Implementación y gestión de una base de datos

Query y JSON

{desafío} latam_

Implementar la conexión a una base de datos PostgreSQL en una aplicación Node utilizando buenas prácticas.

Programar instrucciones para la obtención y manipulación de información desde una base de datos utilizando buenas prácticas, acorde al entorno Node.js.

{desafío} latam_

- Unidad 1:
  Implementación y gestión de una base de datos
- Unidad 2:
  Transacciones y API REST
- Unidad 3:
  Trabajo práctico

Te encuentras aquí

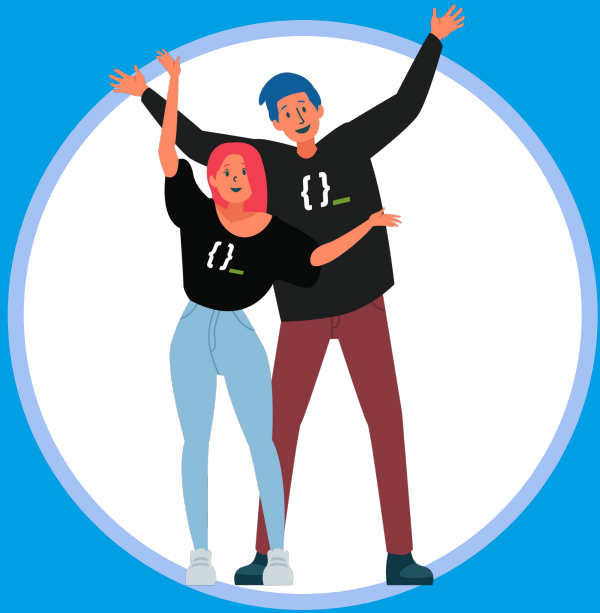
¿Qué aprenderás en esta sesión?

- *Realizar consultas con texto parametrizado utilizando un JSON como argumento del método query.*

{desafío}
latam_

¿Tienes alguna noción sobre lo que veremos hoy?

¡Comentemos!



{desafío} latam_

/* Query utilizando texto plano con parámetros */

Query utilizando texto plano con parámetros

El método query además de permitir realizar consultas mediante un texto plano, también nos permite realizarlas en forma parametrizada. De esta forma, el servidor PostgreSQL recibe la consulta sin modificaciones con los parámetros en forma separada, estos son sustituidos en forma segura en la base de datos.

Query utilizando texto plano con parámetros

¿Por qué utilizaría texto plano parametrizado?

En SQL se popularizó un ataque a los sistemas y bases de datos llamado SQL Injection, este consiste en lograr ingresar una instrucción que vulnere, copie, elimine o afecte de alguna manera la base de datos.

```
pool.query(text[String], values[Array])
```

/* Objeto JSON como argumento de consultas */

Objeto JSON como argumento de consultas

Al manejar un objeto JSON o variables independientes en la ejecución de la consulta, la base de datos evaluará en forma separada los parámetros del resto de la consulta y descartará cualquier instrucción SQL.

Objeto JSON como argumento de consultas

¿Y cuál es la estructura de este JSON?

Este objeto sólo debe tener una propiedad “text” que contendrá la sentencia SQL a ejecutar y “values” donde serán definidos los parámetros que recibirá la consulta. A continuación te mostramos cómo sería la sintaxis de esto:

```
{
 text: '<consultas con parámetros $1,$2, ...',
 values: ['valor1', 'valor2', ...],
};
```

Este objeto sería un argumento del método query() y será interpretado por el paquete pg sin problema.

