Lab: Create Tables and Load Data in PostgreSQL using pgAdmin

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.

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

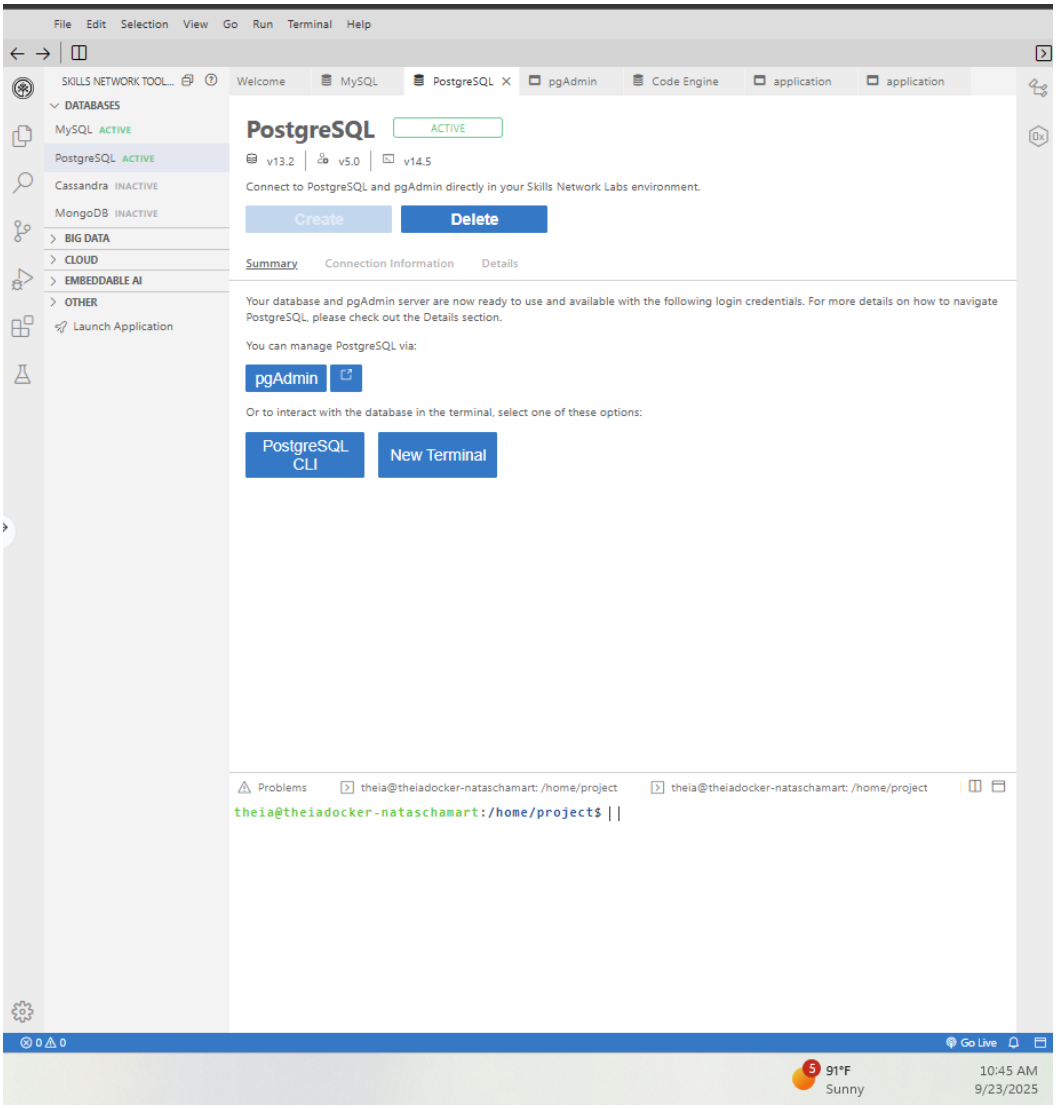
Lab Structure

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.

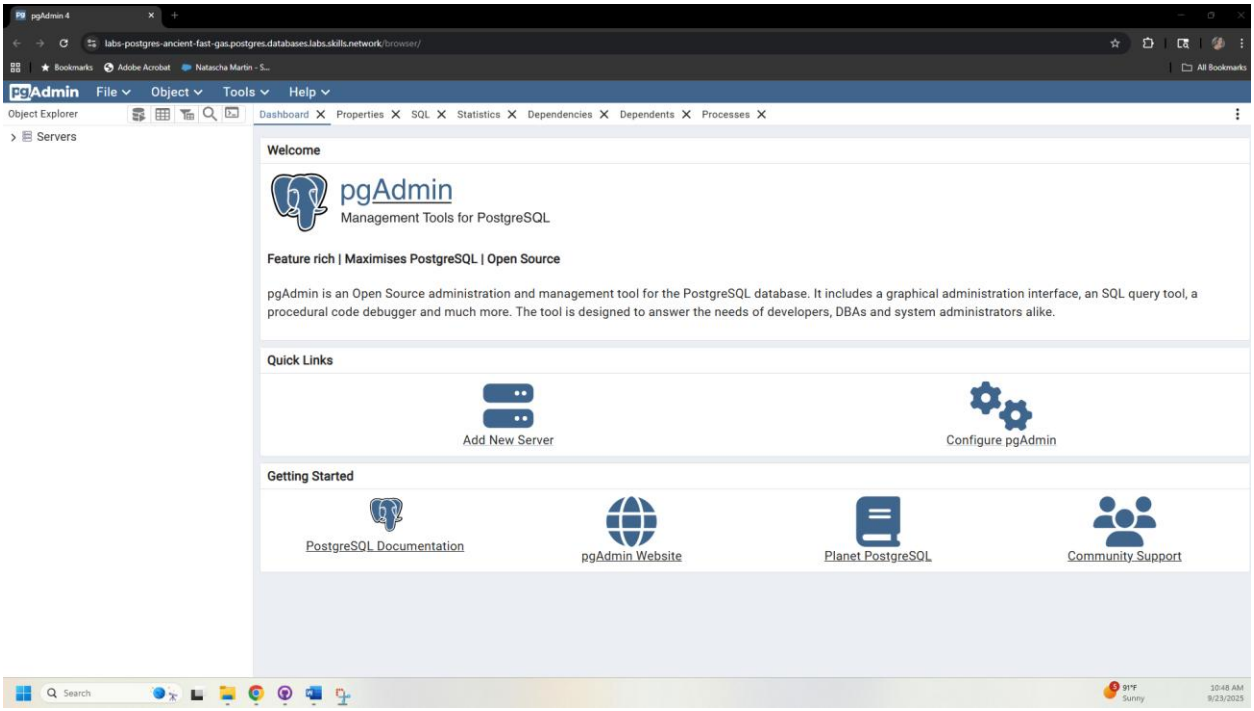
Task A: Create a database

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.

