PostgreSQL Setup Guide

Introduction

PostgreSQL (commonly called Postgres) is a powerful, open-source relational database management system widely used in the industry. In this class, you'll work with a locally hosted Postgres instance. It is also possible to set up a cloud-hosted Postgres instance on AWS.

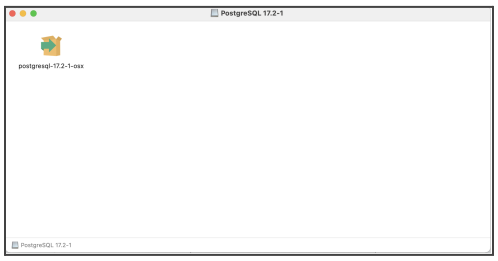
Installing PostgreSQL

1. Download the installer certified by EDB for all supported PostgreSQL versions.

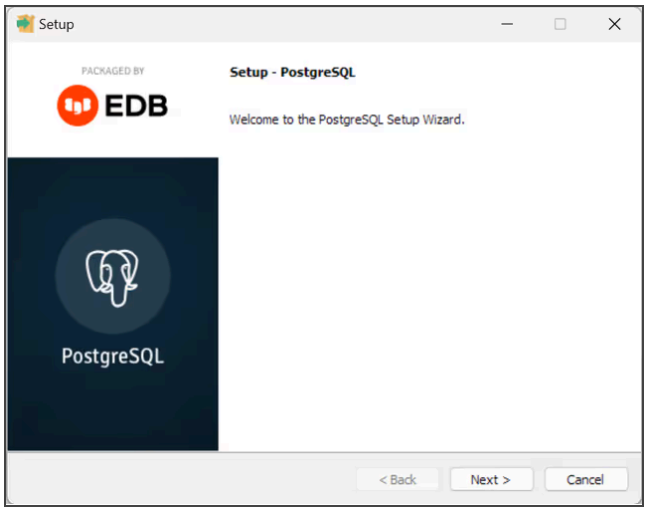
   Note: We recommend the latest version for your OS.

2. Double-click on the installer file and an installation wizard will appear. Click **Next**.

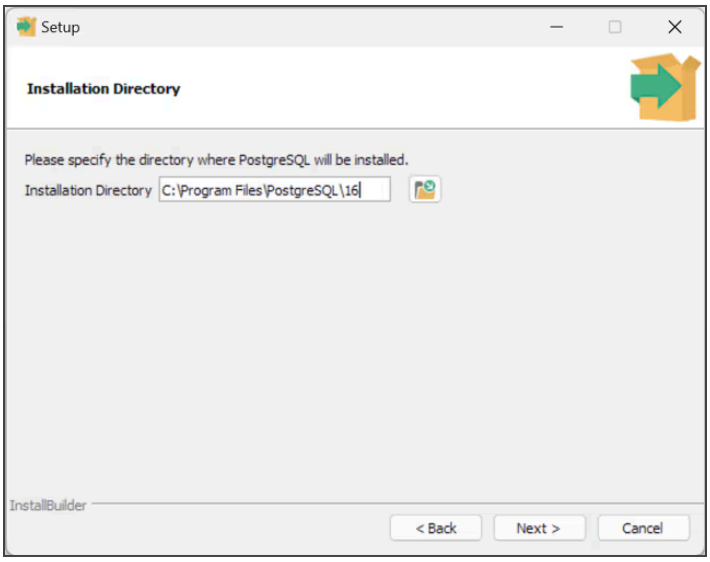
   Note: For Mac you will click on the download and see:



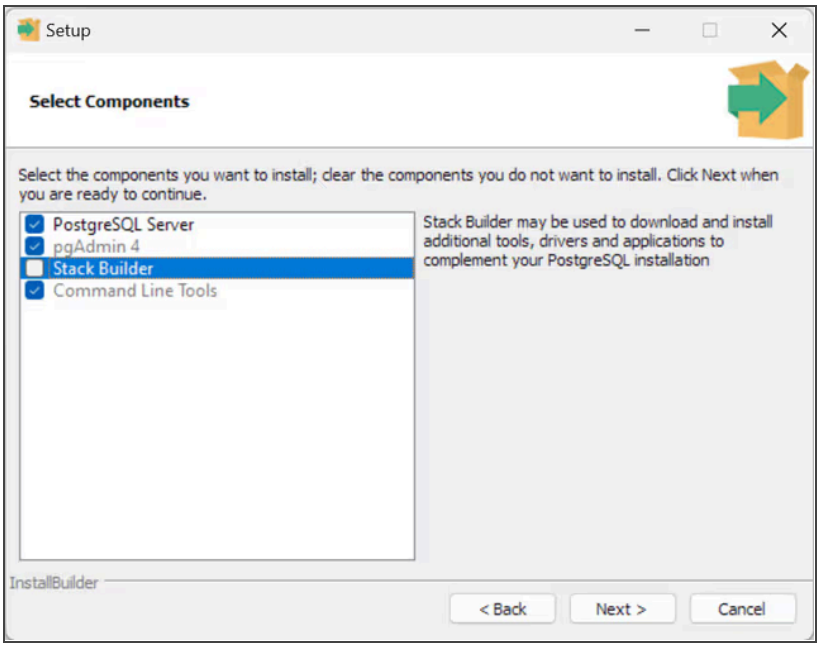
   You'll want to double click on it and enter your password to allow it to open.



