

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

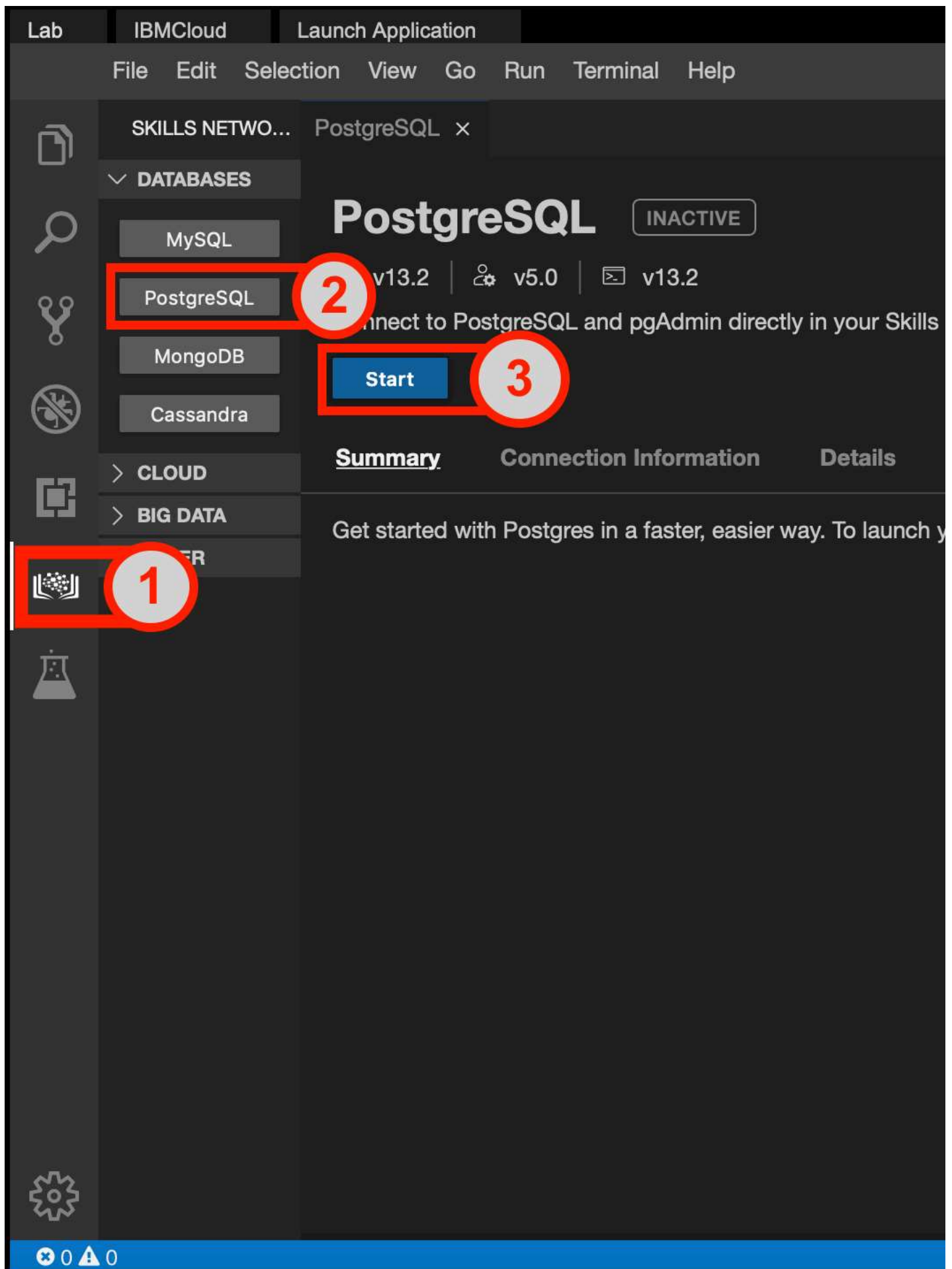
Lab Structure

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.

