



# Hands-on Lab : Views in PostgreSQL

**Estimated time needed:** 15 minutes

In this lab, you will learn how to create and execute views and materialized views in the PostgreSQL database service using the pgAdmin graphical user interface (GUI) tool. Materialized views behave differently compared to regular views. In materialized views, the result set is materialized, or saved for future use. You can't insert, update, or delete rows like in regular views. Essentially, materialized views store the results of a database query as a separate table-like object so that the query results can be accessed at a later time without having to re-run the query. As a result, materialized views can improve database performance compared to regular views.

## Software Used in this Lab

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

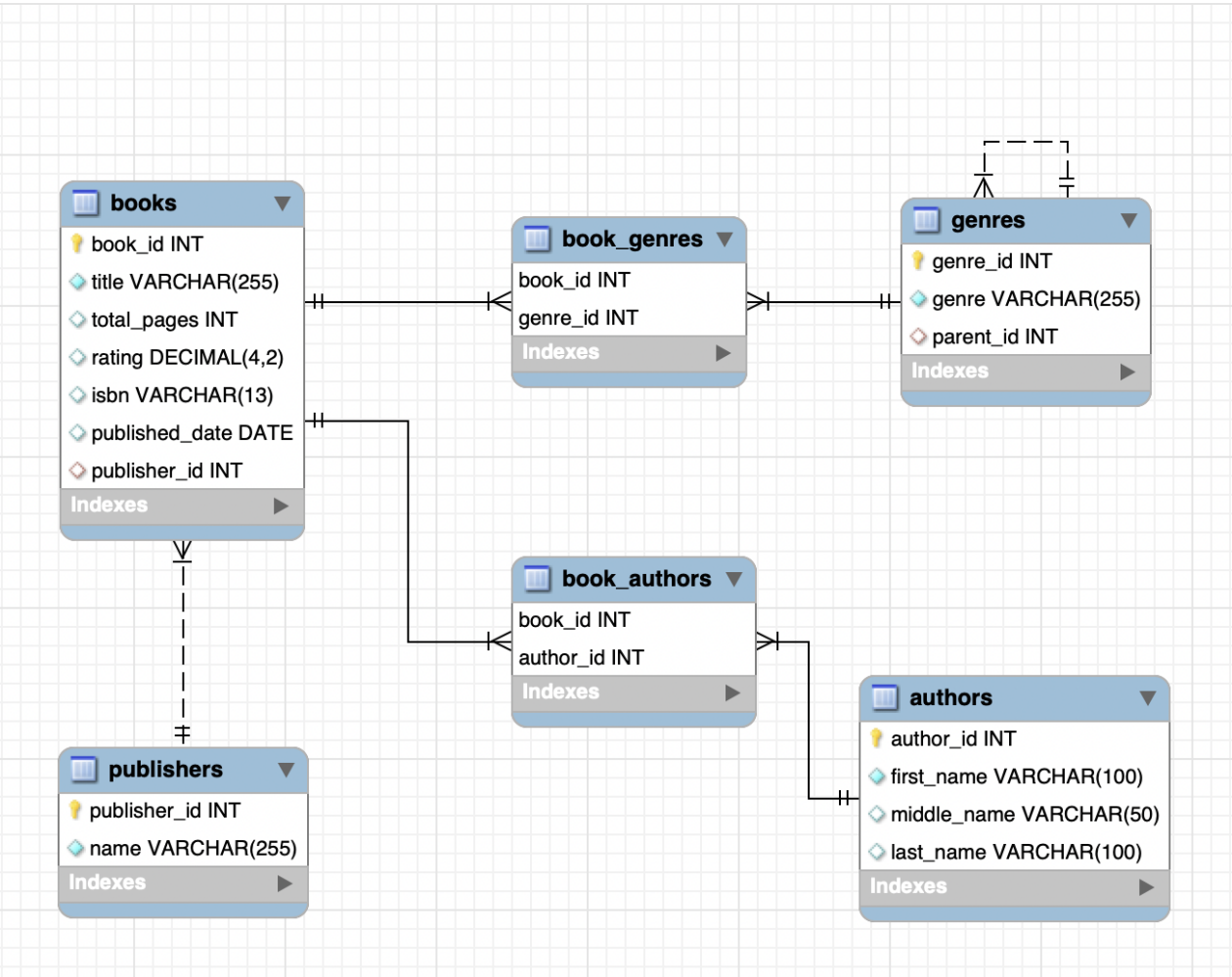


To complete this lab you will utilize the PostgreSQL relational database service available as part of IBM Skills Network Labs (SN Labs) Cloud IDE. SN Labs is a virtual lab environment used in this course.

## Database Used in this Lab

The eBooks database has been used in this lab.

The following ERD diagram shows the schema of the complete eBooks database used in this lab:



# Objectives

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

- Restore a database schema and data.
- Create and execute a view.
- Create and execute a materialized view.

