# Hands-on Lab: Populating a Data Warehouse using PostgreSQL

Estimated time needed: 15 minutes

In this lab, you will learn how to create database and load data in PostgreSQL.

### 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

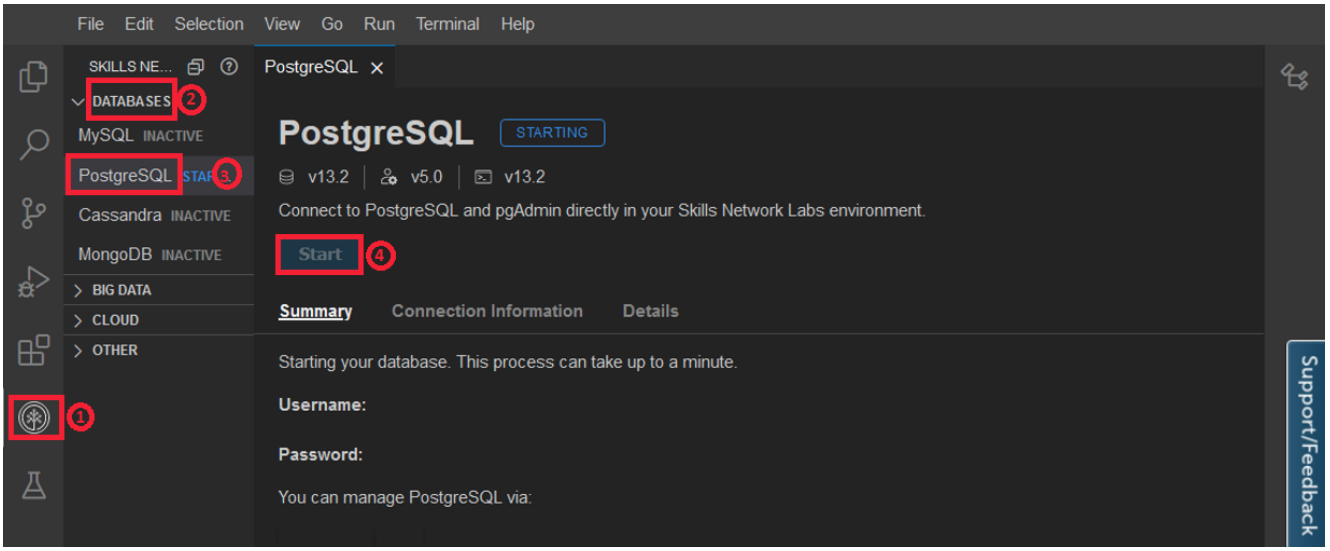
The following are the SQL data files used in this lab.
The production database contains:

- DimCustomer
- DimMonth
- FactBilling
- Star Schema

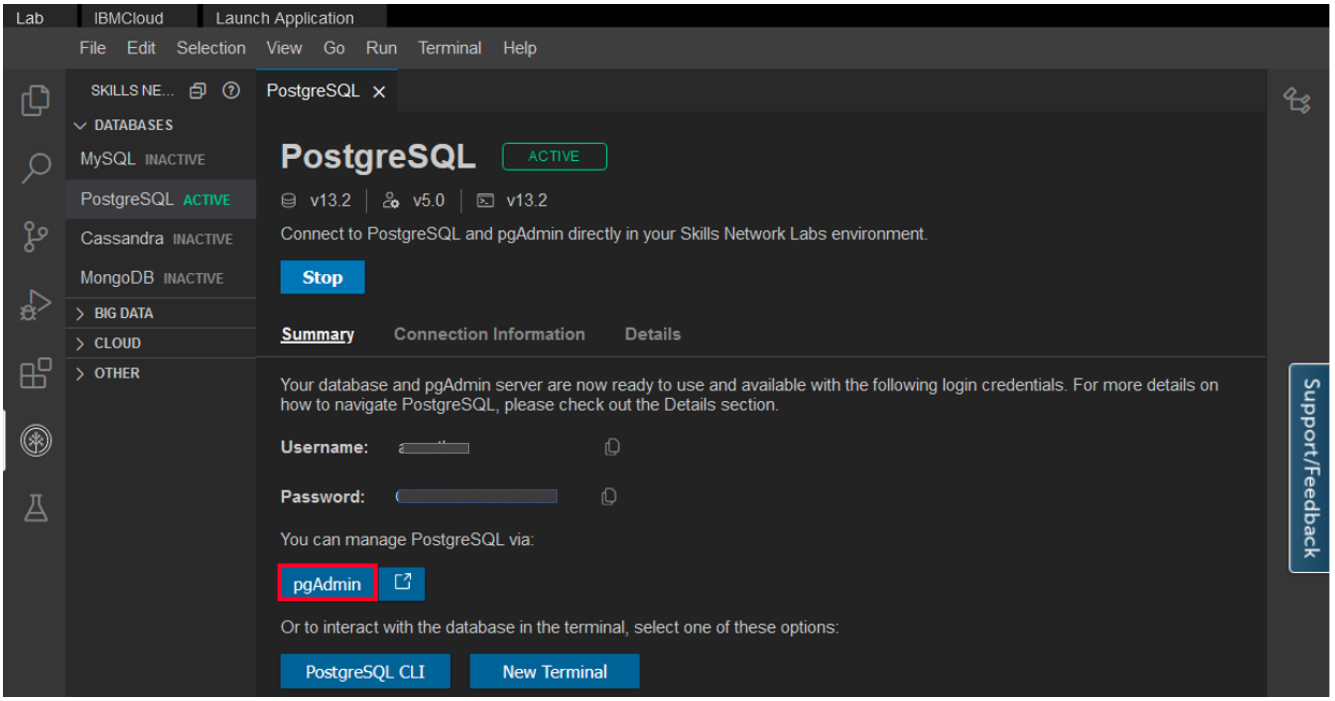
# Task A: Create a database

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.

