



Build a New WordPress Site



FTP Accounts



MySQL® Databases

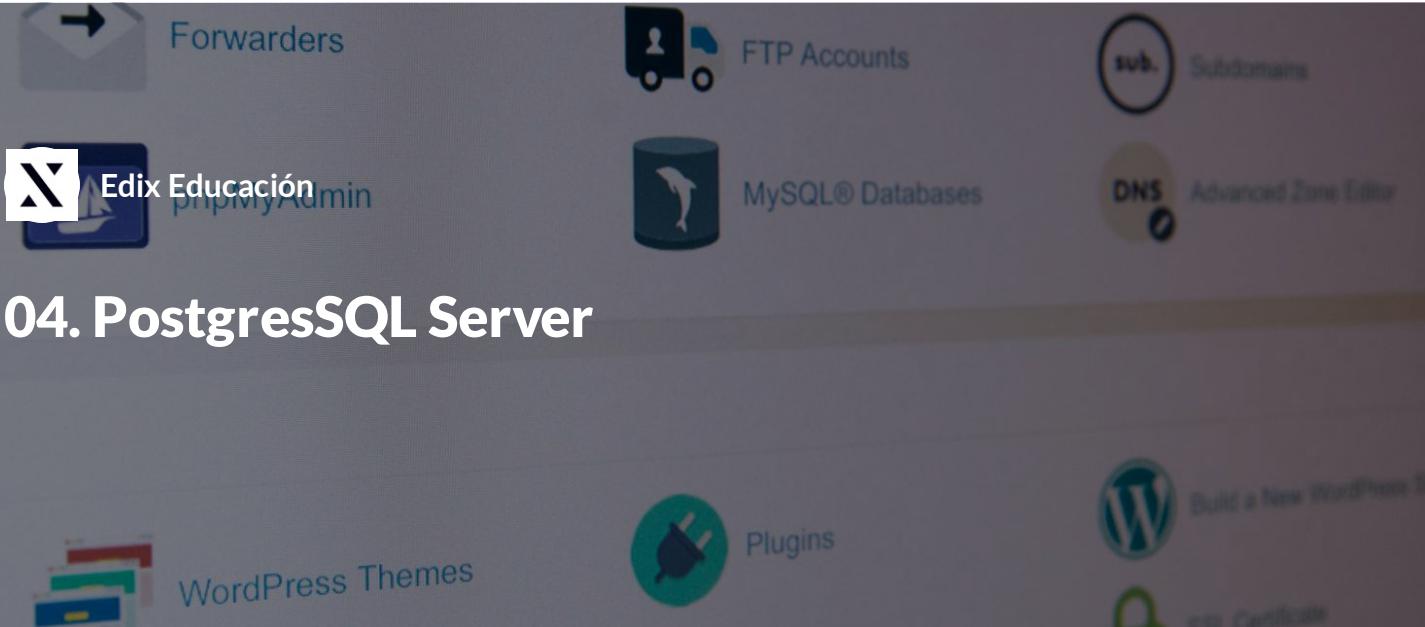


Plugins

# Fastbook 04

## Tratamiento de Datos (Excel y SQL)

PostgresSQL Server



## 04. PostgreSQL Server

PostgreSQL es un SGBD de tipo **relacional y orientado a objetos**. Su licencia y desarrollo es de código abierto, siendo mantenida por una comunidad de desarrolladores, colaboradores y organizaciones de forma libre y desinteresada.

En este fastbook, analizaremos sus principales características y las razones por la que se ha hecho con un nombre importante dentro de la industria. Posteriormente, explicaremos paso a paso cómo instalar el software necesario.

Haremos un breve recorrido por PgAdmin viendo sus principales funcionalidades y, por último, conoceremos algunas utilidades que vienen incluidas en la instalación que hemos realizado.

*Autor: Carlos Manchón y Breogán Cid*

[Introducción a PostgreSQL](#)

[Instalación del entorno de trabajo](#)

[Primer paseo por PgAdmin](#)

[Ejecución por consola](#)

[Últimas configuraciones](#)

[Conclusiones](#)

# Introducción a PostgreSQL

 Edix Educación

---

PostgreSQL nació en la década de los 80, a partir de Ingres, un proyecto de base de datos creado años atrás. Michael Stonebraker, que dirigía el proyecto, decidió llamarlo Post Ingres, acortando su nombre para que finalmente quedara como **Postgres**. Más tarde, en el apogeo de las bases de datos de tipo SQL (relacionales), se decidió volver a cambiar su nombre por **PostgreSQL**.

Entre sus **principales características**, destacamos las siguientes.

1

Código abierto

Como ya hemos visto, PostgreSQL es un **sistema de código abierto**, razón por la cual se ha vuelto tan popular. Debido a la gran comunidad que respalda el proyecto, ha logrado transformarse en uno de los mejores gestores de bases de datos a nivel mundial.

2

Gratis

Se trata de un **software gratuito**, simplemente tenemos que descargarlo de la web oficial y lanzar el programa de instalación.

3

### Multiplataforma

Otro de los factores que ha hecho tan famoso a PostgreSQL es que se trata de un **software multiplataforma**, ejecutándose en distintos entornos y sistemas operativos.

4

### Sencillez

Como veremos, a través del PgAdmin podremos **administrar de una manera muy sencilla la base de datos**. También permite realizar operaciones bastante más complejas, por lo que el programa es válido tanto para usuarios expertos como para usuarios novatos.

5

### Volumen de datos

Otra característica en la que destaca PostgreSQL es en su **capacidad para trabajar con grandes volúmenes de datos**. Otros SGBD competencia de PostgreSQL (como, por ejemplo, MySQL) no destacan en esta faceta, por lo que, si tenemos que trabajar con bases de datos grandes, nuestra elección debe ser PostgreSQL.

6

### Estándar SQL

Si bien hemos hablado de que las **bases de datos relacionales usan SQL** como lenguaje de comunicación, y este a su vez está regido por un estándar, llevarlo a la práctica no es tarea sencilla. No todas las bases de datos cumplen la totalidad del estándar, si bien PostgreSQL es de las **bases de datos más estrictas** en el cumplimiento de este. Además de los tipos de datos estándar, permite soporte nativo o como extensión para muchos tipos de datos (por ejemplo, datos geográficos, geométricos, orientados a las redes – MAC, IP- y un largo etcétera).

Como ya vimos, una de las principales misiones de un SGBD es **controlar el acceso a la base de datos y permitir conexiones concurrentes a esta**. En esta faceta, PostgreSQL destaca porque permite un número elevado de conexiones a la vez.

Además, incluye un **sistema denominado MVCC** que posibilita a los usuarios el acceso a una tabla, mientras que otros realizan escrituras sobre la misma. En definitiva, en todo momento, PostgreSQL ofrece a los distintos usuarios una versión consistente de los datos.

- Existen varios programas (denominados clientes de bases de datos) que nos permiten conectarnos al SGBD de una manera visual (gráfica) y gestionar nuestra base de datos.
- PostgreSQL posee varias opciones, entre ellas PgAdmin, phpPgAdmin, DBeaver, Navicat, etc.

---

**En nuestro caso, trabajaremos con PgAdmin, posiblemente el más conocido y utilizado.**

# Instalación del entorno de trabajo

X Edix Educación

---

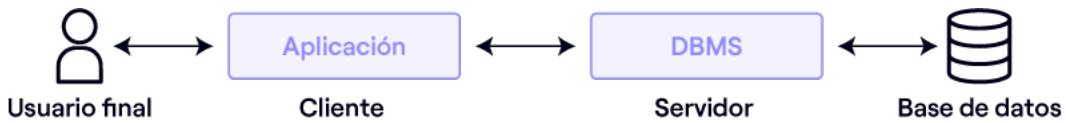
Normalmente, cuando nos conectamos a una base de datos y esta no se encuentra en el mismo ordenador en el que estamos nosotros, decimos que realizamos un **acceso remoto**. En el caso de que sean la misma máquina, decimos que se trata de un **acceso local**.

---

En cuanto al acceso remoto, tan solo necesitamos el software que nos permite conectarnos a la base de datos, comúnmente llamado cliente de base de datos.

---

Dicho nombre se debe a que el software realiza la **función de cliente**, llevando a cabo peticiones, y la base de datos hace la función de servidor respondiendo a las solicitudes.



---

En nuestro caso, al no disponer de un servidor de bases de datos remoto al que conectarnos, tendremos también que montar dicho servidor y alojar internamente la base de datos.

Ahora, realizaremos los **pasos necesarios para dejar todo el software instalado en nuestro ordenador**.

Es importante destacar que esta guía ha sido desarrollada para ordenadores que posean el sistema operativo Windows.

1

### Acceso

Accedemos a través de [este enlace](#) y nos descargaremos la última versión disponible de la plataforma **Windows x86 - 64**.

2

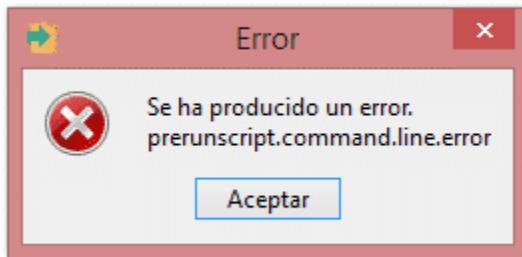
### Instalar

Una vez descargado, hacemos **doble clic sobre el instalador** para iniciar la instalación.

3

### Ventana de error

Es posible que antes de iniciarse el instalador, nos salte **una ventana de error** tal que así:

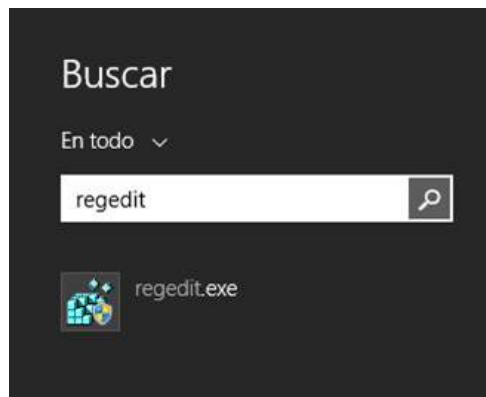


Seguimos este punto solo si nos ha aparecido el error en el punto 3, en caso contrario, saltamos hasta el punto 9. Necesitamos ir al registro de Windows para activar una característica.

4

## Registro

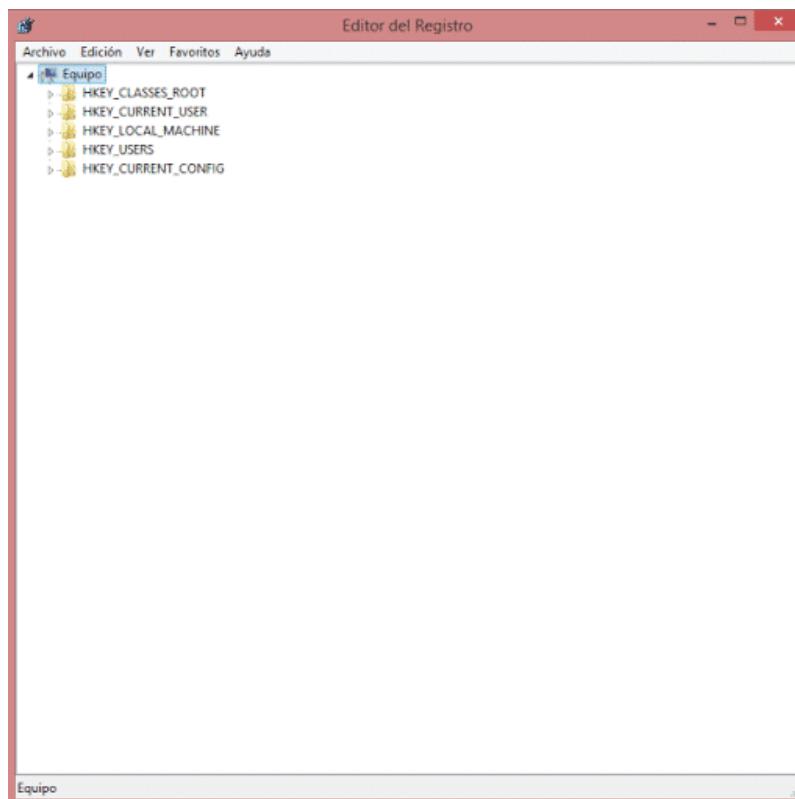
Accedemos al registro, buscando la aplicación Regedit.



