PostgreSQL and PostGIS - Instalation and Configuration on Windows

Alexandre Neto

2022-01-04

On Windows*, the installation of PostgreSQL, PostGIS and PgAdmin4 is done through a convenient "one click installer" from enterpriseDB. [1]

PostgreSQL

1. Run the downloaded file with admin permissions.

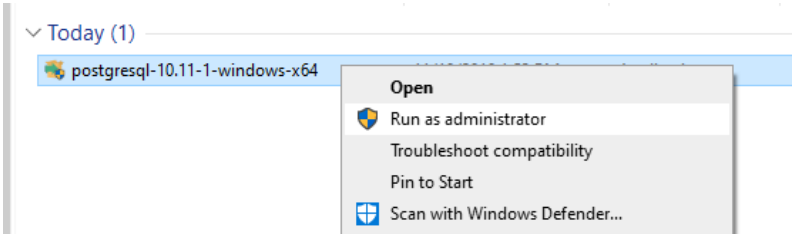

Figure 1: Run the installer as administrator

2. On the first page of setup click Next.

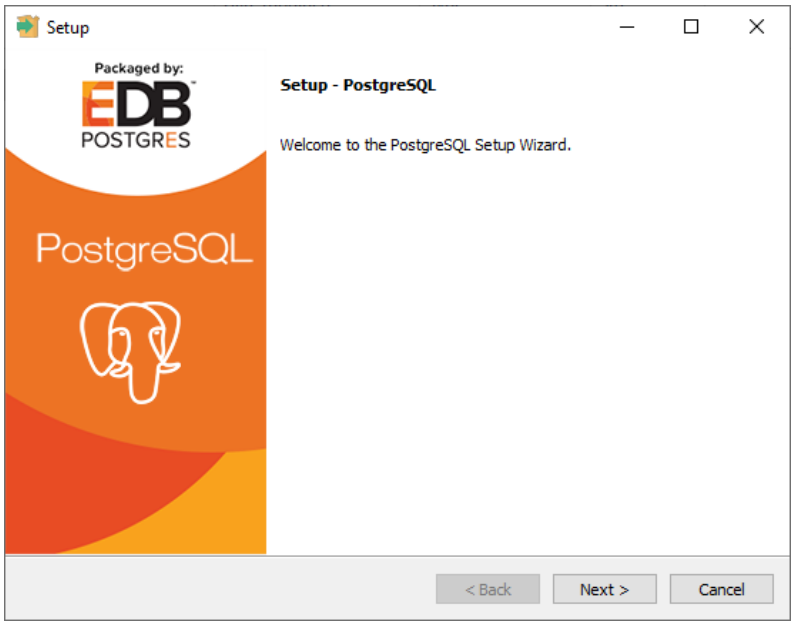

Figure 2: Setup - PostgreSQL

[1] It can be downloaded from the following page: http://www.enterprisedb.com/products-services-training/pgdownload#windows