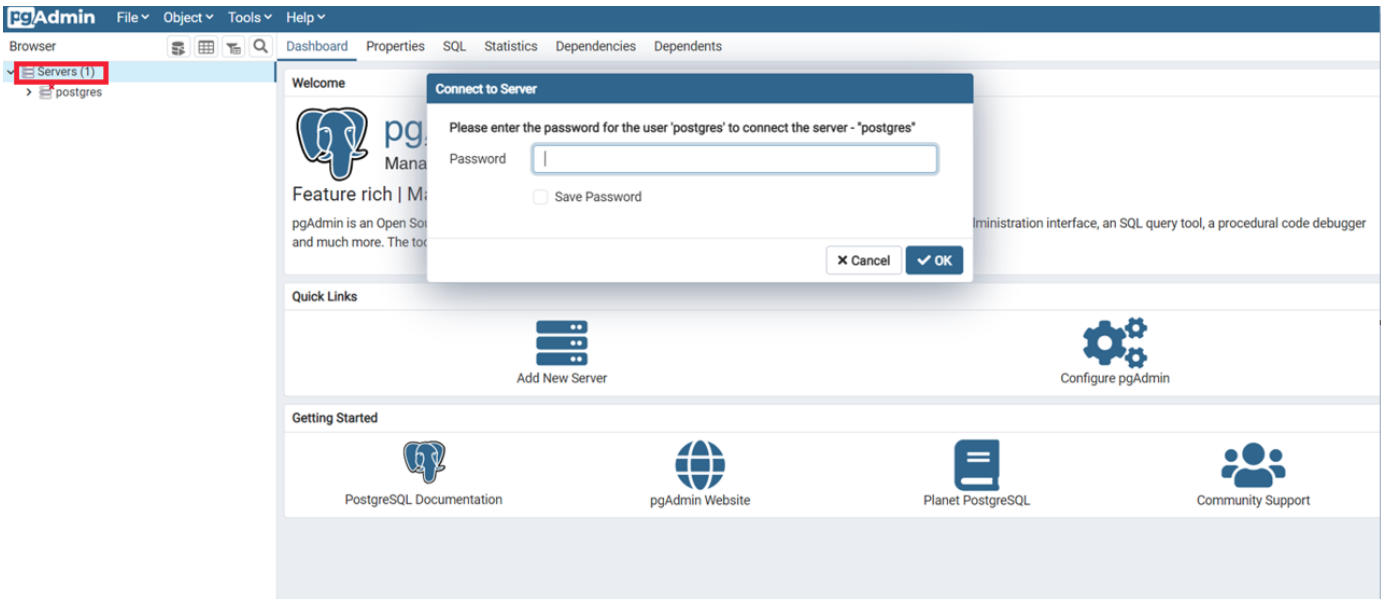
1. Click on the Skills Network extension button on the left side of the window.
2. Open the **DATABASES** drop down menu.
3. Click on **PostgreSQL**
4. Click on the **Start** button. PostgreSQL may take a few moments to start.