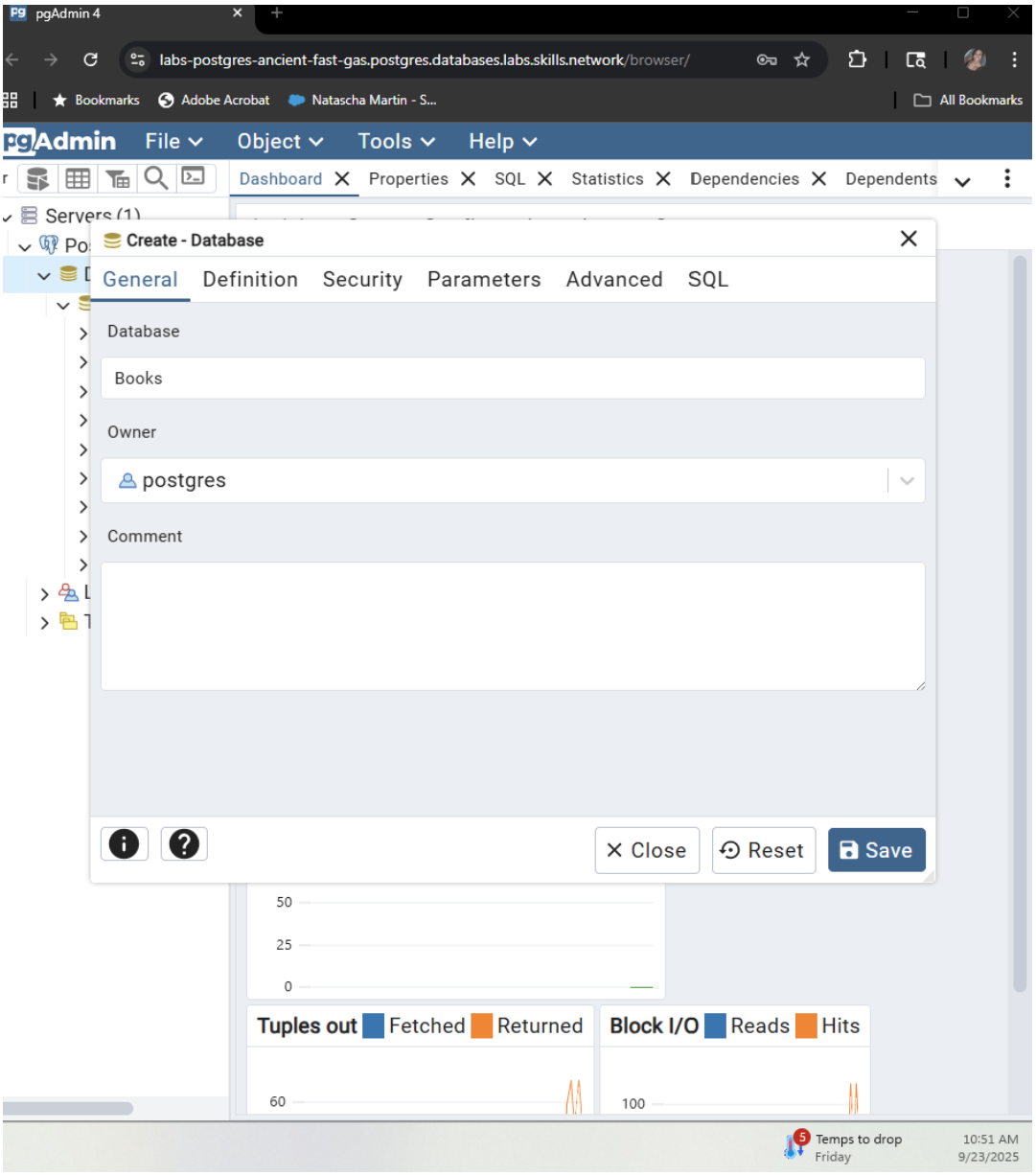
1. Click the Skills Network extension button on the left side of the window.
2. Open the **DATABASES** menu and click **PostgreSQL**.
3. Click **Create**. PostgreSQL may take a few moments to start.



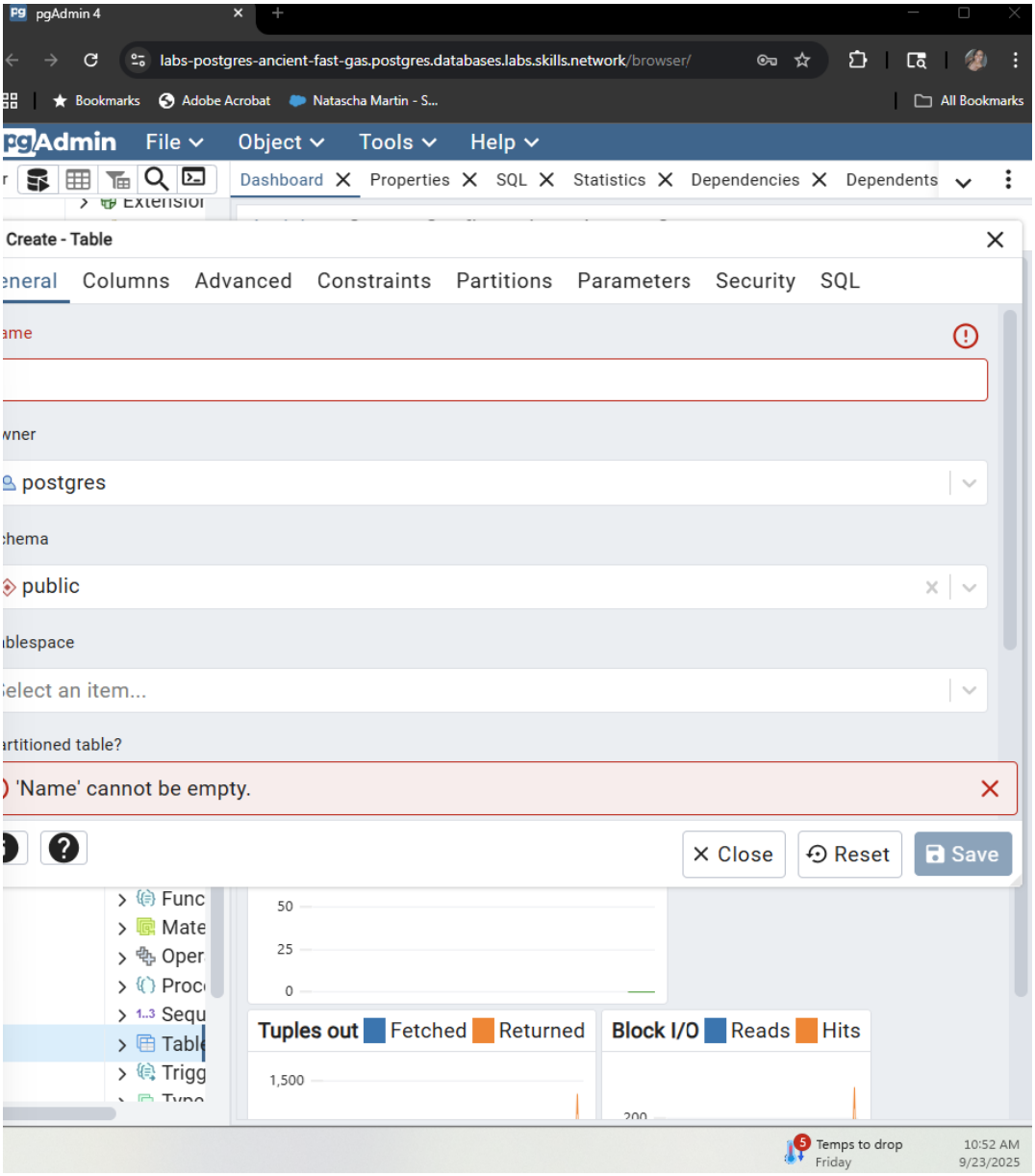
Next, open the pgAdmin Graphical User Interface by clicking **pgAdmin** in the Cloud IDE interface.