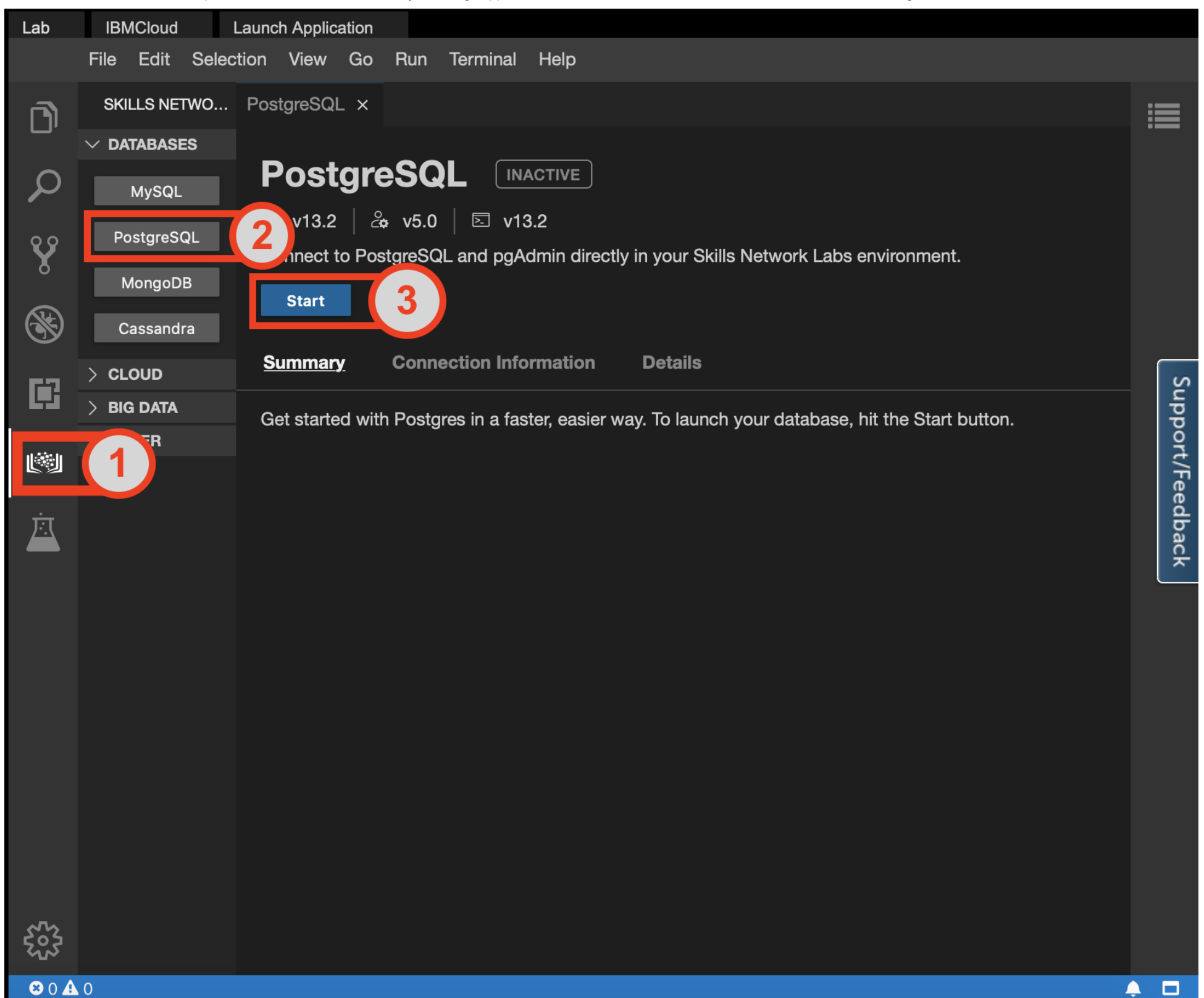
## Lab Structure

In this exercise, you will go through three tasks where you will learn how to create and execute views and materialized views in the PostgreSQL database service using the pgAdmin graphical user interface (GUI) tool.

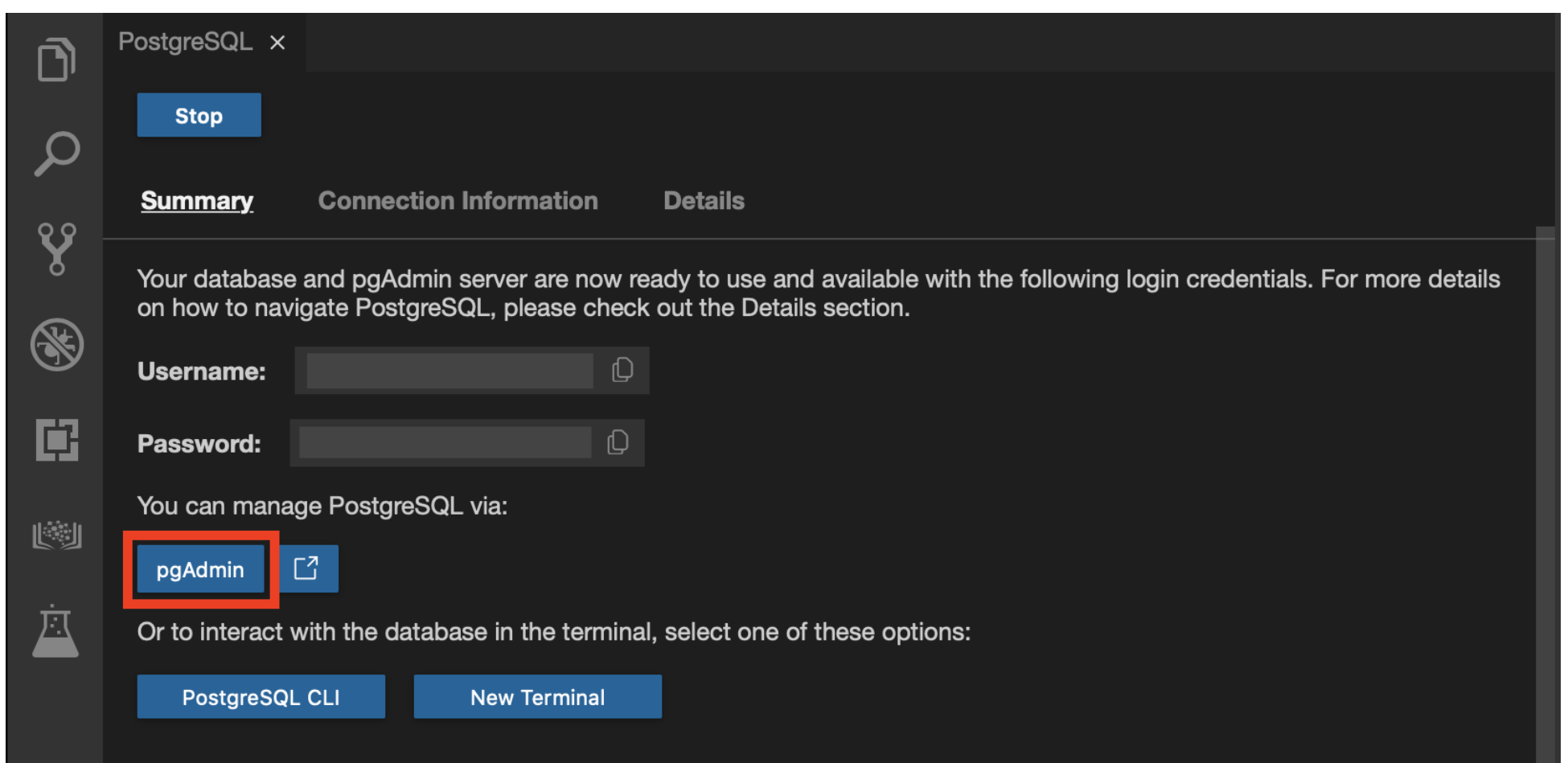
## Task A: Restore a database schema and data

To get started with this lab, you will first download the relevant **eBooks** database dump file, then launch PostgreSQL and pgAdmin using the Cloud IDE. You can do this by following these steps:

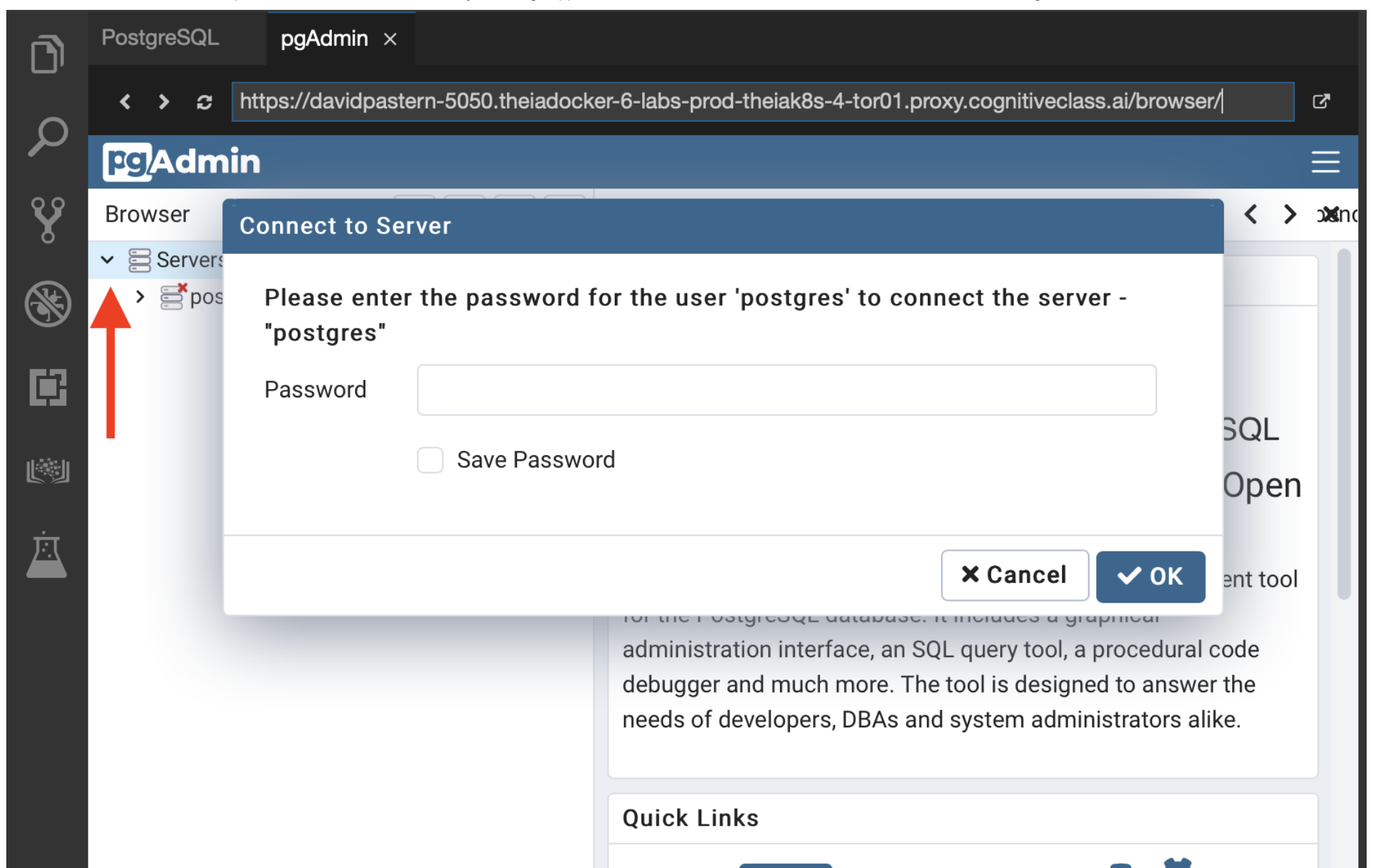
1. Download the **eBooks** PostgreSQL dump file (containing the eBooks database schema and data) below to your local computer storage.
  - [eBooks\\_pgsql\\_dump.tar](#)
2. Click on the Skills Network extension button on the left side of the window.
3. Open the "DATABASES" drop down menu and click on "PostgreSQL"
4. Click on the "Start" button. PostgreSQL may take a few moments to start.



5. Next, open the pgAdmin Graphical User Interface by clicking the "pgAdmin" button in the Cloud IDE interface.

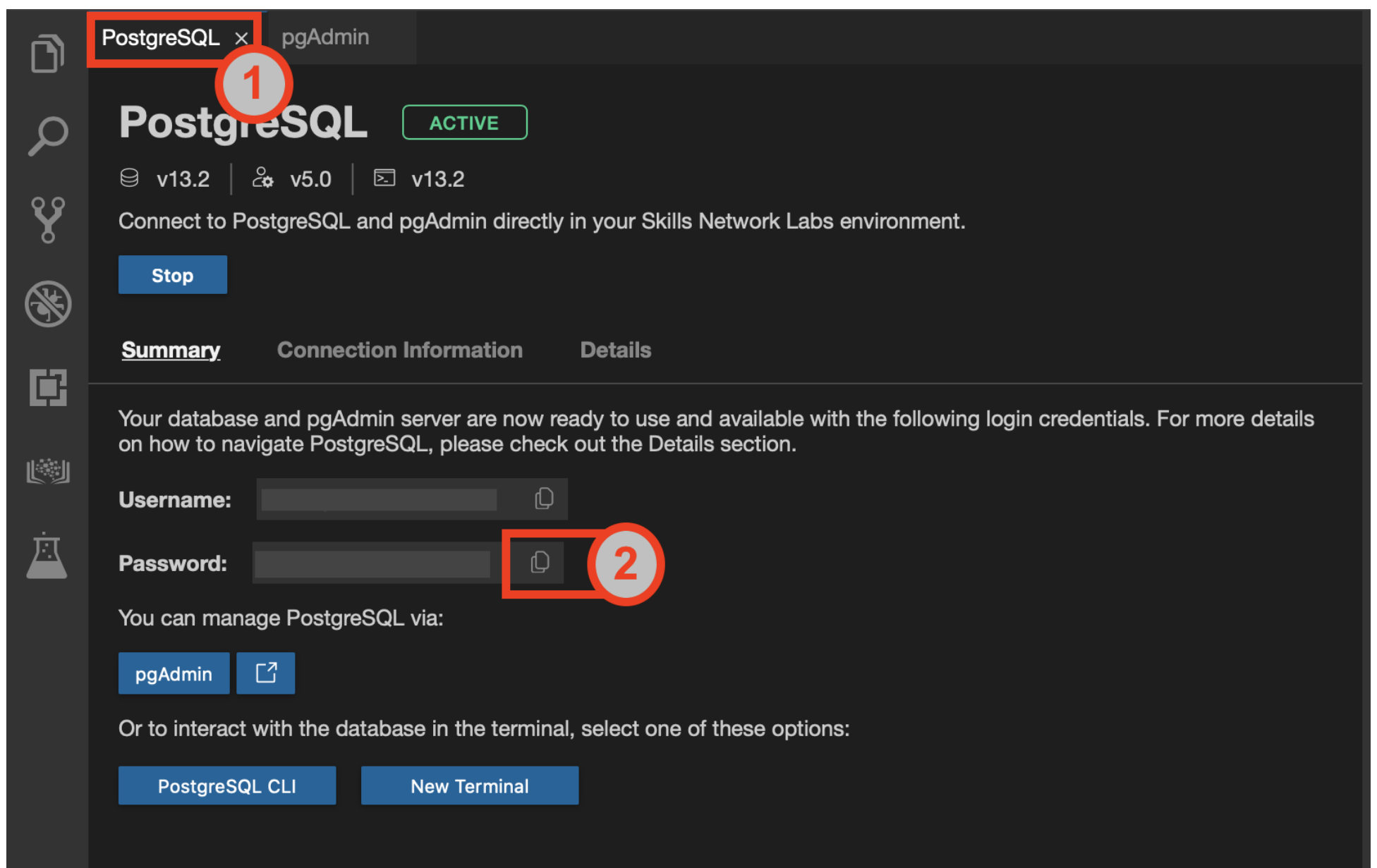


6. Once the pgAdmin GUI opens, click on the **Servers** tab on the left side of the page. You will be prompted to enter a password.



