INSERT to Add Queried Data

by Sophia Tutorial

WHAT'S COVERED

This tutorial explores using the SELECT statement with the INSERT INTO statement to add data from other tables in two parts:

1. INSERT with SELECT
2. Summary Data

1. INSERT with SELECT

The INSERT feature using a SELECT clause can come in handy when we need to load data into a table from another table. It can be used to load data into an existing table, a newly created table, or into a temporary table that includes some calculations. Anything that can be queried can be subsequently added to an INSERT statement to be inserted into a table.

The structure of the statement would look like this:

```
INSERT INTO <tablename> ( <column1>, <column2>, …)
<SELECT …>;
```

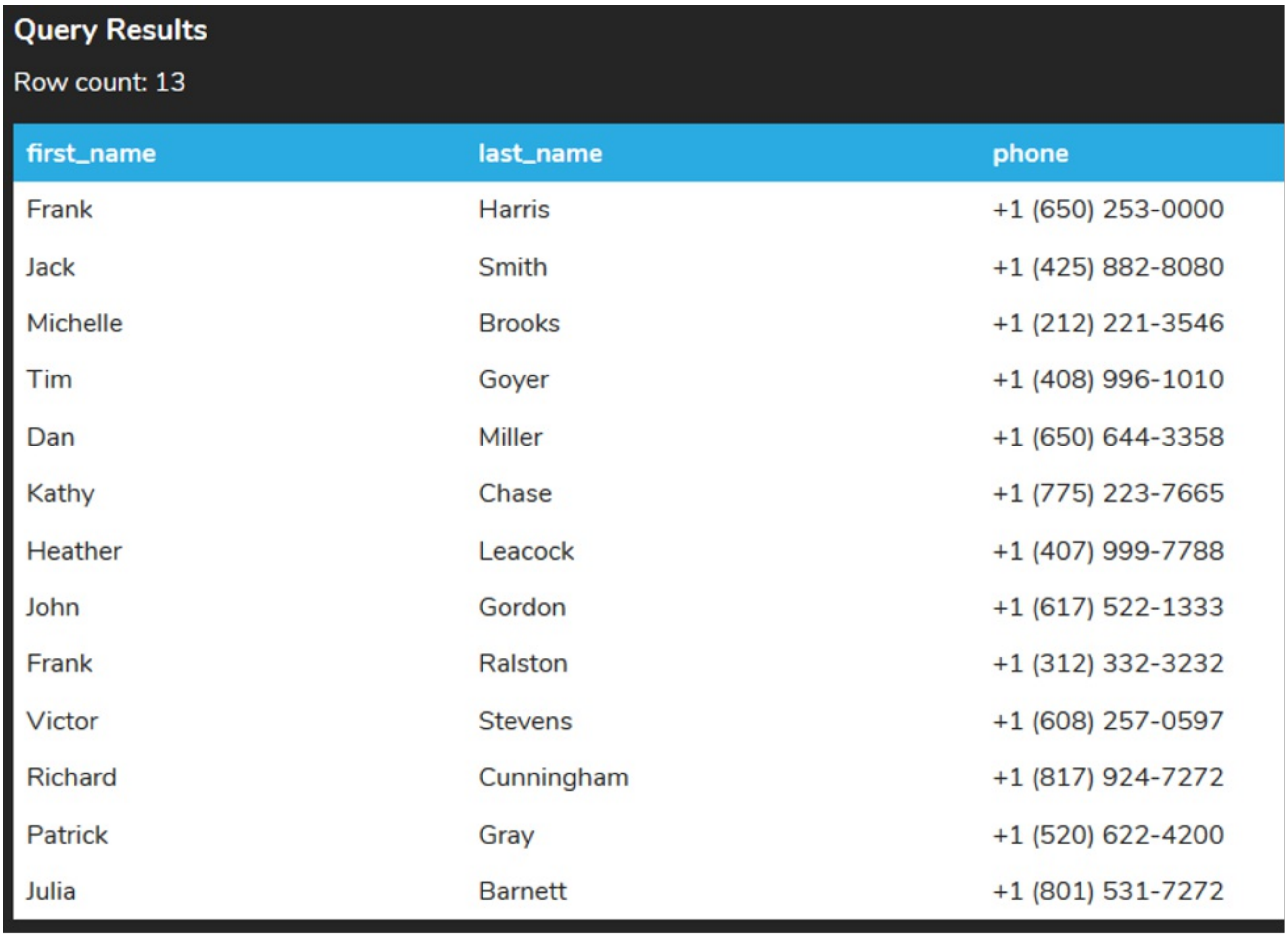
Similar to the INSERT INTO command from the prior tutorials, we want to include the columns and the order in which the query should return their values. In this structure, though, we replace the value list with a complete SELECT statement.