5

## Aplicación

Si pulsamos sobre la aplicación, nos aparecerá una pantalla tal que así:

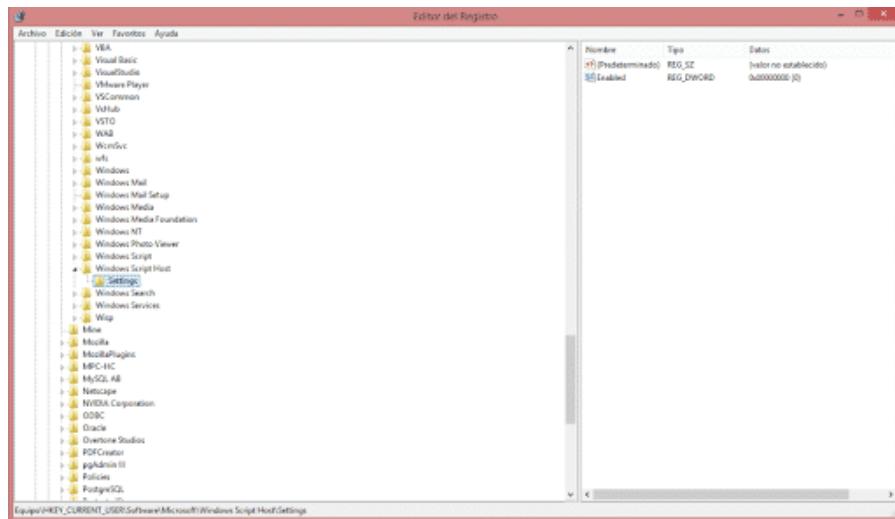


6

## Estructura jerárquica

Nos desplazamos por la estructura jerárquica de carpetas a la dirección siguiente:

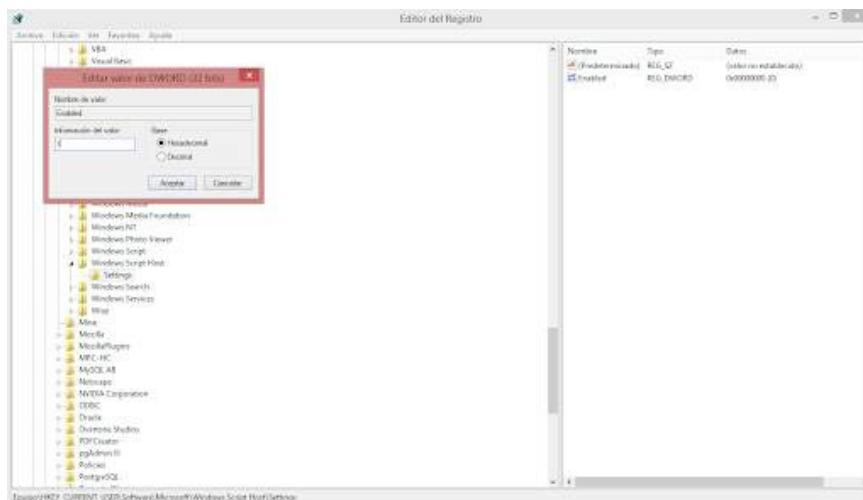
Equipo\HKEY\_CURRENT\_USER\Software\Microsoft\Windows Script Host\Settings



7

## Propiedades

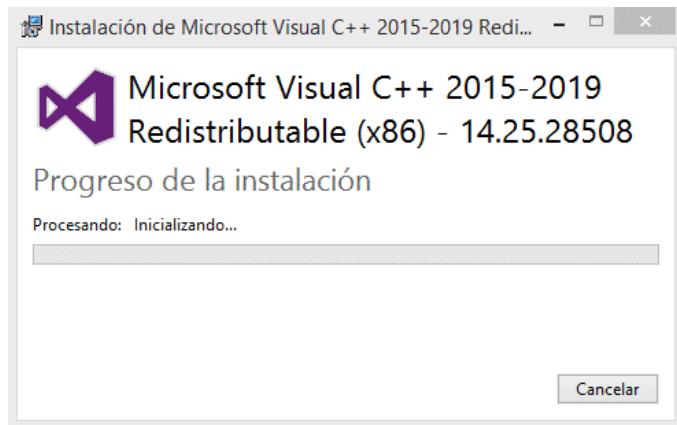
Como observaremos, al pulsar sobre la carpeta 'Settings', la parte derecha de la pantalla mostrará **dos propiedades**. Nos fijaremos en la que se llama 'Enabled'. Hacemos doble clic sobre ella y, en la ventana que aparece, en el campo 'Información del valor', sustituimos el valor 0 por un 1.



8

## Instalar el software

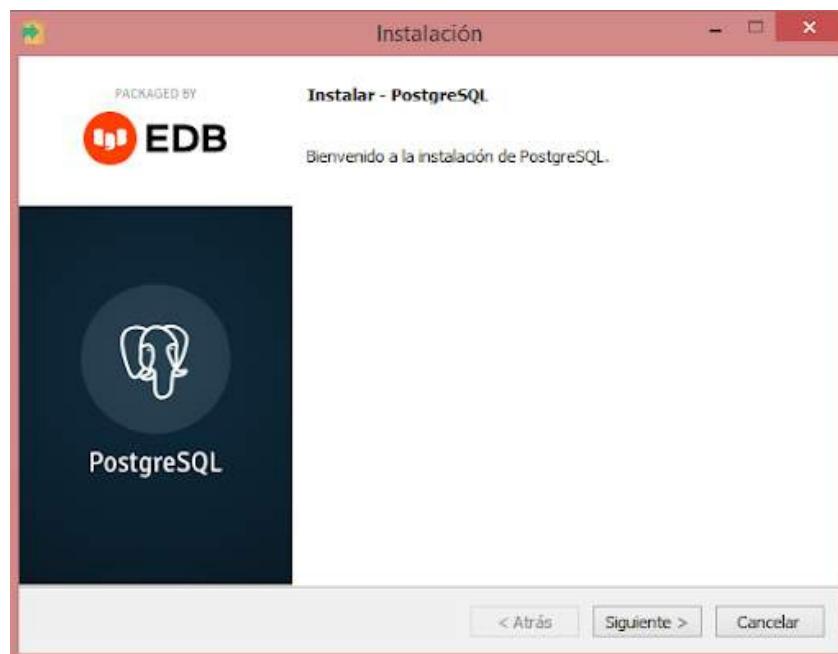
Realizado este cambio, **volvemos a nuestro instalador y le damos doble clic**. El proceso debería ya iniciarse sin problemas, instalando un software necesario previamente (Microsoft Visual C++ 2015–2019).



9

## Instalar PostgreSQL

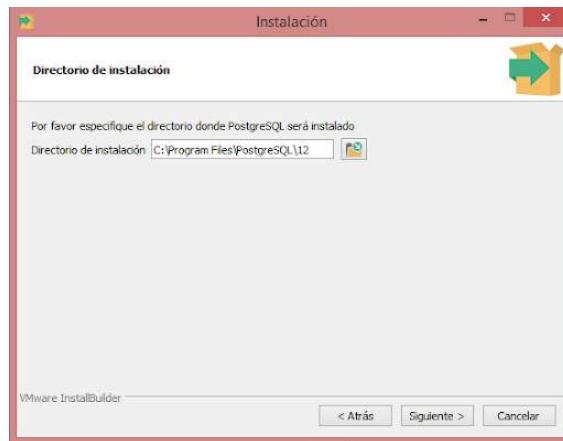
Una vez instalado este software, la instalación ya sí de PostgreSQL se iniciará. Pulsamos sobre 'Siguiente'.



10

## Dirección de instalación

Elegimos una **dirección de instalación**. Por defecto, el instalador nos propondrá una. Volvemos a pulsar sobre el botón ‘Siguiente’.

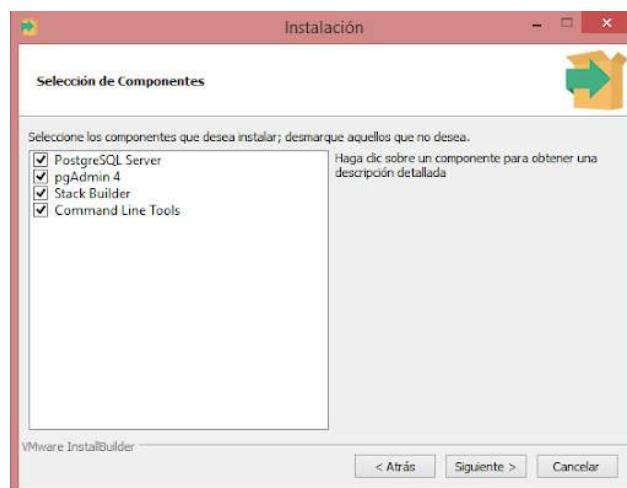


11

## Seleccionar componentes

En la siguiente pantalla, tendremos que **seleccionar los componentes** que deseamos instalar. Dejaremos todos los que están marcados:

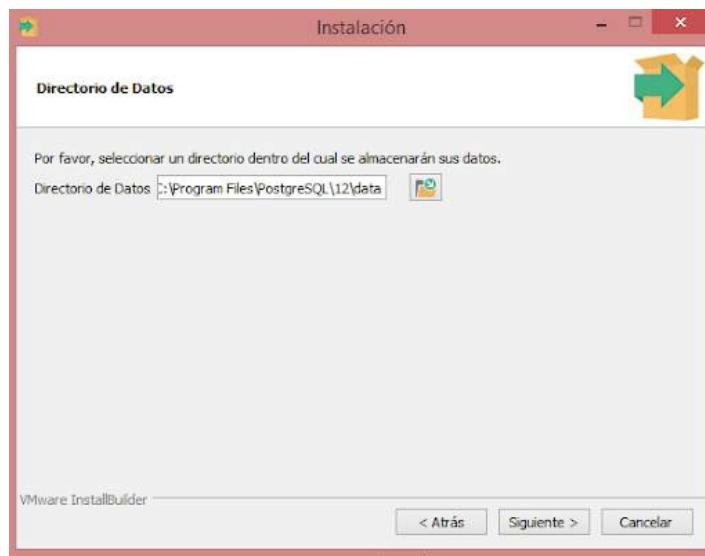
- El servidor de base de datos (PostgreSQL Server).
- El cliente (pgAdmin 4).
- Stack Builder.
- La utilidad de línea de comandos (psql).