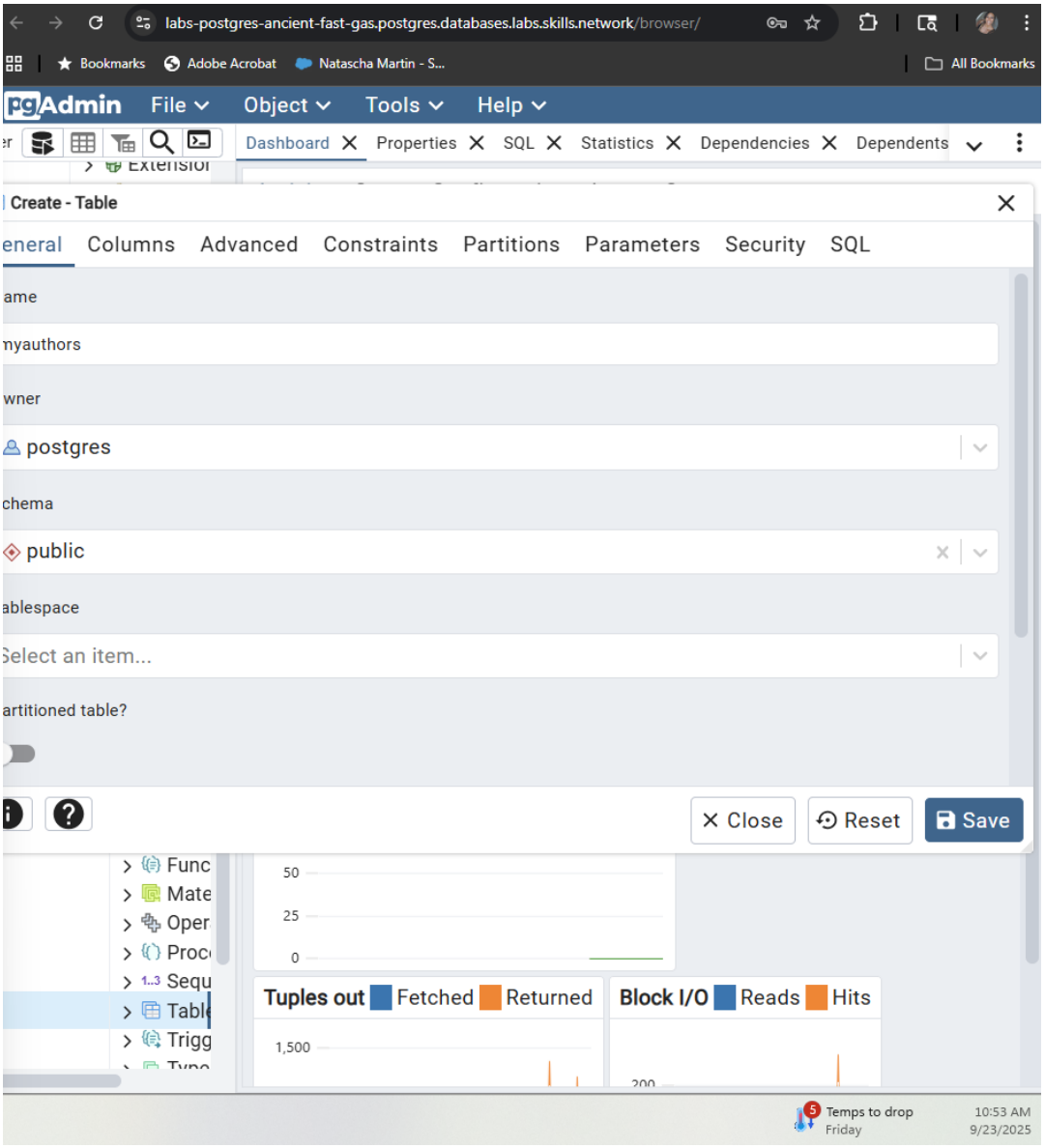
In the tree-view, expand **Servers** > **postgres** > **Databases**. If prompted, enter your PostgreSQL service session password. Right-click on **Databases** and go to **Create > Database**. In the **Database** box, type **Books** as the name for your new database, and then click **Save**. Proceed to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