3. If you want to install PostgreSQL in a directory other than the default one, enter the path here. Click Next.

   Choose installation folder

4. You can choose which components you want to install. Select all and click Next.

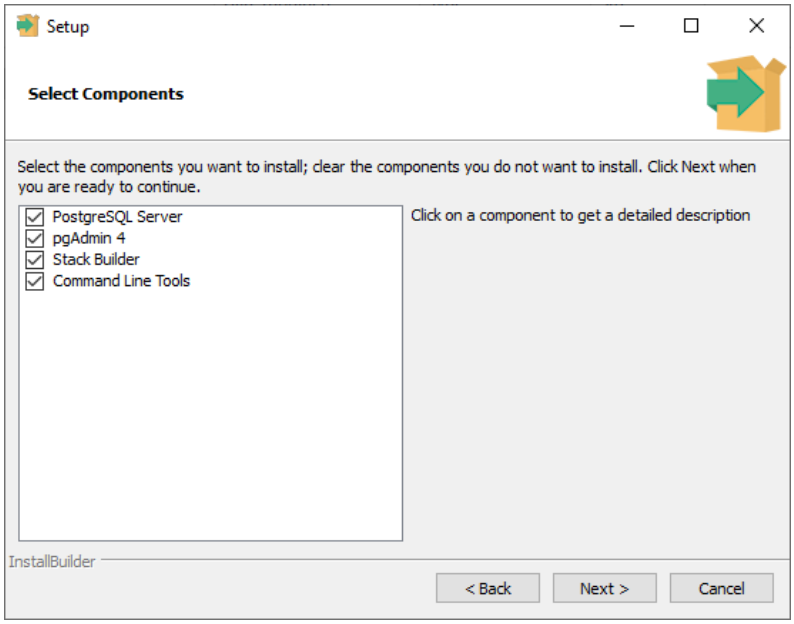

Figure 3: Component selection

5. The location of PostgreSQL data can also be changed to something other than the default one. Click Next.

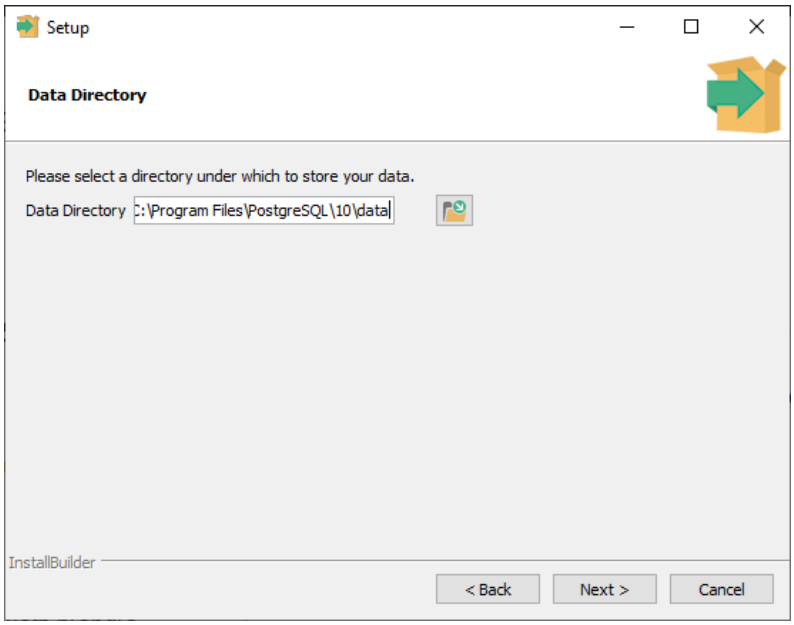

Figure 4: Data location

6. Enter the password for the new superuser account "postgres". [2]

7. Enter the number of the access port (default is 5432) and click Next. [3]

[2] User "postgres" has all permissions to the database. For that reason, in production environment, your password must be strong and only known by database administrators.

[3] In case of using a firewall, it may be necessary to create an exception for the chosen port

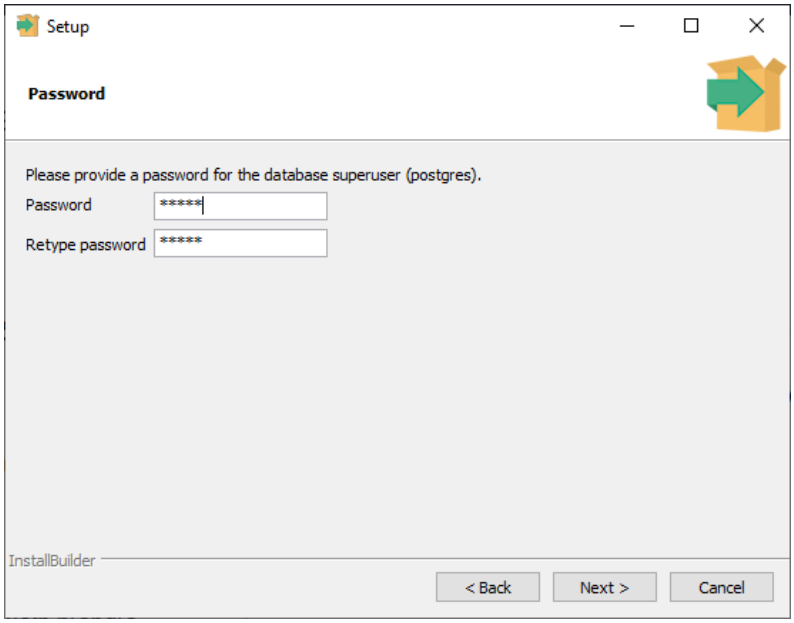

Figure 5: Choose the password of the user “postgres”

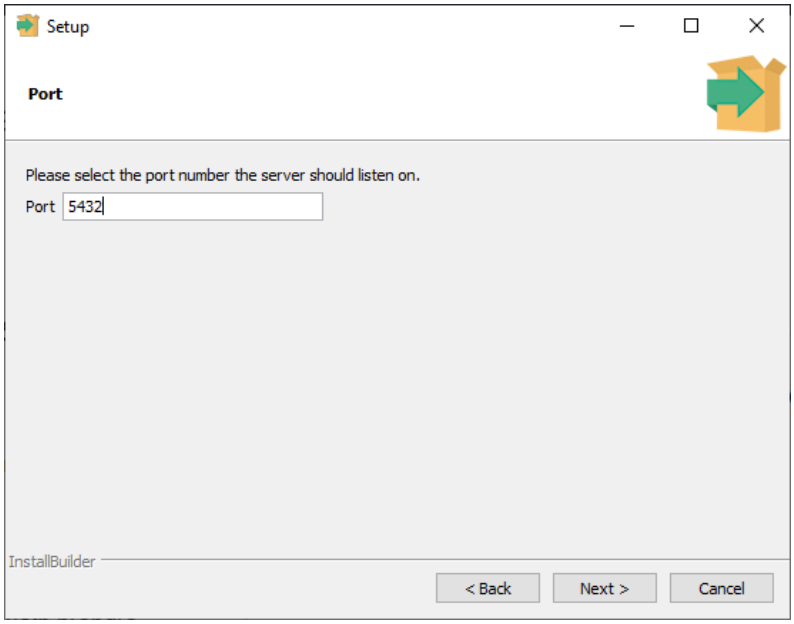

Figure 6: Choice of access port

8. Choose the “locale” used for the new database. The “Default Locale” option will reflect the location of the operating system where you are installing PostgreSQL.