12

## Almacenar información

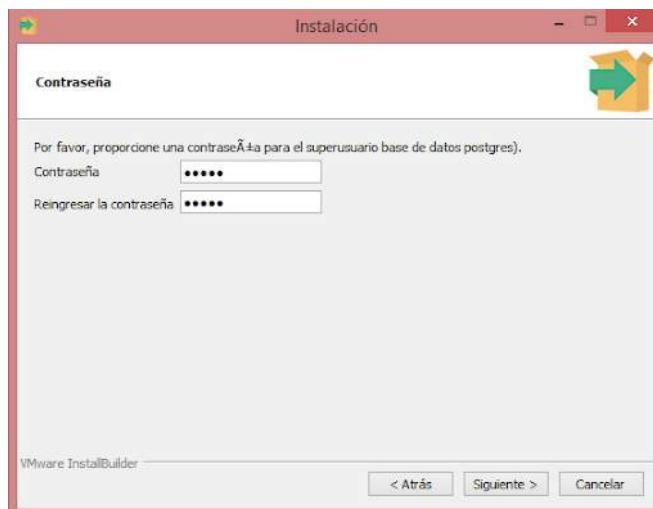
En la pantalla que mostramos a continuación, **seleccionaremos la ruta** donde físicamente se va a almacenar la información de la base de datos. Dejamos como ruta la que se nos propone por defecto y pulsamos sobre el botón ‘Siguiente’.



13

## Introducir la clave

La instalación está casi terminada, ahora nos solicitan que **introduzcamos una clave para el superusuario Postgres** (es un usuario especial con todos los permisos para administrar nuestra base de datos). Es importante guardar a buen recaudo esta información. Después, pulsamos sobre el botón ‘Siguiente’.

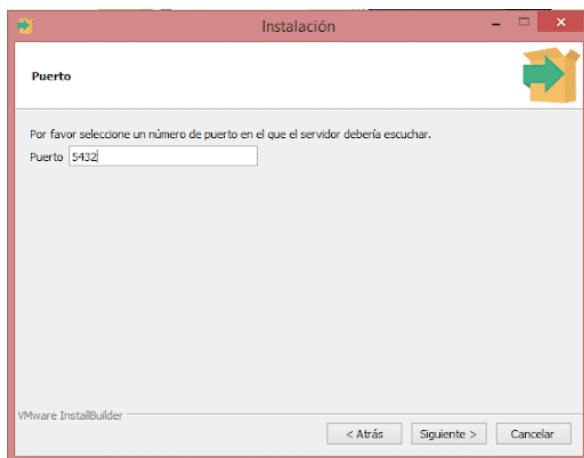


14

## Puerto

Un parámetro más por configurar: el puerto. Debes saber que para establecer una conexión a una base de datos utilizamos un puerto para conectarnos a esta. Piensa que un ordenador dispone de un número determinado de puertos para establecer comunicaciones con el exterior, cada puerto identifica a un servicio.

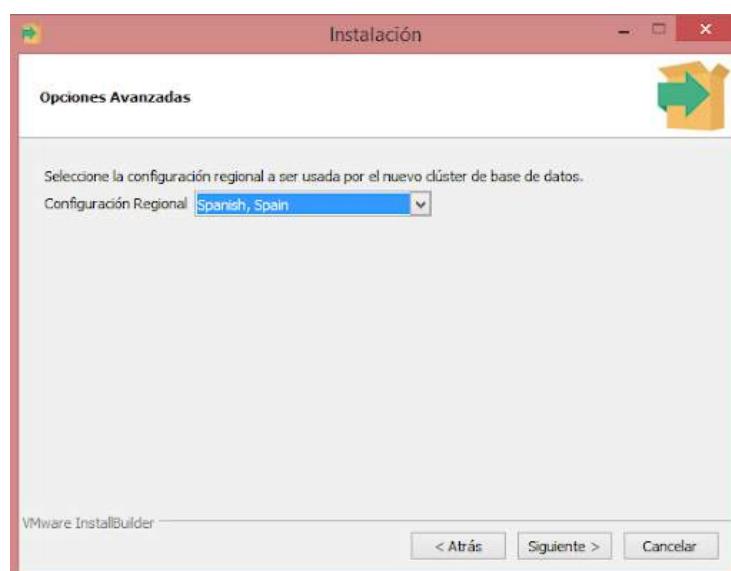
En nuestro caso es el servicio de bases de datos PostgreSQL. Dejamos el valor de puerto por defecto, es decir, 5432, y pulsamos sobre el botón ‘Siguiente’.



15

## Configuración regional

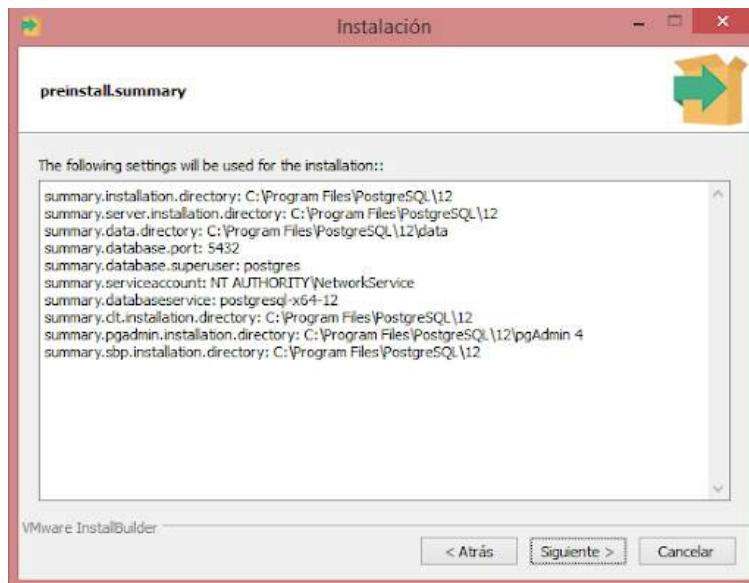
Para la configuración regional, seleccionamos el **español, España**.



16

## Configuración

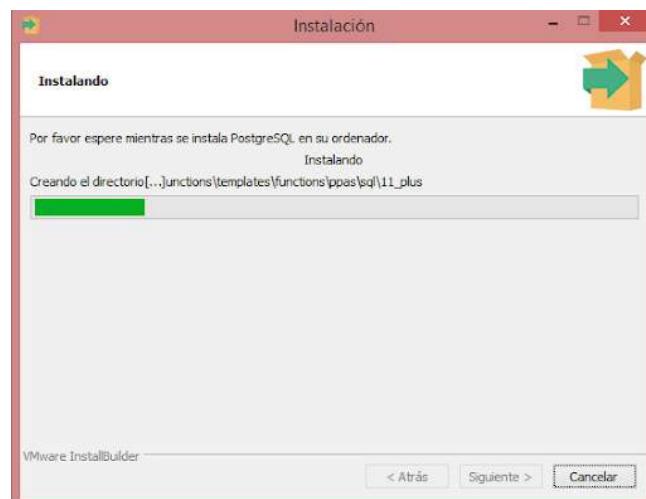
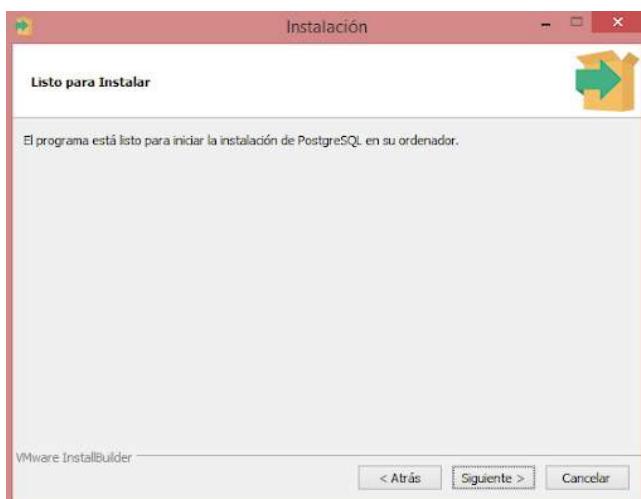
Se nos muestra una pantalla con el resumen de la configuración y...



17

## Instalar todo

¡Ya hemos terminado la configuración! El sistema se dispone a instalar todo lo indicado y tal y como lo hemos indicado.



18

## Descargar herramientas

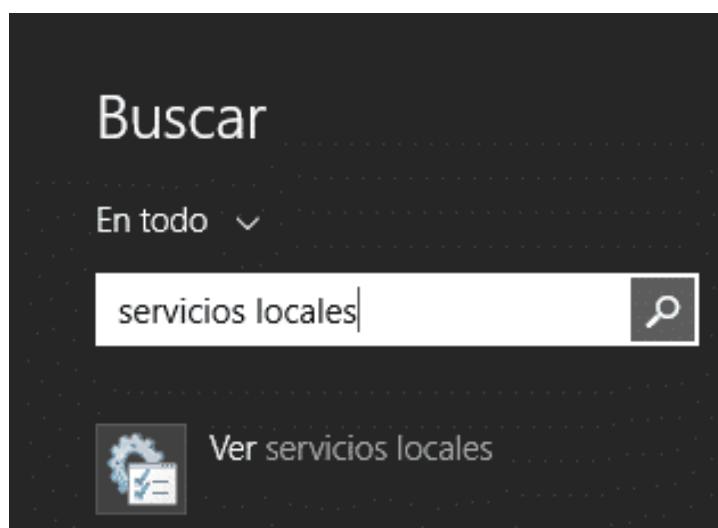
Ahora bien, el proceso ha terminado y tenemos ya disponible el **software indicado**.

Desmarcamos la opción para que StackBuilder nos solicite descargar herramientas adicionales, controladores y otras aplicaciones complementarias.

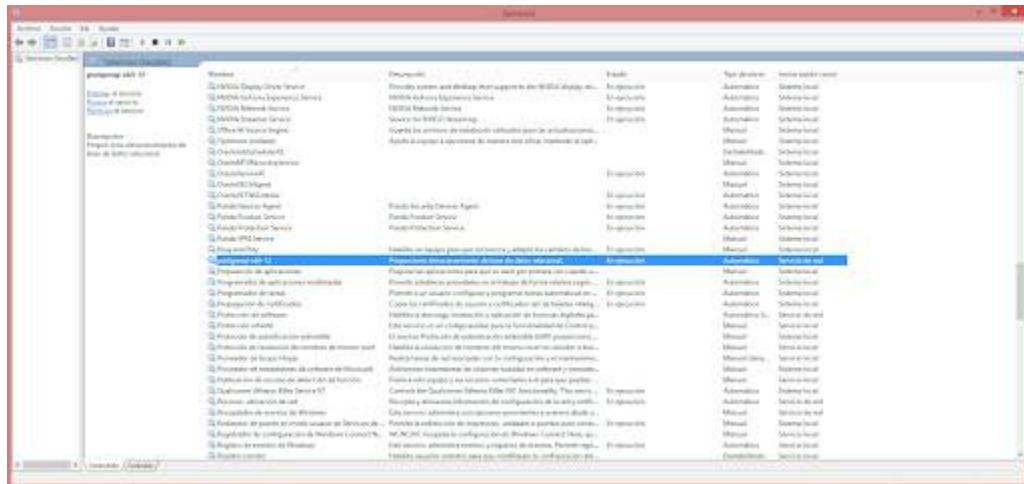


19

Comprobamos que el servidor de base de datos está instalado como servicio e iniciado. Para ello, accedemos al listado de servicios locales como indica la imagen.



Comprobamos que efectivamente el servicio está en el listado en ejecución.



Para otros **sistemas operativos**, se pueden seguir las siguientes guías:

- [Linux \(Ubuntu\)](#).
- [Mac](#).