/* Consultas SQL con JSON */

{desafío}
latam_

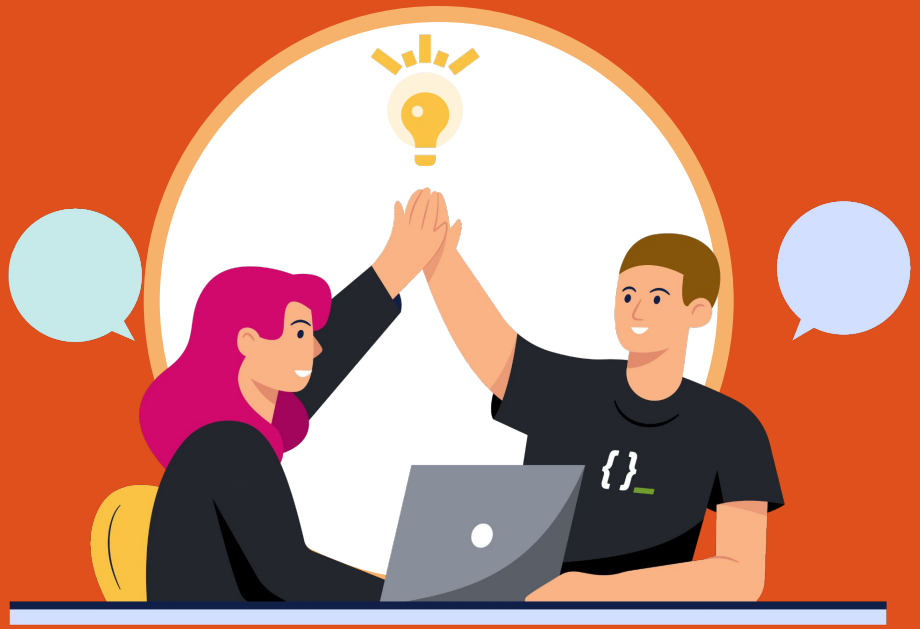
Consultas SQL con JSON

Realizar otra inserción, pero en este caso se tratará de un cinturón color gris de talla 98. Para esto sigue los siguientes pasos:

- **Paso 1:** Crear un objeto con las propiedades text y values en donde deberás definir la consulta parametrizada y los valores en forma de arreglo respectivamente.

- **Paso 2:** Enviar el objeto JSON creado en el paso 1 como argumento del método query

¿Cómo te fue con el ejercicio? Comenta lo que lograste aprender.



{desafío} latam_

/* Row Mode */

Row Mode

Los registros por defecto son retornados en forma de objeto, pero en caso de querer manejar esta data en forma de arreglos independientes podemos hacerlo gracias al Row Mode.

```
const getRopa = async () => {
  const result = await pool.query({
    rowMode: "array",
    text: "SELECT * FROM ropa",
  });

  console.log(result.rows);
};

getRopa();
```

```
PS D:\Node JS\connectObjectClient> node .\index.js
[ [ 10, 'Jean gris', '44' ], [ 1, 'Jean rojo', '46' ] ]
```

El resultado, en vez de ser un arreglo de objetos, se recibe únicamente como arreglo.

/* Obteniendo arreglos como respuesta */

Obteniendo arreglos como respuesta

Obtener todos los registros de la tabla ropa en forma de arreglos, prosigue con los siguientes pasos:

- **Paso 1:** Declarar la propiedad “rowMode” con el valor “array”, en el JSON como argumento al método query.
- **Paso 2:** Cambiar el valor de la propiedad “text” para realizar una consulta que pida todos los registros de la tabla ropa.
- **Paso 3:** Imprimir por consola la propiedad “rows” del objeto recibido como respuesta de la consulta para visualizar la data que se está consultando en formato de arreglo.

¿Cuáles son las ventajas y desventajas de utilizar JSON en consultas SQL en comparación con las consultas SQL tradicionales en aplicaciones React?

Próxima sesión...

- *Identifica los riesgos de SQL Injection y las medidas para mitigarlo en una consulta de obtención de datos.*

- *Programa instrucciones de obtención de datos utilizando Prepared Statements para mitigar el riesgo de SQL Injection.*

{desafío}
latam_

Academia de
talentos digitales