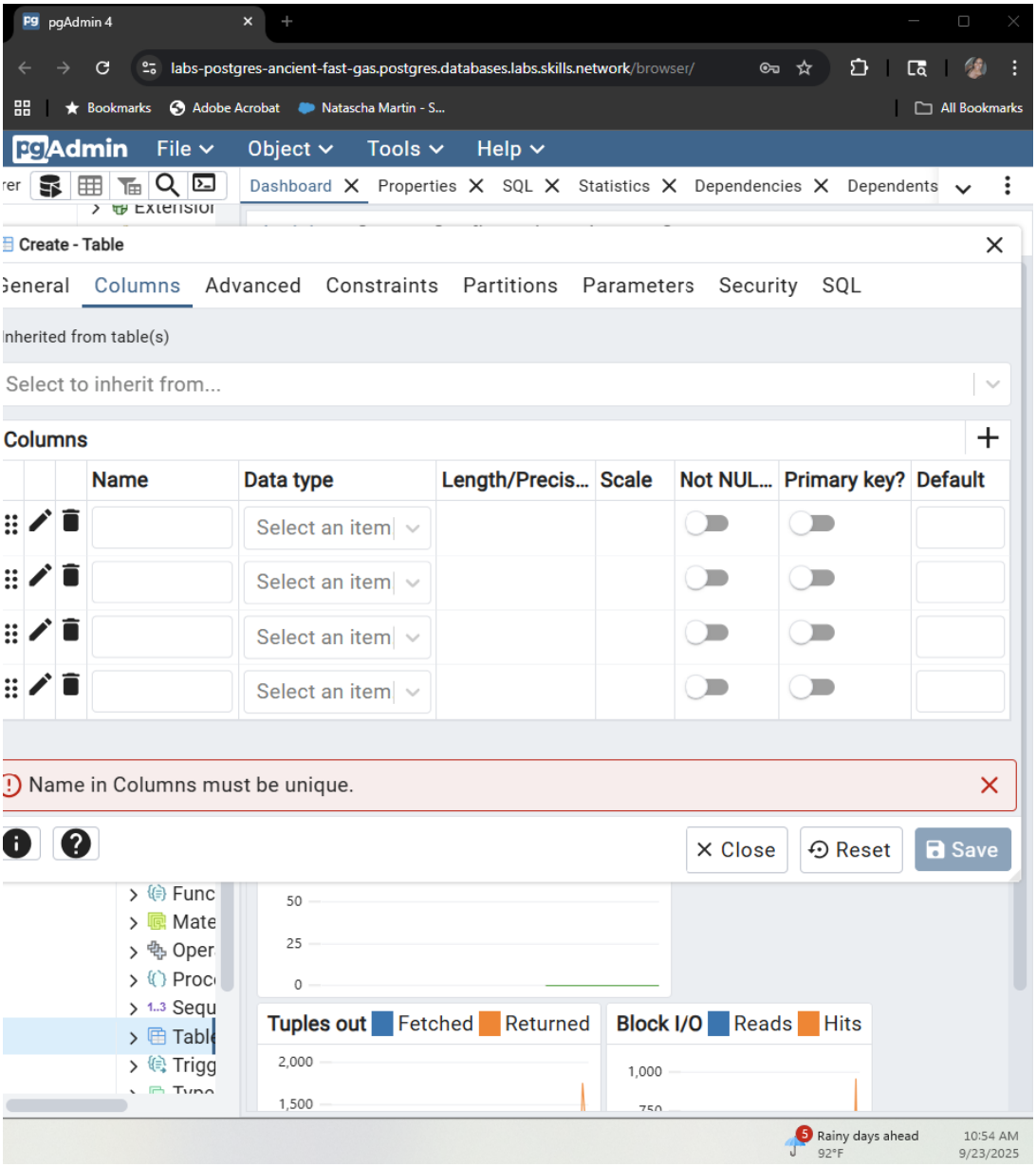
In the tree-view, expand **Books** > **Schemas** > **public**. Right-click on **Tables** and go to **Create > Table**.



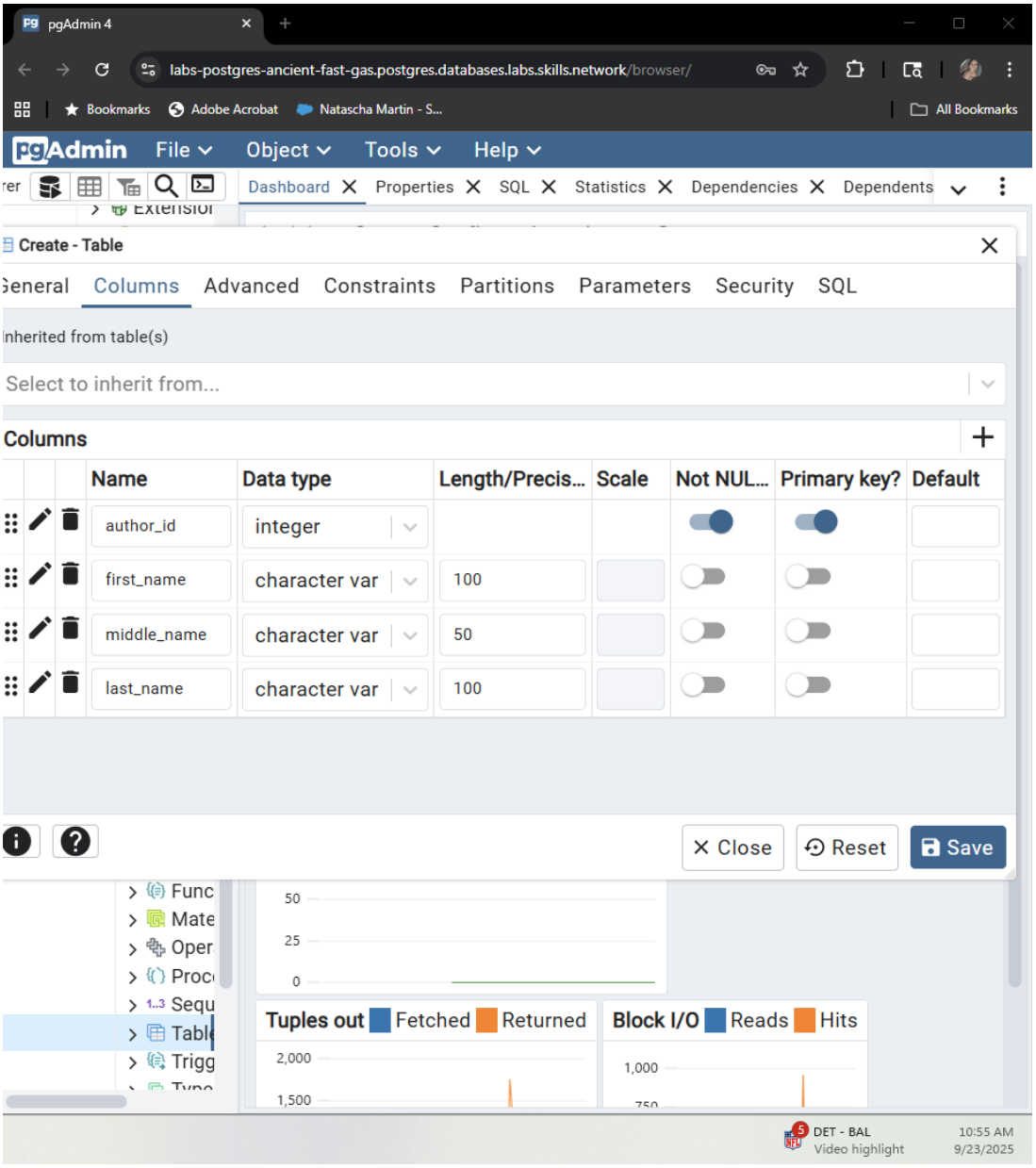
On the **General** tab, in the **Name** box, type **myauthors** as name of the table. Don't click **Save**, proceed to the next step.