Let's say that we've created a contact table and would like to add in the customer's first name, last name, and phone number. We would like to have only the customers that live in the country USA, as the table will be used for customer service representatives to make the calls. The first step is to create the table that we'll be using. We will want to ensure that the data types are the same, and the sizes will be the same or larger to ensure we do not have any issues with truncated data.

```
CREATE TABLE contact( contact_id SERIAL, first_name VARCHAR(40), last_name VARCHAR(40), phone VARCHAR(24) );
```

The next step is to create and validate the SELECT statement, which will look like this:

```
SELECT first_name, last_name, phone
FROM customer
WHERE country = 'USA';
```


Query Results

Row count: 13

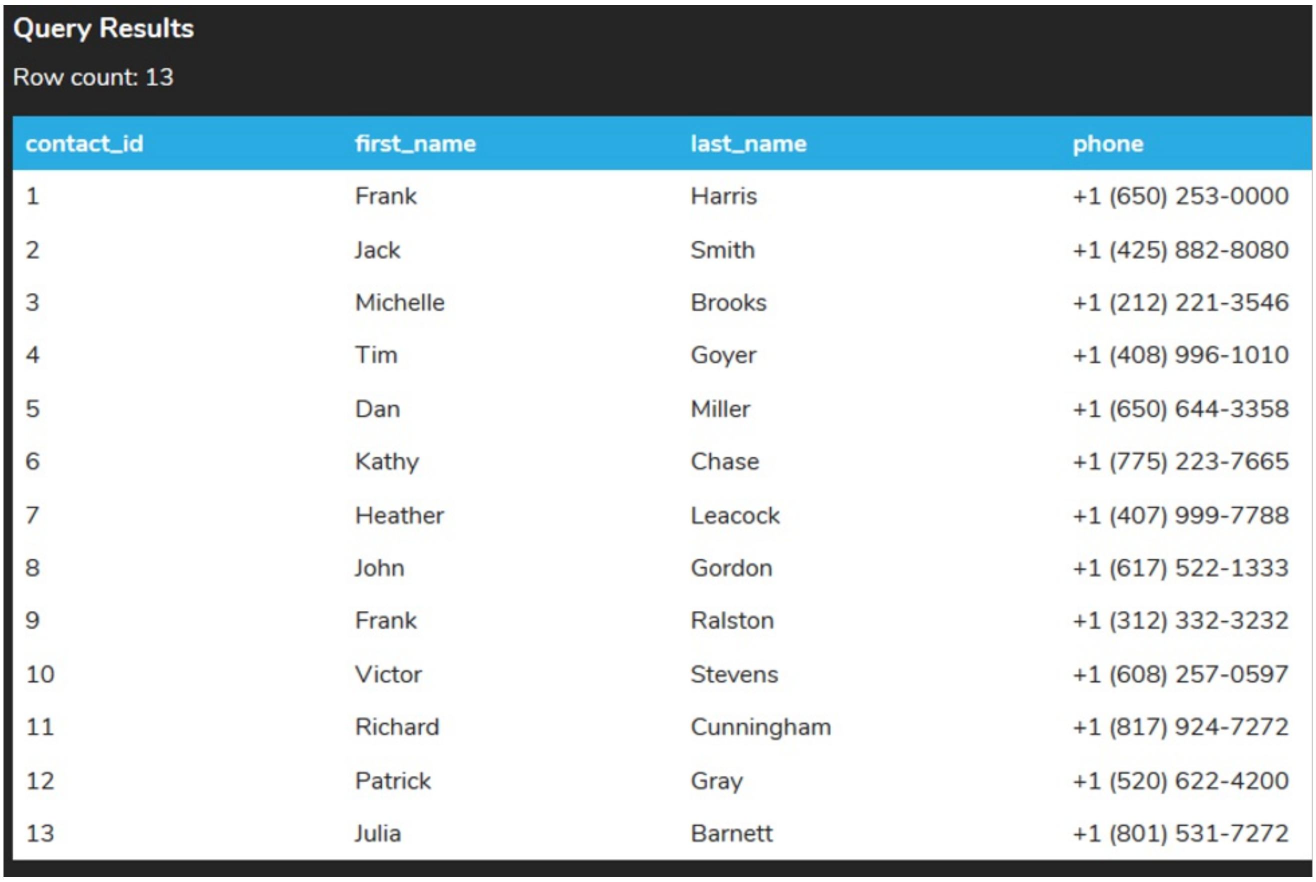
first_name	last_name	phone
Frank	Harris	+1 (650) 253-0000
Jack	Smith	+1 (425) 882-8080
Michelle	Brooks	+1 (212) 221-3546
Tim	Goyer	+1 (408) 996-1010
Dan	Miller	+1 (650) 644-3358
Kathy	Chase	+1 (775) 223-7665
Heather	Leacock	+1 (407) 999-7788
John	Gordon	+1 (617) 522-1333
Frank	Ralston	+1 (312) 332-3232
Victor	Stevens	+1 (608) 257-0597
Richard	Cunningham	+1 (817) 924-7272
Patrick	Gray	+1 (520) 622-4200
Julia	Barnett	+1 (801) 531-7272