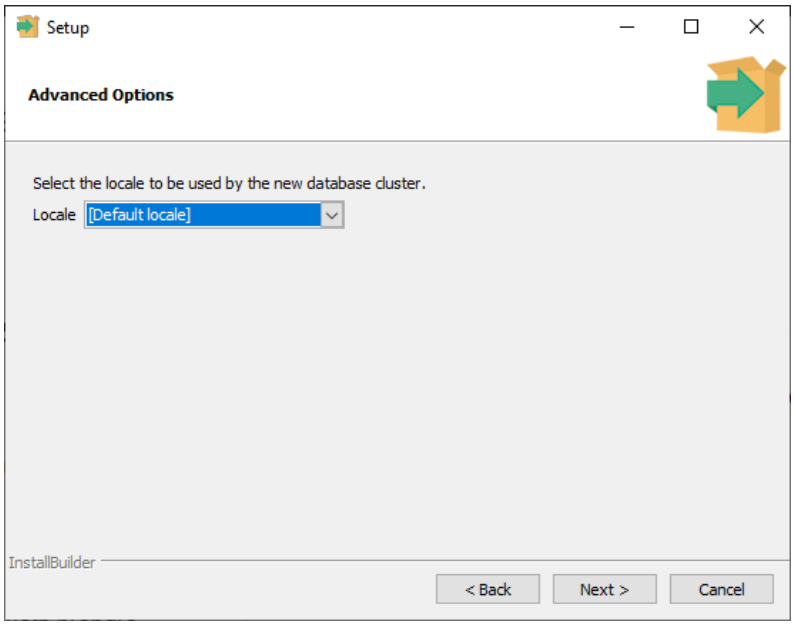

Figure 7: Locale selection

9. Confirm the installation parameters. Click Next.

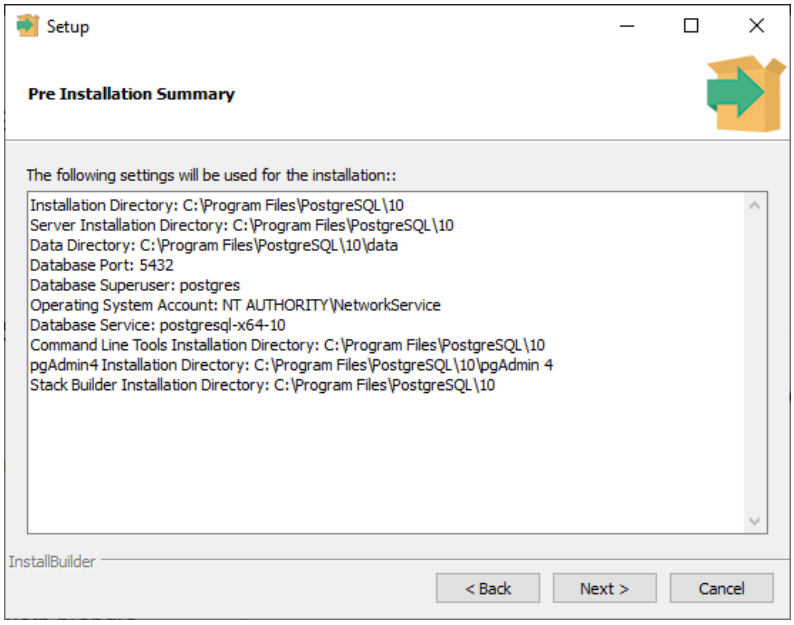

Figure 8: Installation Summary

10. Click Next.
11. Wait for the installation to finish.
12. Before clicking Finish select the option “Stack Builder may be used to download...”.