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.



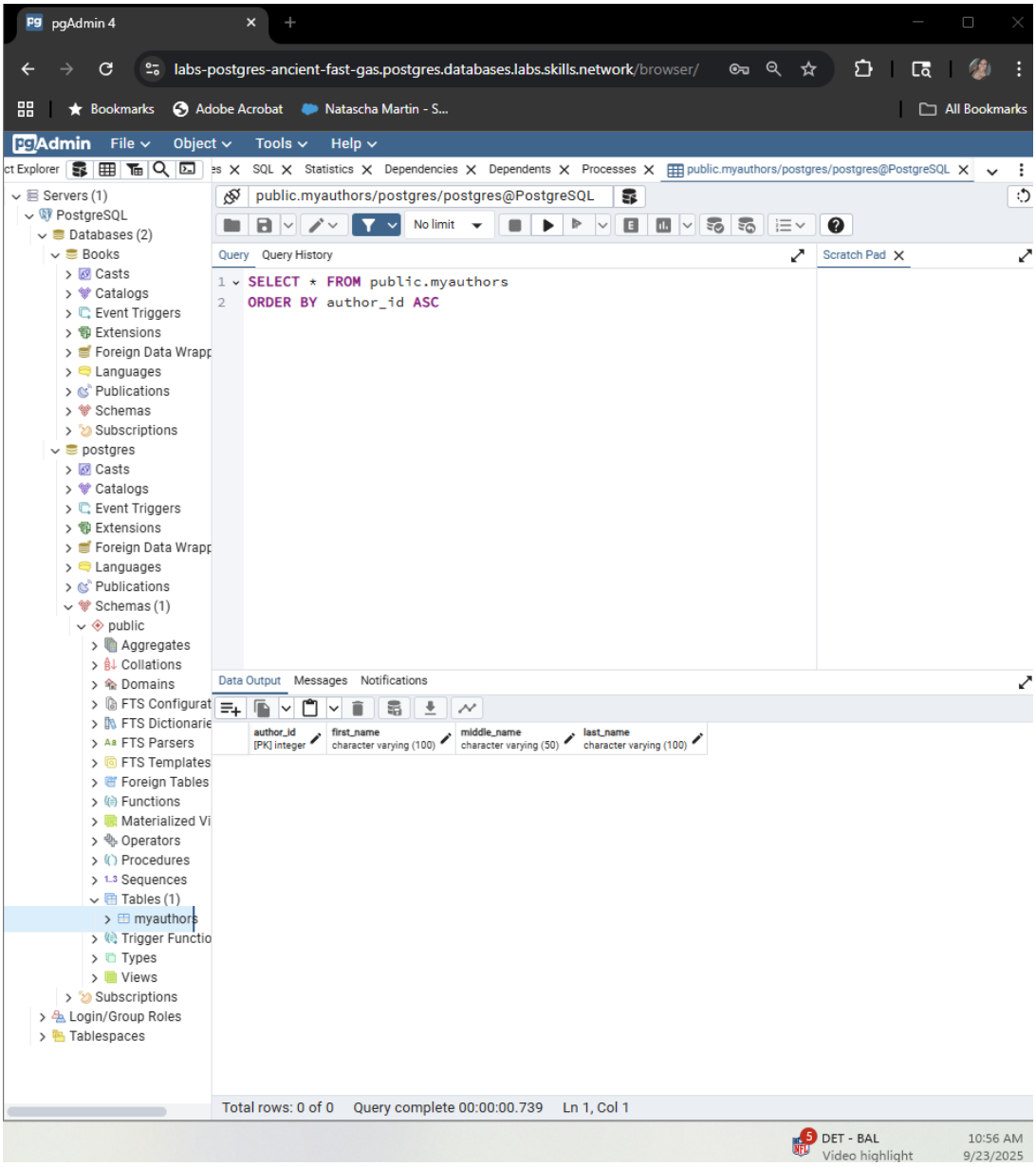
Enter the **myauthors** table definition structure information as shown in the image below in the highlighted boxes. Then click **Save**. Proceed to Task C.



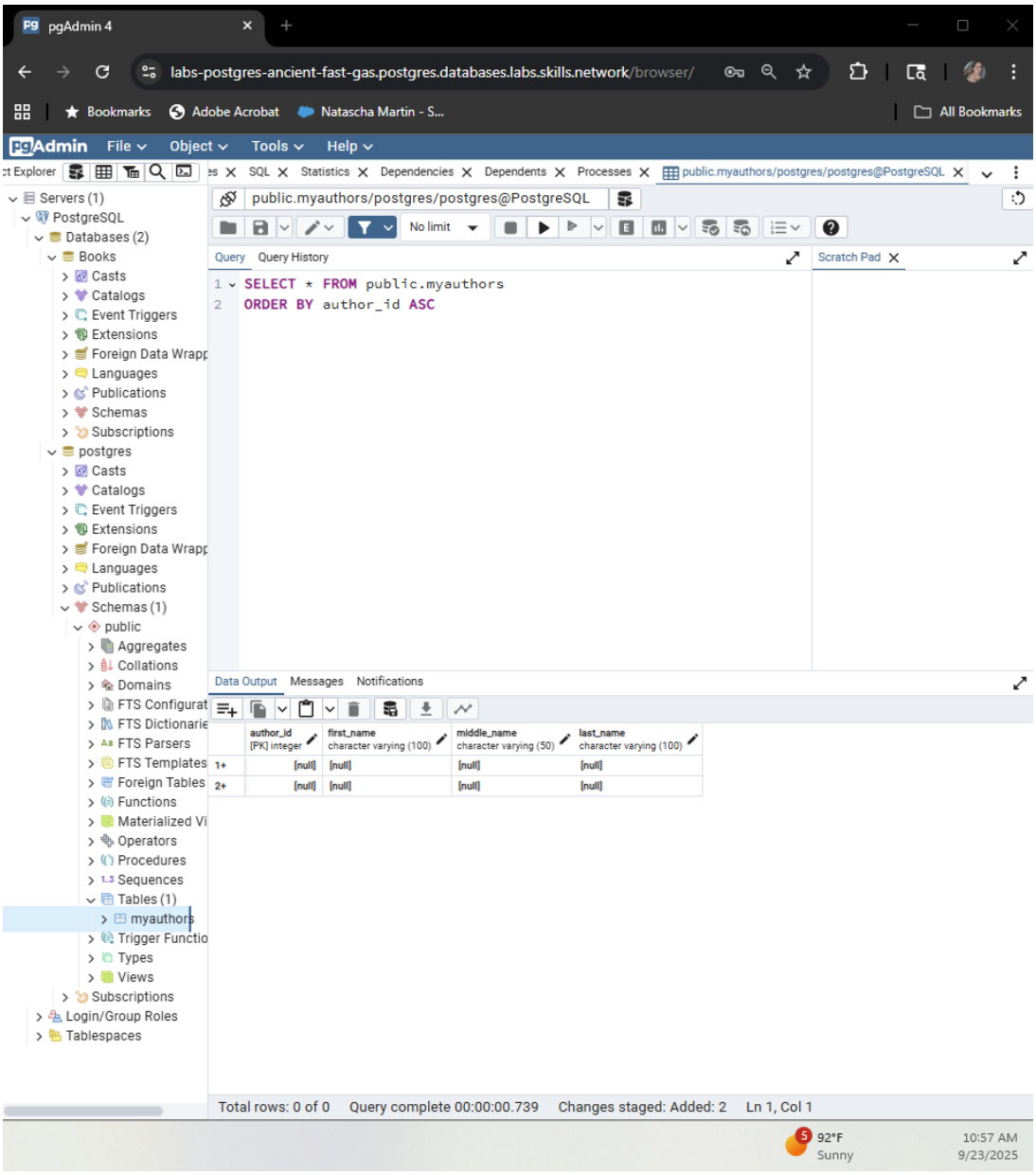
Task C: Load data into tables manually using the pgAdmin GUI

