul 29 05:02:20 2024	
venv	Mon Jul 29 04:58:51 2024	

21 items
File Format All Files

Cancel

Select

- Step 2: Select lib folder.

Select file

Home

Up

/var

Search

Name	Date Modified	Size
cache	Mon Jul 22 14:34:18 2024	
db	Mon Jul 29 05:02:20 2024	
empty	Mon Jul 22 14:34:18 2024	
lib	Mon Jul 29 05:02:26 2024	
local	Mon Jul 22 14:34:18 2024	
lock	Mon Jul 22 14:34:18 2024	
log	Mon Jul 22 14:34:18 2024	
mail	Mon Jul 22 14:34:18 2024	

12 items
File Format All Files

Cancel

Select

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

Select file

Home

Up

/var/lib

Search

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

Home

Up

/var/lib/pgadmin

Search

Name	Date Modified	Size
azurecredentialcache	Thu Sep 5 20:08:53 2024	
pgadmin4.db	Fri Sep 6 01:04:34 2024	164.0 kB
sessions	Thu Sep 5 23:43:26 2024	
storage	Thu Sep 5 20:08:53 2024	

4 items
File Format All Files

Cancel

Select

Rename

Delete

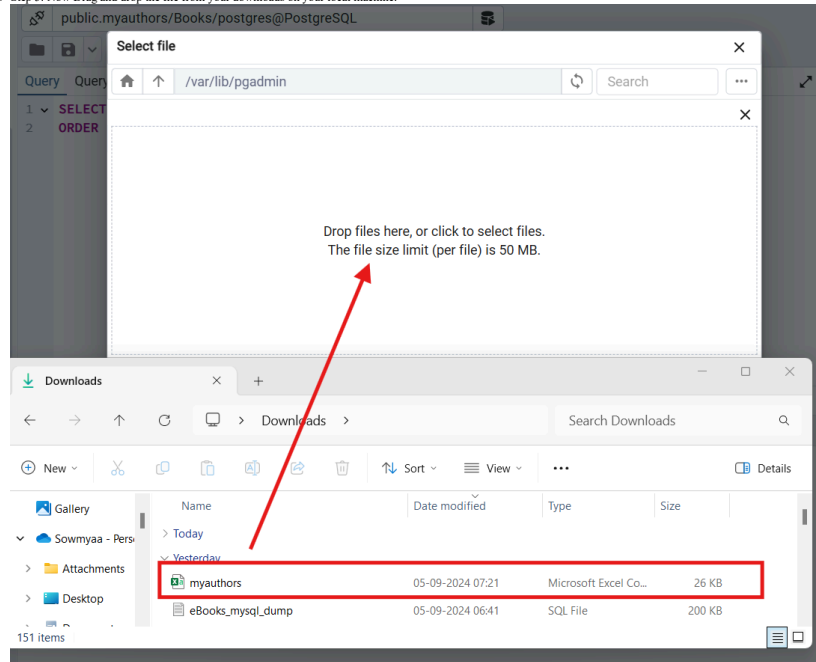
Upload

List View

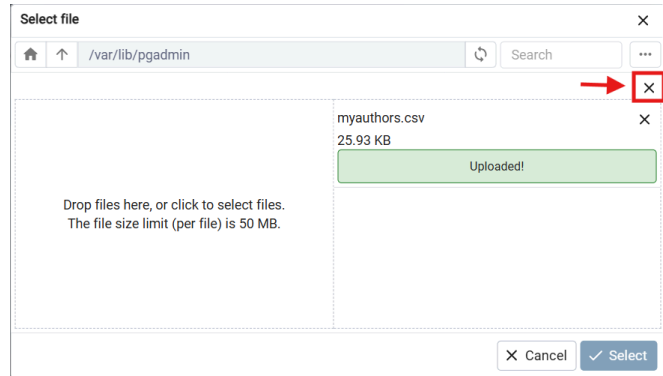
Grid View

Show Hidden Files

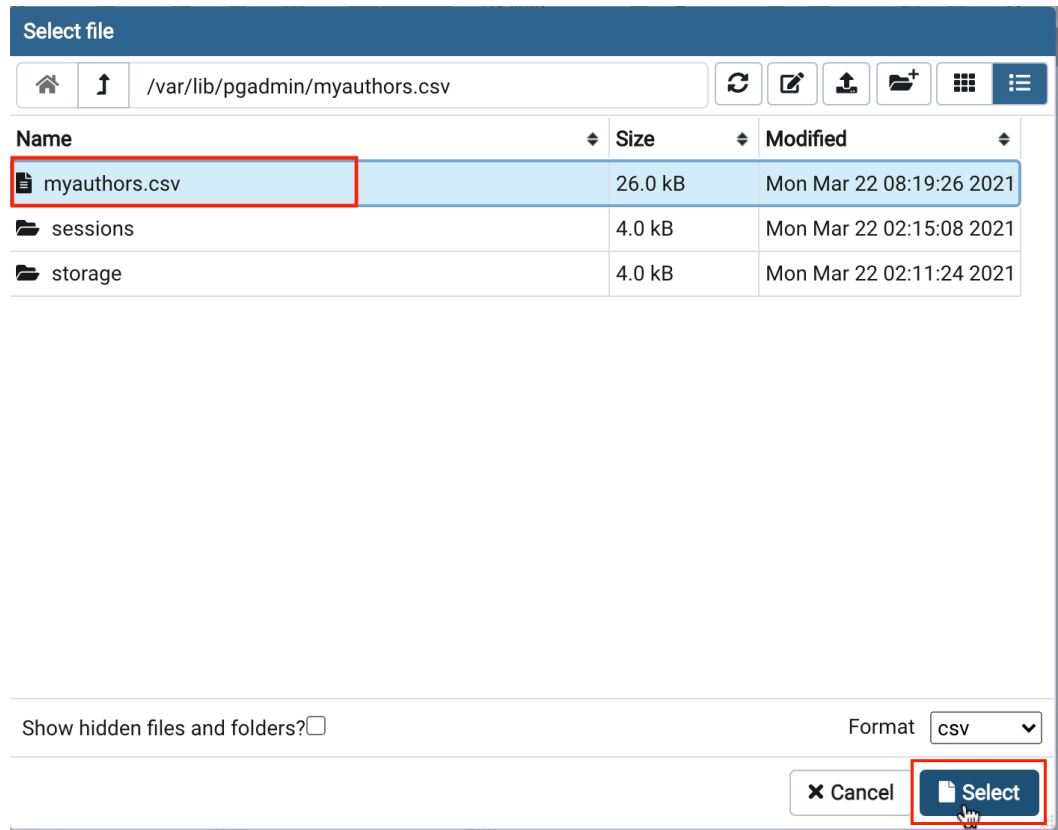
Step 5: Now Drag and drop the file from your downloads on your local machine.



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**.



- o Ensure the file has selected.

Import/Export data - table 'myauthors'

General Options Columns

Import/Export

Filename

Format

Encoding

Close Reset OK

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

Import/Export data - table 'myauthors'

General Options Columns

OID ☐

Header ☒

Delimiter
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.

Quote
Specifies the quoting character to be used when a data value is quoted.

Close Reset OK

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

public.myauthors/Books/postgres@PostgreSQL

Query Query History Scratch Pad

1 SELECT * FROM public.myauthors

2 ORDER BY author_id ASC

Data Output Messages Notifications

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

Process completed

Copying table data 'public.myauthors' on database 'Books' and server 'PostgreSQL (labs-postgres-quiet-low-mechanic:5432)'

View Processes






Process started

Copying table data 'public.myauthors' on database 'Books' and server 'PostgreSQL (labs-postgres-quiet-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.

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

	 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	

Conclusion

Author

- [Sandip Saha Joy](#).

Other Contributors

- [David Pasternak](#)



Skills Network