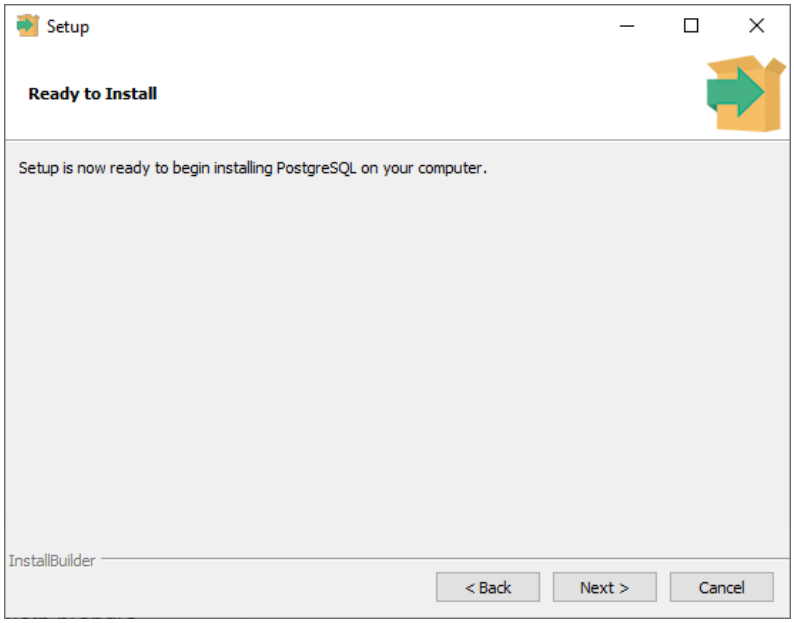

Figure 9: Ready to install

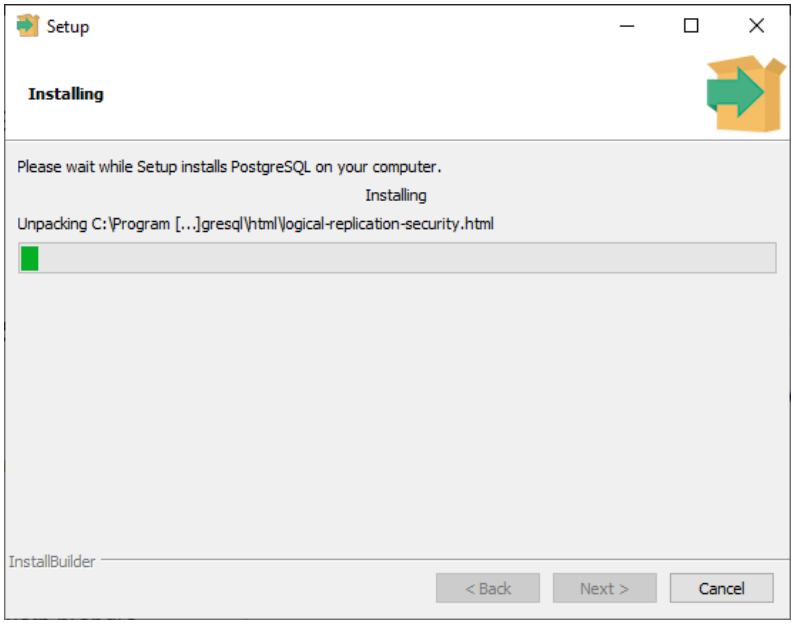

Figure 10: Installation in progress

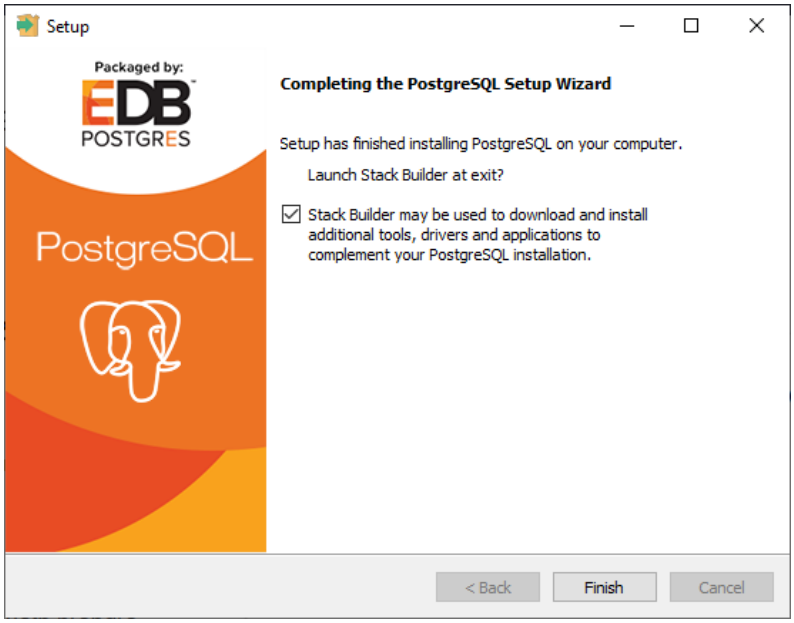

Figure 11: Finish the installation and open Stack Builder

PostGIS

1. The Stack Builder[4] you need to know which PostgreSQL installation to install the extensions on. Choose the location and click Next.

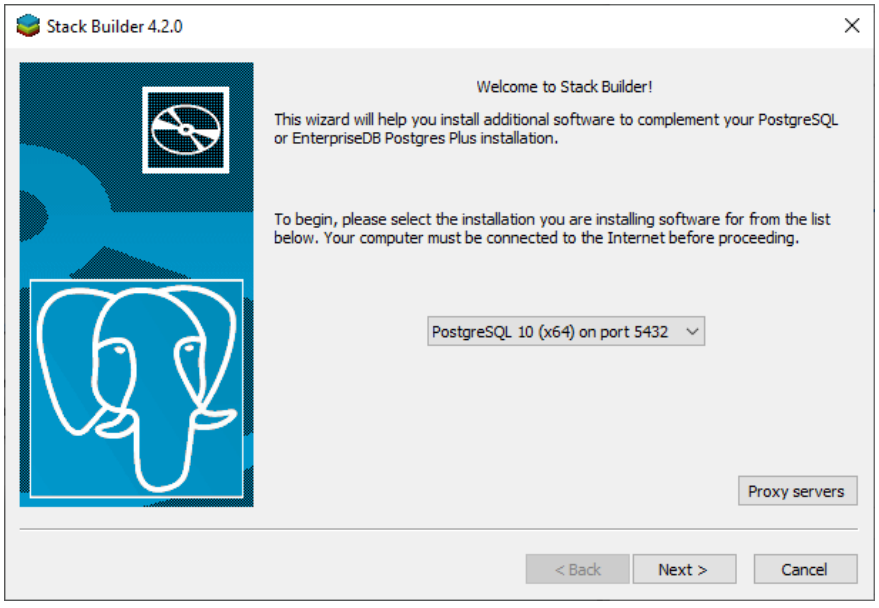

Figure 12: Select PostgreSqL Installation

2. There are a number of categories you can choose from. PostGIS is available in the "Spatial Extensions" group. Select the server and click Next.
3. The review page lists all packages selected for installation. In this case, you must have only the PostGIS selected. Click Next.