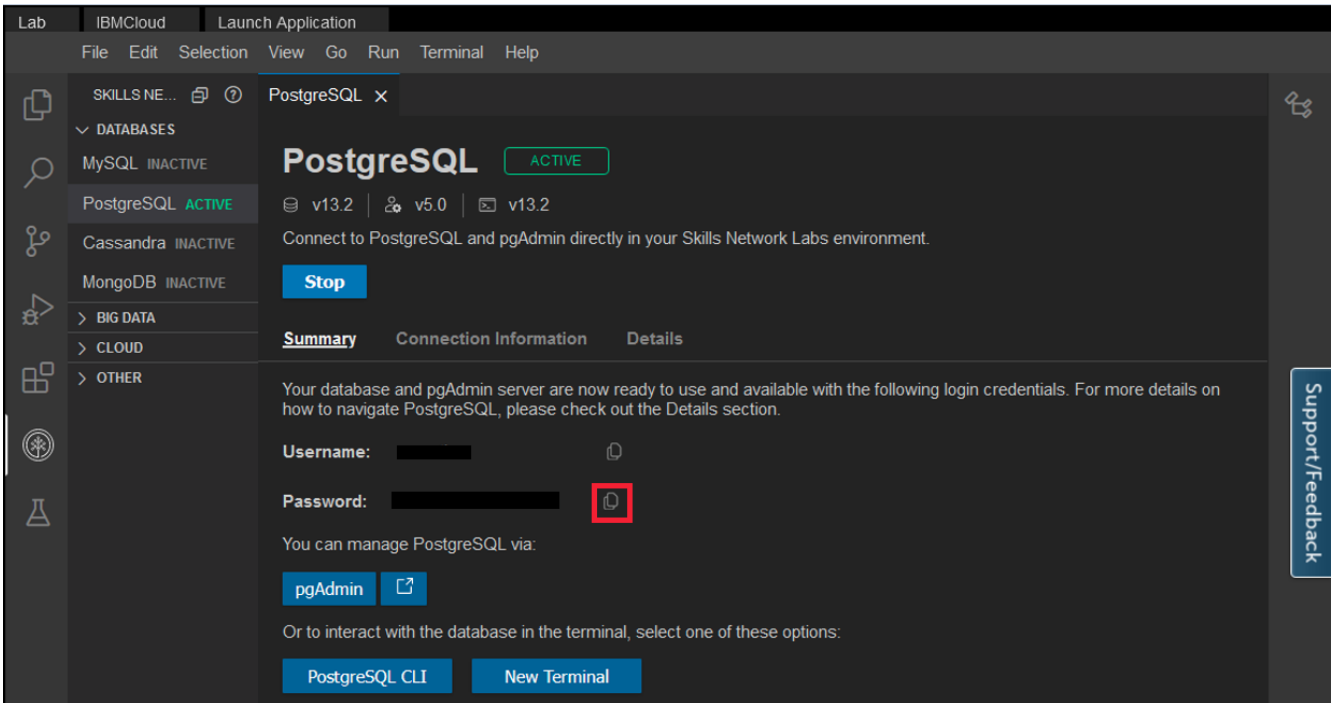

1. Next, open the pgAdmin Graphical User Interface by clicking the **pgAdmin** button in the Cloud IDE interface.



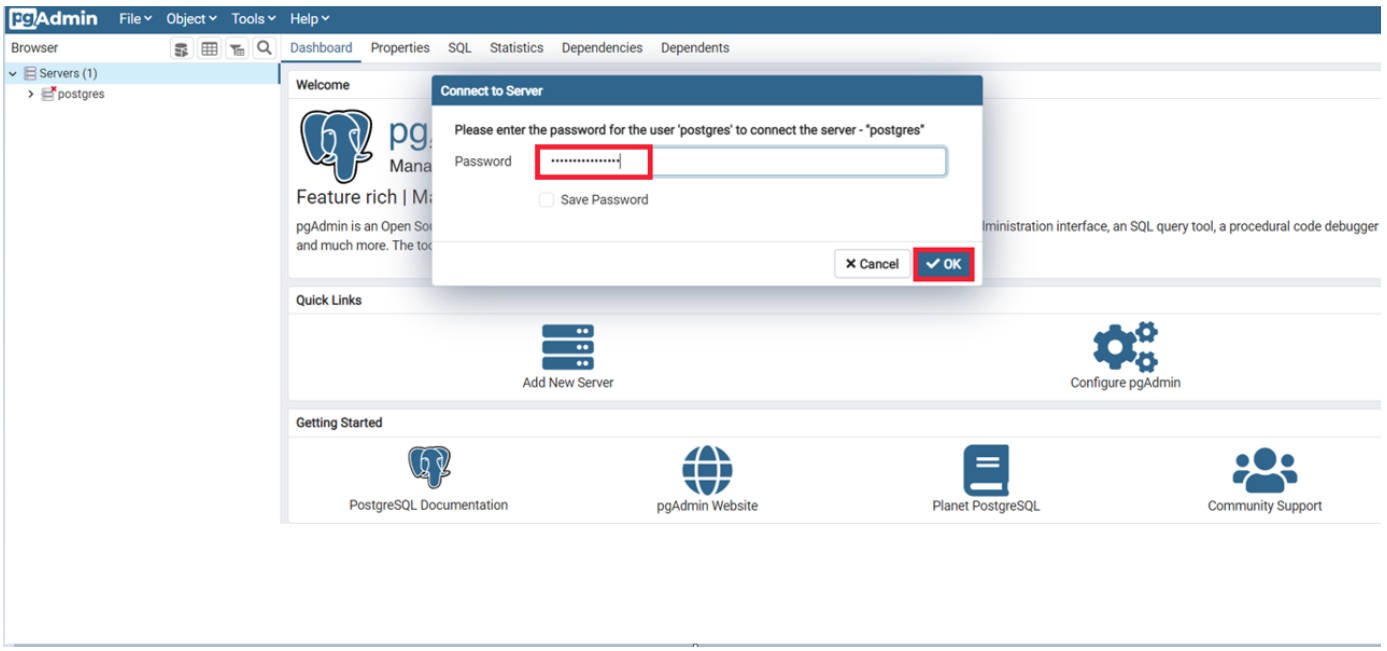

1. Once the pgAdmin GUI opens, click on the **Servers** tab on the left side of the page. You will be prompted to enter a password.



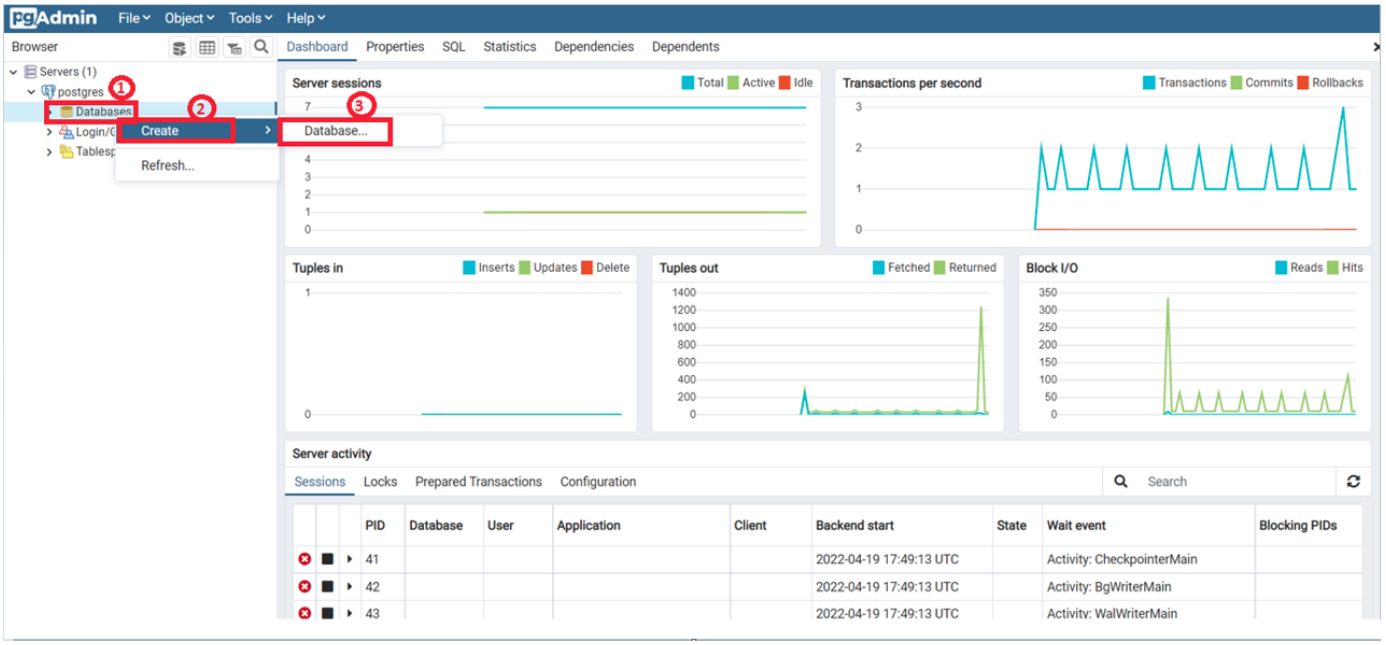

1. To retrieve your password, click on the **PostgreSQL** tab near the top of the interface.
2. Click on the **Copy** icon to the left of your password to copy the session password onto your clipboard.