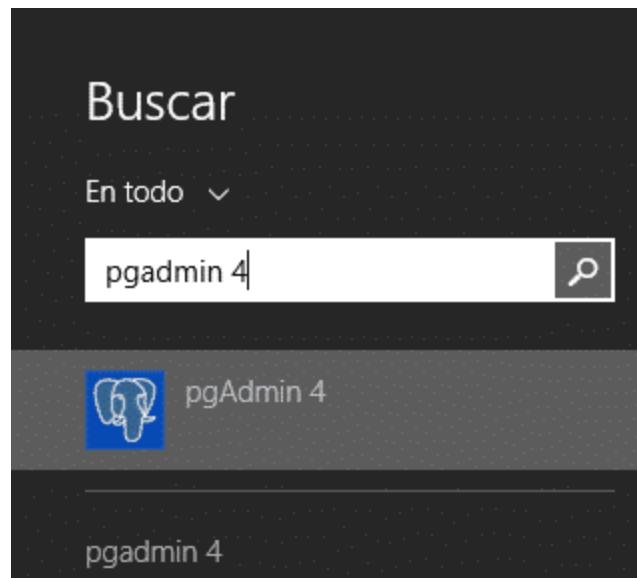
# Primer paseo por PgAdmin

X Edix Educación

---

Ahora que ya disponemos de nuestro entorno montado, daremos un pequeño paseo por la aplicación cliente para ver las posibilidades que nos ofrece. Es importante que tengamos en cuenta que durante el recorrido **ejecutaremos algunas consultas** y no nos pararemos a explicarlas en detalle, ya que lo haremos más adelante.

El primer paso, será entrar en la aplicación. Para ello, comprobaremos que **se ha instalado el cliente PgAdmin 4** y pulsaremos sobre él:



Las primeras versiones de PgAdmin se ejecutaban como aplicación de escritorio, desde la versión 4, se ejecuta como aplicación, por lo que tu navegador predeterminado se iniciará desplegando en una pestaña el software.

La primera vez que accedemos, nos pedirá que **introduzcamos una contraseña máster** para PgAdmin, es muy importante, ya que la necesitaremos para acceder a la herramienta y no la podremos cambiar ni recuperar.

En este punto ya deberemos de tener controladas **dos contraseñas**:

- La creada en el **proceso de instalación**, que es la de nuestro servidor local.
- La creada la primera vez que entramos en PgAdmin, que es la de la **aplicación cliente**.

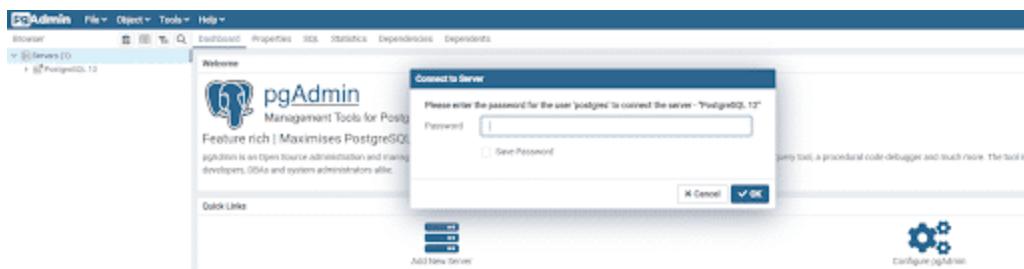


La aplicación muestra todas las conexiones a servidores de bases de datos que tengamos, pero, en nuestro caso, solo tendremos el **acceso al servidor local** que acabamos de configurar e instalar.

1

## Contraseña de acceso

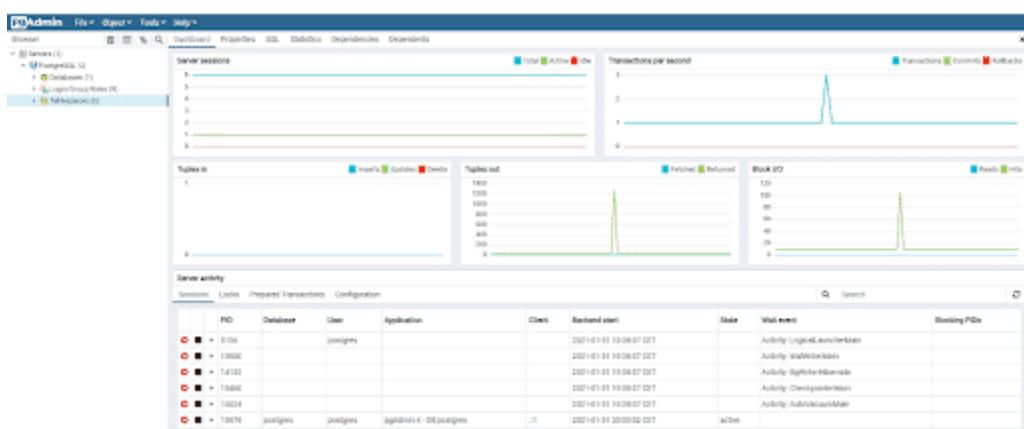
Pulsamos sobre nuestro servidor local y nos aparecerá una ventana solicitándonos la contraseña de acceso.



2

## Conectarse

Una vez hayamos conectado con nuestro SGBD, la aplicación lucirá tal que así:

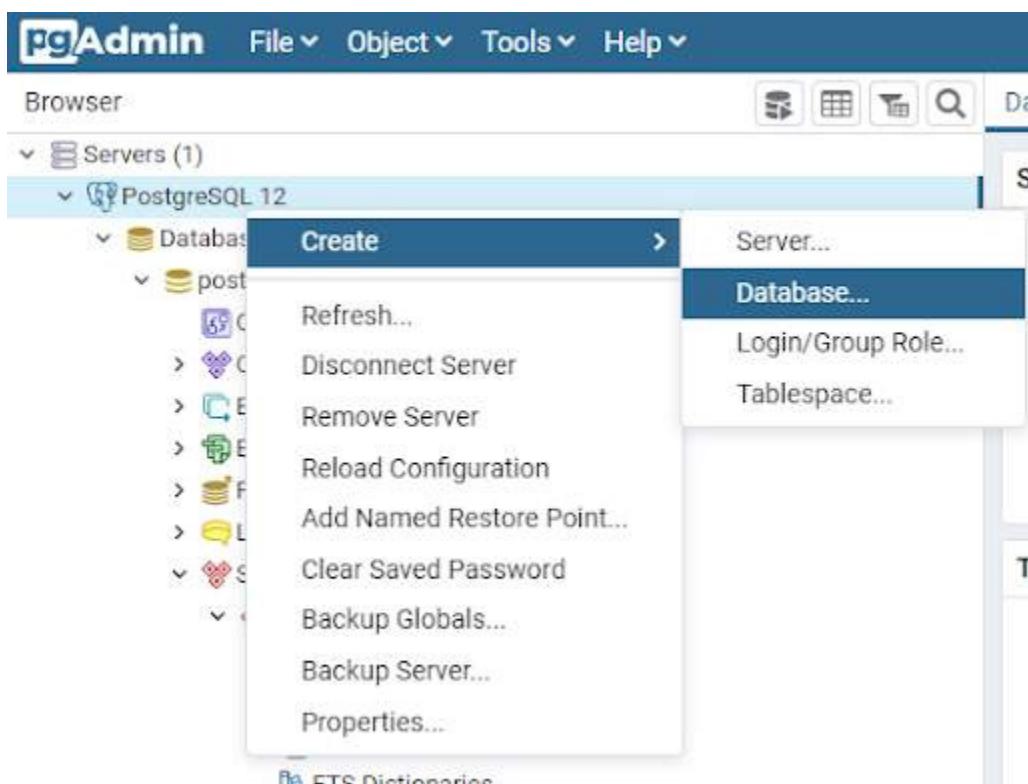


3

### Servidor de base de datos

Comencemos entonces a **realizar operaciones sobre nuestro servidor de base de datos**, por ejemplo, ¡creemos una nueva base de datos!

Nos colocamos encima de nuestro servidor y pulsamos el botón derecho de nuestro ratón. Aparecerá un menú contextual con el siguiente contenido:



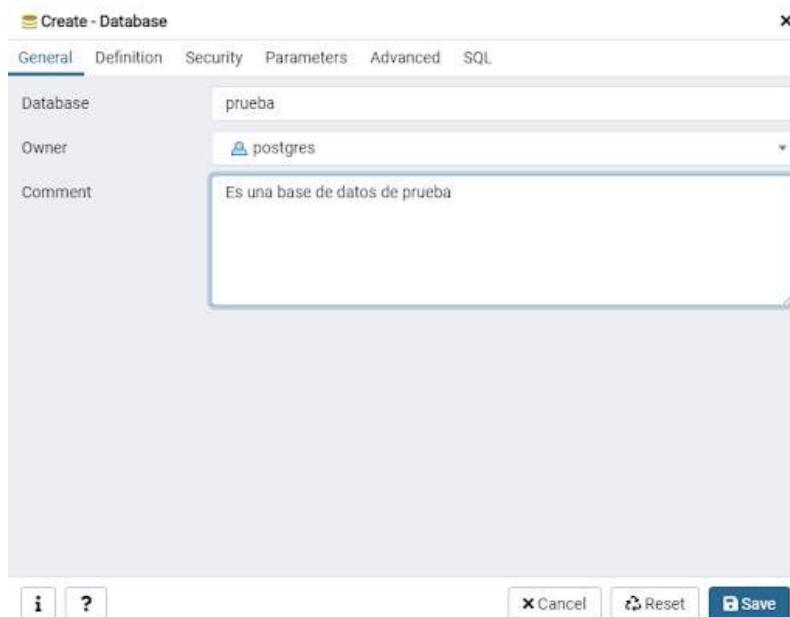
Nos colocamos sobre la **opción ‘Create’** y en el desplegable seleccionamos ‘Database’ para crear una nueva base de datos.

Al pulsar, aparecerá un menú para **configurar nuestra nueva base de datos**. Nos centraremos en la pestaña general para llenar el nombre, por ejemplo: prueba.

4

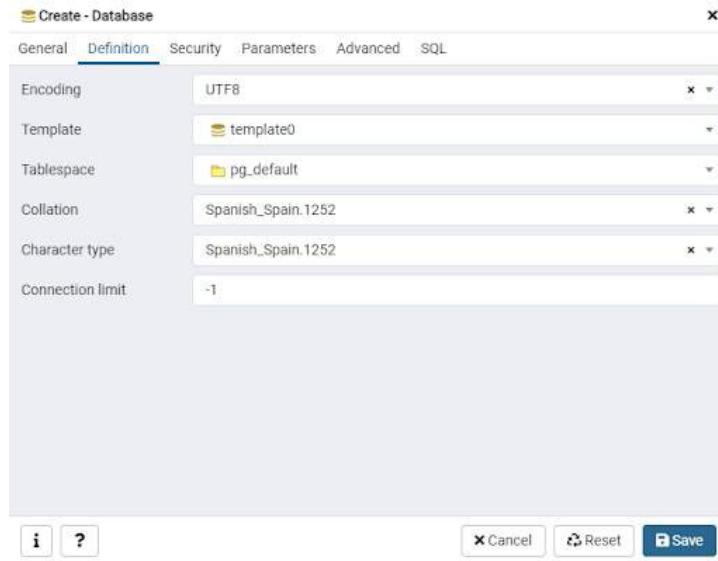
## Superusuario

Como propietario, seleccionaremos nuestro superusuario ‘postgres’ y, por último, añadimos también una breve descripción en el campo comentario.