Confirm selection of packages to install

[4]Note: The Stack Builder application gives you a simple alternative for installing various useful add-on software for the PostgreSQL; PostGIS is one of the many applications.

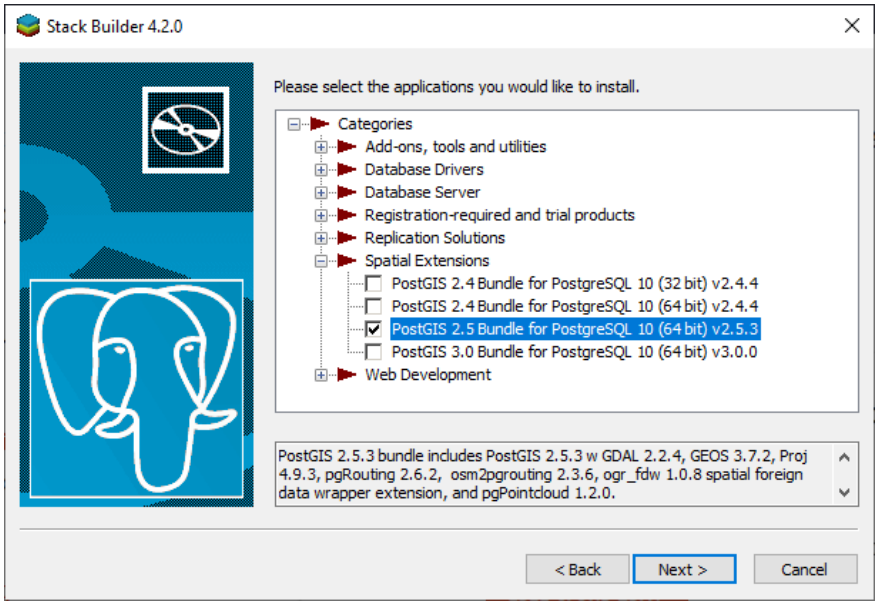

Figure 13: Select PostGIS extension

4. Once the download is complete, the following screen will appear. Click Next to continue.

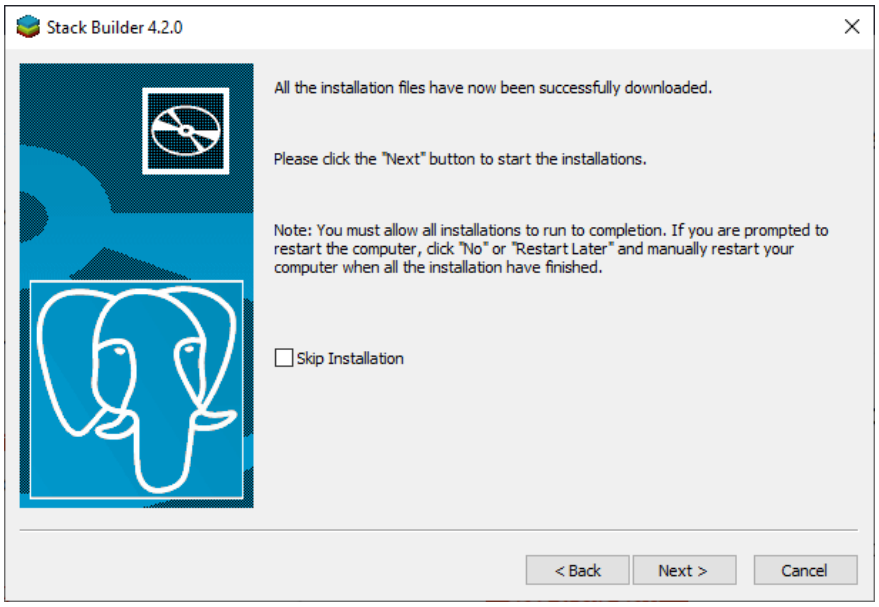

Figure 14: Start installation

5. PostGIS is licensed “GNU General Public License”, the description of which is shown. Click **I Agree**.
6. The PostGIS installer can automatically create a geographic database after installation. Do not select that option for now and click Next.
6. PostGIS needs to know the location of the target PostgreSQL installation. If PostgreSQL has been installed in the location default, the destination for installing PostGIS will be given by default. Click Next to continue.
7. The installer will offer to configure the GDAL_DATA system variable. Click Yes.
7. Click Yes when asked if you want to enable raster drivers.
8. Click Yes when asked if you want to use rasters outside the database. Click Yes.