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.

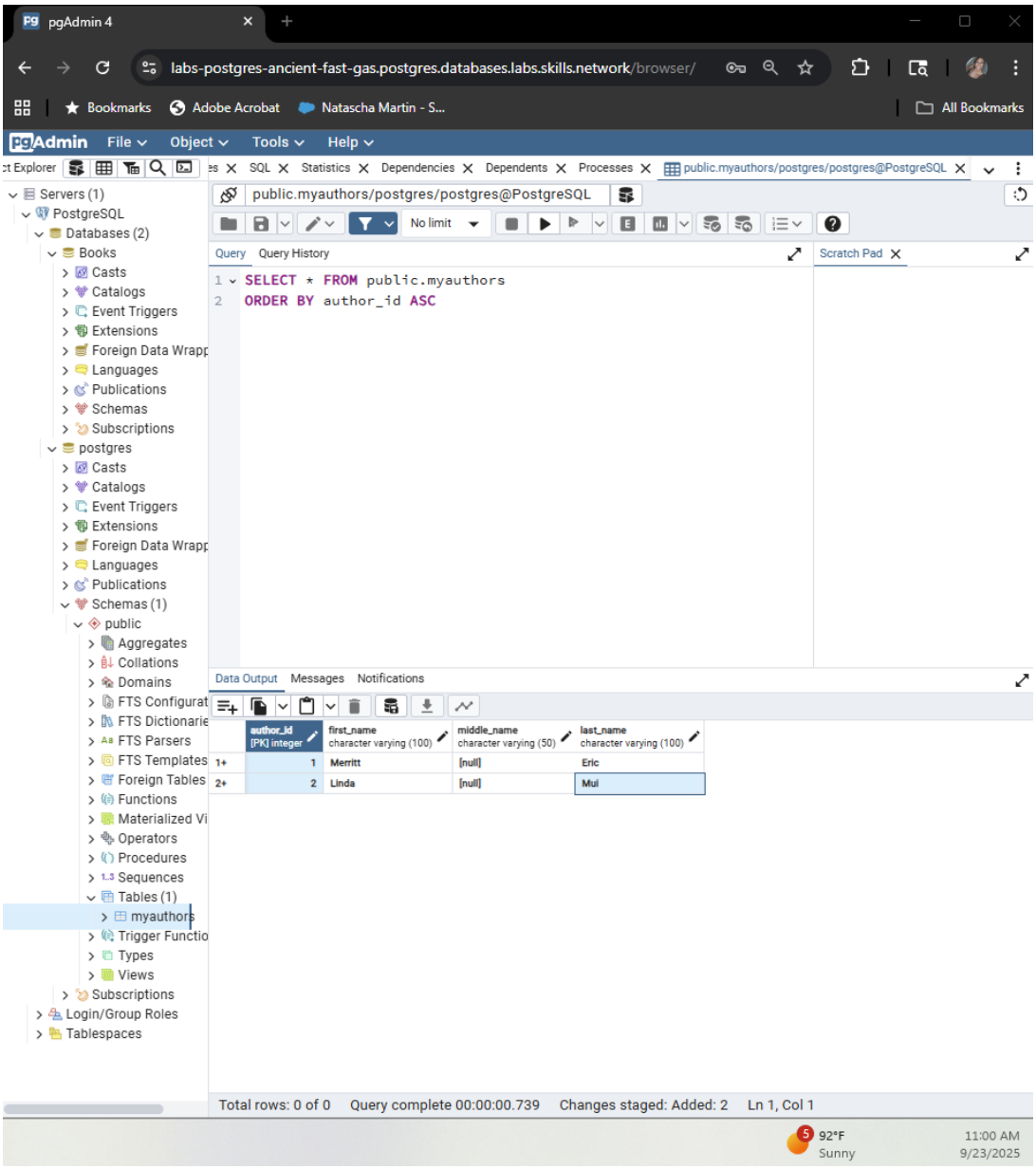
1. In the tree-view, expand **Tables**. Right-click **myauthors** and go to **View/Edit Data > All Rows**.



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.



Enter the values into the table as shown below:

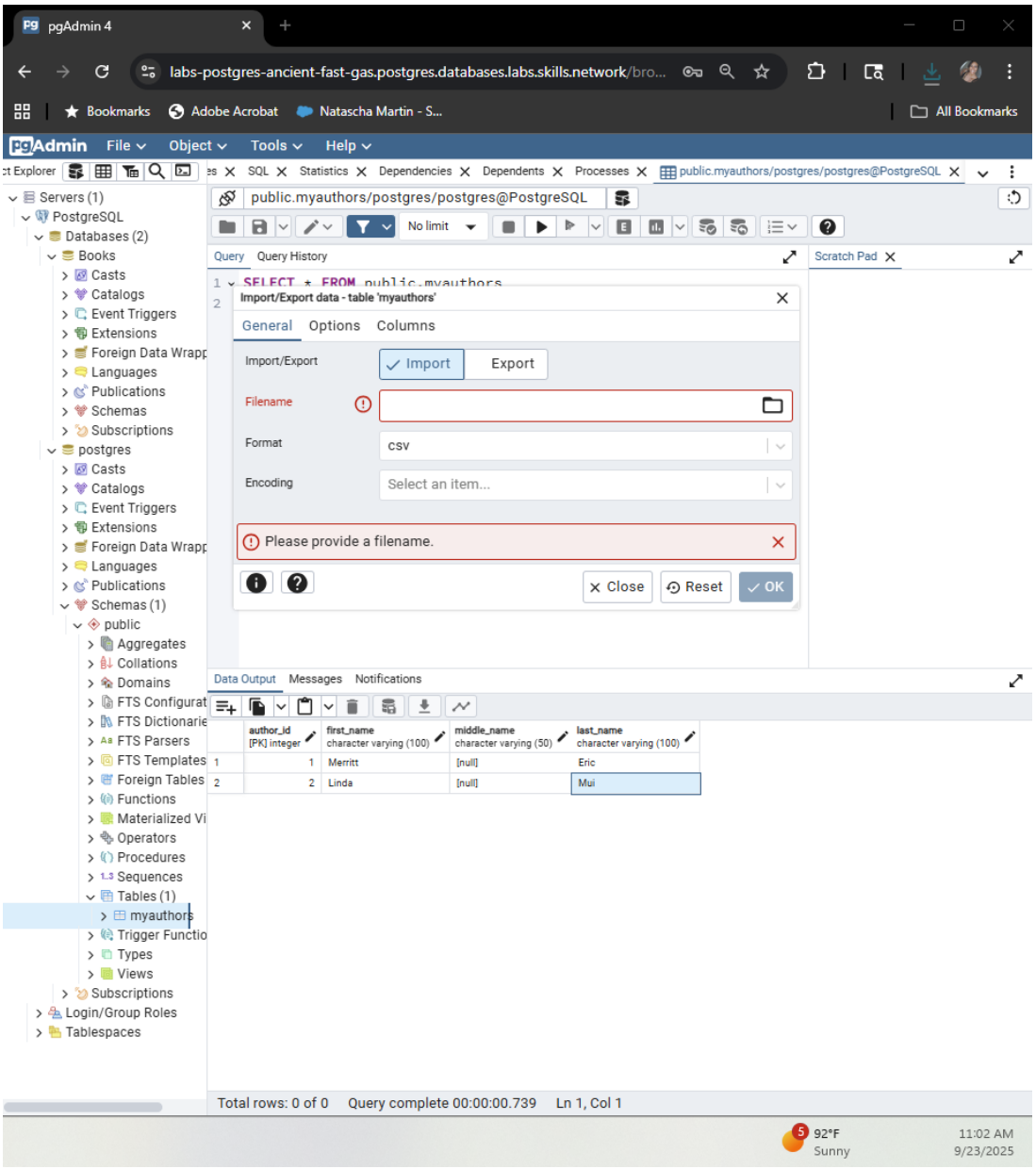


Task D: Load data into tables using a text/script file

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**.



Follow the instructions below to import:

i. Make sure **Import/Export** is set to **Import**,

ii. **Format** = **csv**.

iii. Then click **Select file** icon by the **Filename** box.

i. 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.

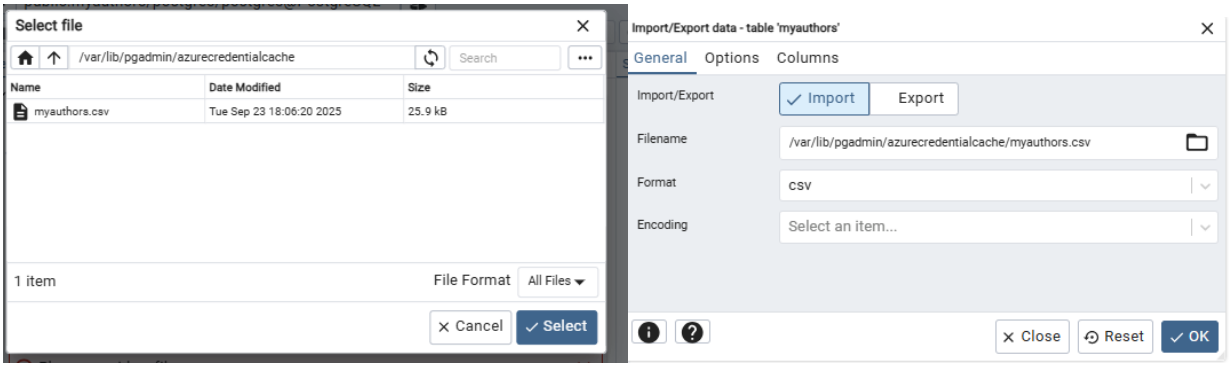
Step 3: Select pgadmin folder. Here you could notice the folders are locked except the pgadmin folder.

Step 4: Now select upload as mentioned here.

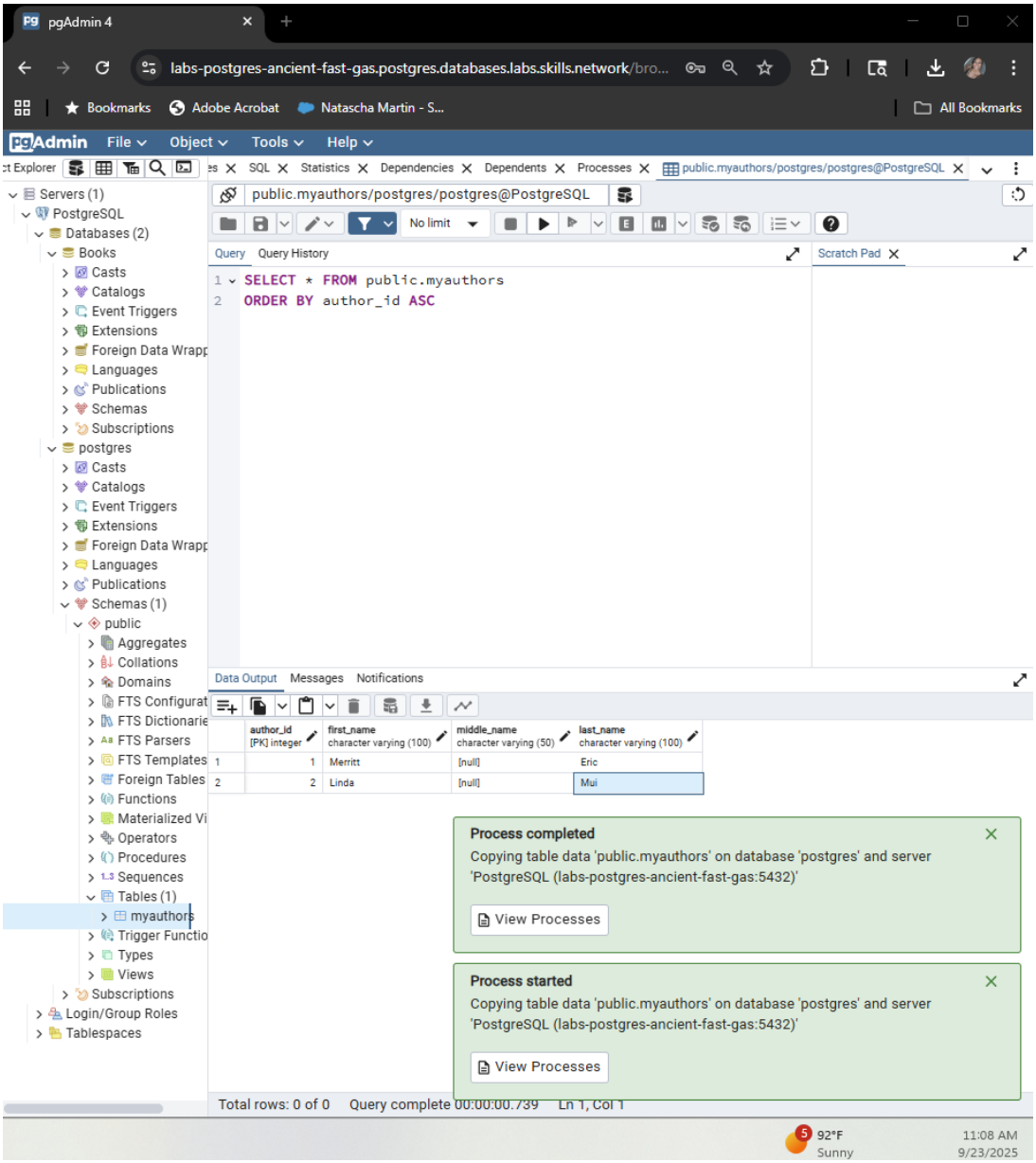
Step 5: Now Drag and drop the file from your downloads on your local machine.