Pulsamos sobre la pestaña ‘Definition’ y rellenamos los campos con los siguientes valores:

Propiedad	Valor
Encoding	UTF8
Template	template0
Tablespace	pg_default
Collation	Spanish_Spain.1252
Character type	Spanish_Spain.1252



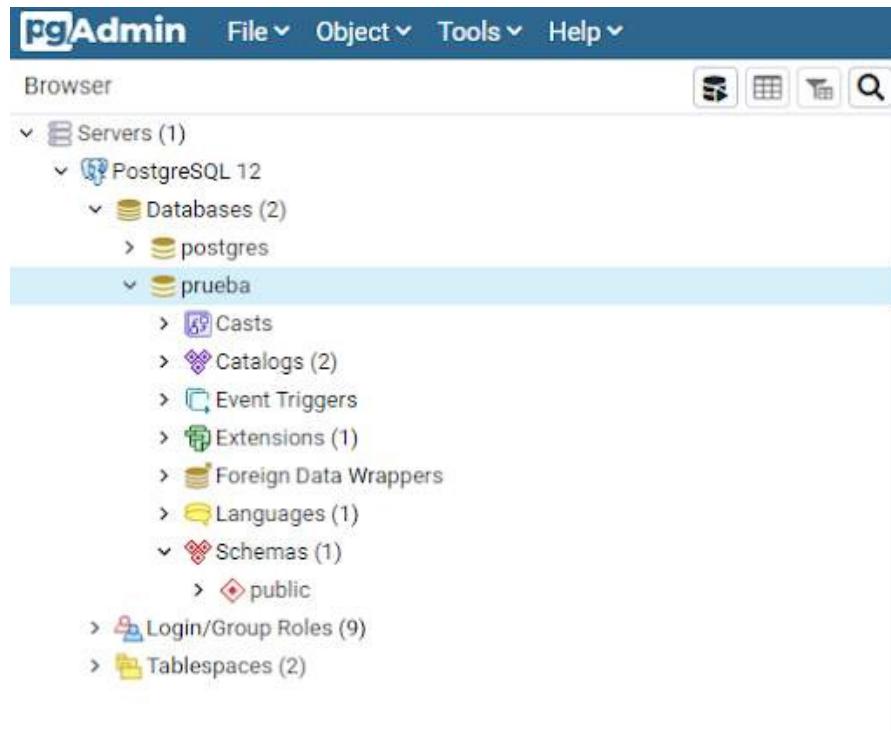
5

## Sentencia SQL

Por último, antes de pulsar sobre el botón ‘Save’, accediendo a la pestaña ‘SQL’, podremos conocer la sentencia SQL equivalente para crear una base de datos (nos resultará muy útil cuando queramos realizar la misma acción desde la consola de comandos, en la que no tendremos una interfaz visual para realizar las operaciones).

```
1 CREATE DATABASE prueba
2   WITH
3     OWNER = postgres
4     TEMPLATE = template0
5     ENCODING = 'UTF8'
6     LC_COLLATE = 'Spanish_Spain.1252'
7     LC_CTYPE = 'Spanish_Spain.1252'
8     TABLESPACE = pg_default
9     CONNECTION LIMIT = -1;
10
11 COMMENT ON DATABASE prueba
12   IS 'Es una base de datos de prueba';
```

Pulsando sobre el botón ‘Save’, se creará nuestra base de datos prueba. Comprobamos en la pantalla principal, en el panel de la izquierda, como se ha añadido un nuevo elemento dentro de Postgresql12 -> Databases.



## 6

### Schemas

Dentro del desplegable de objetos que cuelgan debajo de la base de datos de prueba (casts, catalogs, event triggers, schemas), haremos una parada en este último, en schemas.

Debes saber que PostgreSQL introduce un nivel más de jerarquía entre las bases de datos y las tablas, hablamos de los esquemas (schema).

---

Un schema es un contenedor de objetos, especialmente para nuestro interés, de tablas.

Repasemos, por lo tanto, cómo quedaría la jerarquía de entidades:

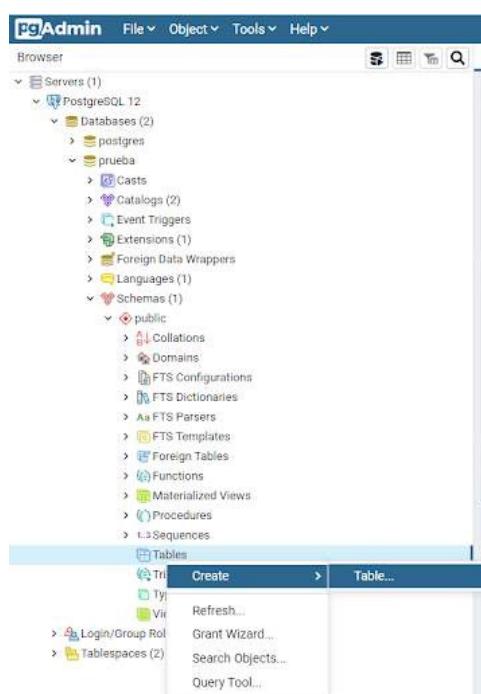
- Hemos creado un servidor de bases de datos, en concreto **PostgreSQL 12.5**.
  - Dentro del servidor de bases de datos, pueden **coexistir** numerosas bases de datos, nosotros hemos creado una base de datos llamada **prueba**.
  - De la misma manera, dentro de una base de datos, existen **distintos schemas**, que aglutinan distintos tipos de objetos. Por defecto, al crear una base de datos, se genera un schema llamado '**public**'.
  - Dentro de un schema, podremos **generar tablas** y otros objetos que ya iremos viendo.
- 

**Vista la jerarquía, creemos, entonces, nuestra primera tabla dentro del ‘schema public’ de nuestra base de datos prueba.**

7

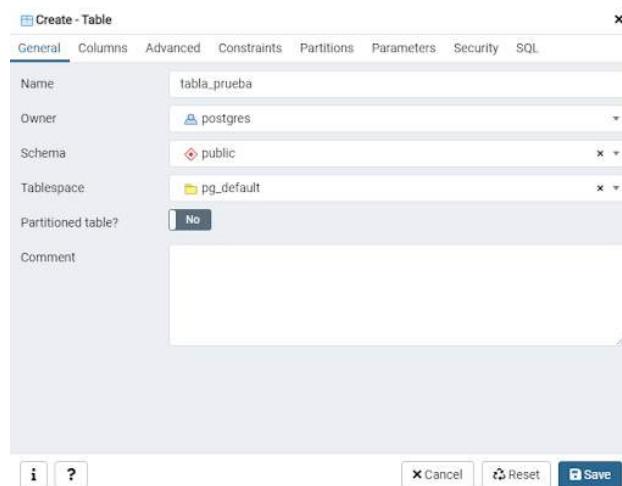
Crear

Para ello, nos **colocamos en el apartado ‘Tables’** dentro de pruebas/public, pulsamos el botón derecho y, en el menú contextual, seleccionamos ‘Create -> Table’.



Aparecerá de nuevo una ventana flotante en la que definiremos las características de nuestra tabla. Dentro de la pestaña inicial ‘General’, rellenaremos los campos de la siguiente manera:

- **Nombre:** tabla\_prueba.
- **Propietario:** postgres.
- **Schema:** public.
- **Tablespace:** pg\_default.



8

### Generar columnas

Continuamos desplazándonos a la pestaña ‘Columns’, donde generaremos las columnas de nuestra tabla. Para ello, iremos pulsando sobre el botón ‘+’ para añadir nuevas columnas. La rellenamos de la siguiente manera:

	Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?
<input checked="" type="checkbox"/>	id	serial			<input type="checkbox"/> No	<input checked="" type="checkbox"/> Yes
<input checked="" type="checkbox"/>	nombre	character varying	50		<input checked="" type="checkbox"/> Yes	<input type="checkbox"/> No
<input checked="" type="checkbox"/>	apellido1	character varying	50		<input checked="" type="checkbox"/> Yes	<input type="checkbox"/> No
<input checked="" type="checkbox"/>	apellido2	character varying	50		<input checked="" type="checkbox"/> Yes	<input type="checkbox"/> No
<input checked="" type="checkbox"/>	edad	integer			<input checked="" type="checkbox"/> Yes	<input type="checkbox"/> No
<input checked="" type="checkbox"/>	direccion	character varying	200		<input checked="" type="checkbox"/> Yes	<input type="checkbox"/> No

De la **configuración de la tabla** podemos destacar las siguientes **características** que pasamos a comentar:

#### Campo 'id'

Para el campo 'id', hemos seleccionado como data type un tipo especial llamado 'serial'. Es muy útil pues, a la hora de realizar inserciones sobre esta tabla, no será necesario **rellenar con valores este campo**, pues se hace automáticamente, por así decirlo es un tipo de dato auto incrementable. Es el tipo que se usa normalmente para columnas identificativas. Además, se ha marcado como Primary Key, es decir, es el campo que va a representar únicamente cada fila de la tabla sobre las demás.

#### Campos de tipo texto

Para los campos de tipo texto, hemos elegido el tipo de datos 'Character varying', para el cual tenemos que definir la longitud máxima. Así pues, para el campo 'nombre' hemos seleccionado como máximo 50 caracteres.

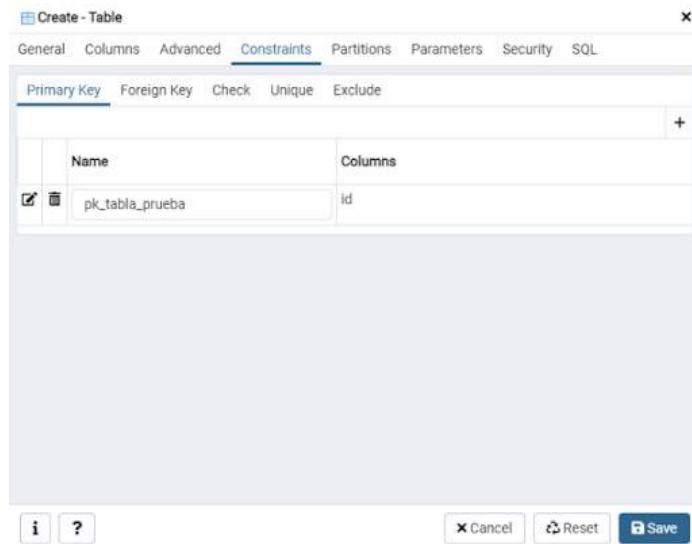
#### Valores NULL

Observarás también que todos los campos salvo 'id' están marcados como 'Not NULL'. Cuando en base de datos hacemos referencia a un valor NULL, nos referimos a que el campo no está informado (está vacío). Imponiendo la restricción 'Not NULL', estamos obligando a que el campo venga siempre lleno. (Podrás pensar por qué no hemos marcado como 'Not NULL' el campo 'id', en realidad no es necesario, ya que, por concepto, una columna marcada como Primary Key debe siempre estar informada).

