Hands-on Lab: Populating a Data Warehouse using PostgreSQL

Estimated time needed: 1 hour

Purpose of the Lab:

The lab is designed to provide hands-on experience in creating and managing a production database using PostgreSQL within the IBM Skills Network Labs (SN Labs) Cloud IDE. You will learn how to launch a PostgreSQL server instance, utilize the pgAdmin graphical user interface (GUI) for database operations, and execute essential tasks like creating a database, designing tables, and loading data. The lab focuses on building a foundation in database management by guiding learners through the process of setting up a 'Production' database and populating it with data following a star schema design.

Benefits of Learning the Lab:

Engaging in this lab offers significant benefits for learners seeking to deepen their understanding of database management systems, particularly PostgreSQL. By working through the lab, you will gain practical skills in SQL, database creation, table design, and data manipulation, which are crucial for roles in data engineering, database administration, and data science. The hands-on approach helps in consolidating knowledge of database schemas and SQL queries, thereby enhancing the learner's ability to manage and analyze data effectively in real-world scenarios. Additionally, familiarity with tools like pgAdmin and the Cloud IDE environment adds valuable experience to your skill set, preparing you for advanced database projects and tasks.

Software Used in this Lab

To complete this lab you will utilize the PostgreSQL Database 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

Production database is used in this lab.
The production database contains:

- DimCustomer
- DimMonth
- FactBilling

Objectives

In this lab you will:

- Create production related database and tables in a PostgreSQL instance.
- Populate the production data warehouse byloading the tables from Scripts.

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.

Data Used in this Lab

The following are the SQL data files used in this lab.

The production database contains:

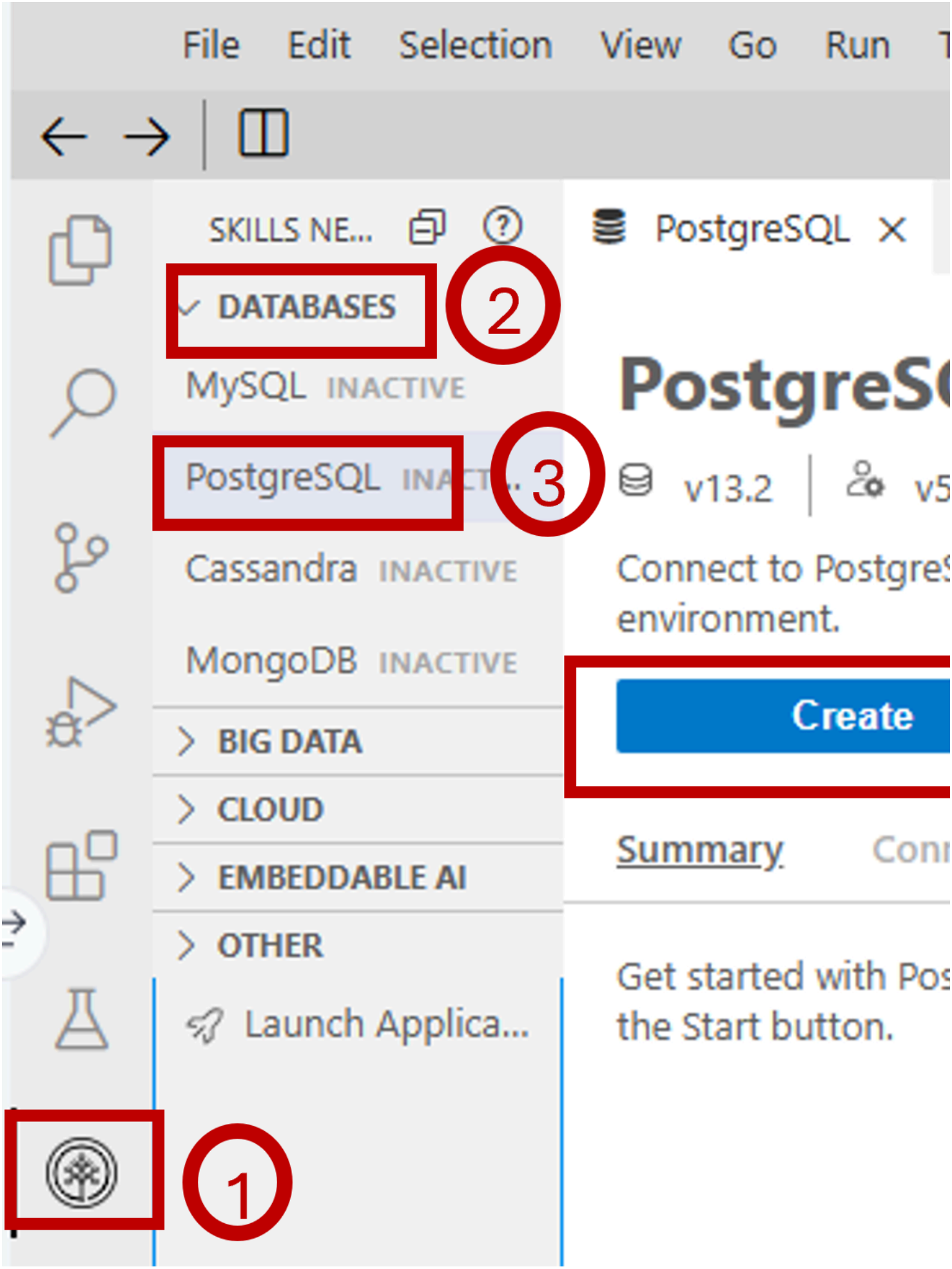
- DimCustomer
- DimMonth
- FactBilling

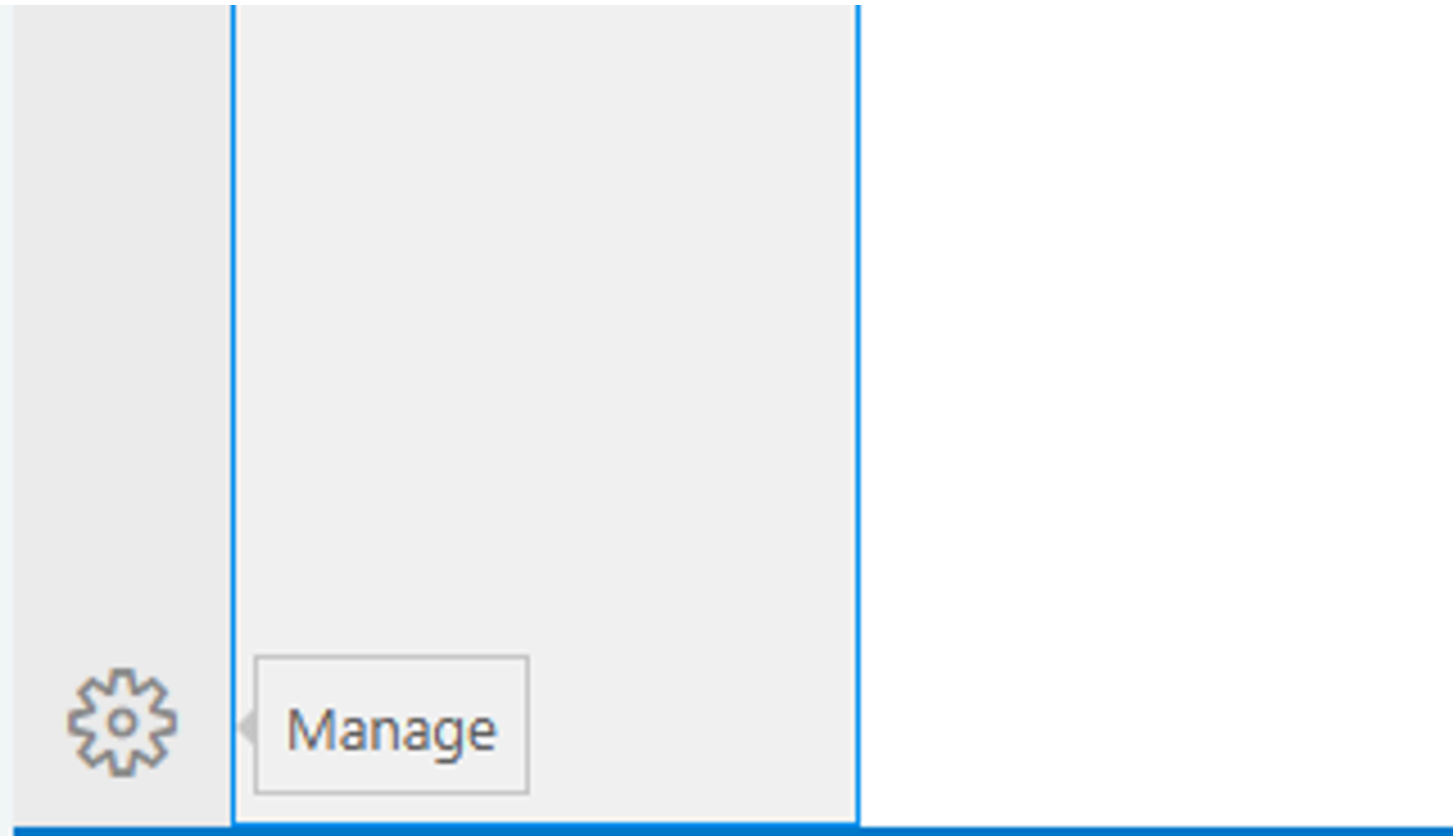
- Star Schema

Task A: Create a database

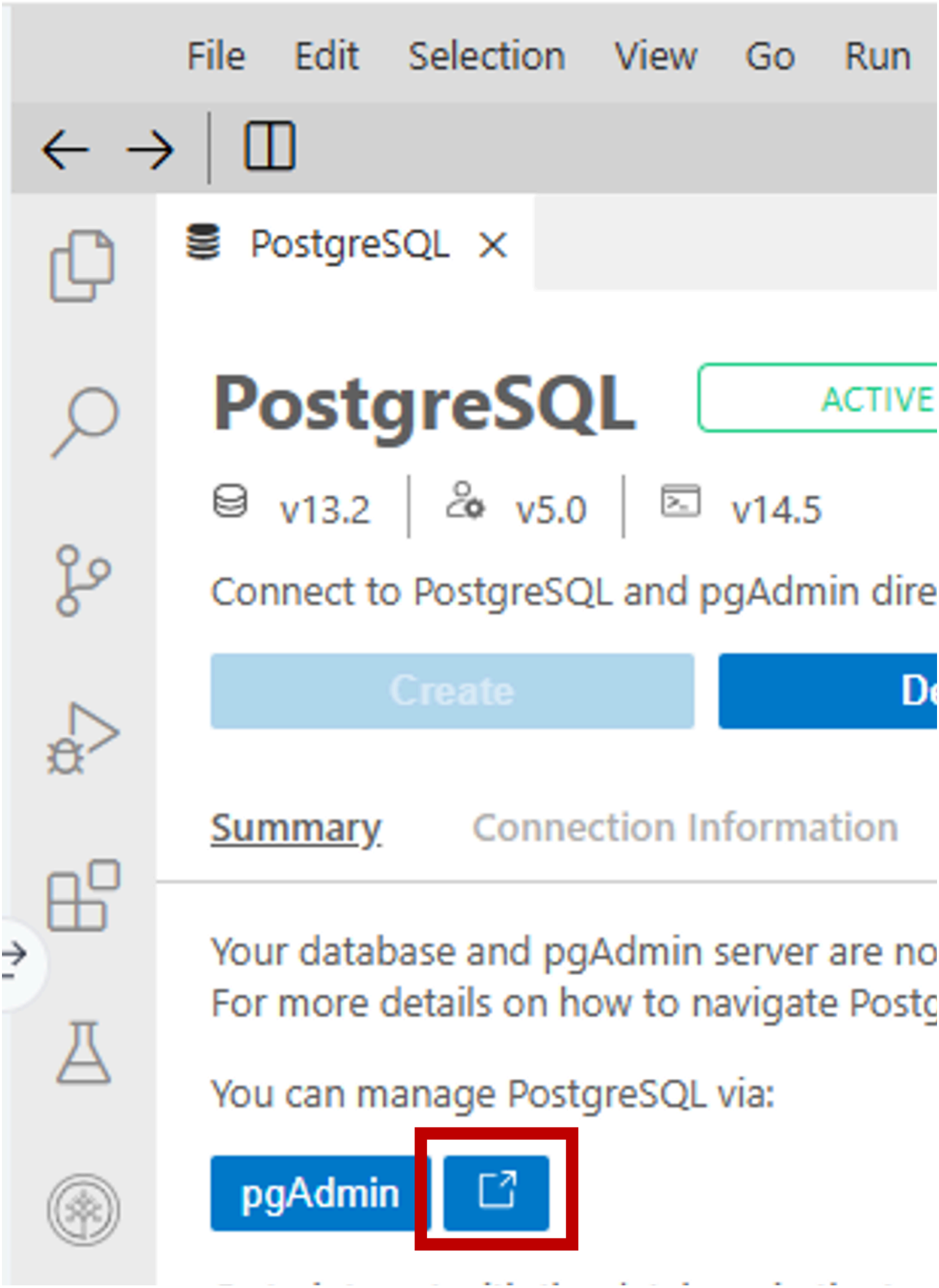
First, to create a database on a PostgreSQL server instance, you'll first want to actually launch a PostgreSQL server instance on Cloud IDE and open up the pgAdmin Graphical User Interface.