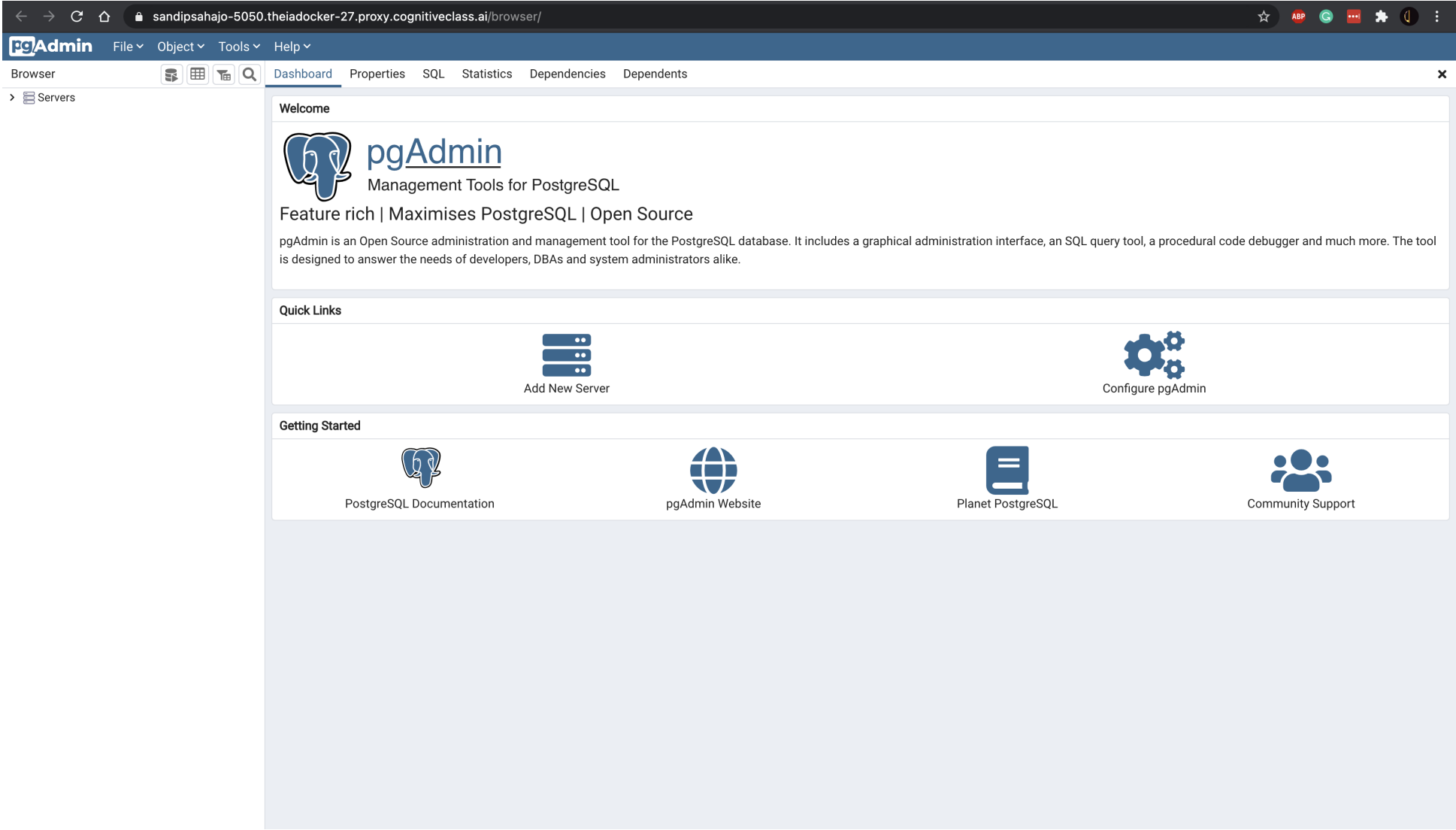
7. To retrieve your password, click on the "PostgreSQL" tab near the top of the interface.