9

## Pestaña 'Constraints'

Rellenadas ya las columnas, seguimos navegando por las pestañas superiores de la ventana desplegada, en concreto avanzamos hacia la pestaña 'Constraints'. En esta pestaña, añadiremos **todas las restricciones referenciales**. En nuestro caso, solo añadiremos la restricción 'Primary Key' para el campo 'id'. Lo haremos de la misma manera que hemos ido añadiendo columnas:



10

## Pestaña 'SQL'

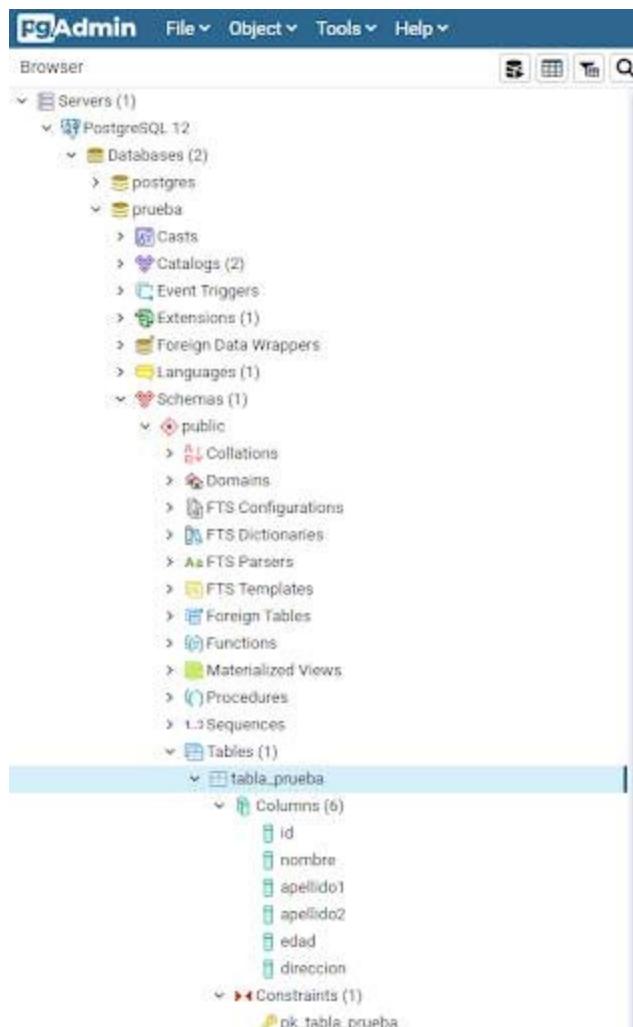
Por último, antes de pulsar sobre el botón aceptar para generar la tabla, navegamos hasta la pestaña 'SQL' para conocer la **instrucción SQL equivalente** que genera la tabla.

```

CREATE TABLE public.tabla_prueba
(
    id serial,
    nombre character varying(50) NOT NULL,
    apellido1 character varying(50) NOT NULL,
    apellido2 character varying(50) NOT NULL,
    edad integer NOT NULL,
    direccion character varying(200) NOT NULL,
    CONSTRAINT pk_tabla_prueba PRIMARY KEY (id)
)
TABLESPACE pg_default;
ALTER TABLE public.tabla_prueba
    OWNER to postgres;

```

Si observamos el menú de la izquierda, vemos que la tabla ya se encuentra disponible dentro del listado de tablas. Además, vemos que tenemos el detalle de columnas con su nombre y tipo, además de las restricciones referenciales que hayamos introducido:



11

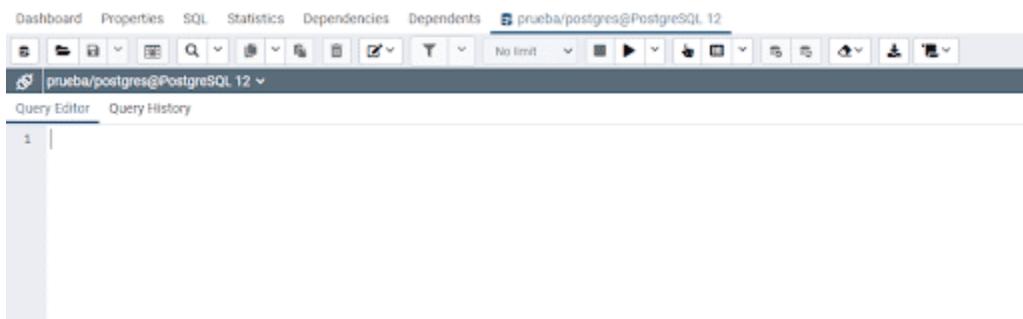
## Consultar tablas

Una vez creada la tabla, ¿cuál será el siguiente paso? Efectivamente, primero consultaremos la tabla, para **posteriormente insertar datos**. Si seleccionamos la tabla del menú de la izquierda y nos fijamos en el siguiente menú:

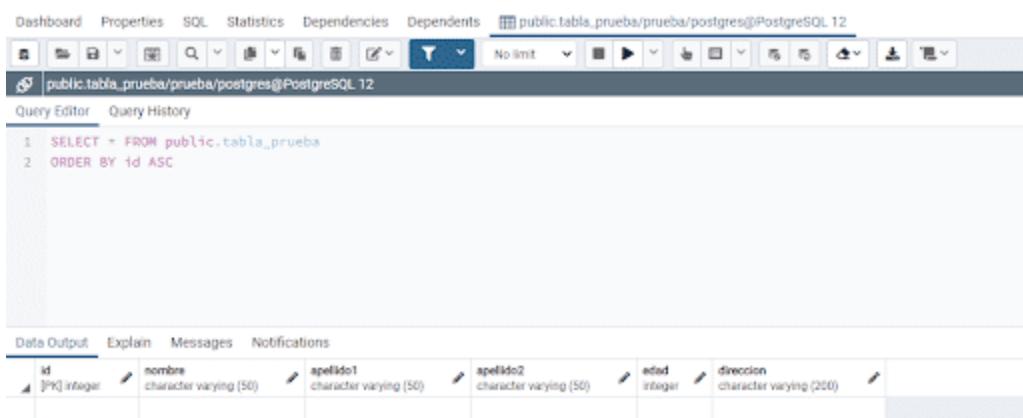


Observamos los siguientes botones:

- El **primer botón** nos abrirá el editor de consultas para poder construirlas y posteriormente ejecutarlas.



- El **segundo botón** consultará directamente la tabla que tengamos seleccionada en el menú de la izquierda. Si no estuviera seleccionada ninguna, el botón aparecería desmarcado. En nuestro caso, en el **panel superior**, aparece la consulta SQL que realiza el botón y debajo en el panel inferior nos muestra el resultado de ejecutar la consulta. Como esperamos, nuestra tabla está vacía, ya que todavía no hemos realizado ninguna inserción.



- El **tercer botón** nos permitirá consultar la tabla que tengamos seleccionada en el menú de la izquierda, pero añadiendo una condición que nos permita filtrar. De la misma manera que en el punto anterior, en el panel superior aparece la consulta que se ejecuta al pulsar sobre el botón e introducir la condición del filtro y, en el panel de abajo, el resultado de la ejecución.

The screenshot shows the pgAdmin 4 interface. At the top, there's a toolbar with various icons. Below it is a menu bar with 'Dashboard', 'Properties', 'SQL', 'Statistics', 'Dependencies', 'Dependents', and a connection status for 'public.table.pr... public.table.prueba/postgres@PostgreSQL 12'. The main area has tabs for 'Query Editor' and 'Query History'. In the 'Query Editor' tab, there is a code editor containing the following SQL:

```

1: SELECT * FROM public.table_prueba
2: WHERE id = 12
3:

```

Below the code editor is a 'Data Output' tab, which displays a table schema:

<b>id</b>	<b>nombre</b>	<b>apellido1</b>	<b>apellido2</b>	<b>edad</b>	<b>direccion</b>

- Por último, el **cuarto botón** es un buscador de objetos muy útil en situaciones en las que nuestra base de datos tenga un número elevado de objetos. Por ejemplo, buscando la cadena de caracteres “pru” aparecen todos los objetos que contengan en su nombre el trío de caracteres “pru”:

The screenshot shows the 'Search Objects' dialog in pgAdmin 4. The search term 'pru' is entered in the search input field. The results table has columns for 'Object name', 'Type', and 'Browser path'. There are three matches found:

Object name	Type	Browser path
pk_tabla_prueba	Primary Key	Schemas/public/Tables/tabla_prueba/Constraints/...
tabla_prueba_id_seq	Sequences	Schemas/public/Sequences/tabla_prueba_id_seq
tabla_prueba	Tables	Schemas/public/Tables/tabla_prueba

At the bottom of the dialog, it says '3 matches found.' and has 'Close' and 'Help' buttons.

Si navegamos hasta nuestra tabla en el menú de la izquierda y pulsamos sobre esta con el botón derecho de nuestro ratón, aparecerá un menú contextual en el que podremos ver todas las opciones que tenemos de ejecutar sobre nuestra tabla.

Nos centraremos en la opción ‘Scripts’ y seleccionaremos la opción ‘INSERT Script’. PostgreSQL automáticamente genera la consulta SQL que sirve para insertar datos en la tabla (fíjate bien en la sintaxis). Tan solo tenemos que sustituir los interrogantes (?) por el valor real que queramos insertar.



¡Ah!, y recuerda que para los campos de tipo cadena de texto hay que iniciar y finalizar el texto con el carácter comilla simple ('').

Por ejemplo:

```
INSERT INTO public.tabla_prueba(  
    nombre, apellido1, apellido2, edad, direccion)  
VALUES ('Luis', 'Aguilar', 'Martínez', 25, 'Calle Buenavista 13 3A CP 28012');
```

Y pulsamos sobre el botón ‘Execute/Refresh’ de la botonera superior.

O si preferimos con el teclado, pulsando el botón F5 para poder ejecutar la consulta.

Si todo ha ido bien, deberá devolver el siguiente mensaje:

**INSERT 0 1 Query returned successfully in 70 msec.**

Para comprobar que efectivamente todo ha funcionado, seleccionemos nuestra tabla en el menú lateral izquierdo y pulsemos sobre el botón de consultar contenido de tabla para ver que la información está insertada. En nuestro caso, es así:

Data Output					
	<b>id</b> (PK) integer	<b>nombre</b> character varying (50)	<b>apellido1</b> character varying (50)	<b>apellido2</b> character varying (50)	<b>edad</b> integer
1	1	Luis	Aguilar	Martinez	25

**Hagamos algunos experimentos...**

Reutilicemos la consulta que utilizamos antes para **insertar nuestros primeros datos** y eliminemos la referencia al campo dirección tal que así:

```
INSERT INTO public.tabla_prueba(
    nombre, apellido1, apellido2, edad)
VALUES ('María', 'Fernández', 'Pérez', 32);
```

¿Qué piensas que ocurrirá? ¿Se insertará o no? Aunque ahora mismo no conozcas la sintaxis, ¿ves algo raro? Ejecutemos la consulta y observemos qué ocurre:

```
ERROR: el valor null para la columna «direccion» viola la restricción not null
DETAIL: La fila que falla contiene (2, María, Fernández, Pérez, 32, null).
SQL state: 23502
```

---

El error es bastante explicativo, hemos impuesto una restricción que no se ha cumplido para el campo ‘dirección’, ya que al no haber hecho referencia alguna a él en la consulta SQL, lo hemos dejado vacío (es decir, es igual a NULL) y, como recordarás, al definir la tabla impusimos que ningún campo de esta quedara vacío.

---

Realicemos otra prueba.

¿Qué pasa si insertamos un nuevo registro en la tabla, indicando explícitamente el valor 1 para el campo ‘id’? (Ya existe un registro en la tabla con valor 1 para el campo ‘id’).