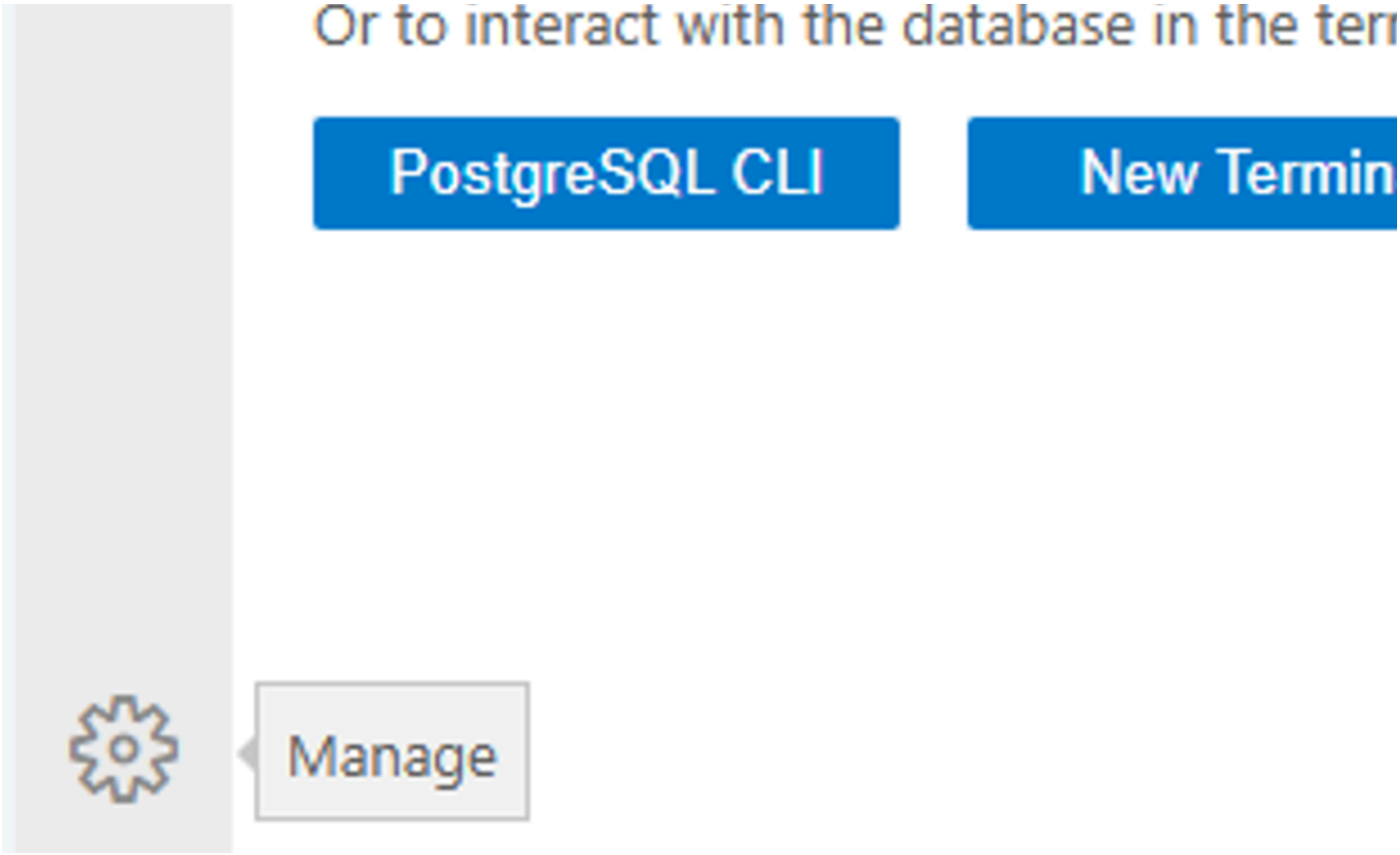
1. Click on the Skills Network extension button on the left side of the window.
2. Open the **DATABASES** drop down menu.
3. Click on **PostgreSQL**
4. Click on the **Create** button. PostgreSQL may take a few moments to start.


File Edit Selection View Go Run
SKILLS NE...
DATABASES
2
MySQL INACTIVE
PostgreSQL INA... 3
Cassandra INACTIVE
MongoDB INACTIVE
BIG DATA
CLOUD
EMBEDDABLE AI
OTHER
Launch Applica...
1
PostgreSQL
v13.2
Connect to PostgreS
environment.
Create
Summary
Get started with Pos
the Start button.

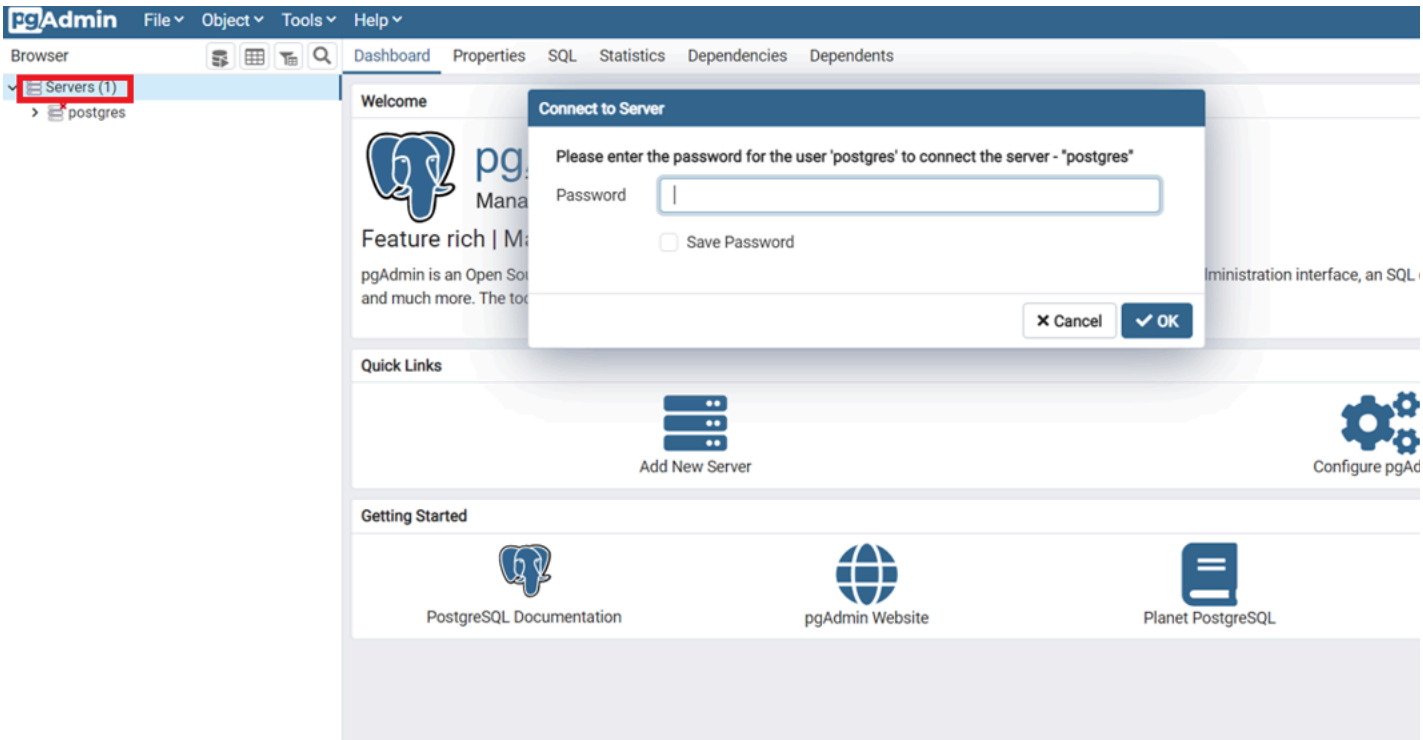


5. Next, open the pgAdmin Graphical User Interface by clicking the **pgAdmin launch** button in the Cloud IDE interface.

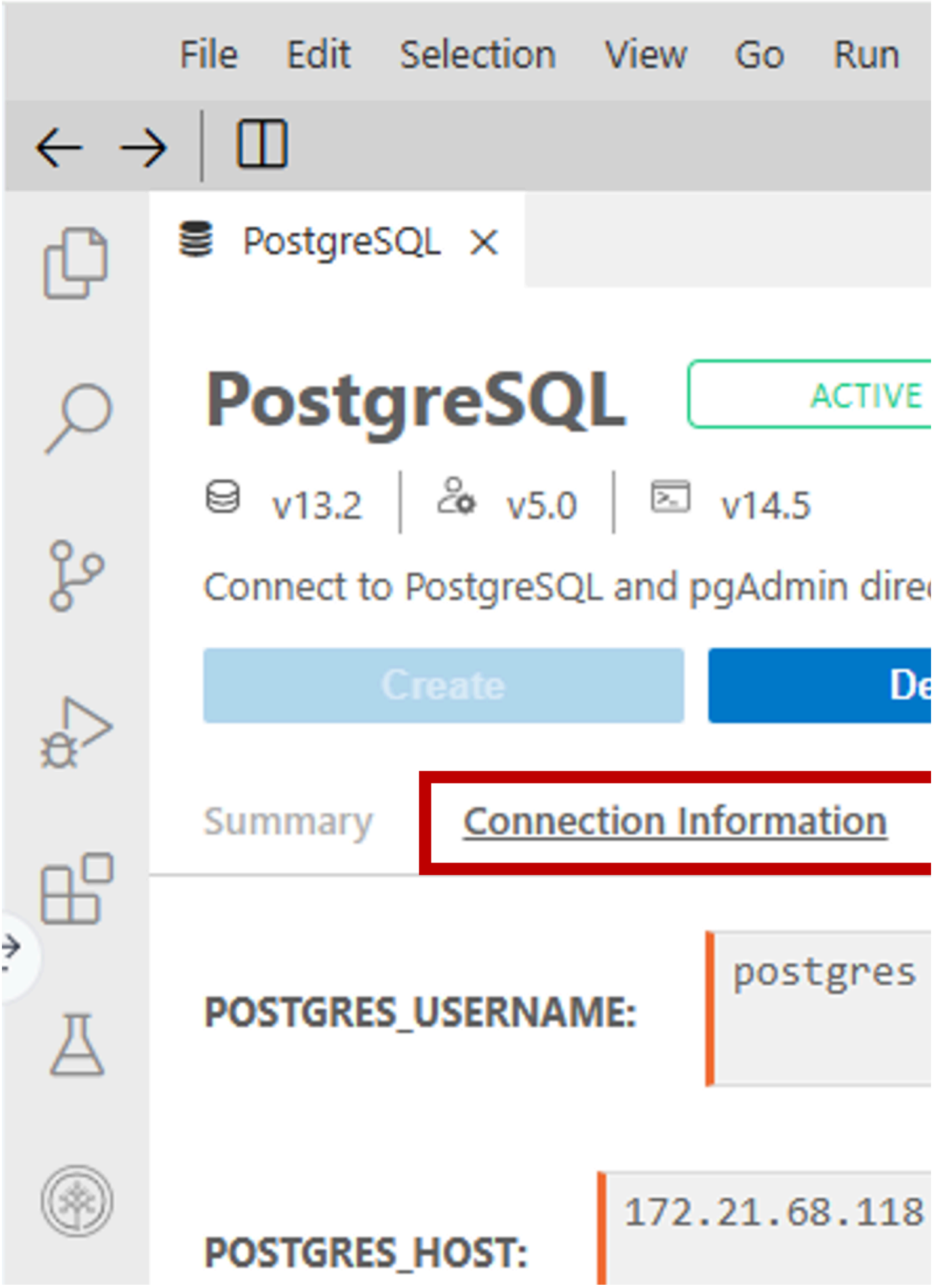

File Edit Selection View Go Run
PostgreSQL
PostgreSQL
ACTIVE
v13.2
v5.0
v14.5
Connect to PostgreSQL and pgAdmin dire
Create
Summary
Connection Information
Your database and pgAdmin server are no
For more details on how to navigate Postg
You can manage PostgreSQL via:
pgAdmin