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

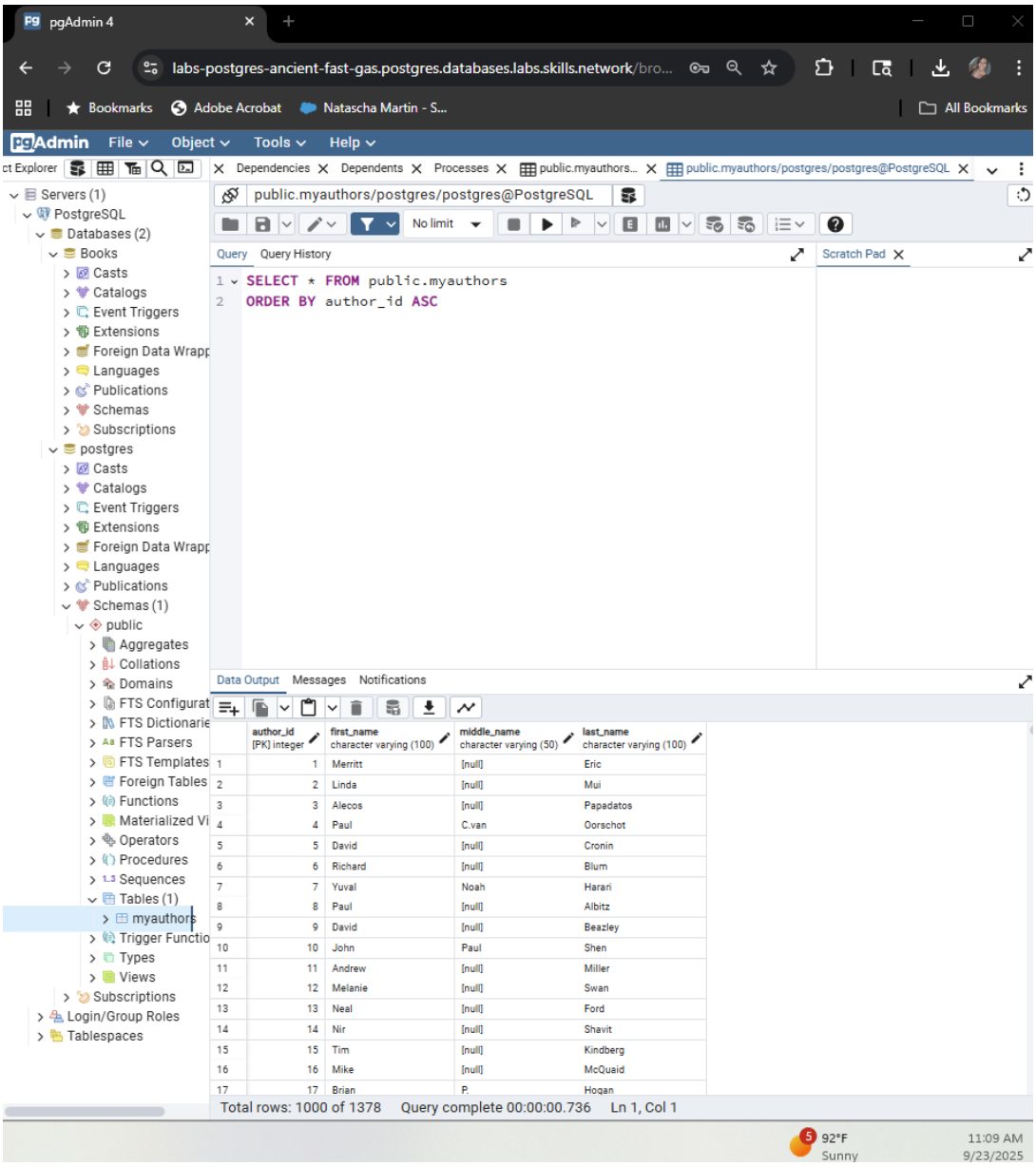
Select the uploaded **myauthors.csv** file from the list and click **Select**.



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



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



Lab Summary: Create Tables and Load Data in PostgreSQL using pgAdmin

Objective

The purpose of this lab was to practice creating and managing tables in a PostgreSQL database using the pgAdmin graphical interface. The focus was on building a database, defining a table schema, and loading data both manually and through a bulk import.

Steps Completed

1. **Database Creation**
   A new database named **Books** was created in pgAdmin.

2. **Table Creation**
   A table called **myauthors** was defined under the *public* schema with four columns:

- author_id (integer, primary key)
- first_name (varchar 100)
- middle_name (varchar 50)
- last_name (varchar 100)

3. **Manual Data Entry**
   Two rows of data were inserted manually into the **myauthors** table using the View/Edit Data → All Rows option, and the changes were saved successfully.

4. **Bulk Data Import**
   The provided myauthors.csv file was uploaded and imported using the Import/Export option in pgAdmin. The import was configured as CSV with the header option enabled, and the remaining rows were successfully added to the table.

Key Learnings

pgAdmin offers an intuitive GUI for creating and managing PostgreSQL databases as an alternative to command-line tools. Data can be entered manually for small additions or imported in bulk via CSV files for efficiency. Correct file format selection and header configuration are essential for successful imports.

Completion Status

The lab was completed successfully. Both manual and script-based data loads were executed and verified in the **myauthors** table.