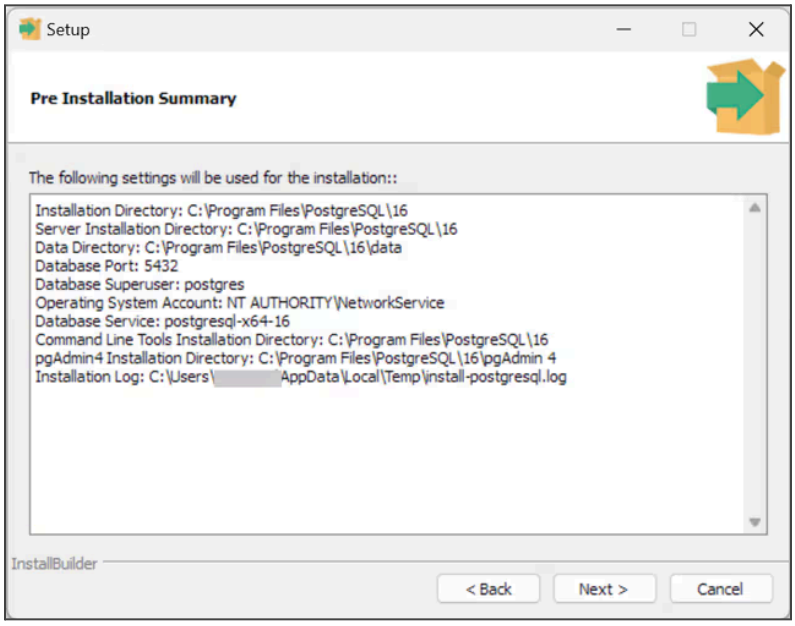
3. Specify the installation directory by either choosing your preferred location or using the default folder suggested by the PostgreSQL installer. Take note of this directory and click **Next**.



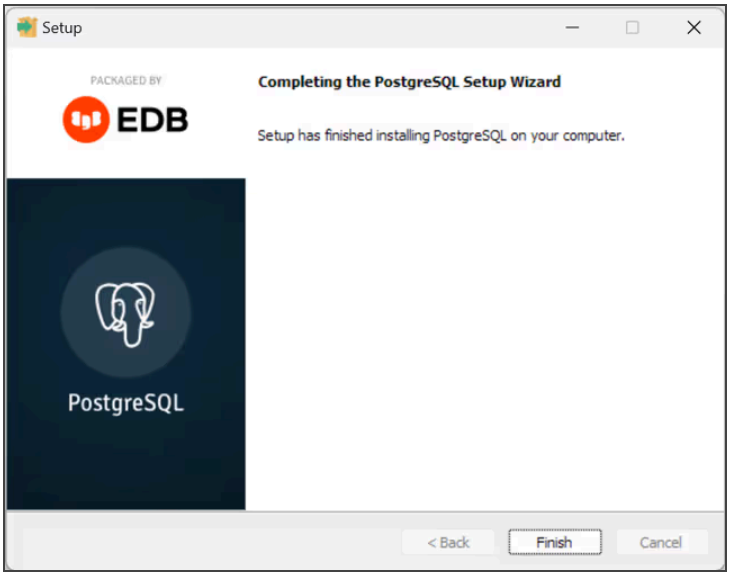
4. Uncheck **Stack builder** and click **Next**.



5. Pick a directory to store the Postgres data, or use the default directory. Click **Next**.
6. Pick a password for the Postgres superuser (postgres). It's important you remember your password. After entering the password, retype it for confirmation, then click **Next**.
7. Use the default port **5432**. Click **Next**.
8. Use the **default locale**. Click **Next**.
9. If the pre-installation summary is correct, click **Next**.



10. You'll see a message that installation is ready. Click **Next** to begin the installation.
11. Click **Finish** and you're done with the installation!


Setup
PACKAGED BY
EDB
Completing the PostgreSQL Setup Wizard
Setup has finished installing PostgreSQL on your computer.
PostgreSQL
< Back
Finish
Cancel

Edit PATH

Adding the bin directory of PostgreSQL to the PATH environment variable allows you to execute the PostgreSQL client psql from any directory, without the need to navigate the bin directory first.