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

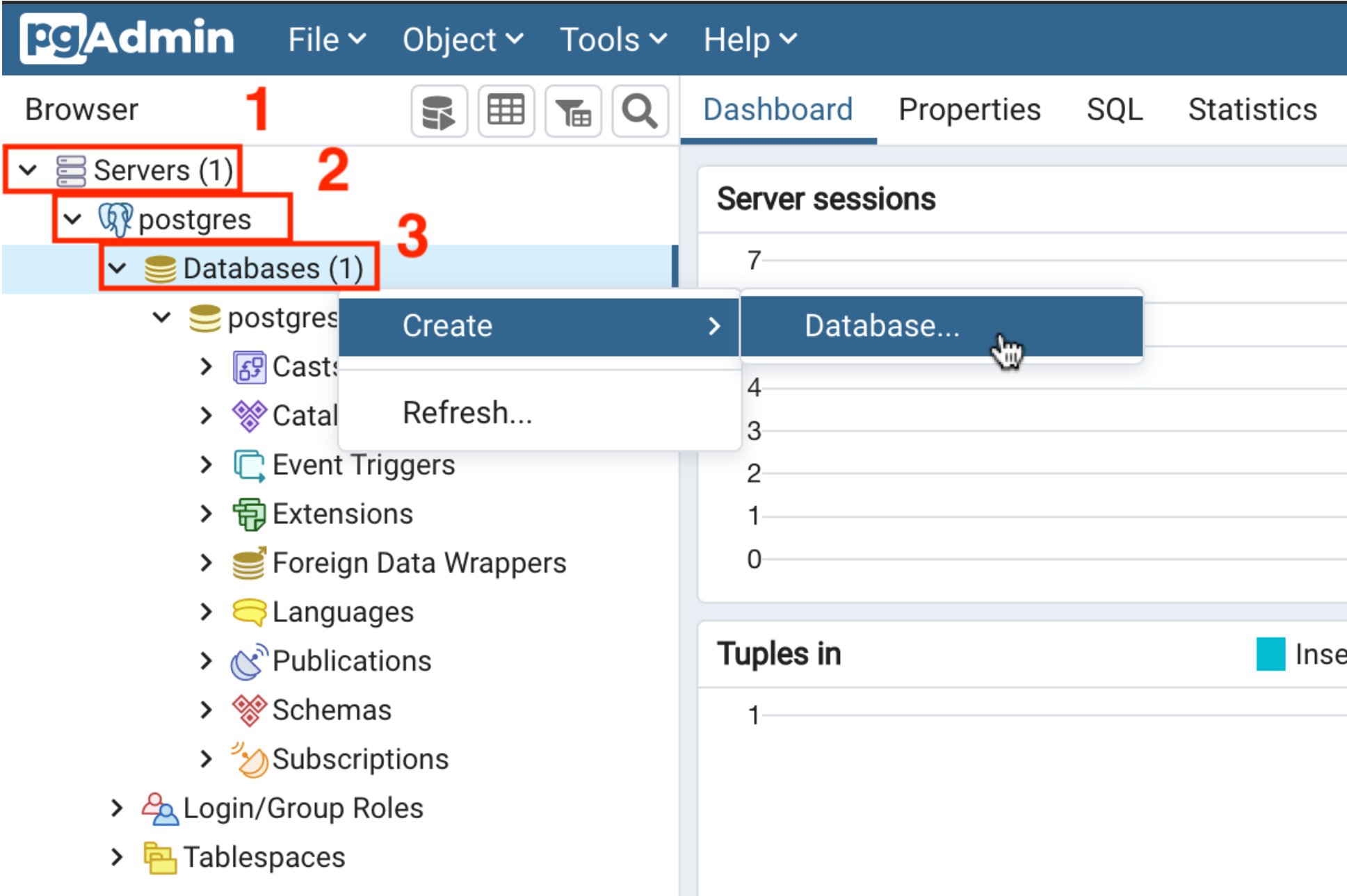



9. Navigate back to the "pgAdmin" tab and paste in your password, then click **OK**

10. You will then be able to access the pgAdmin GUI tool.



11. In the tree-view, expand **Servers** > **postgres** > **Databases**. Enter your PostgreSQL service session password if prompted during the process. Right-click on **Databases** and go to **Create** > **Database**. Type **eBooks** as name of the database and click **Save**.



 **Create - Database** ✕

General

Definition

Security

Parameters

Advanced


SQL



Database

Owner


Comment


eBooks

 postgres

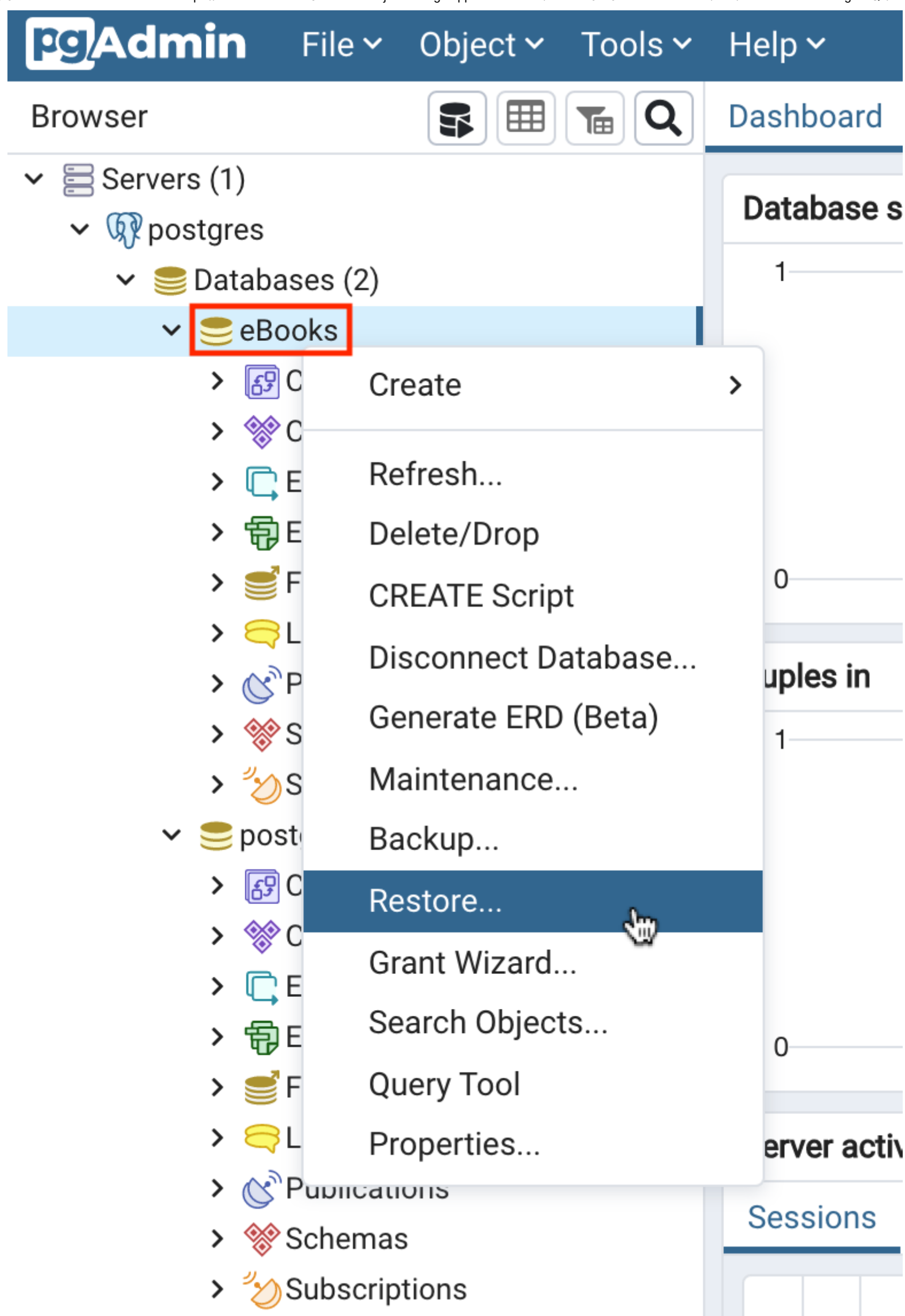
✕ Cancel

 Reset

 Save

12. In the tree-view, expand **eBooks**. Right-click on **eBooks** and select **Restore**.





13. Follow the instructions below to restore and proceed to Task B:

- On the **General** tab, click on the **Select file** button by the Filename box.

Restore (Database: eBooks)

General

Restore options

Format

Custom or tar

Filename

...

Number of jobs

Role name

Select an item...

i

?

✕ Cancel

Restore

- Click the **Upload File** button.

