In this lesson we are looking at the DELETE command, which allows us to delete rows from our tables. The DELETE command is dangerous. It allows us to delete everything from a table unless we specify specific rows to delete.

If we do not include a clause after writing the table name in our query with the DELETE command, then all data in the table will be deleted.

Let’s check it on our items’ table.

*DELETE FROM items;*

|  |  |  |
| --- | --- | --- |
| name  character varying (100) | id  integer | price  numeric |

As we can see that as we did not include a WHERE clause or any other clause in our query all the data from our items’ table got deleted.

Now let’s try another query with DELETE command and this time we will include the WHERE clause to delete one specific item from the table instead of the whole table.

*DELETE FROM items WHERE id=4;*

*ERROR: update or delete on table “items” violates foreign key constraint “fk\_purchase\_item” on table “purchases”*

*DETAIL: Key (id)=(4) is still referenced from table “purchases”.*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*ERROR\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*ERROR: update or delete on table “items” violates foreign key constraint “fk\_purchase\_item” on table “purchases”*

*SQL state: 23503*

*DETAIL: Key (id)=(4) is still referenced from table “purchases”.*

Postgres tells us that our query is not successful because we are deleting an item from our items’ table that has reference in another table called purchases table. Postgres knows that we are deleting an item that has got reference at another table, so it does not let us delete that item in case we are doing it by accident.

To overcome this error, we need to go to the purchases’ table and delete the item that is using the item with id = 4 in the items’ table. Then we can delete the item with id = 4 in items’ table.

This relationship between two tables is what makes Postgres so fantastic, it means that a user has to go through some extra step in case of executing a DELETE command.

Let’s display our purchases’ table to see where is the item, that we are trying to DELETE is being used.

*SELECT \* FROM purchases;*

|  |  |  |
| --- | --- | --- |
| id  integer | item\_id  integer | customer\_id  integer |
| 1 | 4 | 1 |
| 2 | 5 | 1 |
| 3 | 6 | 1 |
| 4 | 1 | 3 |
| 5 | 3 | 5 |
| 6 | 2 | 5 |
| 7 | 4 | 2 |
| 8 | 2 | 4 |
| 9 | 3 | 4 |
| 10 | 6 | 5 |

As we can see that the purchase with id 7 is using item with id 4 in its purchase and deleting this item from the items’ table will contradict with this transaction. That is why we need to delete this purchase transaction in order to delete the item with id 4 in our items’ table.

*DELETE FROM purchases WHERE item\_id=4;*

*Query returned successfully: 2 rows affected, 18 msec execution time.*

Postgres tells us that our query got executed successfully, and now we can go to our items table to view our items.

*SELECT \* FROM items;*

|  |  |  |
| --- | --- | --- |
| name  character varying (255) | id  integer | price  numeric (10,2) |
| Fountain Pen | 2 | 11.30 |
| Laptop | 4 | 899.00 |
| Screen | 5 | 275.50 |
| Hard Drive | 6 | 89.99 |
| Pen | 1 | 5.00 |
| Ink | 3 | 5.00 |

We can see the item with id 4 with the name “Laptop”.

*DELETE FROM items WHERE id=4;*

*Query returned successfully: one row affected, 167 msec execution time.*

Postgres tells us that our query got successfully executed. Now we will check our items’ table.

*SELECT \* FROM items;*

|  |  |  |
| --- | --- | --- |
| name  character varying (255) | id  integer | price  numeric (10,2) |
| Fountain Pen | 2 | 11.30 |
| Screen | 5 | 275.50 |
| Hard Drive | 6 | 89.99 |
| Pen | 1 | 5.00 |
| Ink | 3 | 5.00 |