Lo hacemos de la siguiente manera:

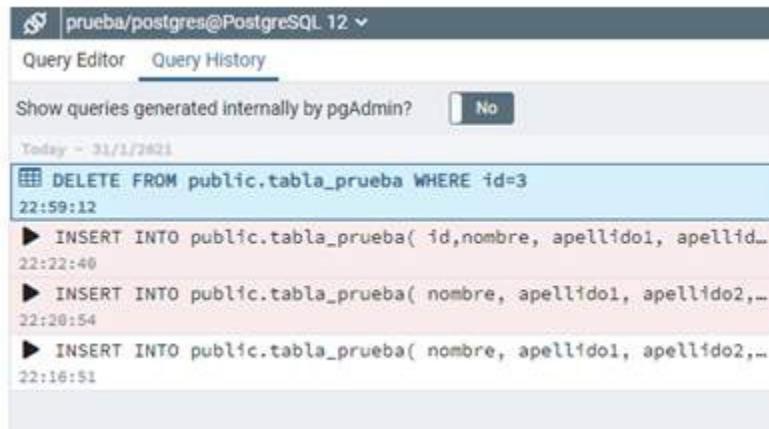
```
INSERT INTO public.tabla_prueba(  
    id,nombre, apellido1, apellido2, edad,direccion)  
VALUES (1,'María', 'Fernández', 'Pérez', 32,'Avenida de la Ilustración N°123 4ºD');
```

Como tu intuición te estará diciendo, se produce un error:

```
ERROR: llave duplicada viola restricción de unicidad «pk_tabla_prueba»  
DETAIL: Ya existe la llave (id)=(1).  
SQL state: 23505
```

Esto se debe a que hemos definido una restricción al crear la tabla, indicando que el campo ‘id’ era la Primary Key de la tabla, por lo que no puede contener valores repetidos.

Por último, veamos **una funcionalidad muy útil** para cuando cojamos soltura realizando operaciones sobre la base de datos. La funcionalidad del histórico de consultas:



The screenshot shows the pgAdmin interface with the 'Query History' tab selected. At the top, there is a checkbox labeled 'Show queries generated internally by pgAdmin?' with the value 'No' checked. Below this, it says 'Today - 31/12/2021'. The history list contains the following entries:

- 22:59:12 ► DELETE FROM public.tabla\_prueba WHERE id=3
- 22:22:40 ► INSERT INTO public.tabla\_prueba( id,nombre, apellido1, apellido2,... ) VALUES ( 1,'Juan','Perez','Garcia',... )
- 22:20:54 ► INSERT INTO public.tabla\_prueba( nombre, apellido1, apellido2,... ) VALUES ( 'Juan','Perez','Garcia',... )
- 22:16:51 ► INSERT INTO public.tabla\_prueba( nombre, apellido1, apellido2,... ) VALUES ( 'Juan','Perez','Garcia',... )

---

Puedes encontrarlo como una pestaña, justo al lado del editor de consultas. En ella, podemos visualizar las últimas consultas que hemos ejecutado.

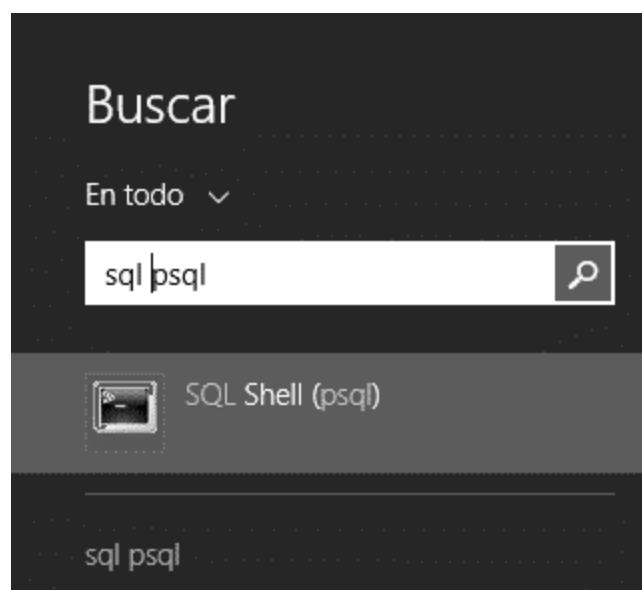
# Ejecución por consola

X Edix Educación

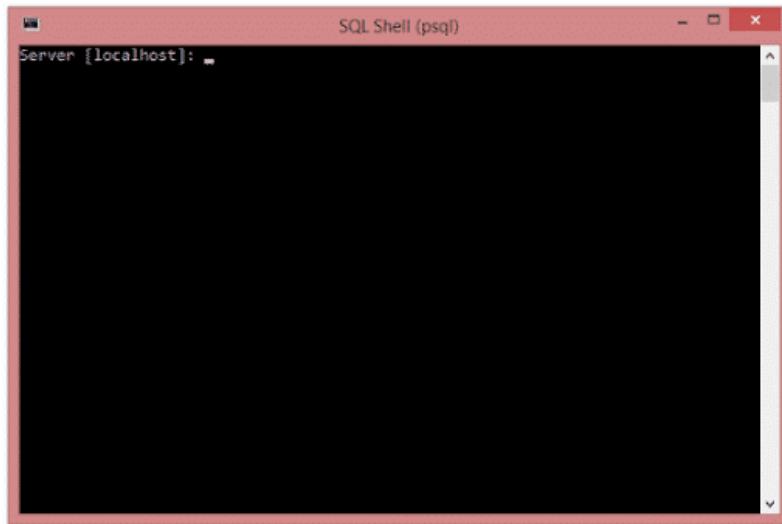
---

La totalidad de operaciones que hemos realizado a través de PgAdmin podemos realizarlas a través de la consola de comandos que **se instala junto al resto del software**. Esta utilidad es muy valiosa para sistemas operativos que no poseen un entorno gráfico, donde la gestión del sistema operativo se realiza a través de shells (líneas de comando). Esta opción suele ser puesta en práctica por expertos en la materia, por lo que, si en algún momento tienes que trabajar en estos sistemas, puedes pedir al técnico correspondiente que te ayude a conectar tu PgAdmin a la base de datos de ese servidor remoto.

Comprobemos que se ha instalado la utilidad, seleccionamos en el buscador de programas lo siguiente:

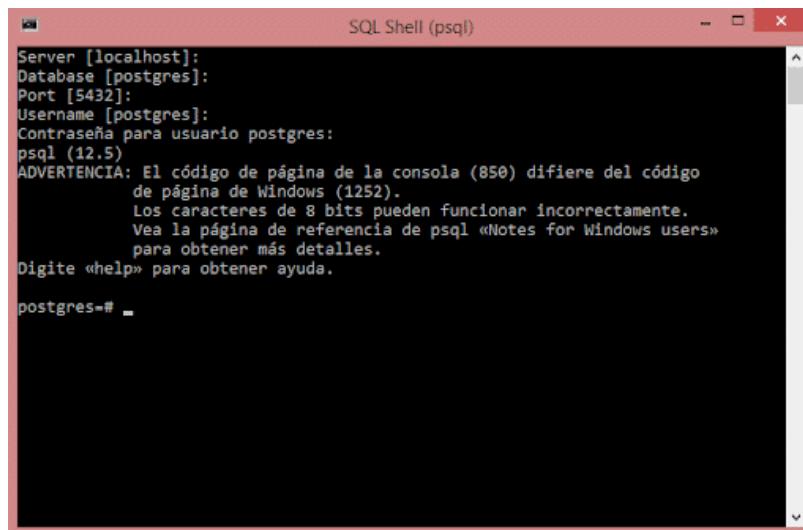


Si pulsamos sobre la **SQL Shell (psql)**, nos aparecerá una ventana tal que así (el diseño puede variar según la versión de Windows que estemos utilizando).



En esta pantalla, **la utilidad nos está solicitando los parámetros de conexión**, indicando entre corchetes el valor por defecto. Si no introducimos ningún dato y pulsamos sobre el botón enter del teclado tomará ese valor.

Así pues, pulsando sucesivamente, introduciremos los parámetros por defecto para el servidor, base de datos, puerto y usuario. Eso sí, tendremos que **introducir la contraseña que generamos al comienzo del tutorial**, cuando instalamos el software. Si todo ha ido bien, se nos presentará la siguiente pantalla:



Si no estamos muy familiarizados con las terminales, básicamente podríamos definirlo como un **programa que espera**, de manera indefinida, las instrucciones por parte de un usuario para interpretarlas y devolver resultados.

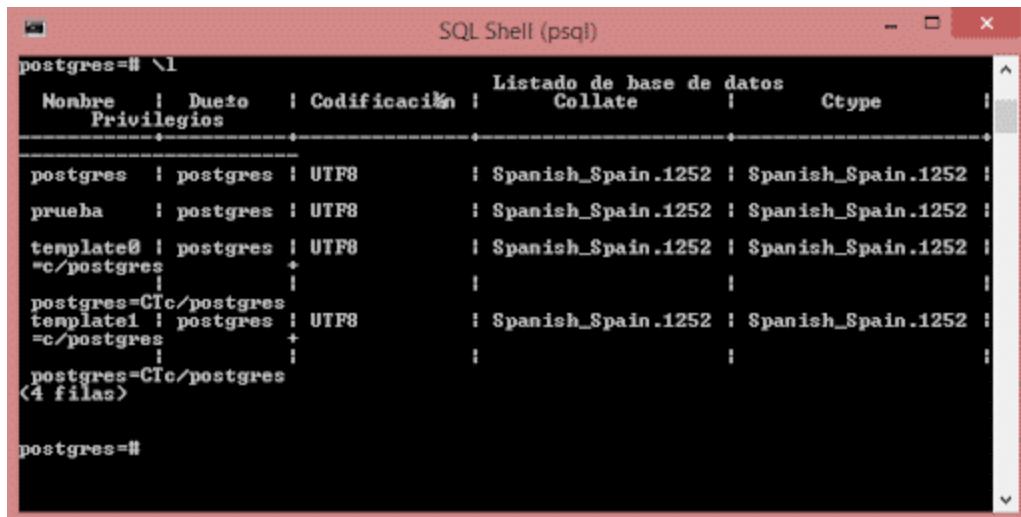
Así pues, vemos que la última línea nos indica primeramente en qué base de datos nos encontramos (postgres) y, por último, el carácter parpadeante nos indica que el programa está a la espera de nuestras instrucciones.

---

**Ejecutaremos algunos comandos para demostrar que la consola tiene el mismo potencial que PgAdmin.**

\l

Nos permitirá listar las bases de datos disponibles.



The screenshot shows a terminal window titled "SQL Shell (psql)". The command "\l" is entered, followed by a list of databases:

Nombre	Dueño	Codificación	Collate	Ctype
postgres	postgres	UTF8	: Spanish_Spain.1252	: Spanish_Spain.1252
prueba	postgres	UTF8	: Spanish_Spain.1252	: Spanish_Spain.1252
template0	postgres	UTF8	: Spanish_Spain.1252	: Spanish_Spain.1252
template1	postgres	UTF8	: Spanish_Spain.1252	: Spanish_Spain.1252

At the bottom, it says "postgres=CTc/postgres <4 filas>" and "postgres=#".

\l+

Igual que la instrucción anterior, pero con más detalle.

