INSERT INTO to Add Multiple Rows

by Sophia

WHAT'S COVERED

In this lesson, you will use the INSERT INTO command to add multiple rows into a table. Specifically, this lesson will cover:

1. Inserting Multiple Rows at Once

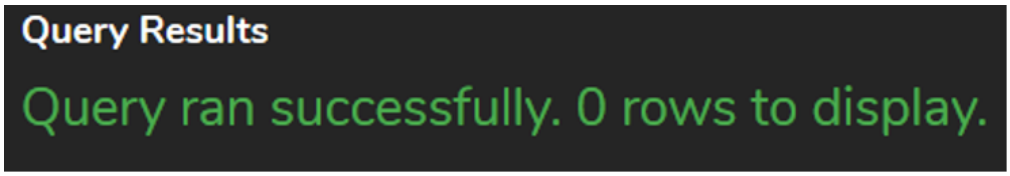
As you learned in the previous lesson, the INSERT INTO statement enables you to create new records in a table. Although so far, we have been inserting individual records, in real-life usage you will more often be inserting many new records at a time—sometimes even hundreds or thousands at a time. Most database systems, including PostgreSQL, enable you to combine multiple value lists in a single INSERT INTO statement. The syntax will look like the following:

```
INSERT INTO <tablename> ( <column1>, <column2>, ...)
VALUES (<value1>,<value2>, ...),
(<value1>,<value2>, ...),
(<value1>,<value2>, ...),
(<value1>,<value2>, ...);
```

It is quite similar to a regular INSERT INTO statement, but instead of having just one set of values, there is a list of values. Each new record's values are enclosed in parentheses, with the columns separated by commas. This statement will create a new table called referral to store individuals' first name, last name, and email.

Let's create a new table to test this on:

```
CREATE TABLE referral( referral_id SERIAL, first_name VARCHAR(50), last_name VARCHAR(50), email VARCHAR(50) );
```



A standard INSERT INTO statement with one record would look like the following:

```
INSERT INTO referral (first_name,last_name,email)
VALUES ('Sandra','Boynton','s.boy@email.com');
```

If we wanted to insert multiple rows at once, the INSERT INTO statement would look like the following:

```
INSERT INTO referral (first_name,last_name,email)
VALUES ('Randall','Faustino','s.boy@email.com'),
('Park','Deanna','p.deanna@email.com'),
('Sunil','Carrie','s.carrie@email.com'),
('Jon','Brianna','j.brianna@email.com'),
('Lana','Jakoba','l.jakoba@email.com'),
('Tiffany','Walker','t.walk@email.com');
```

Note, though, what happens if we forgot to include commas between each value list, like this:

```
INSERT INTO referral (first_name,last_name,email)
VALUES ('Randall','Faustino','s.boy@email.com')
('Park','Deanna','p.deanna@email.com')
('Sunil','Carrie','s.carrie@email.com')
('Jon','Brianna','j.brianna@email.com')
('Lana','Jakoba','l.jakoba@email.com')
('Tiffany','Walker','t.walk@email.com');
```

We would get this error:

Query Results

Query failed because of: error: syntax error at or near "("

Once inserted correctly, we should be able to take a look at what is in the table:

```
SELECT *
FROM referral;
```

Query Results

Row count: 7

referral_id	first_name	last_name	email
1	Sandra	Boynton	s.boy@email.com
2	Randall	Faustino	s.boy@email.com
3	Park	Deanna	p.deanna@email.com
4	Sunil	Carrie	s.carrie@email.com
5	Jon	Brianna	j.brianna@email.com
6	Lana	Jakoba	l.jakoba@email.com
7	Tiffany	Walker	t.walk@email.com

Notice that we have the referral_id incrementing automatically. One of the biggest issues in data accuracy is that mistakes are always made, even when it is automated. You might have seen that there is an error in Randall Faustino's email. You will learn how to make data corrections later in the course.

---

2. RETURNING Selected Columns

Let's drop and recreate the table to look at another example. Sometimes when you are testing and using test data, dropping a table and recreating it is easier and faster than trying to clear all the data from it. Run the following statement to drop the table:

```
DROP TABLE referral;
```

Then we want to recreate that table from scratch.

```
CREATE TABLE referral( referral_id SERIAL, first_name VARCHAR(50), last_name VARCHAR(50), email VARCHAR(50) );
```

As you learned in the previous lesson, adding RETURNING * at the end of the INSERT INTO statement can be useful because it will query the data you just entered and present it to the screen, so you can see if it was entered correctly. This is great when you are working with small datasets, but not so great when working with large datasets, or large amounts of data being entered. It is a very useful command to know when debugging or testing new database programs.

```
INSERT INTO referral (first_name,last_name,email)
VALUES ('Randall','Faustino','s.boy@email.com'),
('Park','Deanna','p.deanna@email.com'),
('Sunil','Carrie','s.carrie@email.com'),
('Jon','Brianna','j.brianna@email.com'),
('Lana','Jakoba','l.jakoba@email.com'),
('Tiffany','Walker','t.walk@email.com')
RETURNING *;
```

Query Results

Row count: 6

referral_id	first_name	last_name	email
1	Randall	Faustino	s.boy@email.com
2	Park	Deanna	p.deanna@email.com
3	Sunil	Carrie	s.carrie@email.com
4	Jon	Brianna	j.brianna@email.com
5	Lana	Jakoba	l.jakoba@email.com
6	Tiffany	Walker	t.walk@email.com

Instead of having RETURNING *, we can specify the return columns. For example, we could use the referral_id, as it is being auto-generated:

```
INSERT INTO referral (first_name,last_name,email)
VALUES ('Randall','Faustino','s.boy@email.com'),
('Park','Deanna','p.deanna@email.com'),
('Sunil','Carrie','s.carrie@email.com'),
```

```
('Jon','Brianna','j.brianna@email.com'),
('Lana','Jakoba','l.jakoba@email.com'),
('Tiffany','Walker','t.walk@email.com')
RETURNING referral_id;
```

WATCH

TRY IT

Your turn! Open the SQL tool by clicking on the LAUNCH DATABASE button below. Then, enter in one of the examples above and see how it works. Next, try your own choices for which columns you want the query to provide.

SUMMARY

In this lesson, you learned how to use the INSERT INTO statement to **insert multiple new records at once**. You also learned to use the **RETURNING** clause to display the data you just entered onscreen immediately after adding it. RETURNING * returns all columns, but you can alternatively request to see only the columns you specify.

Source: THIS TUTORIAL WAS AUTHORED BY DR. VINCENT TRAN, PHD (2020) AND Faithe Wempen (2024) FOR SOPHIA LEARNING. PLEASE SEE OUR TERMS OF USE.