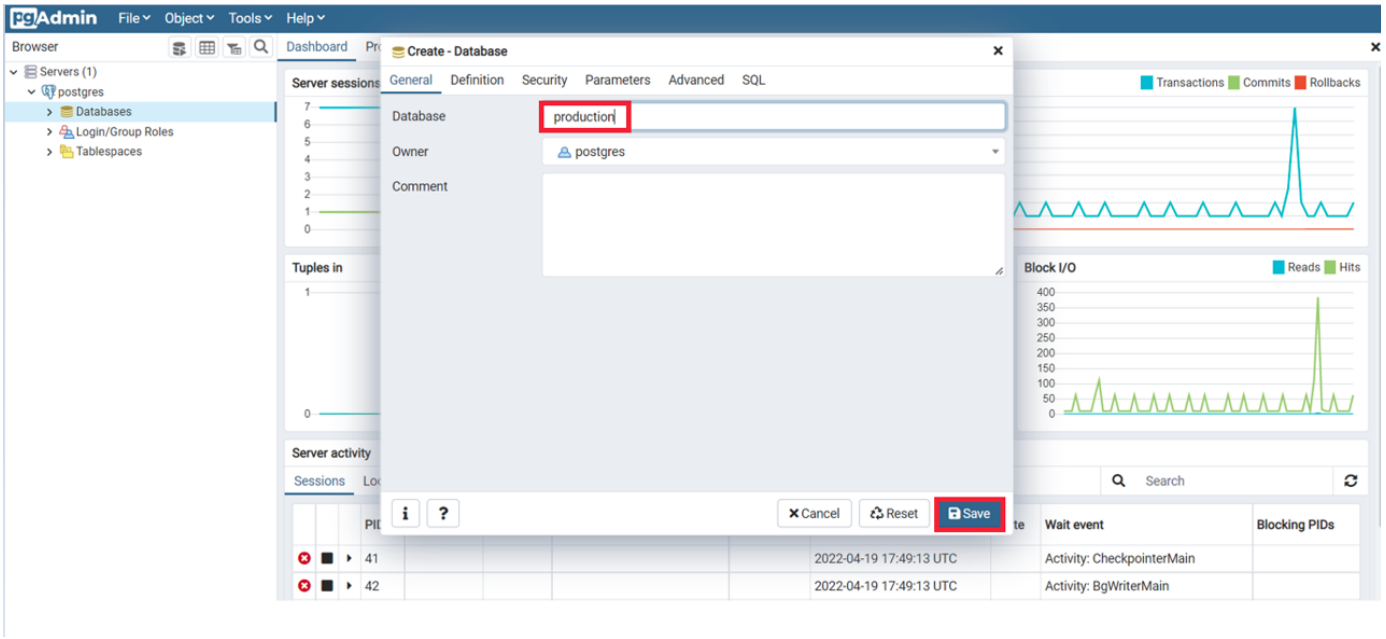

1. Navigate back to the **pgAdmin** tab and paste in your password, then click OK.




1. You will then be able to access the pgAdmin GUI tool.
2. In the left tree-view, right-click on **Databases> Create > Database**.



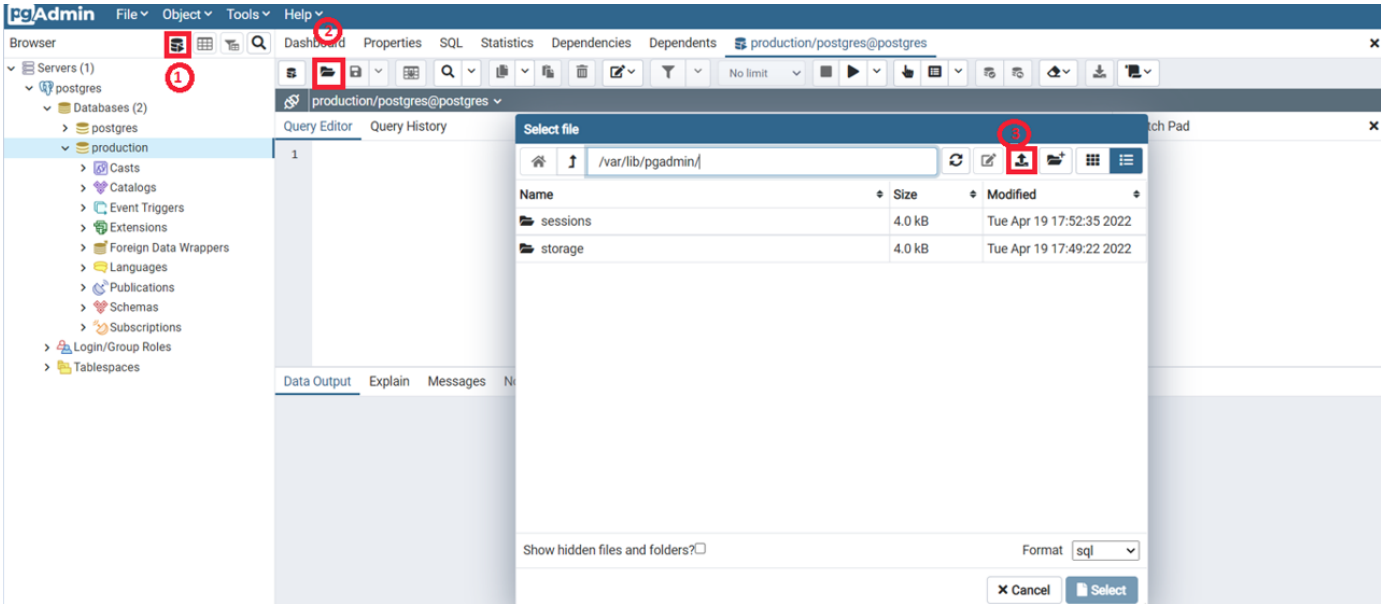

In the Database box, type **Production** as the name for your new database, and then click **Save**. Proceed to Task B.



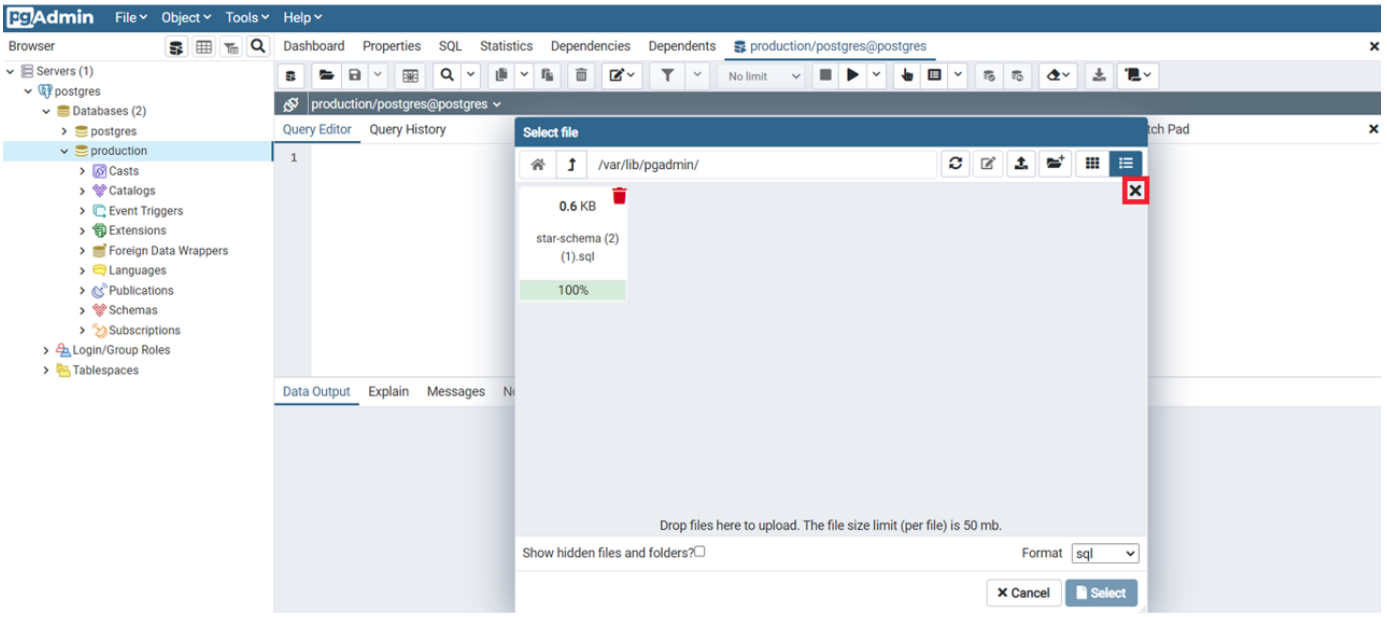

# Task B: Create tables

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.

