

Portfolio 6 – Yuri Garcia Campos

1- Execute:

- a. CREATE TABLE testeSQLi (id serial, data text);
- b. INSERT INTO testeSQLi (data) VALUES ('Important data');

2- Crie a função:

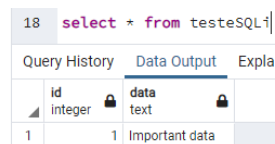
```
CREATE or replace FUNCTION unsafe_add_table(text)
RETURNS void AS $func$
BEGIN
EXECUTE 'CREATE TABLE ' || $1 || '(item_1 int, item_2 int)';
END
$func$ LANGUAGE plpgsql;
```

3- Use a função:

```
SELECT unsafe_add_table('T1(id int); DELETE FROM testeSQLi; --'); -- malicious call with SQL injection
```

4- Selecione os dados da tabela testeSQLi. O que aconteceu?

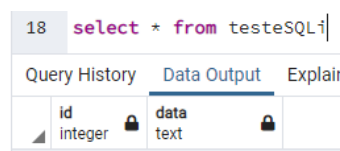
Antes do item 3:



The screenshot shows a SQL query editor with the query `select * from testeSQLi` and its results in a table. The table has two columns: 'id' (integer) and 'data' (text). There is one row with 'id' 1 and 'data' 'Important data'.

id	data
1	Important data

Depois:



The screenshot shows the same SQL query editor with the query `select * from testeSQLi`. The results table is now empty, indicating that the data has been deleted.

id	data
----	------

Todo conteúdo da tabela foi apagado.

5- Execute o Passo 1b) de novo!

6- Substitua a função por:

```

CREATE or replace FUNCTION unsafe_add_table(text)
RETURNS void AS
$func$
BEGIN
EXECUTE 'CREATE TABLE ' || quote_ident($1) || '(item_1 int, item_2 int)';
END
$func$ LANGUAGE plpgsql;

```

6.1- Chame a função de novo:

```
SELECT unsafe_add_table('T1(id int); DELETE FROM testeSQLi; --');
```

7- Selecione os dados da tabela testeSQLi. O que aconteceu?

Dessa vez ao executar a função a tabela permanece inalterada.

8- Verifique as tabelas do seu BD! Tem alguma tabela nova?

Surgiu uma tabela nova.

9- Consegue explicar o que aconteceu? Justifique.

Na primeira função o que era pra ser o nome da tabela a ser criada foi escrito um código malicioso que apagou os dados da minha tabela existente, mas na segunda versão da função o código malicioso foi encarado como uma string e virou o nome da tabela que a função criou, como devia ser.

10- Execute as chamadas abaixo, usando o código seguinte. O que acontece e por que?

```

CREATE OR REPLACE FUNCTION get_staff(_param text, _orderby text, _limit int)
RETURNS SETOF empregado AS $func$
BEGIN
RETURN QUERY EXECUTE '
SELECT *
FROM empregado
WHERE nome_empregado = $1

```

```
ORDER BY ' || quote_ident(_orderby) || ' ASC
```

```
LIMIT $2'
```

```
USING _param, _limit;
```

```
END
```

```
$func$ LANGUAGE plpgsql;
```

```
--Call:
```

```
SELECT * FROM get_staff('Pedro; DELETE FROM testeSQLi; --', 'salario', 100);
```

```
SELECT * FROM get_staff('Pedro', 'salario', 100);
```

R: Quando o código malicioso é chamado nada acontece, pois a função está protegida e quando a segunda chamada acontece é mostrado o resultado correto do select.