We can now wrap it into the INSERT statement as follows:

```
INSERT INTO contact (first_name, last_name, phone)
SELECT first_name, last_name, phone
FROM customer
WHERE country = 'USA';
```

Note that the first_name, last_name, and phone in the first line represent the columns from the contact table rather than from the customer table, although they are named the same. Now that this is complete, we can run a query on the contact table to verify that the rows have been inserted:

```
SELECT *
FROM contact;
```


Query Results

Row count: 13

contact_id	first_name	last_name	phone
1	Frank	Harris	+1 (650) 253-0000
2	Jack	Smith	+1 (425) 882-8080
3	Michelle	Brooks	+1 (212) 221-3546
4	Tim	Goyer	+1 (408) 996-1010
5	Dan	Miller	+1 (650) 644-3358
6	Kathy	Chase	+1 (775) 223-7665
7	Heather	Leacock	+1 (407) 999-7788
8	John	Gordon	+1 (617) 522-1333
9	Frank	Ralston	+1 (312) 332-3232
10	Victor	Stevens	+1 (608) 257-0597
11	Richard	Cunningham	+1 (817) 924-7272
12	Patrick	Gray	+1 (520) 622-4200
13	Julia	Barnett	+1 (801) 531-7272

2. Summary Data

There are many instances when this can be useful. For example, if we're loading data from another table/database/file but we need to have the data formatted in a specific way, rather than ALTER an existing table's properties, we can build the table as we would want it and then copy the data in. We can also use this to build a summary table to get point-in-time data recorded.

Let's take a look at an example of that. We may want to have a summary table that has the invoice total and a number of invoices up to the current date with the added date inserted. Since this is point-in-time data, it can be harder to capture those changes without a more complex set of criteria.

```
CREATE TABLE invoice_summary( summary_date date, all_total numeric, num_of_invoice integer );
```

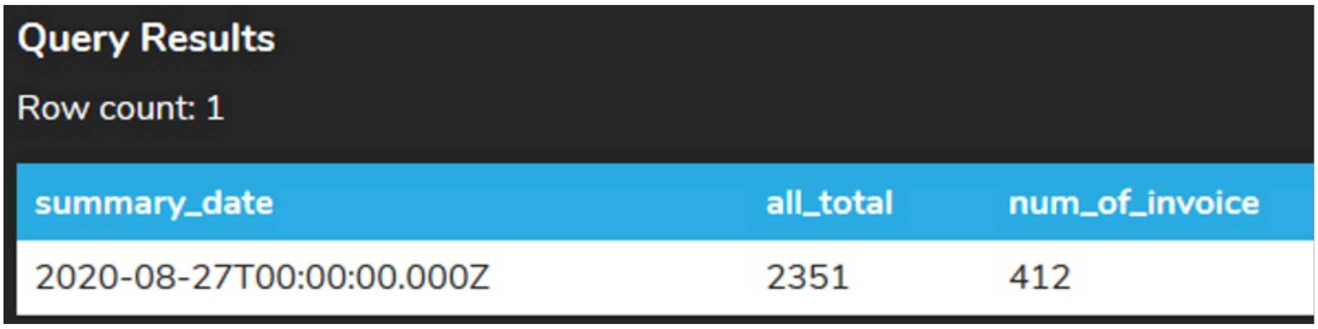
With the table created, we can now build the SELECT statement to generate the point-in-time data that includes the date:

```
SELECT now(), SUM(total), COUNT(invoice_id)
FROM invoice;
```

As a reminder, the now() function returns the current date/time. Putting this together with the SELECT statement, it would look like this:

```
INSERT INTO invoice_summary(summary_date,all_total,num_of_invoice)
SELECT now(), SUM(total), COUNT(invoice_id)
FROM invoice;
```

We can now query the invoice_summary table to see the new record that was added:



Query Results

Row count: 1

summary_date	all_total	num_of_invoice
2020-08-27T00:00:00.000Z	2351	412

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

We can use a SELECT statement to INSERT data into a table using data from another table.

Source: Authored by Vincent Tran