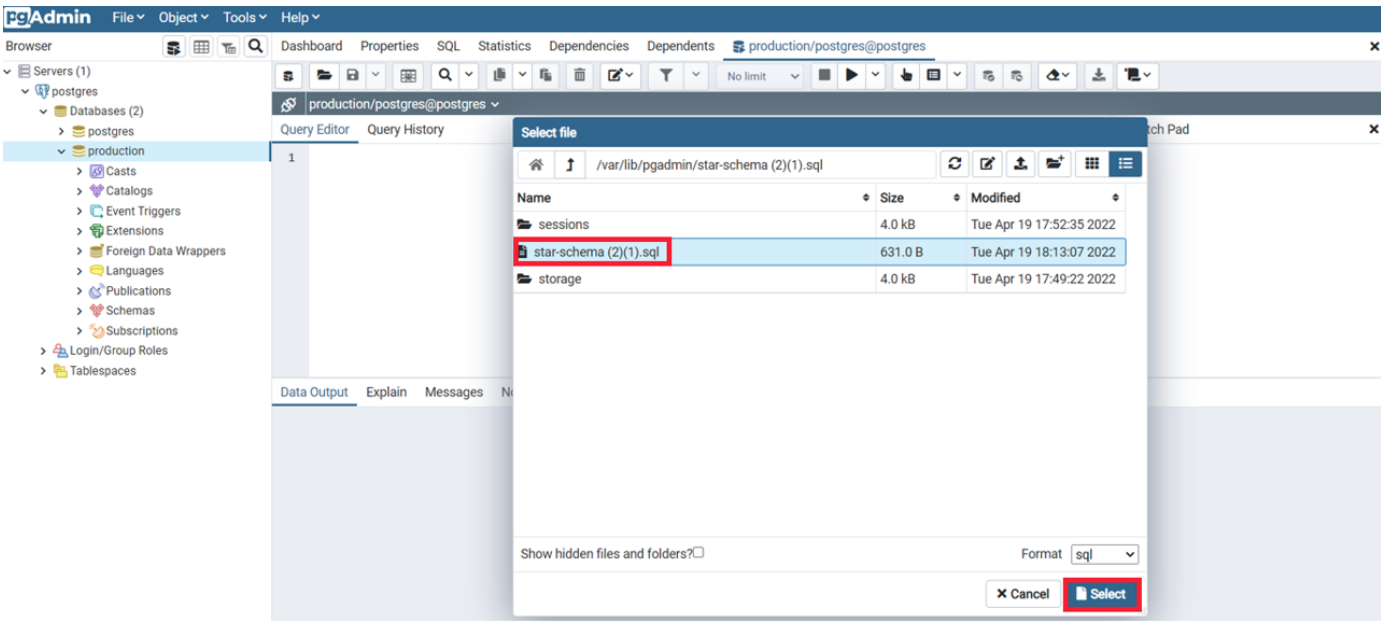
1. In the top of the page go to **Query tool"** and then click on **Open File**. Next a new page pops up called **Select File**. Click on **Upload** icon as shown in the screenshot.



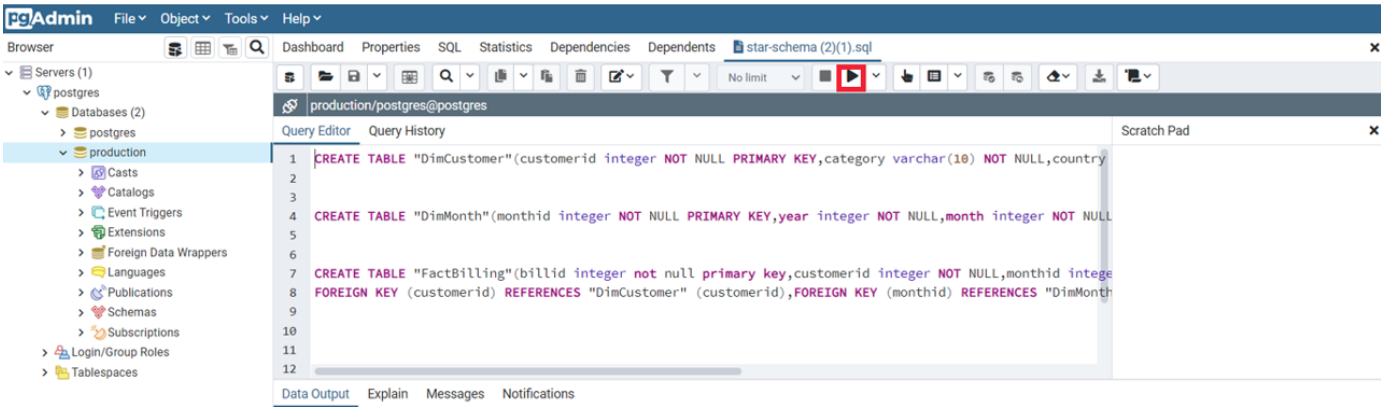

1. In the new blank page that appears drag and drop the **star-schema.sql** file inside the blank page. Once the **star-schema.sql** file is successfully loaded, click on the **X** icon on the left hand side of the page as shown in the screenshot.




1. 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 on **Select** tab.