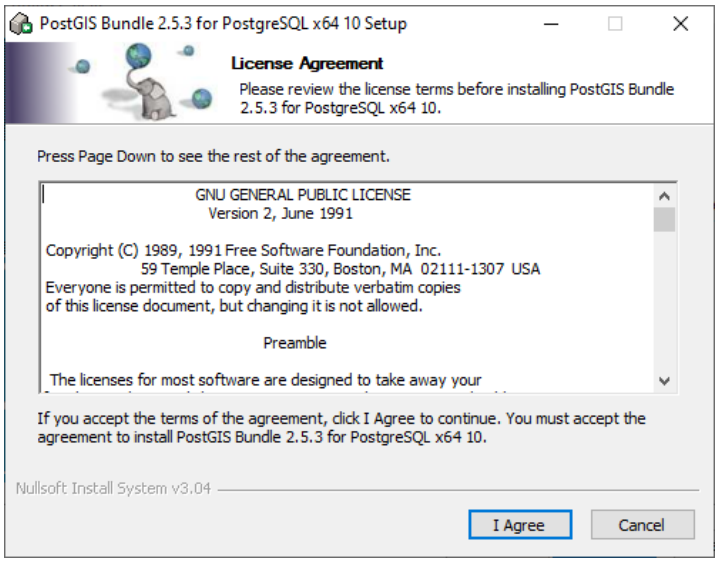

Figure 15: Accept license

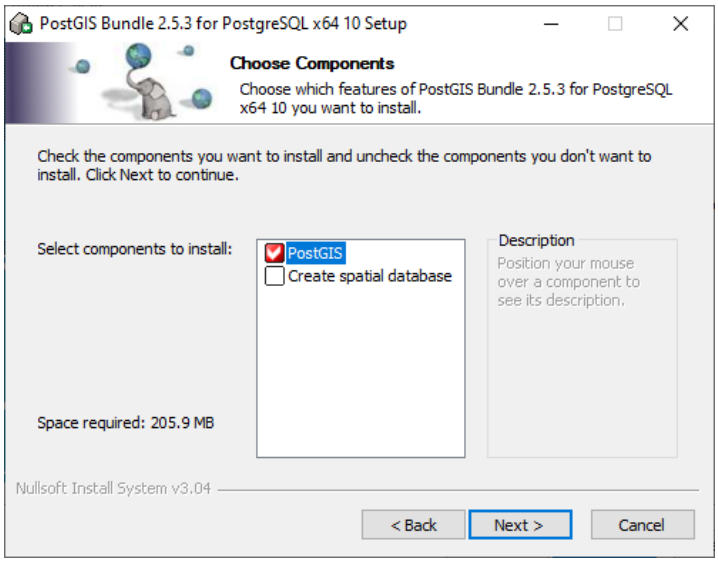

Figure 16: Choose Components

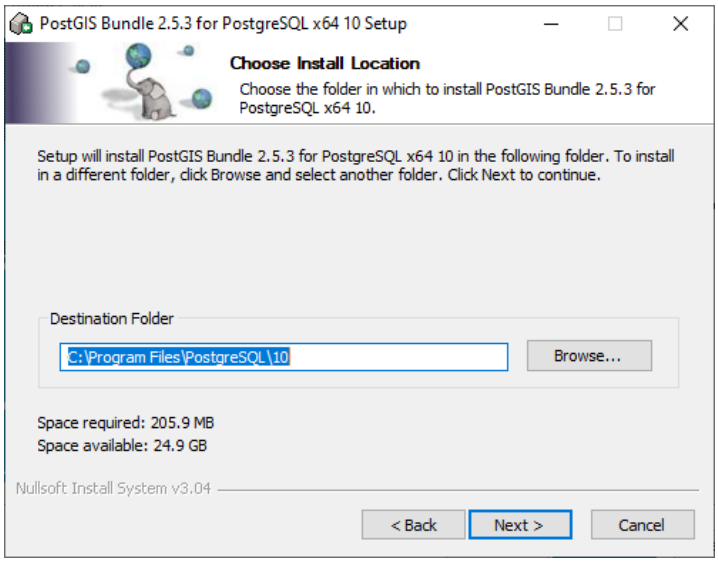

Figure 17: Select the installation folder

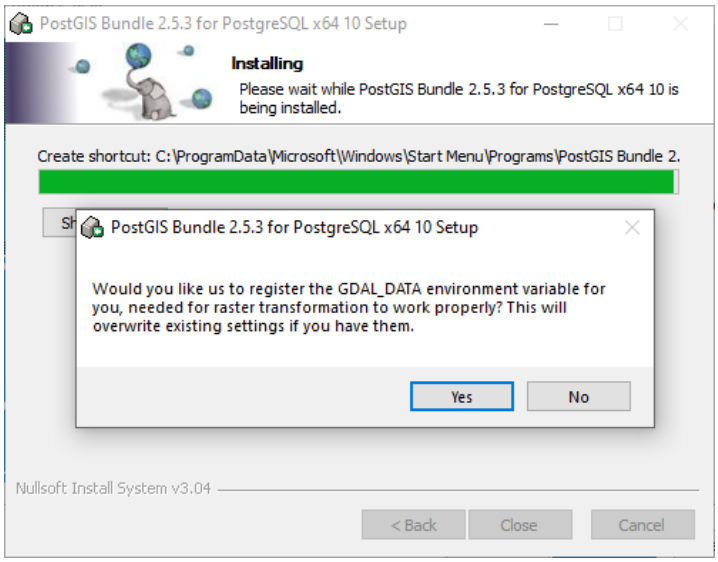

Figure 18: Accept to register the GDAL_DATA system variable

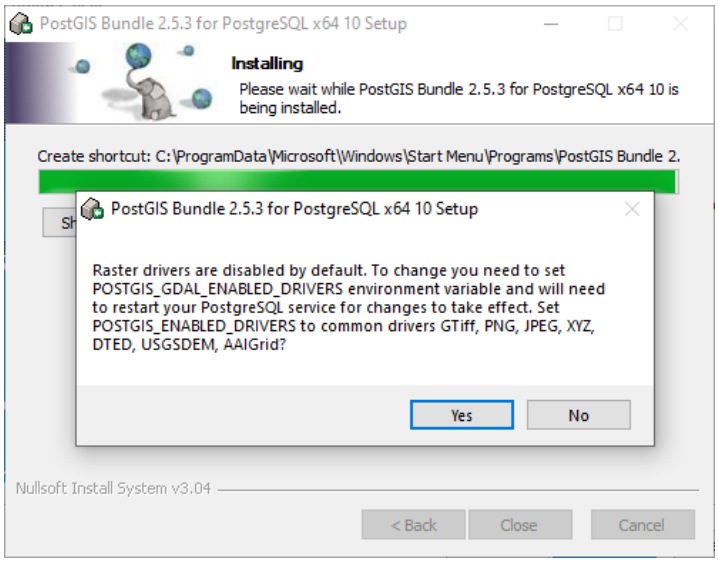

Figure 19: Enable raster drivers

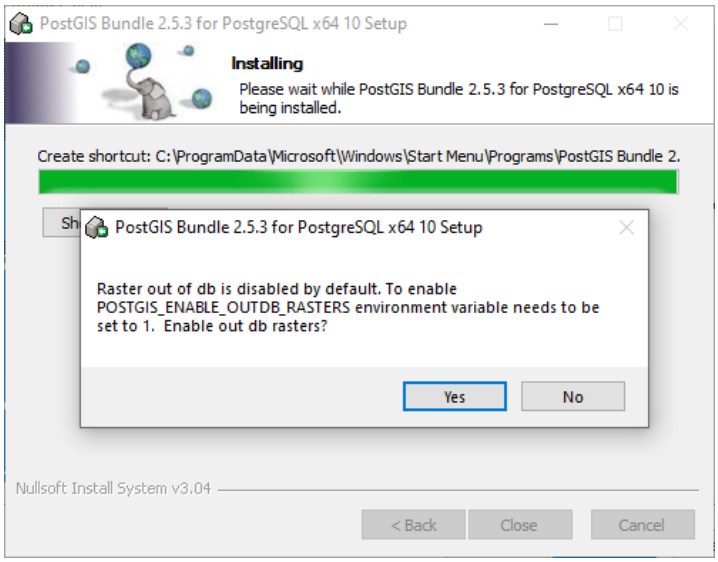

Figure 20: Enable raster out of database

8. After installation, click Finish.

Finish installation

Note that after installation, although several new shortcuts are added, there is no proper shortcut to start PostgreSQL. This is because PostgreSQL is constantly running as a system service, which responds whenever requested.

1. Open the system services[5] and confirm that PostgreSQL is active.

First configuration

After installation, and depending on the operating system, you must:

- configure the connections to access the database (hereafter referred to as "DB")
- configure firewall rule to allow external connections[6]
- create the necessary elements (BD and users) essential to start working.

PostgreSQL configuration files

On MS Windows systems, the configuration files are located in "C:\Program Files\PostgreSQL\10\data".

The **postgresql.conf** file, among other things, allows the activation of the TCP/IP connection. Example:

```
# - Connection Settings -
listen_addresses = '*' # what IP interface(s) to listen on;
# defaults to localhost, '*' = any
port = 5432
```

If not already, it is necessary to change the parameter `listen_addresses` to "*" (or to the IP address of the server) in order to allow the connection of others computers (connection via TCP/IP protocol).

The file **pg_hba.conf**[7] determines the host-based access conditions. In postgreSQL the default file is the following, and it only allows connections to `localhost (127.0.0.1)`:

```
# TYPE DATABASE USER ADDRESS METHOD

# IPv4 local connections:
host all all 127.0.0.1/32 md5