Listado de bases de datos									
nombre	Dueño	Codificación	Collate	Ctype	privilegios	Tamaño	tablespace	Descripción	
postgres	postgres	UTF8	Spanish_Spain.1252	Spanish_Spanish.1253		7901 kB	pg_default	default administrative connection database	
prueba	postgres	UTF8	Spanish_Spain.1252	Spanish_Spanish.1253		7603 kB	pg_default	Es una base de datos de prueba	
template0	postgres	UTF8	Spanish_Spain.1252	Spanish_Spanish.1252	-c/postgres	+ 7769 kB	pg_default	unmodifiable empty database	
template1	postgres	UTF8	Spanish_Spain.1252	Spanish_Spanish.1252	-c/postgres	+ 7769 kB	pg_default	default template for new databases	

(4 filas)

\c nombre\_base\_datos

Nos conectará a la base de datos que le indiquemos en ‘nombre\_base\_datos’. Observa cómo después de ejecutarla, se ha cambiado el nombre de ‘postgres’, por ‘prueba’:

```
postgres=# \c prueba
Ahora está conectado a la base de datos «prueba» con el usuario «postgres».
prueba# -
```

\dt

Lista las tablas disponibles dentro de la base de datos a la que nos hemos conectado previamente.

```
prueba# \dt
      Listado de relaciones
 Esquema |    Nombre     | Tipo | Dueño
-----+-----+-----+-----
 public | tabla_prueba | tabla | postgres
(1 fila)
```

\dt+

Igual que la instrucción anterior, pero con más detalle.

```
prueba=# \dt+
              Listado de relaciones
 Esquema |     Nombre      | Tipo   | Dueño | Tamano | Descripción
-----+-----+-----+-----+-----+-----+
 public | tabla_prueba | tabla | postgres | 8192 bytes |
(1 fila)
```

\d nombre\_tabla

Detalle de la tabla ‘nombre\_tabla’ (esquema de la tabla):

```
prueba=# \d tabla_prueba
           Tabla Xpublic.tabla_prueba
 Columna |      Tipo       | Ordenamiento | Nutable | Por omisión
-----+-----+-----+-----+-----+
 id    | integer        |             | not null | nextval('tabla_prueba_id_seq'::regclass)
 nombre | character varying(50) |             | not null |
 apellido1 | character varying(50) |             | not null |
 apellido2 | character varying(50) |             | not null |
 edad   | integer        |             | not null |
 dirección | character varying(200) |             | not null |
Indices:
 "pk_tabla_prueba" PRIMARY KEY, btree (id)
```

## Ejecutar SQL

Por supuesto, podemos ejecutar SQL para consultar, insertar, borrar y actualizar información:

a. Consultamos.

```
prueba=# select * from tabla_prueba;
 id | nombre | apellido1 | apellido2 | edad |         dirección
----+-----+-----+-----+-----+-----+
 1 | Luis   | Aguilar  | Martínez | 25  | Calle Buenavista 13 3A CP 28012
(1 fila)
```

b.

Insertamos.

```
prueba=# insert into tabla_prueba(nombre,apellido1,apellido2,edad,direccion) values ('Laura','Casanova','Verde',18,'Calle de la Esperanza N12 3D')  
INSERT 0 1
```

c.

Consultamos de nuevo para chequear si se ha realizado la inserción.

```
prueba=# select * from tabla_prueba;  
 id | nombre | apellido1 | apellido2 | edad |          direccion  
---+-----+-----+-----+-----+-----+  
  1 | Luis  | Aguilar | Martínez |  25 | Calle Buenavista 13 3A CP 28012  
  2 | Laura | Casanova | Verde   |  18 | Calle de la Esperanza N12 3D  
(2 filas)
```

# Últimas configuraciones

X Edix Educación

---

Por último, entraremos en la carpeta de instalación de PostgreSQL y buscaremos la carpeta 'bin', para investigar el resto de las utilidades que se encuentran disponibles y que nos pueden ayudar mucho en nuestro trabajo, por ejemplo, las utilidades 'pg\_dump' y 'pg\_restore', las cuales se encargan de generar copias de seguridad de nuestras bases de datos para posteriormente poder restaurarlos (respectivamente).

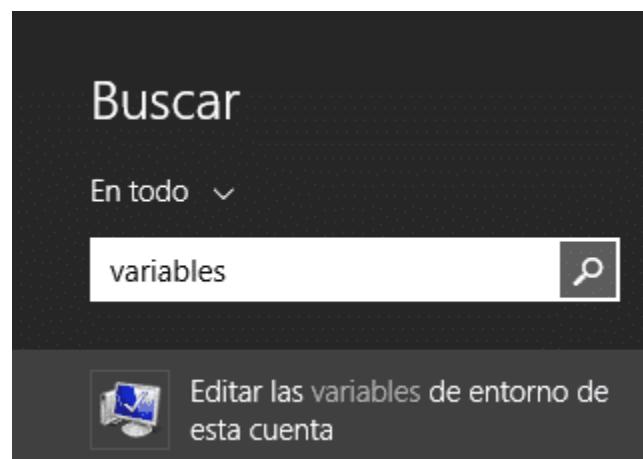
---

Para poder usarlos, añadiremos la ruta donde se encuentran dichos programas al PATH de tu usuario en Windows.

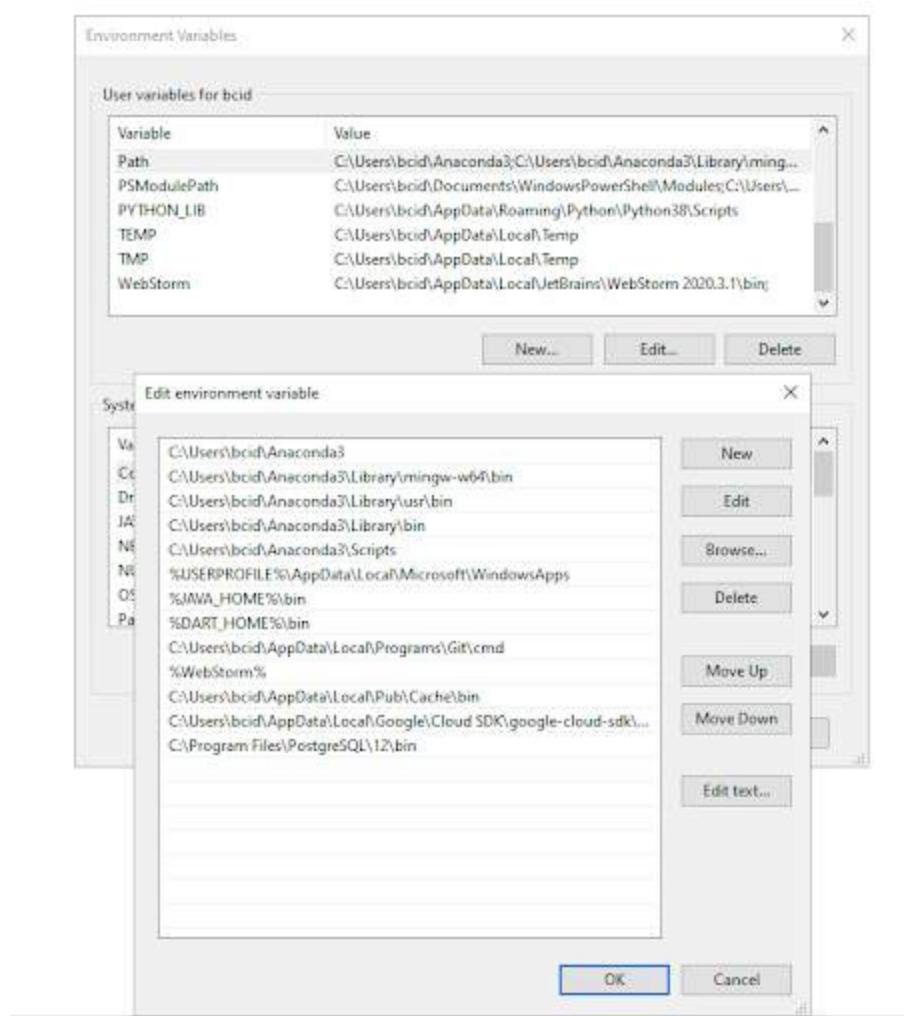
---

Esto se hace de la siguiente manera:

Accedemos a las variables de entorno de nuestro sistema.

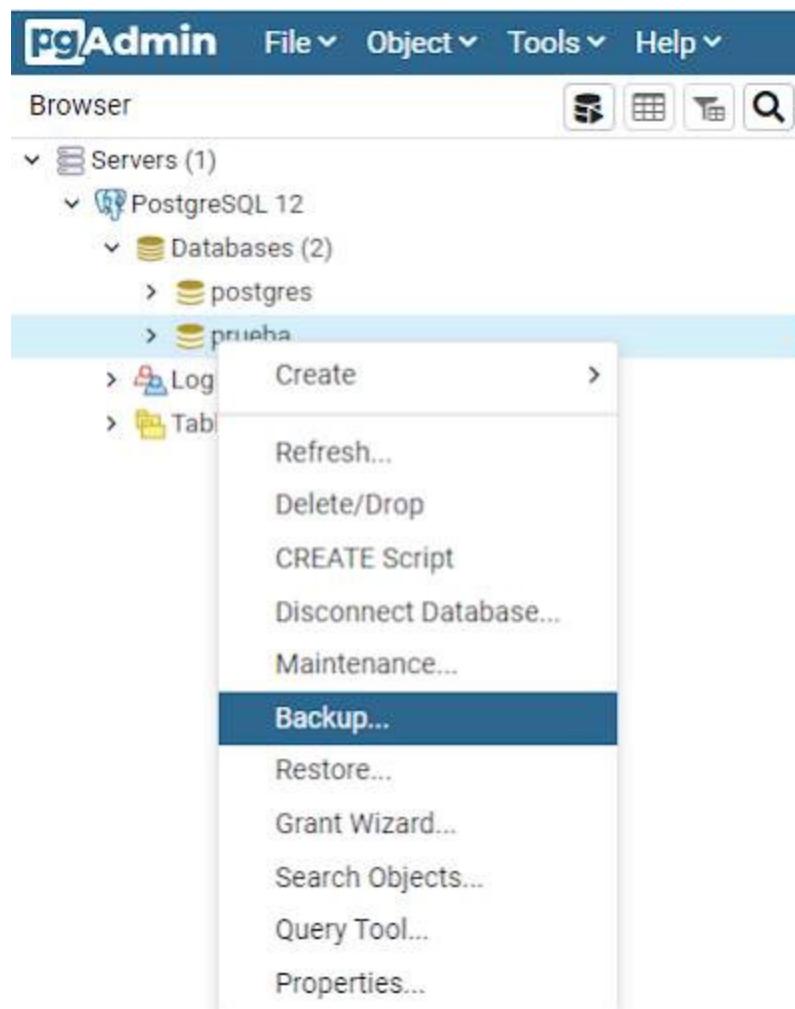


Editamos la variable **Path**, dentro del apartado ‘Variables de usuario para XXXX’ y añadimos (siempre añadimos, nunca borramos) en ‘Valor de variable’ la ruta **C:/Program Files/PostgreSQL/12/bin:**



Con esto, conseguimos que, para **ejecutar los programas** que se encuentran en la ruta especificada desde la línea de comandos, no haga falta más que poner su nombre, sin añadir la ruta.

No os preocupéis, ya que todas las opciones estarán también disponibles desde 'PgAdmin'.



# Conclusiones

X Edix Educación

---

Como **pequeño resumen** de lo comentado en este fastbook destacamos los aprendizajes fundamentales:

- Vimos los **principales motivos** por los que hemos elegido el SGDB PostgreSQL para su uso en la parte práctica de esta asignatura.
  - Hemos realizado todas las **instalaciones y configuraciones** necesarias para realizar los ejercicios y ejecutar la práctica de los próximos fastbooks.
  - Hicimos un **primer paseo por la herramienta PgAdmin**, que será la que usemos como interfaz gráfica de este SGDB, en el que hemos realizado las operaciones básicas que podremos realizar en nuestro día a día, como crear bases de datos o realizar consultas.
  - Realizamos las mismas **operaciones desde la terminal**, aunque hemos visto que este método es el reservado para los expertos de la materia.
- 

**Por todo esto, podríamos decir que ya estamos preparados para empezar a trabajar con PostgreSQL realizando los primeros ejercicios en los siguientes fastbooks.**

¡Enhorabuena! Fastbook superado

edix

Creamos Digital Workers