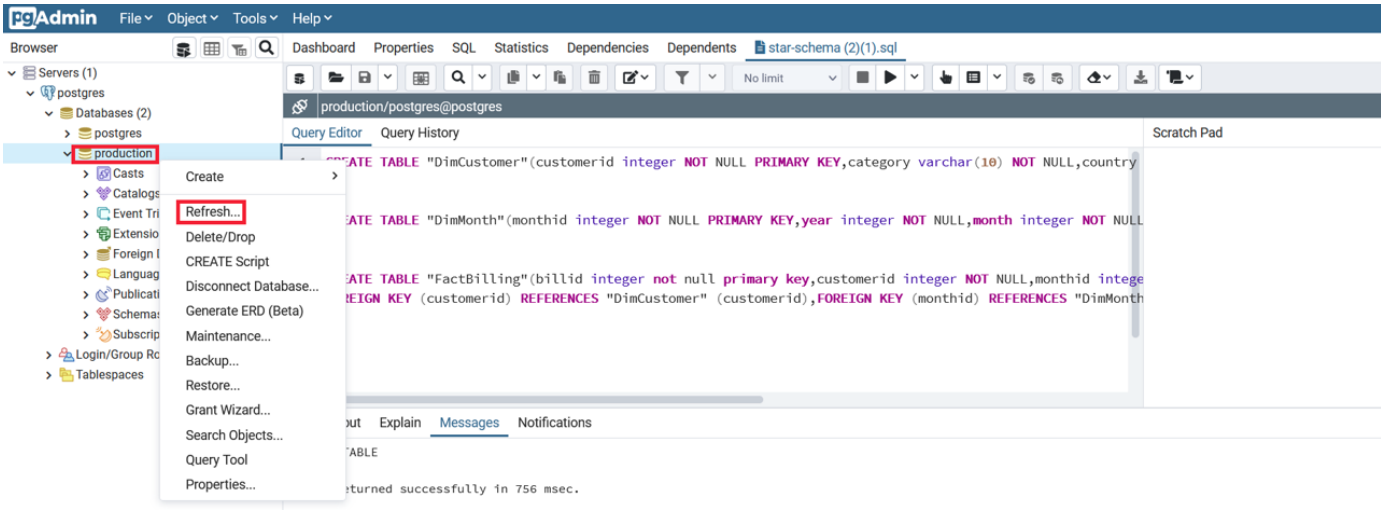


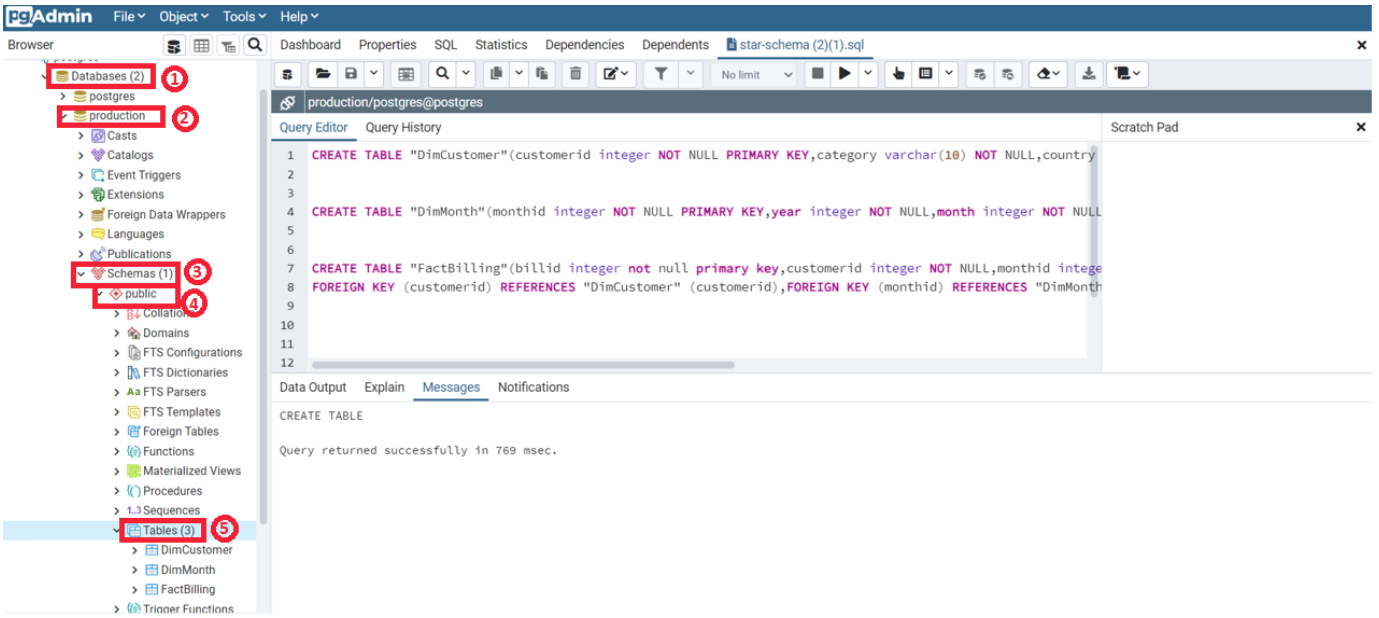

1. Once the file opens up click on the **Run** option to execute the **star-schema.sql** file.




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

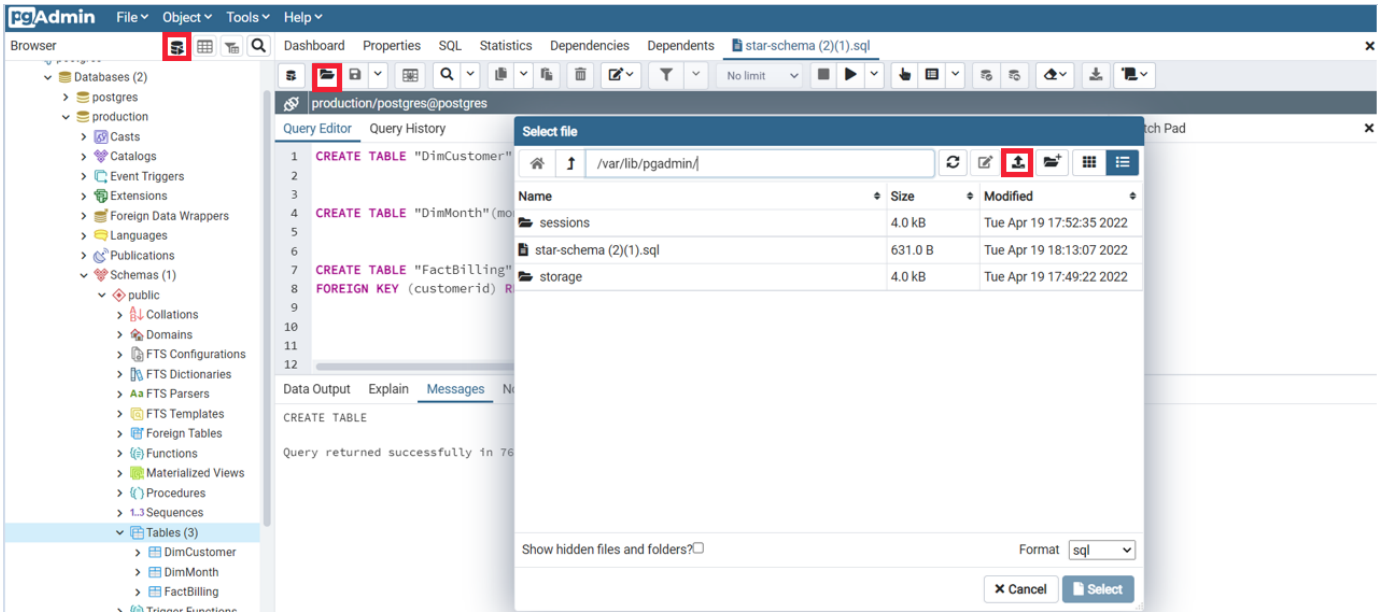
After the database is refreshed the 3 tables(DimCustomer, DimMonth,FactBilling) are created under the **Databases > Production > Schema > Public > Tables**.