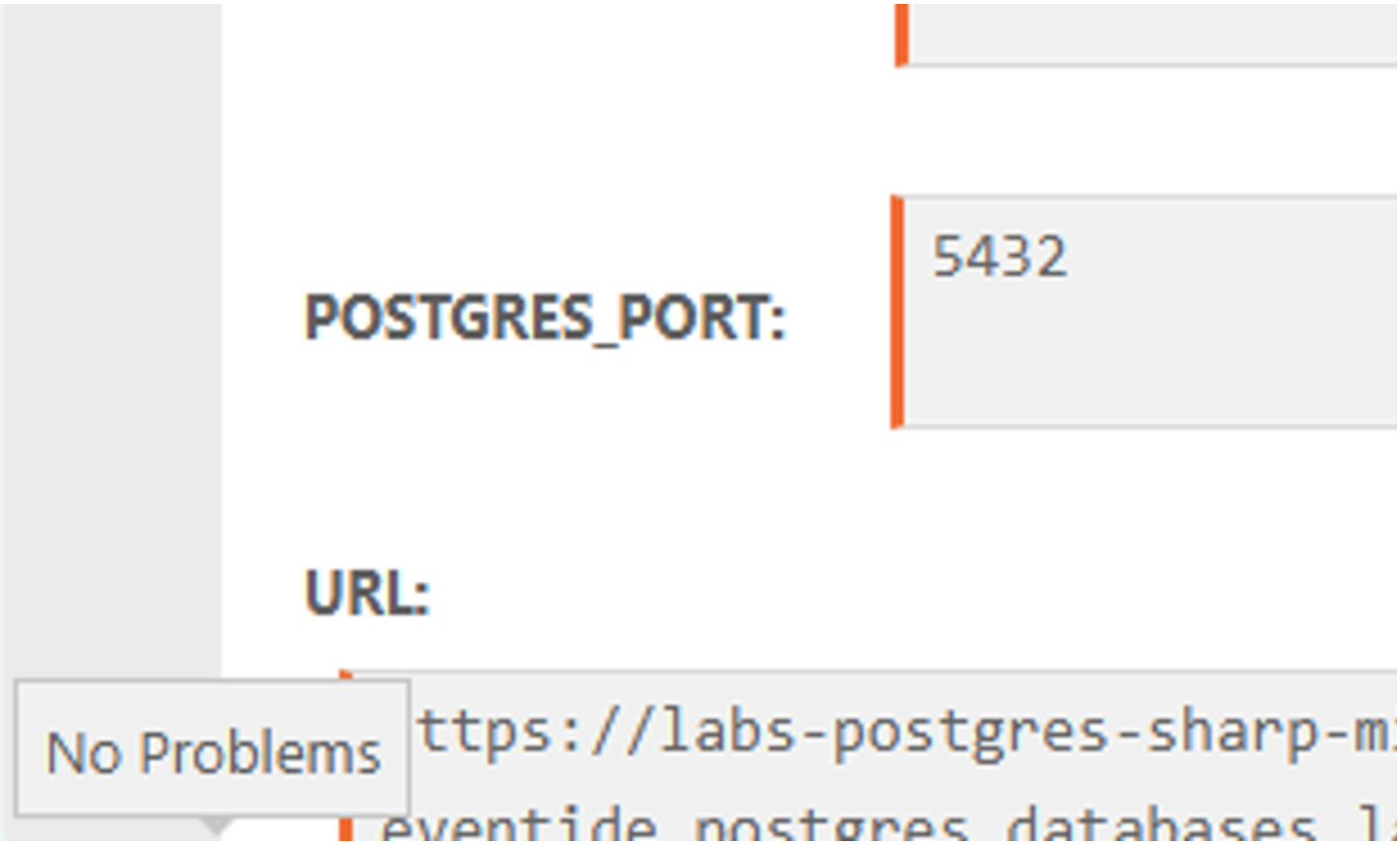


6. Once the pgAdmin GUI opens, click on the **Servers** tab on the left side of the page. You will be prompted to enter a password.

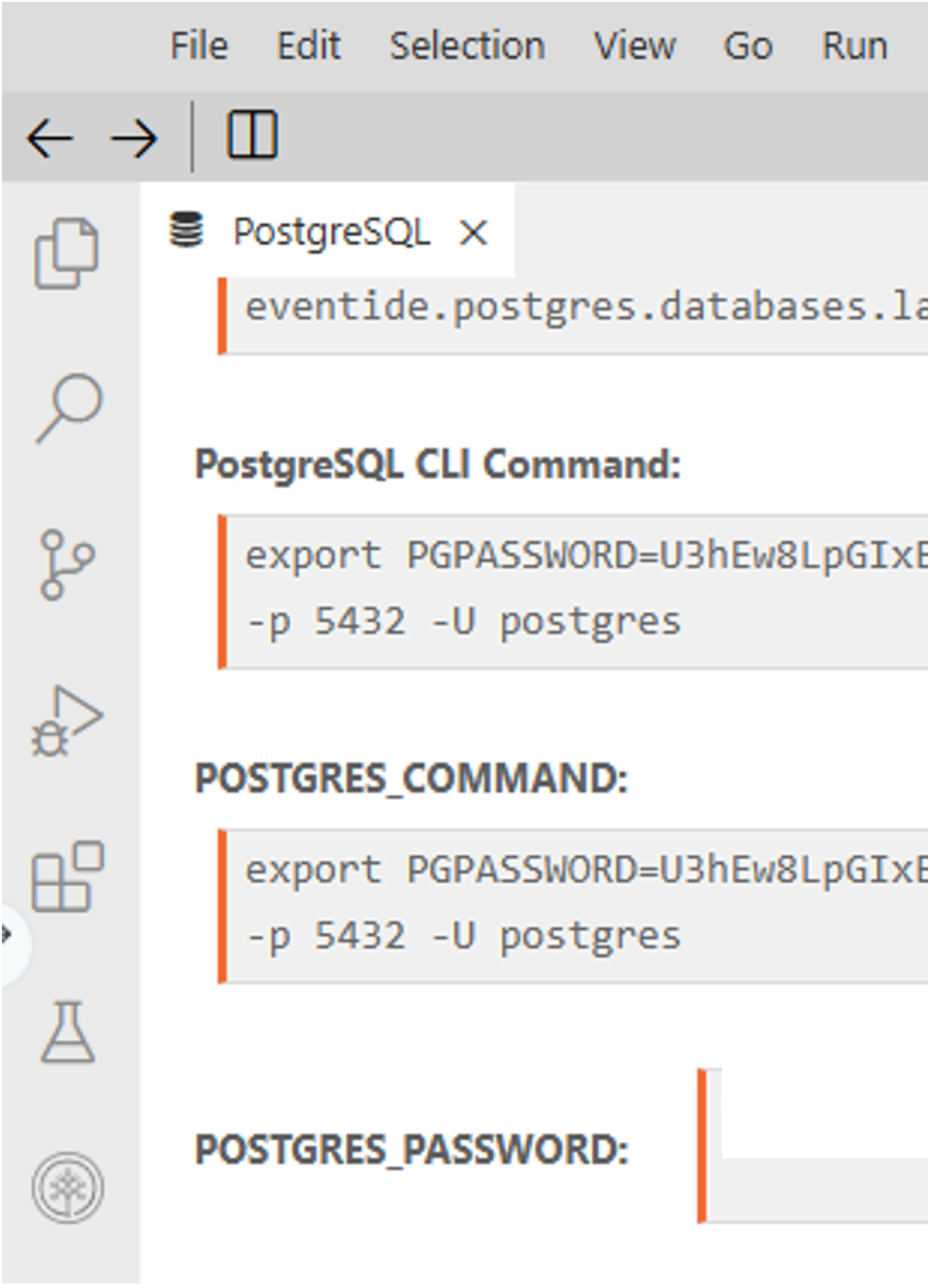


7. To retrieve your password, click on the **PostgreSQL** and go to **Conection Information** tab on the top of the interface.

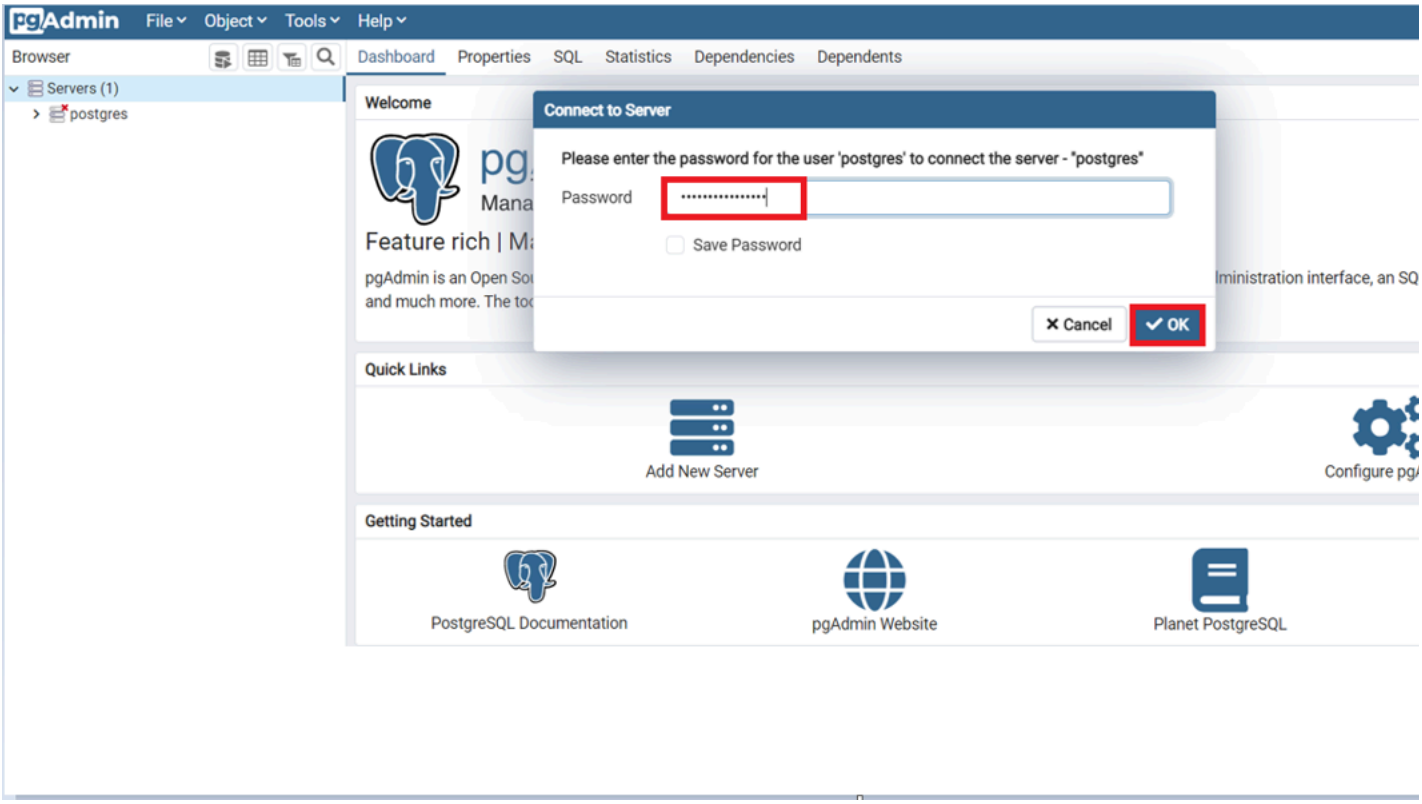

File Edit Selection View Go Run
PostgreSQL
PostgreSQL
ACTIVE
v13.2
v5.0
v14.5
Connect to PostgreSQL and pgAdmin dire
Create
De
Summary
Connection Information
POSTGRES_USERNAME:
postgres
POSTGRES_HOST:
172.21.68.118



8. Scroll down to the session password and click on the **Copy** icon to the right of your password to copy onto your clipboard.