Select file

🏠

↑

/var/lib/pgadmin/

↺

✎

⬆

+

⌵

⌵

Name	Size	Modified
<div>📁 sessions</div>	4.0 kB	Mon Mar 29 10:20:20 2021
<div>📁 storage</div>	4.0 kB	Mon Mar 29 10:04:10 2021

Show hidden files and folders?☐

Format 

backup

✕ Cancel

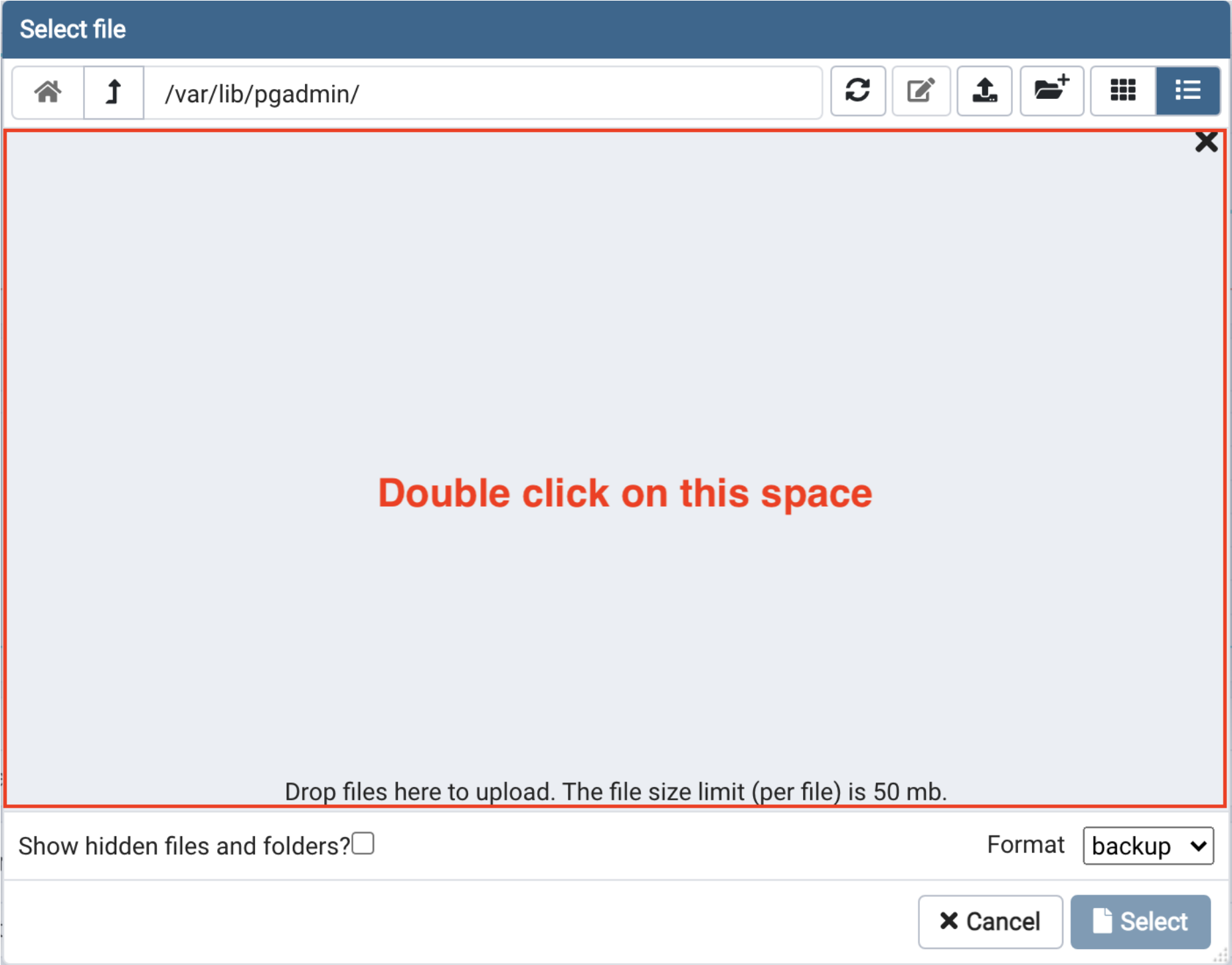
Select

- Double-click on the drop files area and load the **eBooks\_pgsql\_dump.tar** you downloaded earlier from your local computer storage.

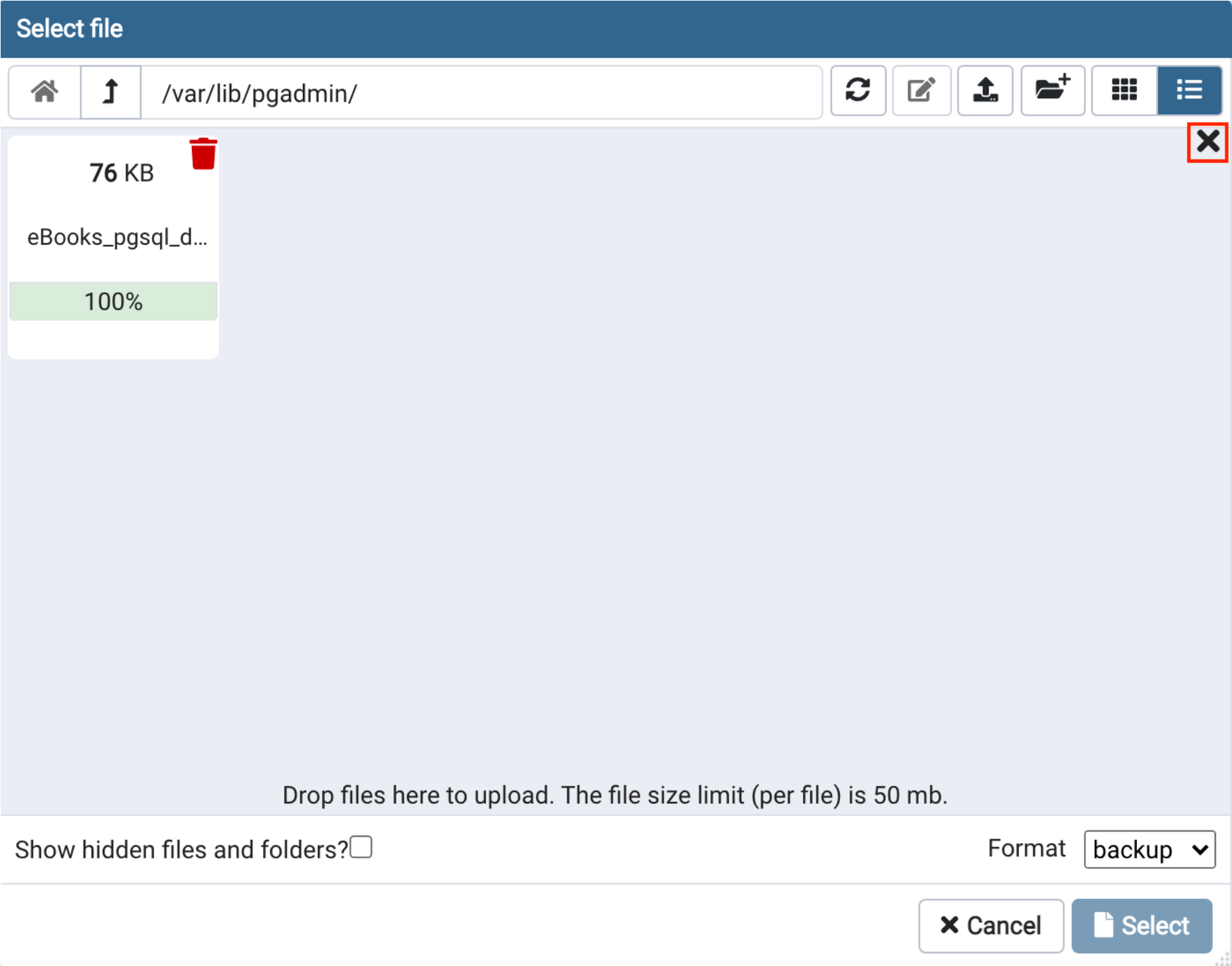
https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB0110EN-SkillsNetwork/labs/Lab - Views in PostgreSQL/instructional-labs.md.html