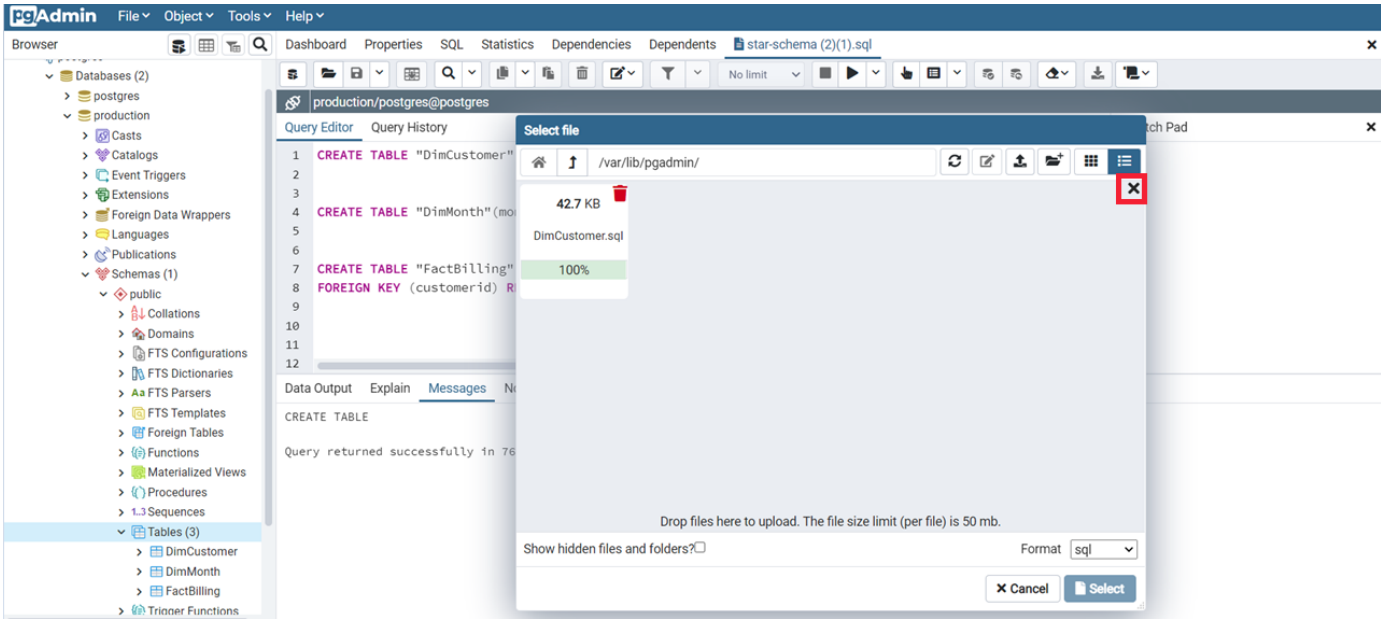

# Task C: Load tables

1. Click on **Query tool** and then click **Open** file and click on **Upload** icon.

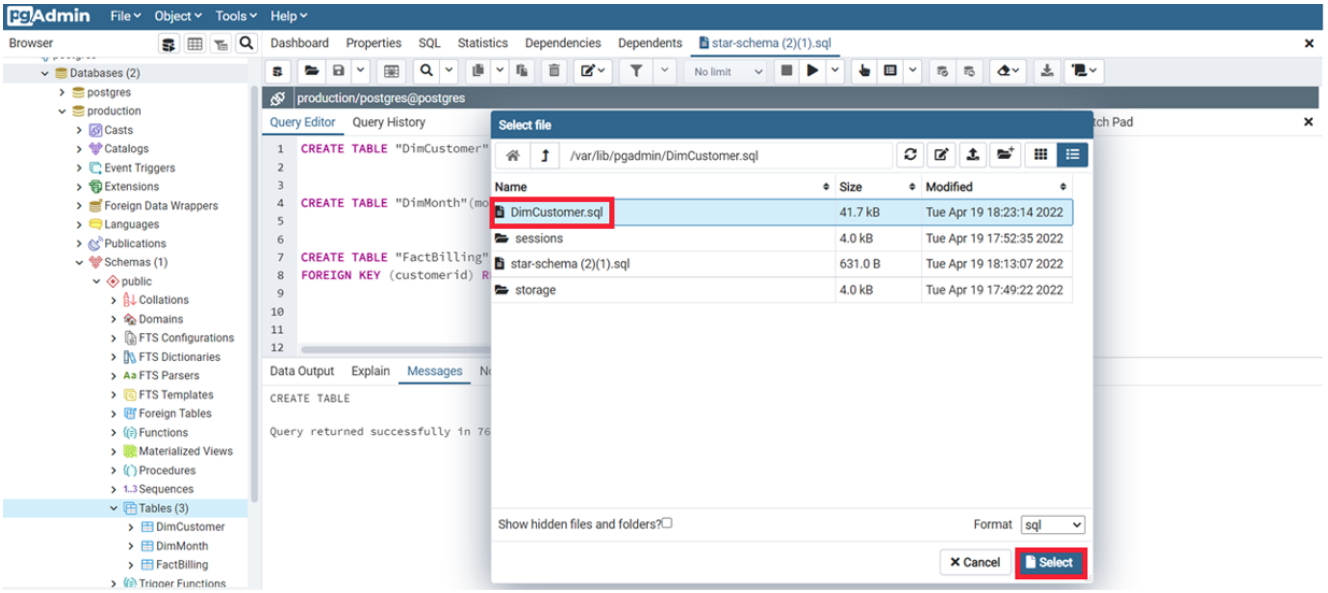



1. In the new blank page that appears drag and drop the **DimCustomer.sql** file inside the blank page. Once the **DimCustomer.sql** file is successfully loaded.