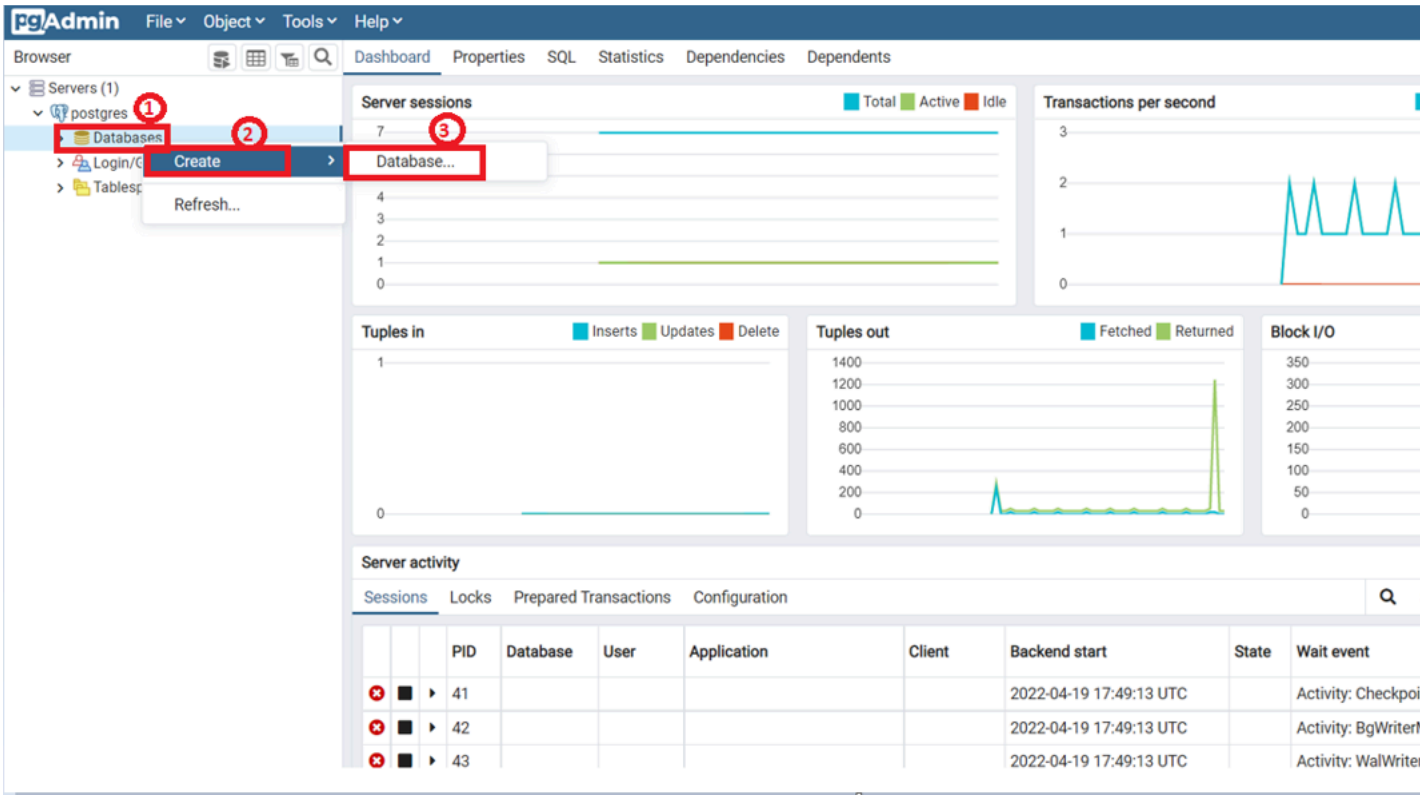

File Edit Selection View Go Run
PostgreSQL
eventide.postgres.databases.la
PostgreSQL CLI Command:
export PGPASSWORD=U3hEw8LpGIxE
-p 5432 -U postgres
POSTGRES_COMMAND:
export PGPASSWORD=U3hEw8LpGIxE
-p 5432 -U postgres
POSTGRES_PASSWORD:

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

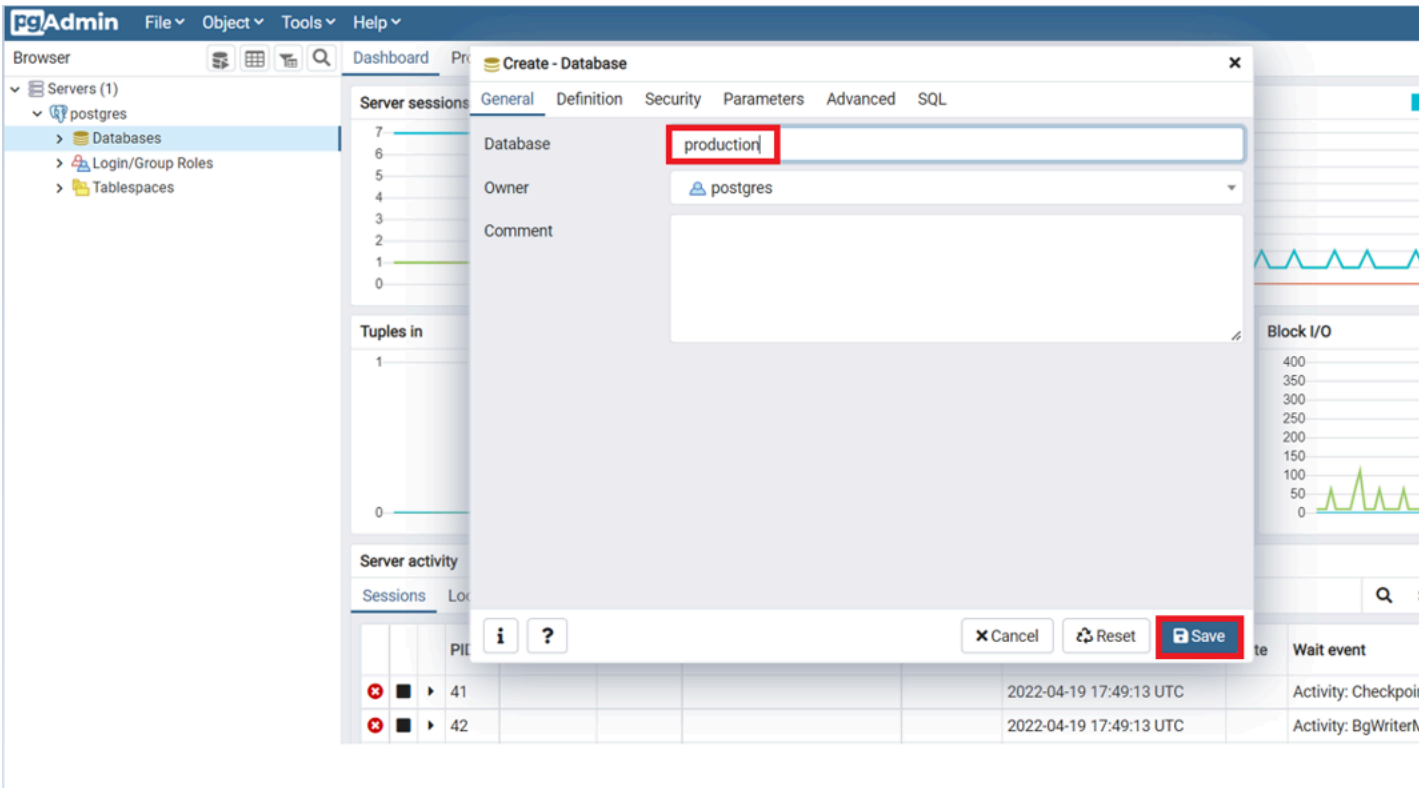


10. You will then be able to access the pgAdmin GUI tool.

11. In the left tree-view, right-click on **Databases> Create > Database**.



In the Database box, type **Production** as the name for your new database, and then click **Save**. Proceed to Task B.

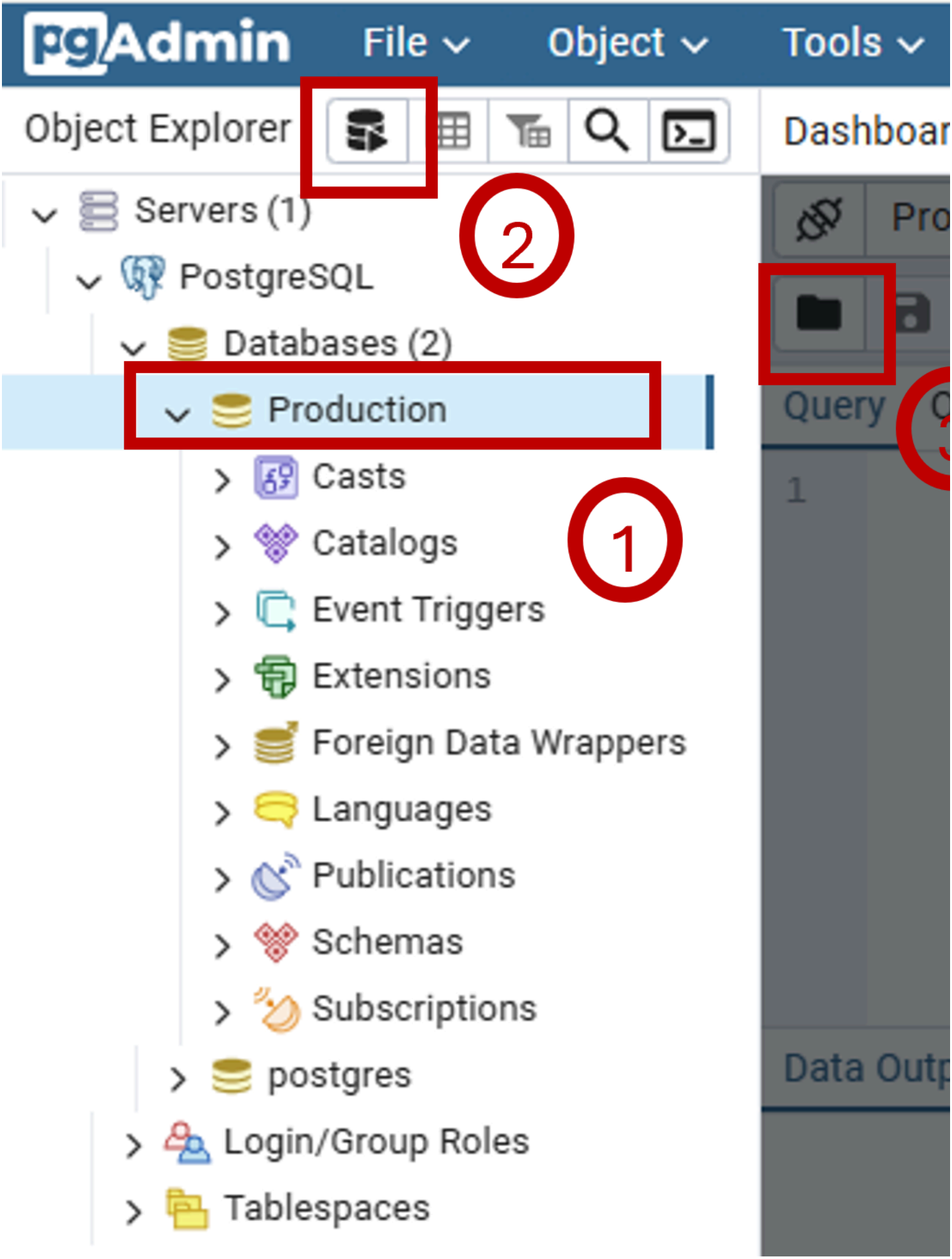


Task B: Create tables

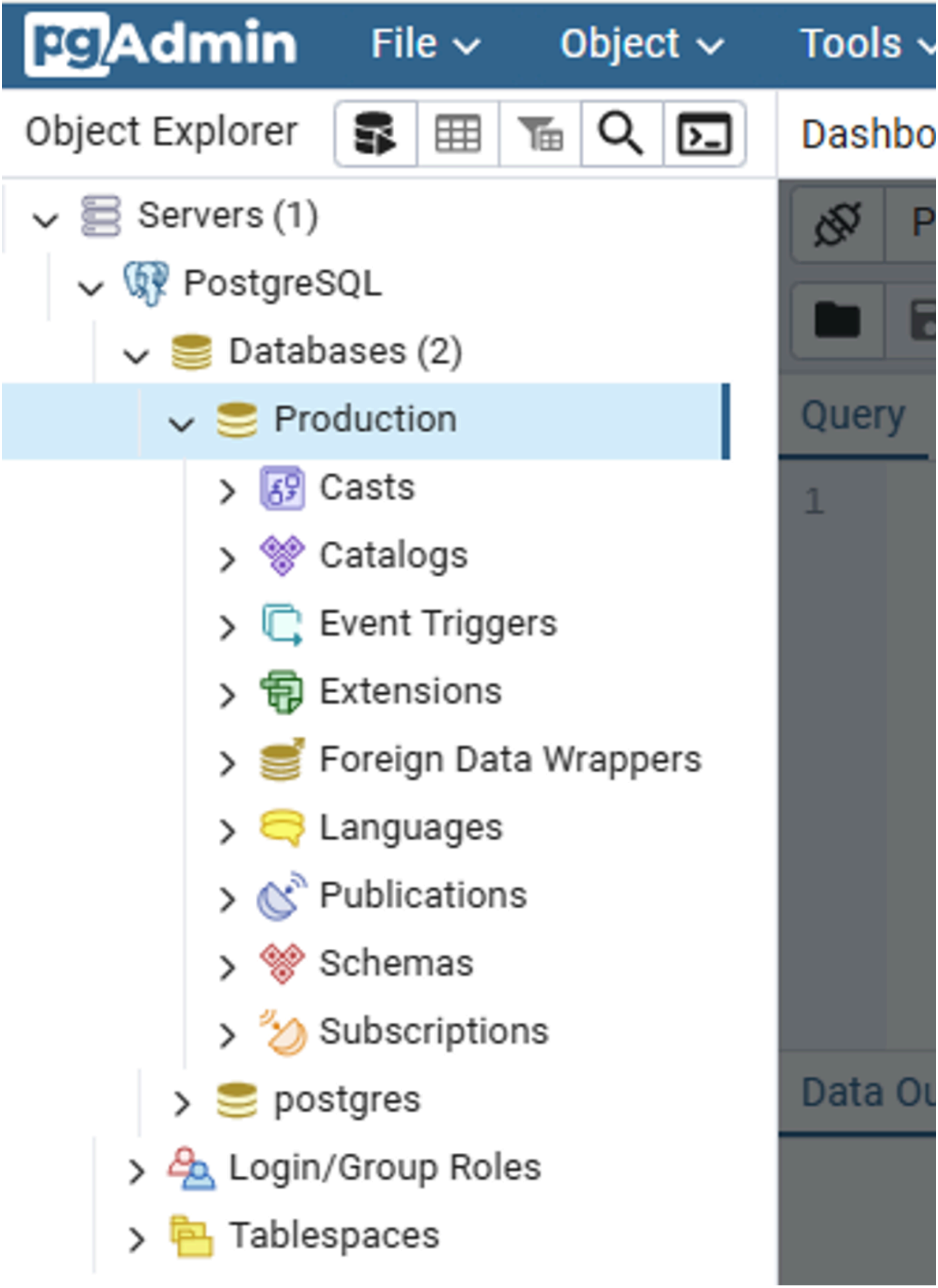
Now, that you have your PostgreSQL service active and have created the **Production database** using pgAdmin, let's go ahead and create a few tables to populate the database and store the data that we wish to eventually upload into it.