8/23





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



- Make sure Format is set to **All Files**, select the uploaded **eBooks\_pgsql\_dump.tar** file from the list, and then click the **Select** button.

Select file

/var/lib/pgadmin/eBooks\_pgsql\_dump.tar

Name	Size	Modified
<div><div></div>eBooks_pgsql_dump.tar</div>	74.2 kB	Mon Mar 29 10:18:23 2021
<div><div></div>pgadmin4.db</div>	156.0 kB	Mon Mar 29 10:15:25 2021
<div><div></div>sessions</div>	4.0 kB	Mon Mar 29 10:04:48 2021
<div><div></div>storage</div>	4.0 kB	Mon Mar 29 10:04:10 2021

Show hidden files and folders?☐

Format

All Files

✕ Cancel

Select

- Now switch to **Restore options** tab.

Restore (Database: eBooks)

General

Restore options

Format

Custom or tar

Filename

/var/lib/pgadmin/eBooks\_pgsql\_dump.tar

Number of jobs

Role name

Select an item...

i

?

✕ Cancel

Restore

- Under Disable, set the Trigger option to **Yes**. Then click **Restore** button.

https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB0110EN-SkillsNetwork/labs/Lab - Views in PostgreSQL/instructional-labs.md.html

10/23

Restore (Database: eBooks)

General

Restore options

Queries

Include CREATE DATABASE statement

No

Clean before restore

No

Single transaction

No

Disable

Trigger

Yes

No data for Failed Tables

No

i?

✕ Cancel

Restore

# Task B: Create and execute a view

1. In the tree-view, expand **eBooks** > **Schemas** > **public**. Right-click on **Views** and go to **Create** > **View**.

The screenshot shows the pgAdmin interface. The top navigation bar includes 'pgAdmin', 'File', 'Object', 'Tools', and 'Help'. Below this is a 'Browser' section with icons for server, table, view, and search. The main panel on the left displays a tree structure of database objects. The tree is expanded to show the 'public' schema under the 'eBooks' database. The 'Views' object is highlighted with a red box and the number 4. A context menu is open over the 'Views' object, showing options: 'Create', 'View...', 'Refresh...', 'Grant Wizard...', 'Search Objects...', and 'Query Tool'. The 'Create' option is highlighted with a red box and the number 5, and the 'View...' option is highlighted with a red box and the number 6. The main panel on the right is empty.

pgAdmin File Object Tools Help

Browser Dashboard Properties SQL Statistics Depend

Servers (1)

- postgres
  - Databases (2)
    - 1** eBooks
      - Casts
      - Catalogs
      - Event Triggers
      - Extensions
      - Foreign Data Wrappers
      - Languages
      - Publications
      - 2** Schemas (1)
        - 3** public
          - Collations
          - Domains
          - FTS Configurations
          - FTS Dictionaries
          - FTS Parsers
          - FTS Templates
          - Foreign Tables
          - Functions
          - Materialized Views
          - Procedures
          - 1.3 Sequences
          - Tables (6)
          - Trigger Functions
          - Types
          - 4** Views


Subscriptions

- postgres
  - Casts
  - Catalogs
  - Event Triggers
  - Extensions
  - Foreign Data Wrappers
  - Languages

**5** Create > **6** View...

Refresh...  
Grant Wizard...  
Search Objects...  
Query Tool

2. On the **General** tab, type **publisher\_and\_rating\_view** as name of the view. Then switch to **Code** tab.


 **Create - View** ✕

**General** Definition Code Security SQL


Name

publisher\_and\_rating\_view



Owner

 postgres


Schema


 public

Comment

✕ Cancel

 Reset


 Save

3. On the **Code** tab, copy and paste the code below. Then click **Save**.

```
SELECT books.title, books.rating, publishers.name
FROM books INNER JOIN publishers ON books.publisher_id = publishers.publisher_id
```

https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB0110EN-SkillsNetwork/labs/Lab - Views in PostgreSQL/instructional-labs.md.html

13/23

 **Create - View** ✕

General

Definition

**Code**

Security

SQL

```
1 SELECT books.title, books.rating, publishers.name
2 FROM books INNER JOIN publishers ON books.publisher_id = publishers.publisher_id
3
```

**i**

**?**

✕ Cancel

↺ Reset

**Save**

4. In the tree-view, expand **Views**. Right-click on **publisher\_and\_rating\_view** and go to **View/Edit Data > All Rows**.

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

Browser Dashboard Properties SQL Statisti

- Servers (1)
  - postgres
    - Databases (2)
      - eBooks
        - 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 (6)
            - Trigger Functions
            - Types
            - Views (1)**
              - publisher\_and\_rating\_view**
                - Columns
                - Rules
                - Triggers

1 2

3 View/Edit Data > 4 All Rows

First 100 Rows  
Last 100 Rows  
Filtered Rows...