On Windows:

1. Go to **Control Panel > System Properties > Advanced > Environment Variables**.
2. Select **PATH** and click **Edit**.
3. Append a semi-colon (;) followed by the "bin" directory path of PostgreSQL.
    a. Depending on your version of windows, you might see rows of existing paths, click new, paste your path maybe be different than -> (C:\Program Files\PostgreSQL\17\bin) without a (;)
    b. Note: “bin” is short for “binary” which is an executable file, or *.exe on Windows
4. Click **OK** twice to save.

On Mac:

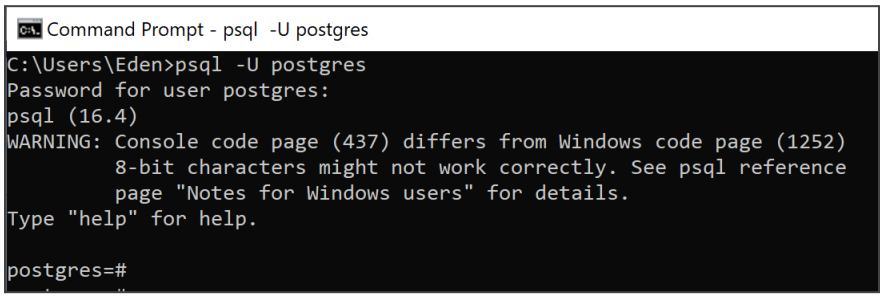
1. Run `echo 'export PATH=$PATH:/Library/PostgreSQL/17/bin' >> ~/.zshrc && source ~/.zshrc`
    a. The default installation location is /Library/PostgreSQL/17/bin
        i. If you choose another installation location, use that for this step.
    b. This command will not produce any output. To double check that it was added correctly you can run `cat ~/.zshrc`

Note: if you have an older mac you might need to replace .zshrc with .bash_profile in the above steps

Using PostgreSQL Locally

You may wish to orient yourself to the specifics of Postgres by reviewing the documentation, or one of the tutorials. Notice that the commands in Postgres begin with a `\`.

1. Launch your terminal. (Launch a new one to use your new path.)
2. Run `psql -U postgres`
3. For the **Password**, type the password you chose for the superuser.
4. At this point, your terminal should look similar to this:



   This is an interactive environment where you can create databases and connect to them.

5. To create a new database, run `CREATE DATABASE hw3db;`
6. To navigate to this database, run `\c hw3db;`
7. Create your tables by running your create table commands from HW2.
   a. You can do this by cutting and pasting your commands from your file
   b. Or by loading the file. You will need to include the path to the file. My command looks like this: `\i 'C:/Users/mh75/OneDrive - UW/Documents/DATA514/Homework/hw3sqls/create-tables.sql'`
   c. Notice that 'PRAGMA' is specific to sqlite, so you would remove that from your files.
8. Check that the tables have been created by running `\dt` You should see:

```
          List of relations
 Schema |   Name   | Type  |  Owner
--------+----------+-------+----------
 public | carriers | table | postgres
 public | flights  | table | postgres
 public | months   | table | postgres
 public | weekdays | table | postgres
(4 rows)
```

9. Run the following commands to import your flight data from HW2 (you'll need to adjust the paths). You may notice that it takes some time to load the data.

```
\copy WEEKDAYS FROM /path/to/weekdays.csv (FORMAT CSV)
\copy MONTHS FROM /path/to/months.csv (FORMAT CSV)
\copy CARRIERS FROM /path/to/carriers.csv (FORMAT CSV)
```

```
\copy FLIGHTS FROM /path/to/flights-small.csv (FORMAT CSV)
```

10. To verify that your imports were successful, run some SELECT COUNT(*) queries. As you know, this SELECT statement tells you how many rows are the tables. Postgres additionally tells you the number of rows produced by the query (1). If everything is set up correctly, you should have:
    a. 1594 for Carriers.
    b. 12 for Months.
    c. 8 for Weekdays.
    d. 1148675 for Flights.

11. Run the following commands to create indexes for your tables.

```
create index Flights_idx1 on
Flights(origin_city,dest_city,actual_time);

create index Flights_idx2 on Flights(actual_time);

create index Flights_idx3 on
Flights(dest_city,origin_city,actual_time);
```

12. Check that they were created by running `\di`. You should see:

```
hw3db=# \di
                List of relations
 Schema |     Name      | Type  |  Owner   |  Table
--------+---------------+-------+----------+----------
 public | carriers_pkey | index | postgres | carriers
 public | flights_idx1  | index | postgres | flights
 public | flights_idx2  | index | postgres | flights
 public | flights_idx3  | index | postgres | flights
 public | flights_pkey  | index | postgres | flights
 public | months_pkey   | index | postgres | months
 public | weekdays_pkey | index | postgres | weekdays
(7 rows)
```

13. Congratulations! You're ready to begin writing the queries for the homework.