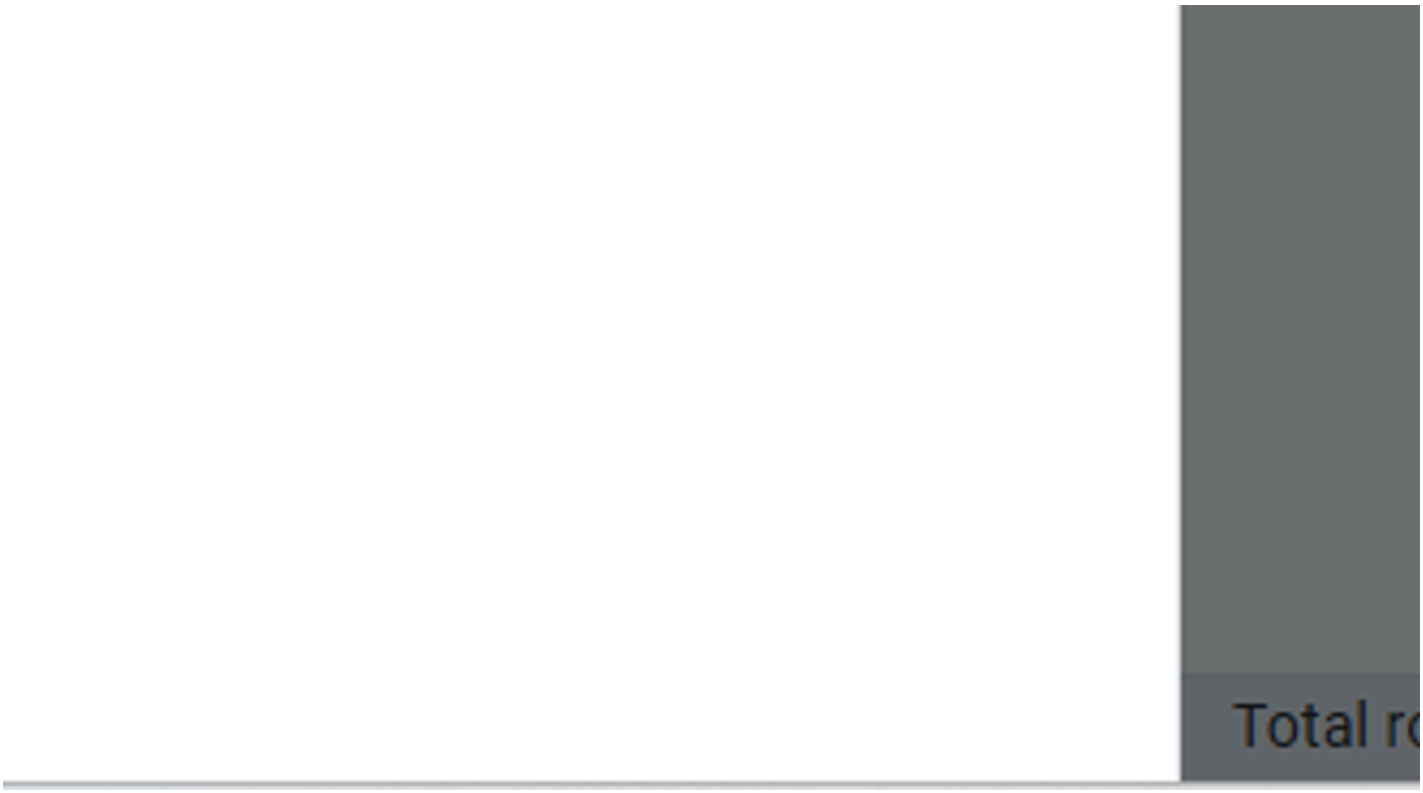
1. Click on the Production database and in the top of the page go to **Query tool** and then click on **Open File**. Next a new page pops up called **Select File**. Click on the three dots and choose **Upload** option as shown in the screenshot.

   Note: Ensure that you upload the files to this path: /var/lib/pgadmin/

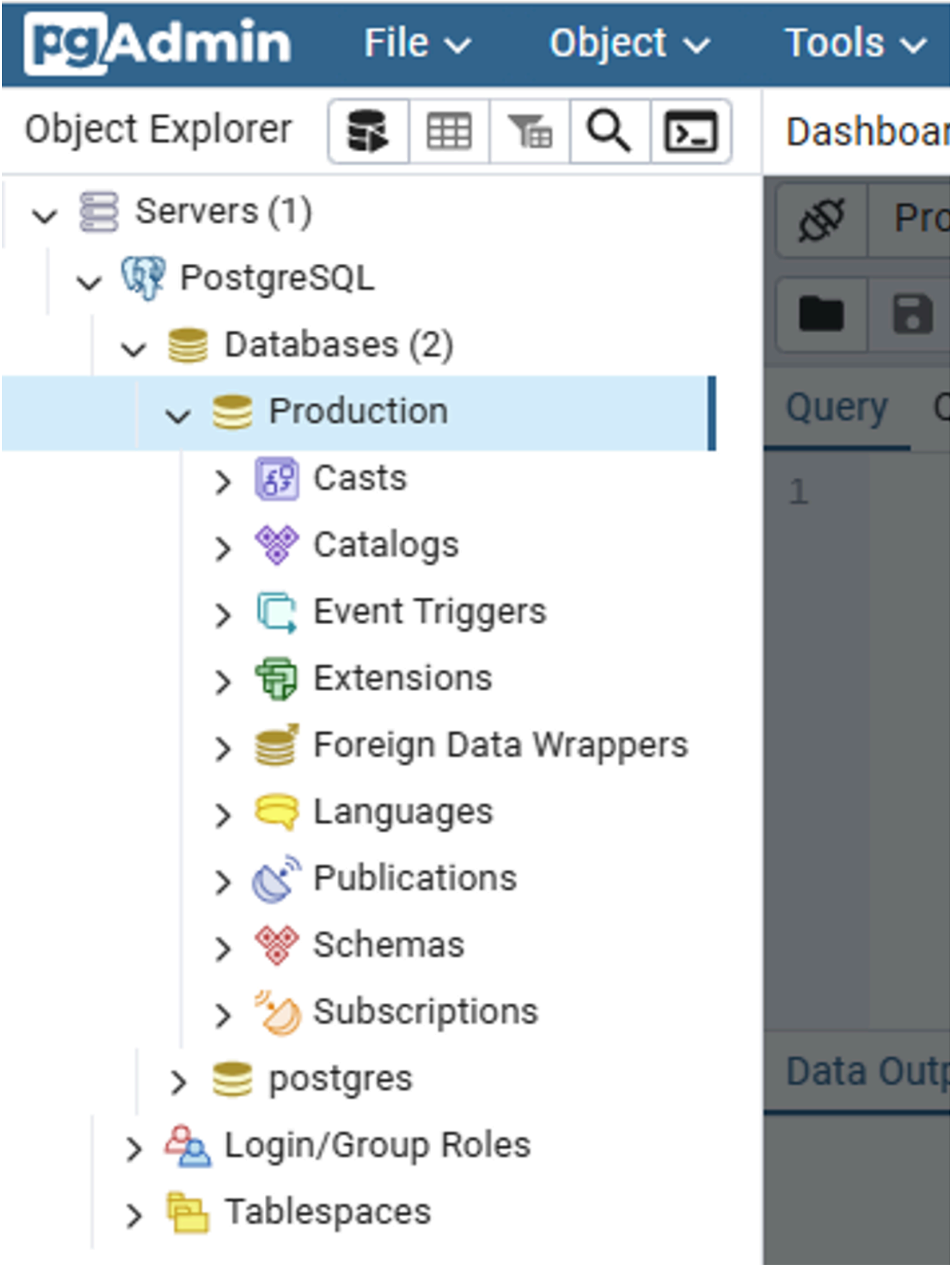

pgAdmin
File
Object
Tools
Object Explorer
Dashboar
Servers (1)
PostgreSQL
Databases (2)
Production
Casts
Catalogs
Event Triggers
Extensions
Foreign Data Wrappers
Languages
Publications
Schemas
Subscriptions
postgres
Login/Group Roles
Tablespaces
Query
Data Outp
1
2

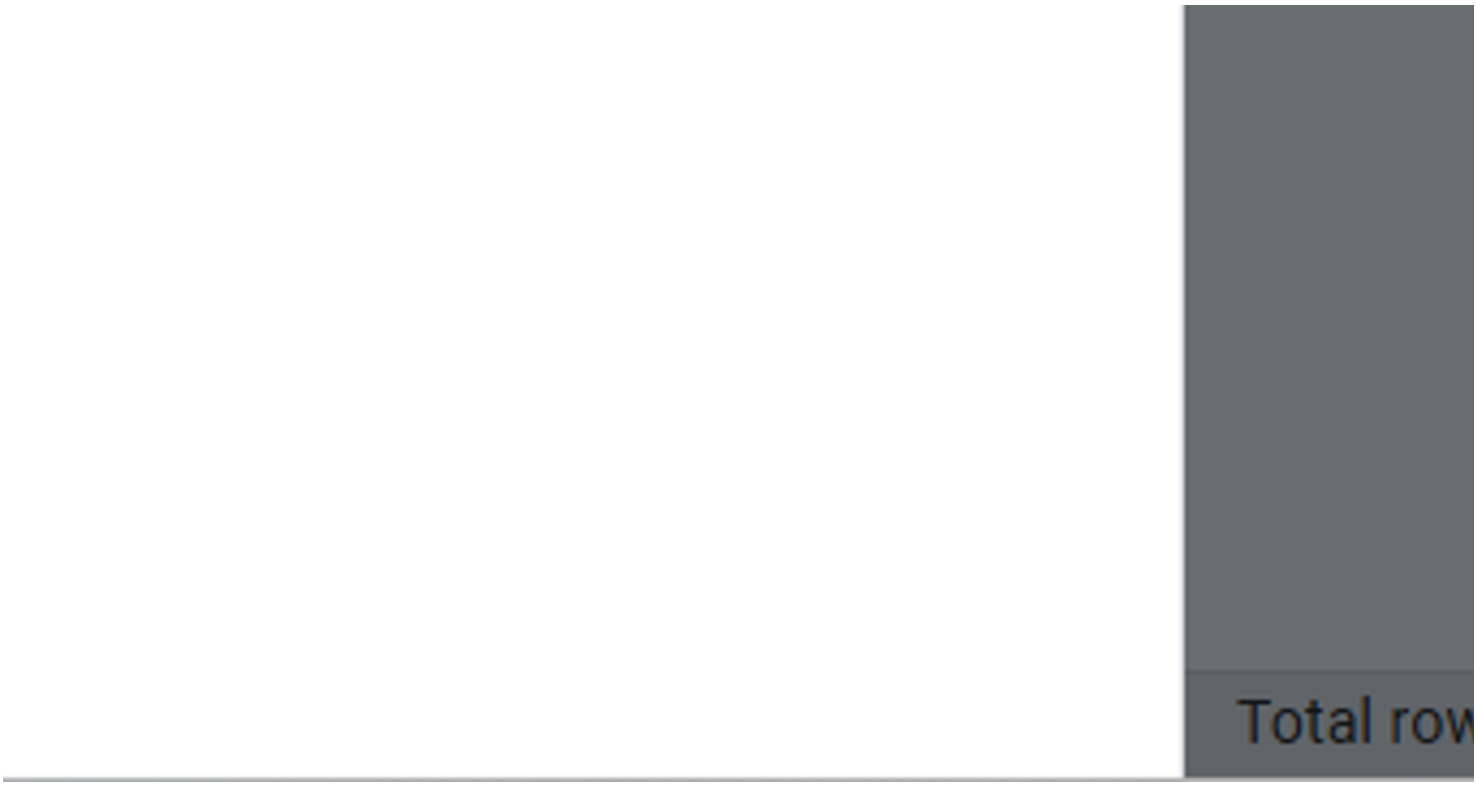
2. In the new blank page that appears drag and drop the **star-schema.sql** file inside the blank page. Once the **star-schema.sql** file is successfully loaded, click on the **X** icon on the right hand side of the page as shown in the screenshot.