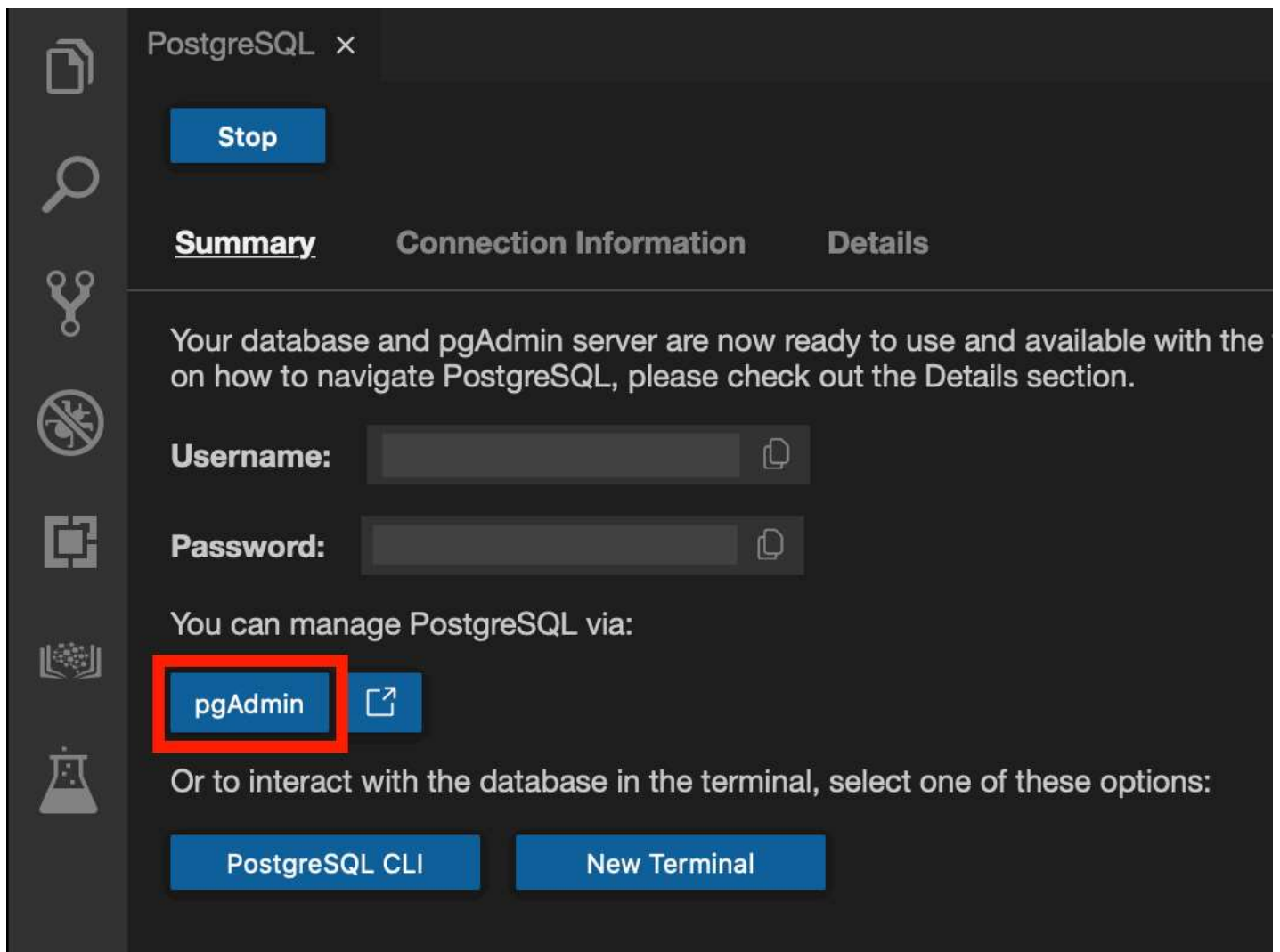
Task A: Create a database

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.