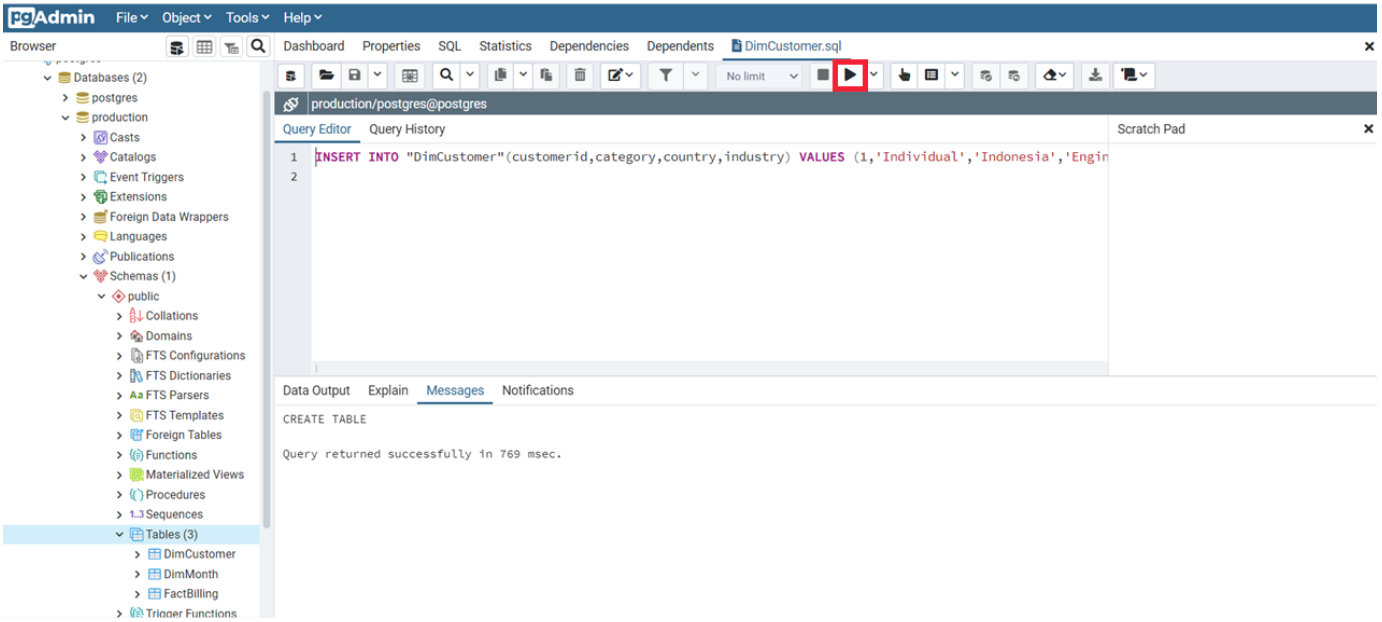
Click on the small **X** icon on the left hand side of the page as shown in the screenshot.




1. 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.




1. 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**.

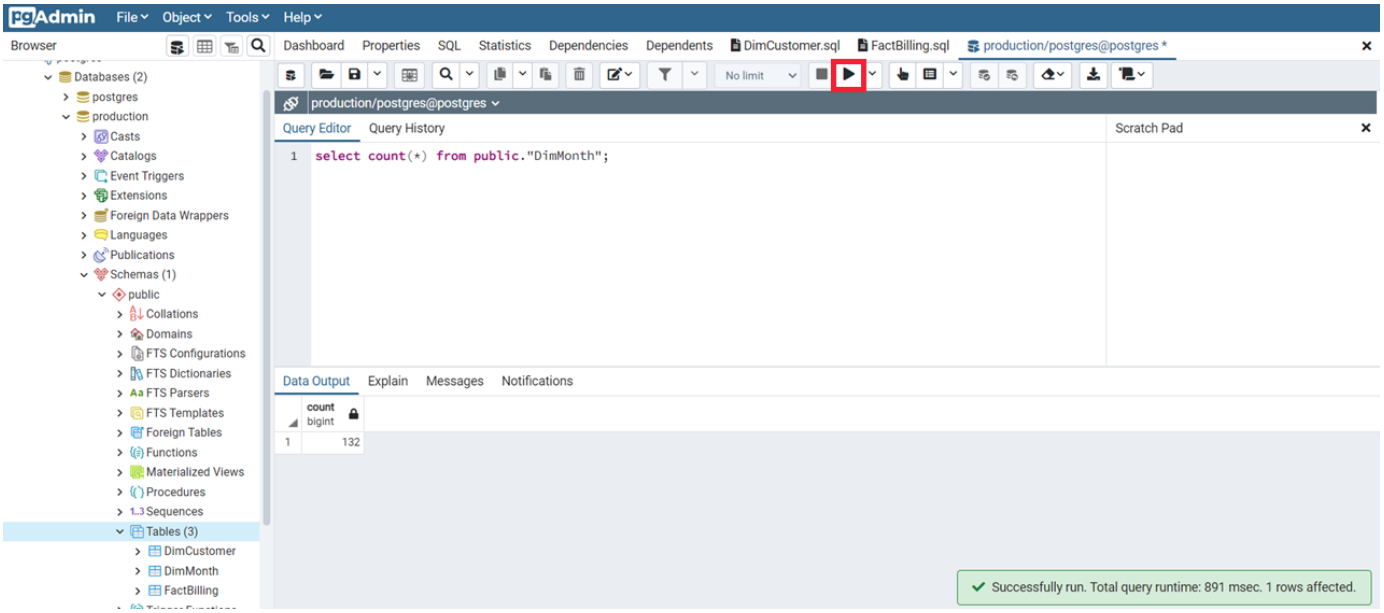
1. Let's run the command below on the PostgreSQL Tool.

```
1
```

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

Copied!

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

Use the select statement along with count function on the table FactBilling.

▾ Click here for Solution

```
1. 1
```

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

Copied!

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

▸ Click here for Hint
▾ Click here for Solution

```
1. 1
2. 2
3. 3
4. 4
5. 5
```

```
1. CREATE MATERIALIZED VIEW  avg_customer_bill (customerid, averagebillamount) AS
2. (select customerid, avg(billedamount)
3. from public."FactBilling"
4. group by customerid
5. );
```

Copied!

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 MQT**

▾ Click here for Hint

Use the refresh materialized view command.

▾ Click here for Solution

```
1. 1
```

```
1. REFRESH MATERIALIZED VIEW avg_customer_bill;
```

Copied!

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

▸ Click here for Hint
▾ Click here for Solution

```
1. 1
```

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

Copied!

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

## Author

Amrutha Rao

## Change Log

| Date (YYYY-MM-DD) | Version | Changed By | Change Description |
|---|---|---|---|
| 2022-04-21 | 0.2 | Amrutha Rao | Converted initial version to postgreSQL workaround. |
| 2021-09-29 | 0.1 | Ramesh Sannareddy | Created initial version of the lab |