pgAdmin
File
Object
Tools
Object Explorer
Dashbo
Servers (1)
PostgreSQL
Databases (2)
Production
Casts
Catalogs
Event Triggers
Extensions
Foreign Data Wrappers
Languages
Publications
Schemas
Subscriptions
postgres
Login/Group Roles
Tablespaces
Query
1
Data Ou

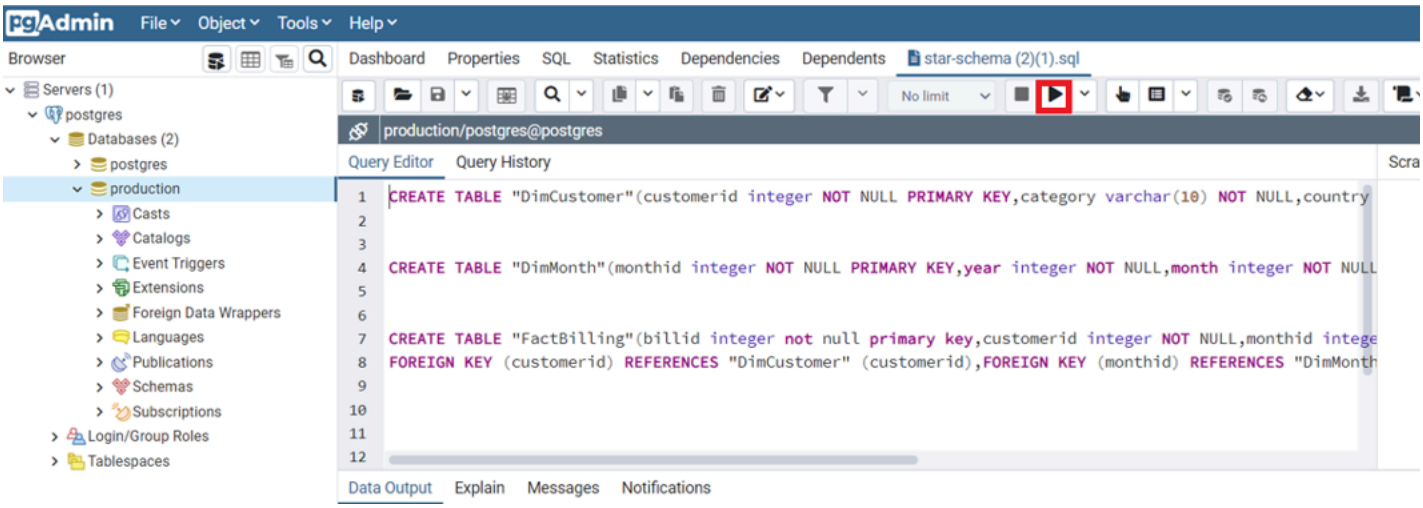


3. Once you click on the **X** icon a new page appears with the file **star-schema.sql**. Select the **star-schema.sql** file from the list and click the **Select** tab.

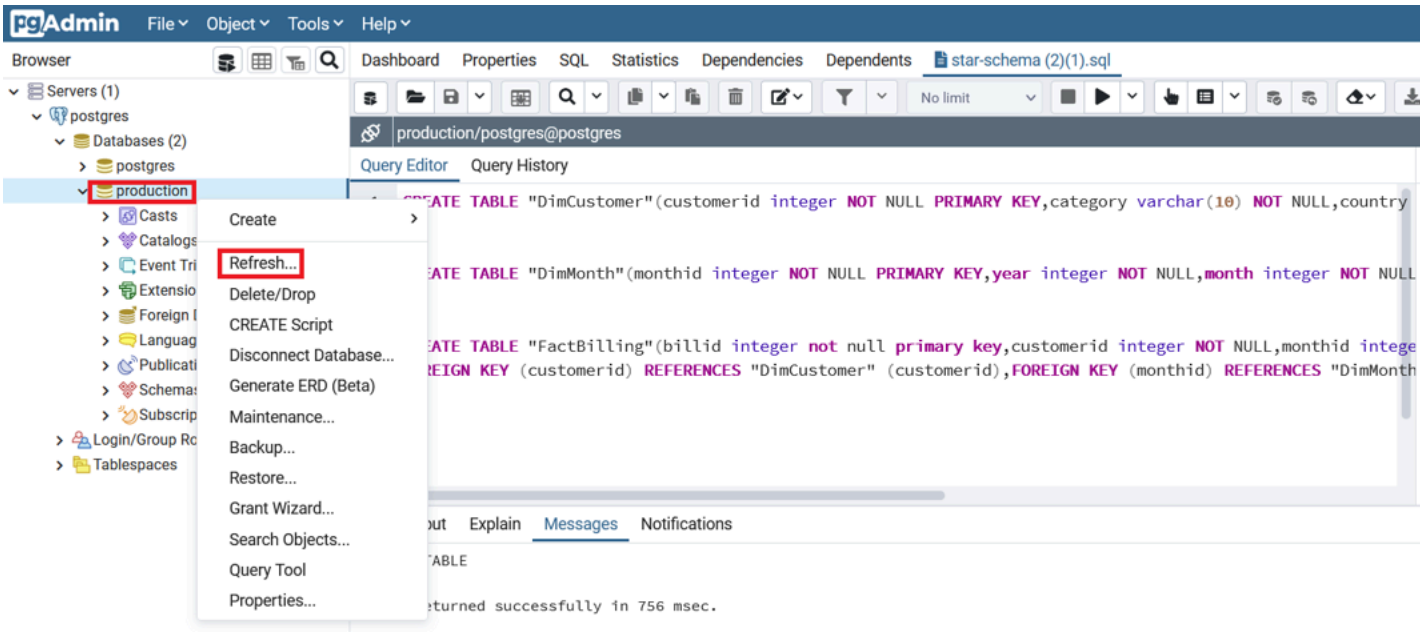

pgAdmin
File
Object
Tools
Object Explorer
Dashboar
Servers (1)
PostgreSQL
Databases (2)
Production
Casts
Catalogs
Event Triggers
Extensions
Foreign Data Wrappers
Languages
Publications
Schemas
Subscriptions
postgres
Login/Group Roles
Tablespaces
Query
1
Data Outp



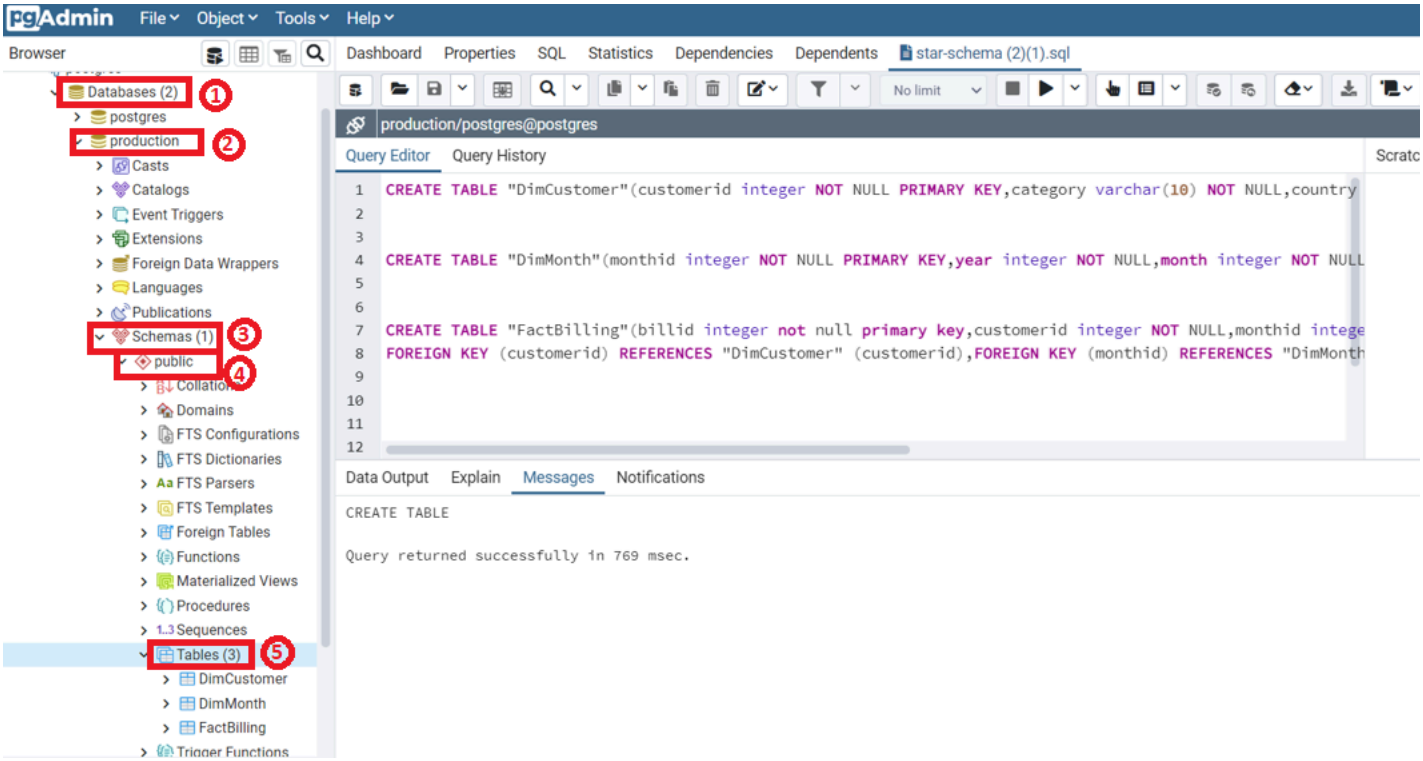
4. Once the file opens up click on the **Run** option to execute the **star-schema.sql** file.



5. Next, right-click on the **Production database** and click on the **Refresh** option from the dropdown.

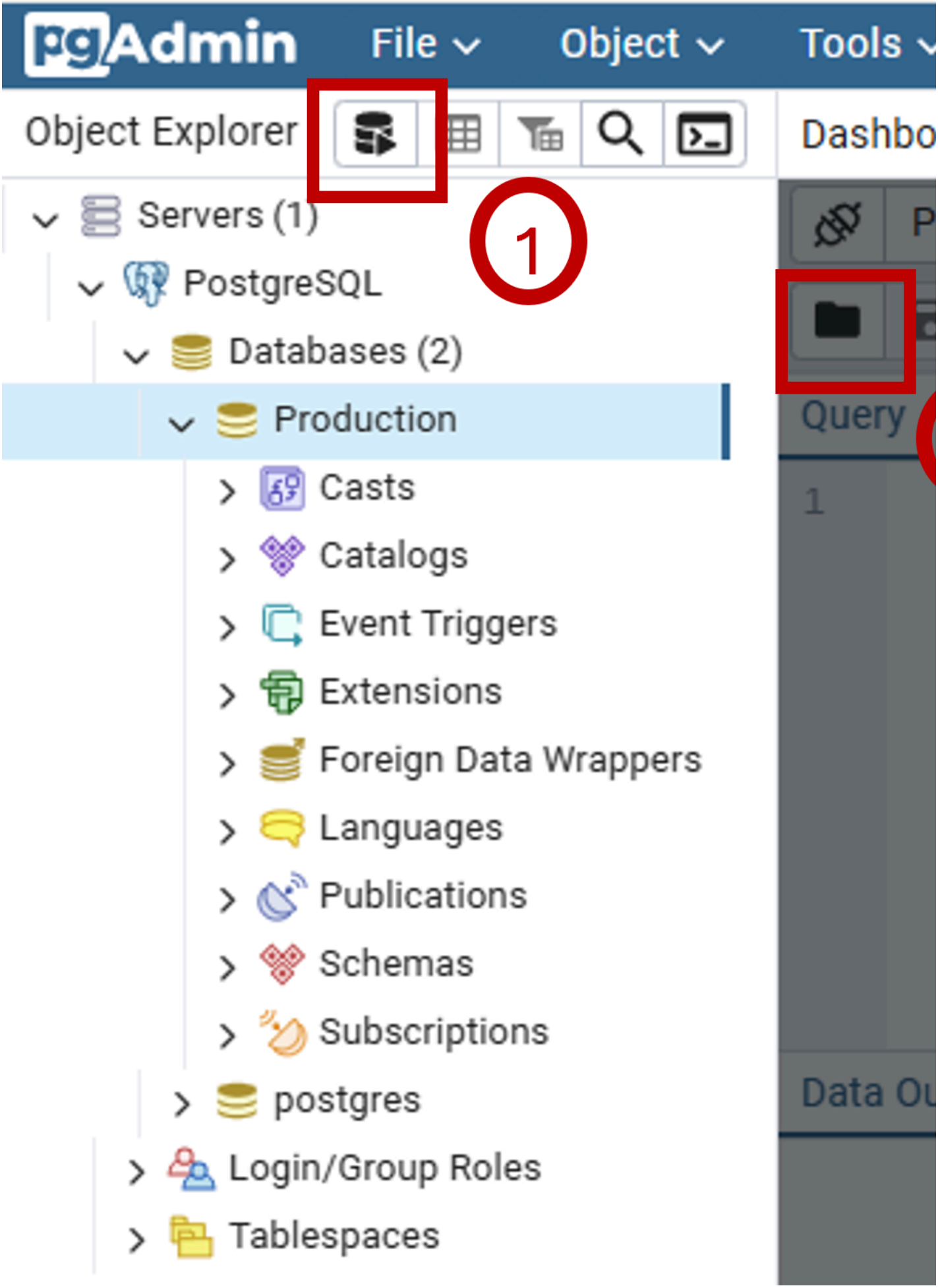


After the database is refreshed the 3 tables (DimCustomer, DimMonth,FactBilling) are created under the **Databases > Production > Schemas > Public > Tables**.