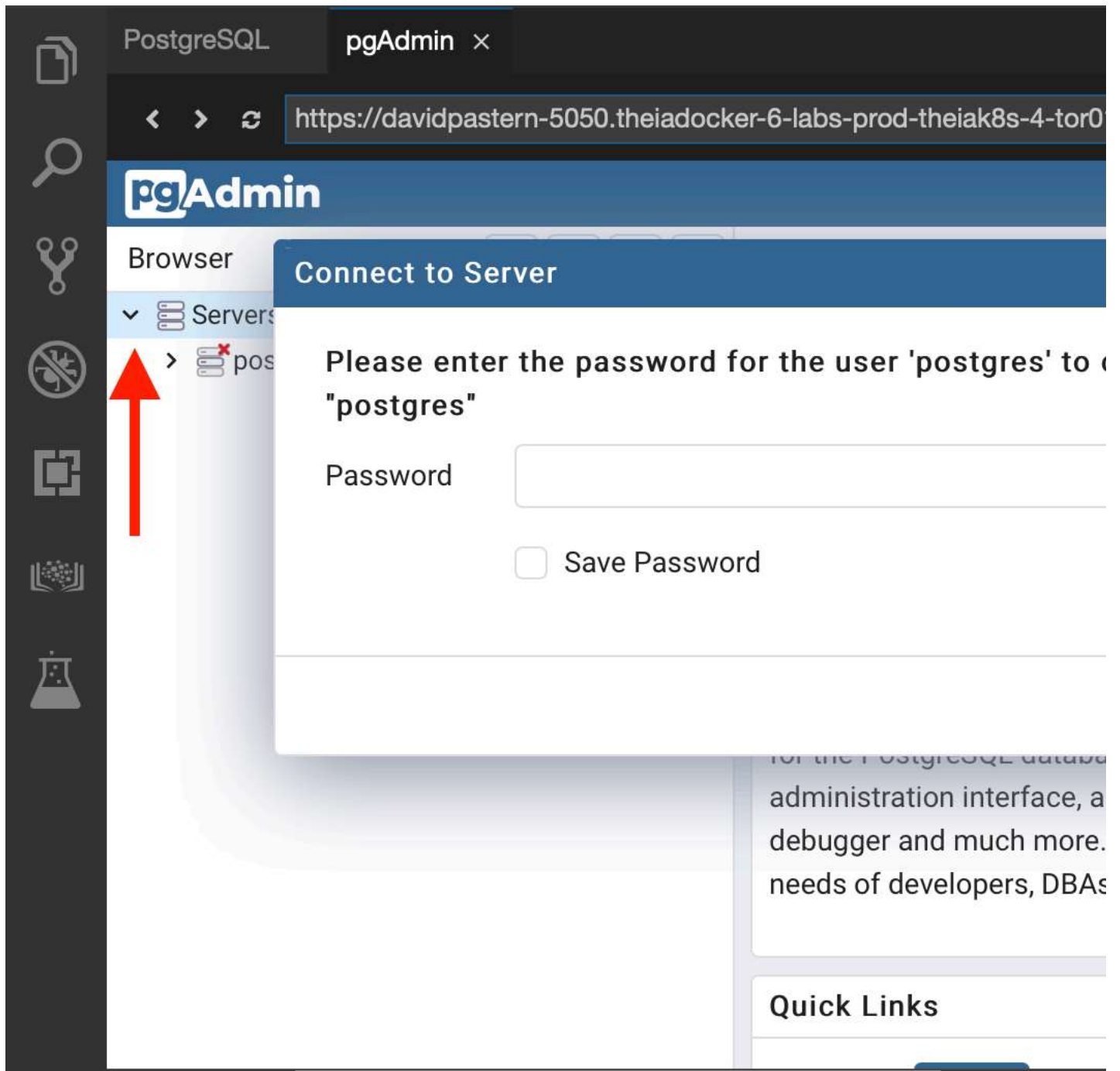
1. Click SKILLS NETWORK extension on the left side of the window.
2. Open the **DATABASES** drop down menu and click **PostgreSQL**.
3. Click **Start**. PostgreSQL may take a few moments to start.



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.

7. Click the Copy icon on the left of your password to copy the session password onto your clipboard.

PostgreSQL x pgAdmin

PostgreSQL

ACTIVE

v13.2 | v5.0 | v13.2

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

Stop

Summary Connection Information Details

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

Username:

Password:

You can manage PostgreSQL via:

pgAdmin [Link](#)

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

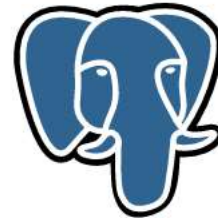
PostgreSQL CLI New Terminal

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.