**Database sessions**

1

0

**Tuples in**

18  
16  
14  
12  
10  
8  
6  
4  
2  
0


**Server activity**

Sessions Locks Prepared Transacti

			PID	User	A
✖	■	▶	83	postgres	p

5. You will access the view you created. This allows you to actually access and view the contents of tables in your database.



 public.publisher\_and\_rating\_view/eBooks/postgres@postgres

Query Editor

Query History

1

2

```
SELECT * FROM public.publisher_and_rating_view
```

Data Output

Explain

Messages

Notifications

	<div>title</div> <div>character varying (255)</div>	<div>rating</div> <div>numeric (4,2)</div>	<div>name</div> <div>character varying (255)</div>
1	Lean Software Development: ...	4.17	Addison Wesley
2	Facing the Intelligence Explosi...	3.87	Machine Intelligence Researc...
3	Scala in Action	3.74	Manning
4	Patterns of Software: Tales fr...	3.84	Oxford University Press, USA
5	Anatomy Of LISP	4.43	McGraw-Hill
6	Computing machinery and int...	4.17	MSAC Philosophy Group
7	XML: Visual QuickStart Guide	3.66	Peachpit Press
8	SQL Cookbook	3.95	O'Reilly Media
9	The Apollo Guidance Comput...	4.29	Praxis Publications Inc
10	Minds and Computers: An Intr...	3.54	Edinburgh University Press
11	The Architecture of Symbolic ...	4.50	McGraw-Hill
12	Nmap Network Scanning: The...	4.32	Nmap Project
13	The It Handbook for Business:...	4.40	Createspace Independent Pub...
14	Accidental Empires	4.00	Harper
15	Introducing HTML5	3.97	New Riders Publishing


# Task C: Create and execute a materialized view

1. In the tree-view, expand **eBooks** > **Schemas** > **public**. Right-click on **Materialized Views** and go to **Create** > **Materialized View**.

The screenshot shows the pgAdmin interface with the following elements:

- Top Bar:** pgAdmin logo, File, Object, Tools, Help menus.
- Browser Panel:**
  - Servers (1)
    - postgres
      - Databases (2)
        - 1** eBooks
          - Casts
          - Catalogs
          - Event Triggers
          - Extensions
          - Foreign Data Wrappers
          - Languages
          - Publications
        - 2** Schemas (1)
          - 3** public
            - Collations
            - Domains
            - FTS Configurations
            - FTS Dictionaries
            - FTS Parsers
            - FTS Templates
            - Foreign Tables
            - Functions
            - 4** Materialized Views
              - 5** Create (context menu)
                - 6** Materialized View...
                - Refresh...
                - Grant Wizard...
                - Search Objects...
                - Query Tool
              - Procedures
              - 1.3 Sequences
              - Tables (6)
              - Trigger Functions
              - Types
              - Views (1)
              - Subscriptions- Right Panel:** Dashboard, Properties (selected), SQL, Statistics.

2. On the **General** tab, type **publisher\_and\_rating\_materialized\_view** as name of the view. Then switch to the **Definition** tab.

 **Create - Materialized View** ✕


General

DefinitionStorageParameterSecuritySQL


Name

publisher\_and\_rating\_materialized\_view



Owner

 postgres

Schema

 public

Comment

✕ Cancel ↺ Reset 💾 Save

3. On the **Definition** tab, copy and paste the code below. Then click **Save**.

```
SELECT books.title, books.rating, publishers.name
FROM books INNER JOIN publishers ON books.publisher_id = publishers.publisher_id
```

Create - Materialized View

General

Definition

Storage

Parameter

Security

SQL

1

2

3

SELECT books.title, books.rating, publishers.name

FROM books INNER JOIN publishers ON books.publisher\_id = publishers.publisher\_id

i

?

Cancel

Reset

Save

4. In the tree-view, expand **Materialized Views**. Right-click on **publisher\_and\_rating\_materialized\_view** and go to **Refresh View > With data**.

eBooks

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 (1)

publisher\_and\_rating\_materialized\_view

Columns

Indexes

Procedures

Sequences

Tables (6)

Trigger Functions

Types

Views (1)

publisher\_and\_

Columns

Rules

Triggers

Subscriptions

Create

Refresh...

Delete/Drop

Drop Cascade

Scripts

Refresh View

View/Edit Data

Search Objects...

Query Tool

Properties...

With data

With no data

With data (concurrently)

With no data (concurrently)

General

Name

OID

Owner

System materialized view?

Comment

Security

Privileges

Storage

Tablespace


Storage settings

5. Right-click on **publisher\_and\_rating\_materialized\_view** again and go to **View/Edit Data > All Rows**.

The screenshot shows the pgAdmin 4 web interface. The left pane displays a tree view of the database structure. Under 'Databases (2)', 'eBooks' is expanded, showing 'Schemas (1)' with 'public' selected. Under 'public', 'Materialized Views (1)' is expanded, and 'publisher\_and\_rating' is selected. A context menu is open over this view, listing actions like 'Create', 'Refresh...', 'Delete/Drop', 'Drop Cascade', 'Scripts', 'Refresh View', 'View/Edit Data', 'Search Objects...', 'Query Tool', and 'Properties...'. The 'View/Edit Data' option is highlighted, and a sub-menu is open showing 'All Rows', 'First 100 Rows', 'Last 100 Rows', and 'Filtered Rows...'. The 'All Rows' option is selected. The right pane shows the 'Properties' tab for the selected view, with sections for 'General', 'Security', and 'Storage'. The 'General' section includes fields for Name, OID, Owner, System materialized view?, and Comment. The 'Security' section includes Privileges. The 'Storage' section includes Tablespace and Storage settings.

6. You will access the materialized view you created.





public.publisher\_and\_rating\_materialized\_view/eBooks/postgres@postgres

Query Editor

Query History

1

2

```
SELECT * FROM public.publisher_and_rating_materialized_view
```

Data Output

Explain

Messages

Notifications

	<div>title</div> <div>character varying (255)</div>	<div>rating</div> <div>numeric (4,2)</div>	<div>name</div> <div>character varying (255)</div>	
1	Lean Software Development: ...	4.17	Addison Wesley	
2	Facing the Intelligence Explosi...	3.87	Machine Intelligence Researc...	
3	Scala in Action	3.74	Manning	
4	Patterns of Software: Tales fr...	3.84	Oxford University Press, USA	
5	Anatomy Of LISP	4.43	McGraw-Hill	
6	Computing machinery and int...	4.17	MSAC Philosophy Group	
7	XML: Visual QuickStart Guide	3.66	Peachpit Press	
8	SQL Cookbook	3.95	O'Reilly Media	
9	The Apollo Guidance Comput...	4.29	Praxis Publications Inc	
10	Minds and Computers: An Intr...	3.54	Edinburgh University Press	
11	The Architecture of Symbolic ...	4.50	McGraw-Hill	
12	Nmap Network Scanning: The...	4.32	Nmap Project	
13	The It Handbook for Business:...	4.40	Createspace Independent Pub...	
14	Accidental Empires	4.00	Harper	
15	Introducing HTML5	3.97	New Riders Publishing	

As you can see, at first glance it doesn't look too different from the regular view you created earlier in this lab - indeed, from the user perspective it's essentially the same: you see the results of a query displayed in a table-like format. The difference is that this materialized view is cached in the database so that the data can be accessed again at a future time without having to re-run the database query, which can be intensive on the server depending on the complexity of the query and the size of the table being queried.

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
2021-03-25	1.0	Sandip Saha Joy	Created initial version
2021-10-18	1.1	David Pasternak	Updated instructions

© IBM Corporation 2021. All rights reserved.