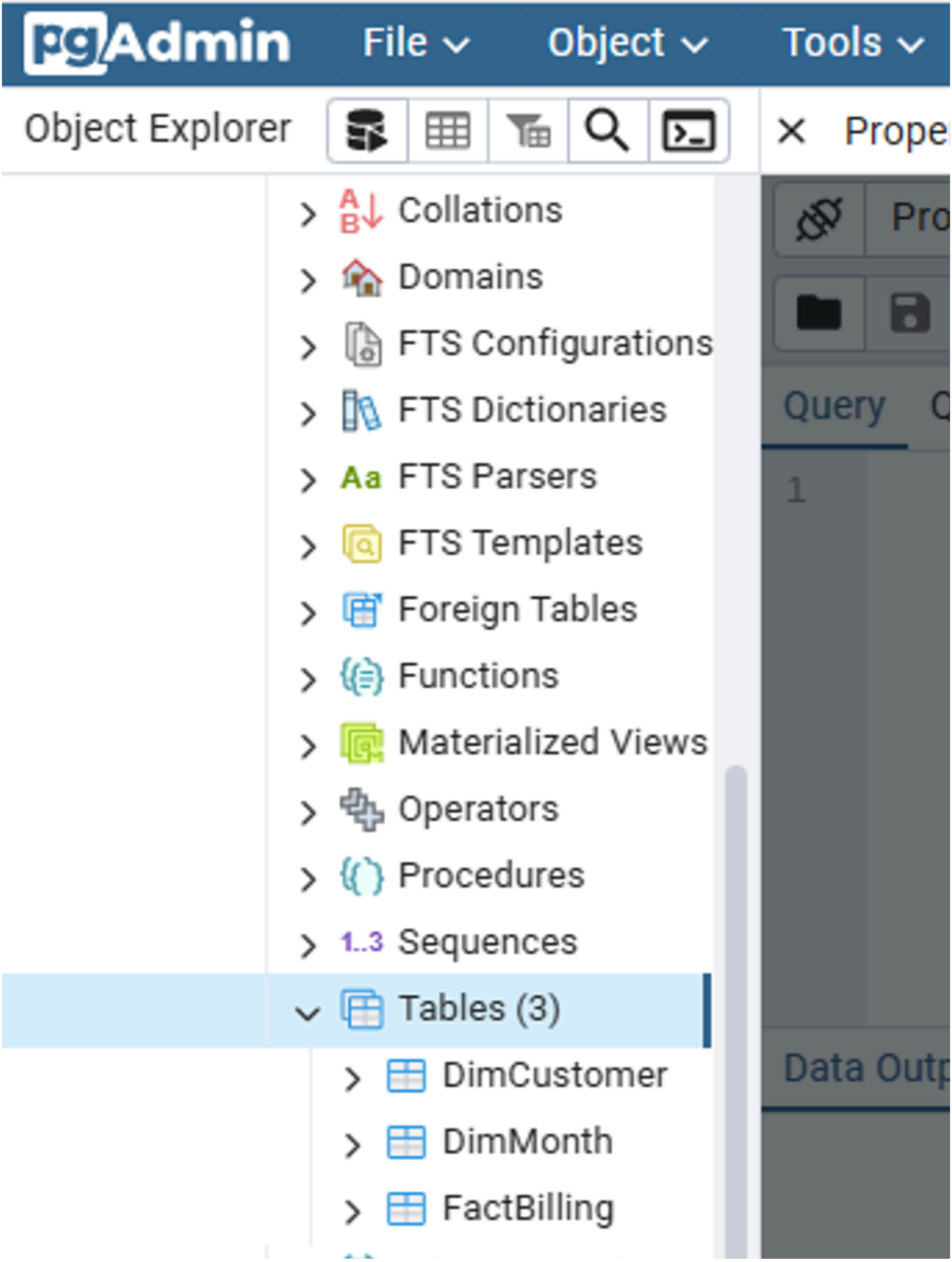
Task C: Load tables

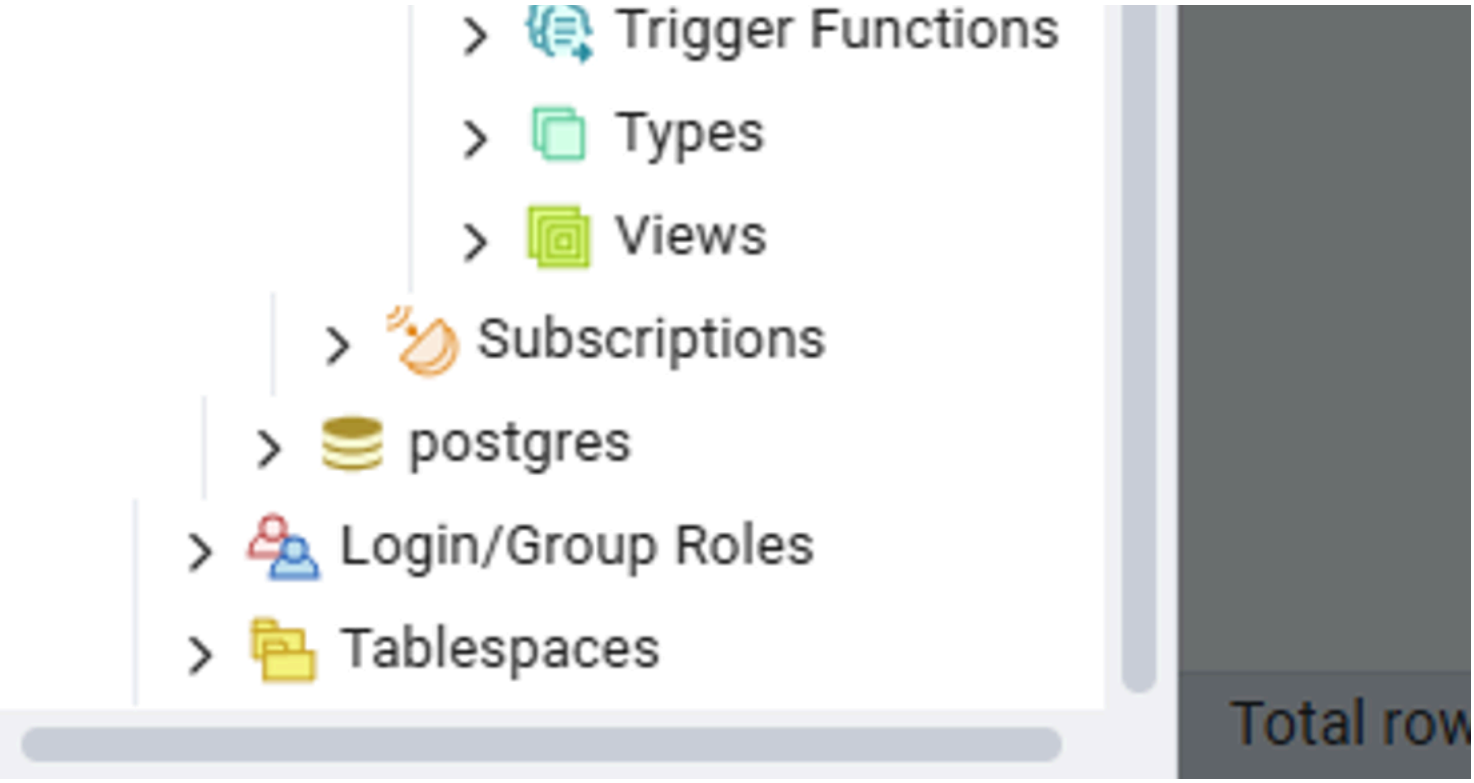
1. Click on **Query tool** and then click **Open** file and click on the three dot and choose **Upload** option.


pgAdmin
File
Object
Tools
Object Explorer
Dashbo
1
Servers (1)
PostgreSQL
Databases (2)
Production
Casts
Catalogs
Event Triggers
Extensions
Foreign Data Wrappers
Languages
Publications
Schemas
Subscriptions
postgres
Login/Group Roles
Tablespaces
Query
1
Data Ou

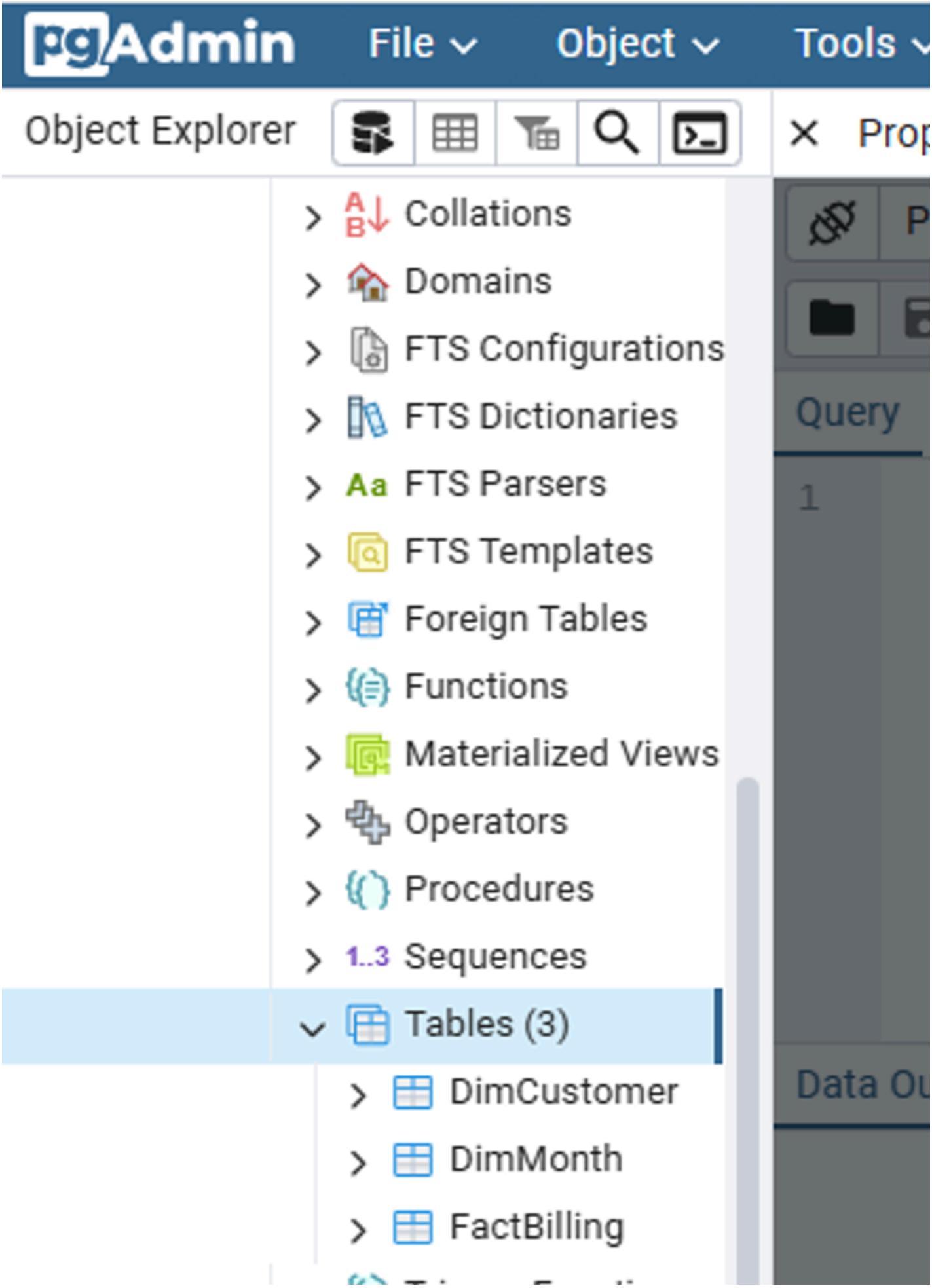
2. In the new blank page that appears drag and drop the **DimCustomer.sql** file inside the blank page. Once the **DimCustomer.sql** file is successfully loaded.