# IPv6 local connections:
host all all ::1/28 md5
# Allow replication connections from localhost, by a user with the
# replication privilege.
host replication all 127.0.0.1/32 md5
host replication all ::1/128 md5
```

In order for the service to accept connections from all other computers, add the following line:

```
# All other IPv4 connections
host all all 0.0.0.0/0 md5
```

Or a more secure alternative, just a certain range of IPs. For example:

```
# All other IPv4 connections
host all all 192.168.1.1/100 md5
```

Note: For changes to the configuration files to take effect the postgreSQL service must be restarted [8]

[5] You can open the services by looking for "services" in the windows start bar.

[6] The first two configurations are only necessary if external connections to the installation machine are foreseen. If only for local use, they can be ignored.

[7] https://www.postgresql.org/docs/10/auth-pg-hba-conf.html

[8] To restart the service, on the slash from tasks, run **PostgreSQL > Reload Configuration**.

Advanced settings (extra)

Depending on the type of use of the service, it is possible to optimize the service by changing the file **postgresql.conf**. It is, for example, possible to define the number of colors to use, RAM memory, etc... [9]

Not being the scope of this course, it is important to mention two tools that can help with this task:

- pg_tune[10]
- pgbench[11]

Generally speaking this is the optimization workflow: 1. Load database with data 2. Run pgbench to get a score 3. Run o pgtune 4. Apply changes to the configuration file[12]. 5. restart the service 6. Rerun pgbench to see if the system is improved.

Create users

Create administration superuser

For security reasons[13], the user "postgres" should just be used to create another superuser and then deleted. For that reason, the first step is to create a new superuser, with admin permissions. This will henceforth be the service administrator.