Welcome



pgAdmin
Management

Feature rich | Maximise

pgAdmin is an Open Source admin
is designed to answer the needs o

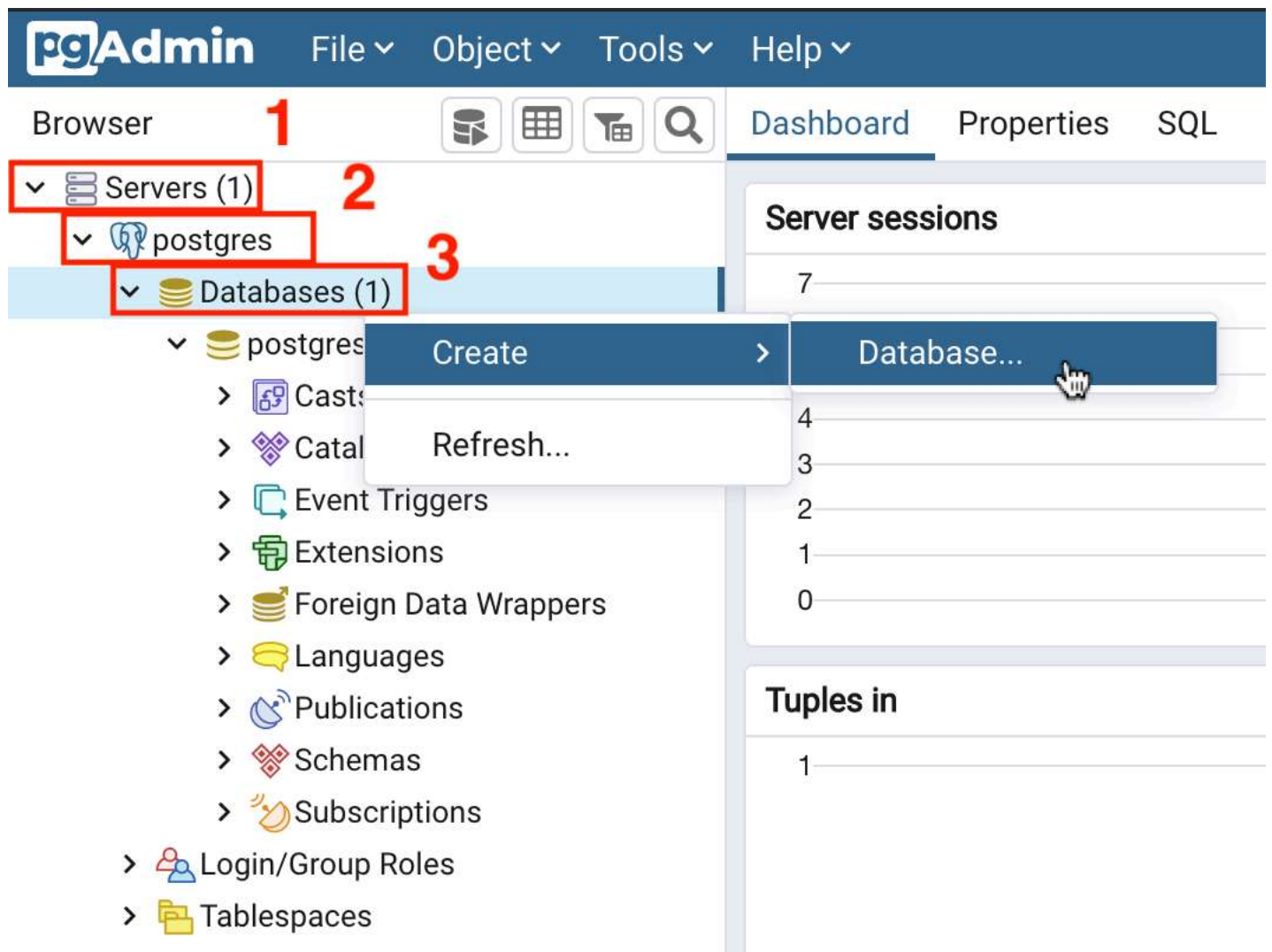
Quick Links


Getting Started



PostgreSQL Documen

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.



 **Create - Database**

General

Definition

Security

Parameters


Advanced

SQL

Database

Books

Owner

 postgres

Comment

i

?

✕ Cancel

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**.

Browser



Dashboard

Properties

SQL

▾ 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

> 1..3 Sequences

4 > Tables

> Trigger

> Types

> Views

> Subscriptions

> postgres

> Login/Group Roles

Database sessions

1

0

Tuples in

1

0

Server activity

Sessions

Locks

Prepared Transactions

PID

User

Create




Table...

Refresh...

Grant Wizard...

Search Objects...

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


Name

myauthors

Owner

 postgres

Schema

 public



Tablespace


Select an item...

Partitioned table?

☒ No

Comment

 Cancel

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 Secur

Inherited from table(s)

Select to inherit from...

Columns

		Name ▲	Data type	Length/Precision	Scale	N
		<input type="text"/>	Select an item... ▼			
		<input type="text"/>	Select an item... ▼			
		<input type="text"/>	Select an item... ▼			
		<input type="text"/>	Select an item... ▼			

i

?

✕ Cancel

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

Inherited from table(s)

Select to inherit from...

Columns

		Name	Data type	Length/Precision	Scale	N
		author_id	integer			
		first_name	character varying	100		
		middle_name	character varying	50		
		last_name	character varying	100		

i

?

Cancel

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**.