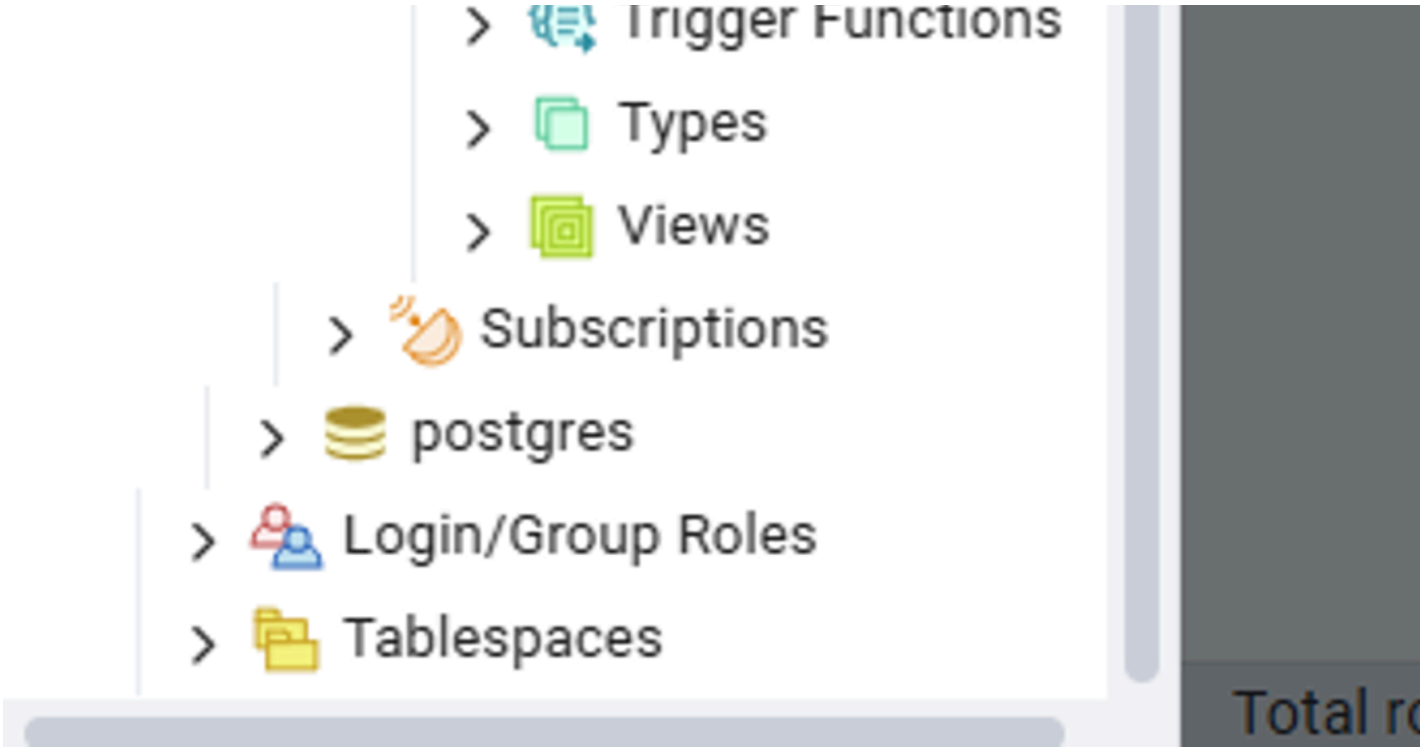
Click on the small **X** icon on the right hand side of the page as shown in the screenshot.


pgAdmin
File
Object
Tools
Object Explorer
Collations
Domains
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Operators
Procedures
Sequences
Tables (3)
DimCustomer
DimMonth
FactBilling
Query
1
Data Outp

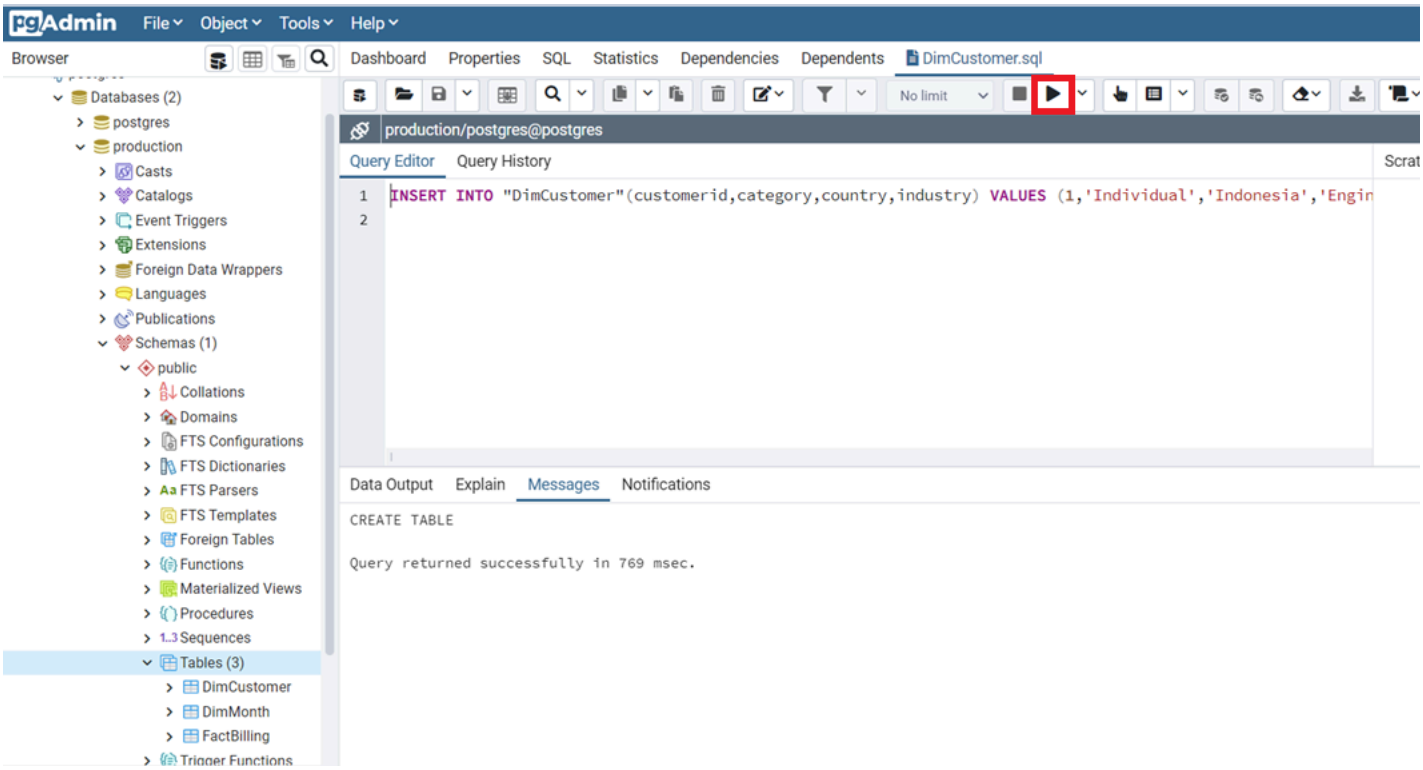


3. Once you click on the **X** icon a new page appears with the file **DimCustomer.sql**. Select the **DimCustomer.sql** file from the list and click on **Select** tab.


pgAdmin
File
Object
Tools
Object Explorer
Collations
Domains
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Operators
Procedures
Sequences
Tables (3)
DimCustomer
DimMonth
FactBilling
Query
Data Ou



4. Once the file opens up, click on the **Run** option to execute the **DimCustomer.sql** file.

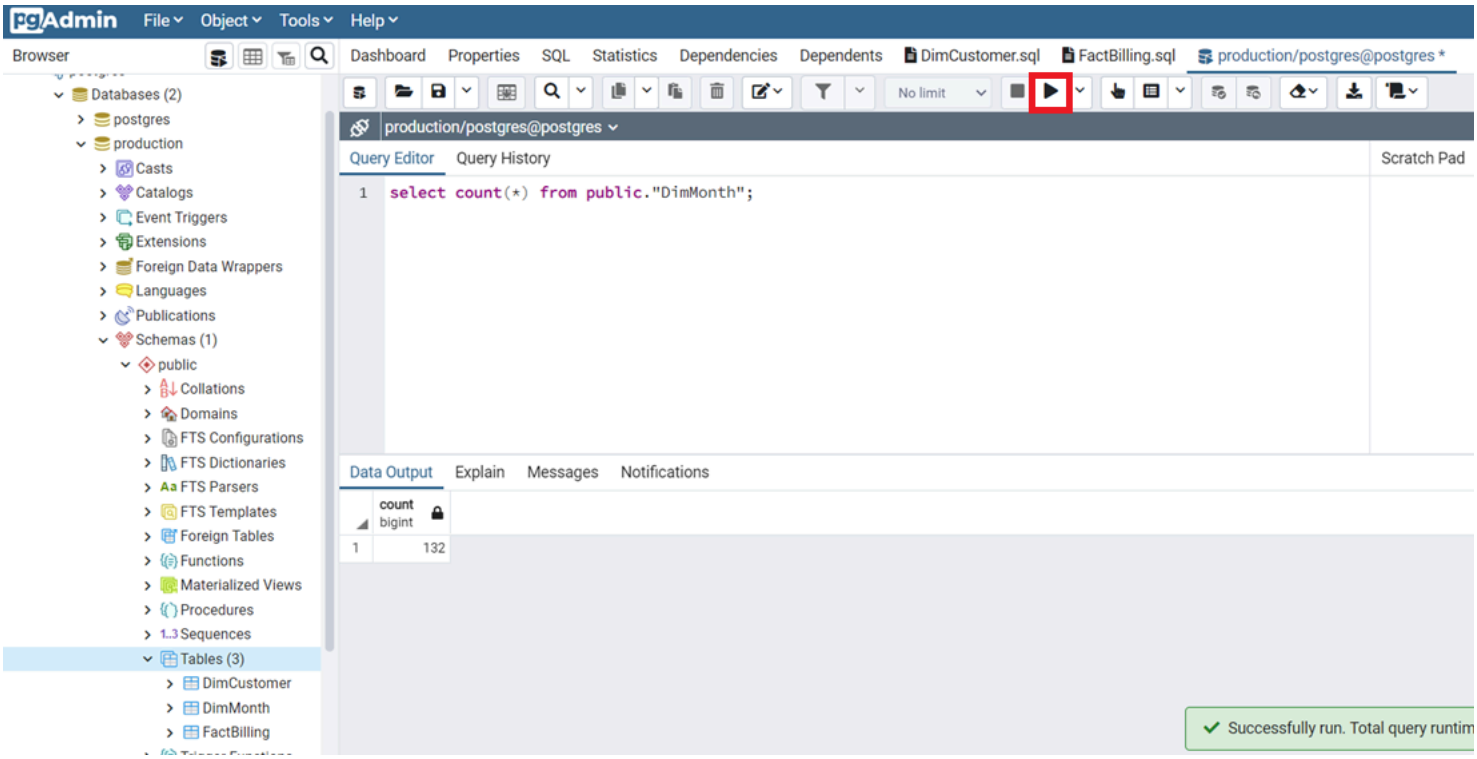


Note: Repeat the steps as given in Task C to upload the remaining sql files to insert data in **DimMonth and FactBilling**.

5. Let's run the command below on the PostgreSQL Tool.

```
select count(*) from public."DimMonth";
```

You should see an output as seen in the image below.



You are encouraged to run more SQL queries.

Practice exercises

Problem 1: Using the PostgreSQL tool, find the count of rows in the table FactBilling

▶ Click here for Hint
▼ Click here for Solution

```
select count(*) from public."FactBilling";
```

Problem 2: Using the PostgreSQL tool, create a simple Materialized views named avg_customer_bill with fields customerid and averagebillamount.

▶ Click here for Hint
▼ Click here for Solution

```
CREATE MATERIALIZED VIEW  avg_customer_bill (customerid, averagebillamount) AS
(select customerid, avg(billedamount)
from public."FactBilling"
group by customerid
);
```

Click the **Run All** Button to run the statement. You should see status as **Success** in the **Result** section.

Problem 3: Refresh the newly created Materialized views

▶ Click here for Hint
▼ Click here for Solution

```
REFRESH MATERIALIZED VIEW avg_customer_bill;
```

Problem 4: Using the newly created Materialized views find the customers whose average billing is more than 11000.

▶ Click here for Hint
▼ Click here for Solution

```
select * from avg_customer_bill where averagebillamount > 11000;
```

Congratulations! You have successfully finished the Populating a Data Warehouse lab.

Author

Amrutha Rao