postgres

Índice

Introducción

PostgreSQL es un motor de base de datos relacional SQL. Será el encargado de almacenar y gestionar los datos de nuestra aplicación. PostgreSQL es el sistema más común entre los sistemas que prestan servicio de hosting de bases de datos de manera gratuita en algunos planes. En particular, esta cheatsheet se enfoca en utilizar PostgreSQL, pgAdmin (una interfaz gráfica para PostgreSQL) y conectarlo a una base de datos hosteada en Vercel.

Creación de una base de datos en Vercel

Una vez que ingresaron a su cuenta de Vercel, y crearon un proyecto en el que desean utilizar una base de datos PostgreSQL, sigan los siguientes pasos para crear una base de datos en Vercel:

Ir a la sección "Storage" del proyecto.

Luego, tendrán que seleccionar para crear una nueva base de datos en PostgreSQL.

Conexión a la base de datos

Para conectarse a la base de datos, tanto desde pgAdmin como desde la aplicación, se necesitan datos que se encontrarán en la sección "Quickstart" de la base de datos creada en Vercel.

pgAdmin

pgAdmin es una interfaz gráfica para PostgreSQL que permite visualizar y manipular los datos de la base de datos.

Conexión a la base de datos desde pgAdmin

Para conectarse a la base de datos desde pgAdmin, sigan los siguientes pasos:

Descargar pgAdmin desde pgAdmin.

Abrir pgAdmin y agregar un nuevo servidor
new_server
Completar los datos del servidor con los datos de la base de datos de Vercel vistos en la sección "Conexión a la base de datos".
server_data

Una vez que se conectaron a la base de datos, podrán visualizar y manipular los datos de las distintas tablas que tengan en la base de datos.

Es importante remarcar que si la base de datos que utilizaremos es la llamada "verceldb", pues no tenemos permisos para editar la llamada "postgres".

Creación de una tabla

Para crear una tabla en la base de datos, deberán acceder desde el menú de la izquierda a "Servers" → "nombre_del_servidor" → "Databases" → "verceldb" → "Schemas" → "public" → "Tables" → "Create" → "Table ... ".

Luego, deberán completar los datos de la tabla que desean crear. Y agregando las columnas que deseen.

table

En general, queremos que todos los campos sean NOT NULL, excepto los que no sean necesarios.

Tipos de datos

- `integer` : Números enteros.
- `real` : Números con coma.
- `character varying` : Es el equivalente a `varchar` en otros motores de base de datos.
- `text` : Texto.
- `date` : Fecha.
- `timestamp with time zone` : Fecha y hora.
- `boolean` : Verdadero o falso.

Autoincremental

Para que un campo sea autoincremental, deberán seleccionar la columna y darle el tipo de dato `serial` . Es importante que el campo que sea autoincremental sea la clave primaria de la tabla.

pg en Node.js

Para conectarse a la base de datos desde Node.js, se puede utilizar el paquete `pg` . Aquí su documentación.

Instalación

Para instalarlo, deberán correr el siguiente comando (dentro de un proyecto de Node.js):

```
npm install pg
```

Utilización

Conexión

Una vez que instalaron el paquete, podrán conectarse a la base de datos y realizar consultas.

```
import { Client } from "pg"; // Importamos el cliente de pg (recordar que para utilizar 'import' es necesario us

// Pueden (y deberían) utilizar variables de entorno para almacenar los datos de conexión (dotenv)
const client = new Client({
    user: "<usuario>",
    host: "<host>",
    database: "<database>",
    password: "<password>",
    port: 5432,
});

client.connect(); // Nos conectamos a la base de datos
```

Consultas

Una vez que se conectaron a la base de datos, podrán realizar consultas. Por ejemplo, para obtener todos los registros de una tabla:

```
const res = await client.query("SELECT * FROM <nombre_tabla>");
console.log(res.rows); // Imprimimos los registros
```

Para insertar un registro en una tabla:

```
await client.query("INSERT INTO <nombre_tabla> (columna1, columna2) VALUES ($1, $2)", [valor1, valor2]);
```

Notar que en el segundo argumento de `query` se pasan los valores que se quieren insertar en la tabla. Esto se escribe un poco distinto a los prepared statements de MySQL (utilizábamos el `?`). En este caso, se utilizan los `$1`, `$2`, etc. para referenciar a los valores que se quieren insertar, nos permite cambiar el orden de los valores sin tener que cambiar el orden de los .