Browser



Dashboard

Properties

SQL

▾ Servers (1)

▾ postgres

▾ Databases (2)

▾ Books

> Casts

> Catalogs

> Event Triggers

> Extensions

> Foreign Data Wraps

> Languages

> Publications

▾ Schemas (1)

▾ public

> Collations

> Domains

> FTS Config

> FTS Dictionaries

> FTS Parsers

> FTS Templates

> Foreign Tables

> Functions

> Materialized Views

> Procedures

> Sequences

1 ▾ Tables (1)

2 ▾ myauthors

> Columns

> Constraints (1)

> Indexes

> RLS Policies

> Rules

Type

🔑 Primary Key

Create >

Refresh...

Count Rows

Delete/Drop

Drop Cascade

Reset Statistics

Import/Export...

Maintenance...

Scripts >

Truncate >

Backup...

Restore...

View/Edit Data >

Search Objects...

Query Tool

Properties...

All Rows

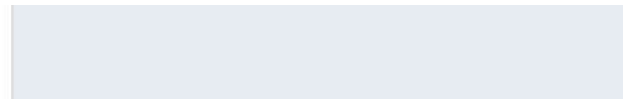
First 100 Rows

Last 100 Rows

Filtered Rows...

>  Rules

>  Triggers



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.

Dashboard Properties SQL Statistics Dependencies Dependents

Save Data Changes icon

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, 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**.

pgAdmin File Object Tools Help

Browser Dashboard Pr

- Servers (1)
 - postgres
 - Databases (2)
 - Books
 - Cast
 - Catalogs
 - Event Triggers
 - Extensions
 - Foreign Data V
 - Languages
 - Publications
 - Schemas (1)
 - public
 - Collatio
 - Domain
 - FTS Co
 - FTS Dic
 - FTS Par
 - FTS Ter
 - Foreign
 - Funcio
 - Materia
 - Procedu
 - 1..3 Sequen
 - 1 Tables
 - 2 mya
 - Columns
 - Constraints (1)
 - Indexes
 - RLS Policies

Create

Refresh...

Count Rows

Delete/Drop

Drop Cascade

Reset Statistics

Import/Export...

Maintenance...

Scripts

Truncate

Backup...

Restore...

View/Edit Data

Search Objects...

Query Tool

Properties...

- >  Rules
- >  Triggers

3. Follow the instructions below to import:


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

Import/Export data - table 'myauthors'

Options Columns

Import/Export ☒ Import **1**

File Info

Filename 

Format **2**

csv

Encoding

Select an item...

Miscellaneous

OID ☐ No

Header **3**

Yes

Delimiter

Select from list...

Specifies the character that separates columns within file. The default is a tab character in text format, a comma in binary format. This option is not available in binary format.

- Click **Upload File**.

Select file


  


Name	Size
 sessions	4.0 kB
 storage	4.0 kB

Show hidden files and folders? ☐


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

Select file





/var/lib/pgadmin/





Double click on this space

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


Show hidden files and folders?☐


- When the upload is complete, close the drop files area by clicking X.

Select file



/var/lib/pgadmin/



26.6 KB

myauthors.csv

100%

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

Show hidden files and folders?☐




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

Select file



/var/lib/pgadmin/myauthors.csv



Name	Size	
 myauthors.csv	26.0 kB	
 sessions	4.0 kB	
 storage	4.0 kB	

Show hidden files and folders? ☐

- Click **OK** and notification of import success will 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 file. The default is a tab character in text format, a comma in binary format. This option is not available in binary format.

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



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.

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]		Mu
3		3	Alecos		[null]		Pap
4		4	Paul		C.van		Oor
5		5	David		[null]		Cro
6		6	Richard		[null]		Blu
7		7	Yuval		Noah		Har
8		8	Paul		[null]		Alb
9		9	David		[null]		Bea
10		10	John		Paul		She
11		11	Andrew		[null]		Mill
12		12	Melanie		[null]		Swi
13		13	Neal		[null]		For
14		14	Nir		[null]		Sha
15		15	Tim		[null]		Kin
16		16	Mike		[null]		Mc
17		17	Brian		P.		Hog
18		18	Jean-Philippe		[null]		Aur
19		19	Lance		[null]		For

19	19	John	[null]	Ho
20	20	Richard	C.	Jef
21	21	William	L.	Sim
22	22	Magnus	Lie	Het
23	23	Mike	[null]	Mc
24	24	Norman	[null]	Ma
25	25	John	E.	Ho
26	26	S.	[null]	Suc

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](#)

© IBM Corporation. All rights reserved.