1. From the toolbar open **PgAdmin4**.[14]
2. Enter a master password for PgAdmin4 [15]
3. Create a new service (**Object > Create > Server**)
4. In the tab **General**:

- **Name**: `PostgreSQL 10 - local - superuser`

5. In thetab **Connection**

- **Host name/address**: `localhost`[16]
- **Port**: `5432`
- **Maintenance database**: `postgres`[17]
- **Username**: `postgres`
- **Password**: `admin`[18]
- Selectoption **Save password**

6. Click **Save**
7. The new connection (server) appears on the left side of pgadmin, double click to expand it.
8. Right-click on the link and choose **Create Login/Group Role**
9. In thetab **General**:

- **Name**: administrator

10. In thetab **Definition**:

- Choose a password for the user administrador‘^ `Suggested course` [admin1‘]

[9] This configuration is only necessary when the volume of data and users increases significantly.
[10] https://pgtune.leopard.in.ua/#/
[11] pgbench can be run via the command line and is found in `C:\\Program Files\\PostgreSQL\\10\\bin>`. For more information https://www.postgresql.org/docs/10/pgbench.html
[12] CAUTION: Before any changes to **postgresql.conf**, make a backup copy
[13] Using theuser `postgres` directly poses a risk as, since all postgreSQL installations come with that user, half of the credential is easily discovered
[14] PgAdmin is a GUI PostgreSQL service administration tool that comes installed by default with the Windows installation. There are other GUI tools for the same purpose. For example **DBeaver**
[15] This password has nothing to do with the password of the user "postgres", but allows you to protect the passwords stored in pgadmin4. For more information about the master password, see https://www.pgadmin.org/docs/pgadmin4/latest/master_password.html
[16] Or the remote server IP.
[17] The database named `postgres` is the service management database, it should never be used directly to store data.
[18] Or another password you chose during the postgreSQL installation process

11. Privileges ****:

- **Can login?** yes
- **Superuser?**: `yes`
- **Can initiate streaming replication and backups**: `yes`

12. Click **Save**

13. Repeat steps 3 to 7 to create a new link, now with the admin role.

14. After confirming that the new connection works correctly, delete the first one.

Create "normal" user and permission groups

Using the above instructions, create multiple users not superusers, but with different permissions. Create a `user1`, `user2` and a `user3`[19] with passwords `pass1`, `pass2` and `pass3`.

For everyone, we must give at least the following permissions:

- **Can login?** `yes`
- **inherit rights from parent roles?**: `yes`

For one of them, `user1` can be given permissions to create databases[20].

- **Create databases**: `yes`

Create a group

Let's now create a group. [21]. We can create a group of users who will be able to view and edit certain databases. Let's call it `editor`[22]. In this case we don't give you what kind of general administration permissions.

1. On thetab **General**:

- **Name**: editor

2. In **Privileges**:

    - **Can login?** `no`
    - **Superuser?**: `no`
    - **create roles?**: `no`
    - **inherit rights from parent roles?: `no`
    - **Can initiate streaming replication and backups**: `no`

3. Click **Save**

4. Repeat the steps to create another group, called `reader`, with the same type of permissions.

5. Now, go to the properties of each of the "normal" users, under **Membership**, add the groups `reader` and `editor` to all, with the exception of `user3`, which should only have permissions to `reader`. In this way users will inherit the permissions of the groups they have been assigned.

Create database

Create a database So

let's finally create a database.[23]

1. Right-click and choose **Create > Database**

[19] You can give peer names to associate with different roles within your organization

[20] This would typically be the data manager role, which although cannot create other users and manage the service, it can create and manage new databases

[21] Groups are created in the same way as a normal user, but without theoption **Can login??**

[22] Avoid using capitalized names, as they will only complicate life later on

[23] For now, the only database that PostgreSQL has installed is called `postgres` and it is the administration database of the system. It should not be used directly for saving data.

2. In thetab **General**, choose the name as `gisdb_curso`. For **Owner** of the database we will choose `user1`.[24]
3. In thetab **Security** we will assign different levels of access to the two groups we created.

- For the `editors` we will giveprivileges **CREATE** and **CONNECT**
- For `readers` we will only giveprivileges **CONNECT**

4. Click **Save**
5. Expand **Databases** and confirm that there is a new database called `gisdb`

Activate the PostGIS extension We will

talk more about Schemas later on, but for now open theschema `public` and look at **functions**. Note that there are no functions in the list.

1. Open the query tool (**Tools > Query Tool**).
2. At the console type:

```
CREATE EXTENTION postgis;
```

3. Click on execute

Refresh the database (or just theschema `public`) and notice that the list of functions now includes hundreds of functions.

At this moment the database is ready to receive spatial data.

PSQL command line

Note that all the tasks we have done so far by PgAdmin4 tool, can also be done through PSQL command line. PgAdmin4 itself contains several ways to query the necessary commands. For example, if you right click on thedatabase `gisdb` and choose **CREATE script** you can see the command needed to create the database:

```
-- Database: gisdb

-- DROP DATABASE gisdb;

CREATE DATABASE gisdb
   WITH
   OWNER = user1
   ENCODING = 'UTF8'
   LC_COLLATE = 'English_United States.1252'
   LC_CTYPE = 'English_United States.1252'
   TABLESPACE = pg_default
   CONNECTION LIMIT = -1;
```

[24] This ensures that `user1` has automatically been able to manage this database.