UPDATE to Edit Multiple Rows

by Sophia

WHAT'S COVERED

In this lesson, you will use the UPDATE statement to bulk update a number of rows in one statement rather than individually. Specifically, this lesson will cover:

1. Using UPDATE to Edit Multiple Rows
2. Using RETURNING

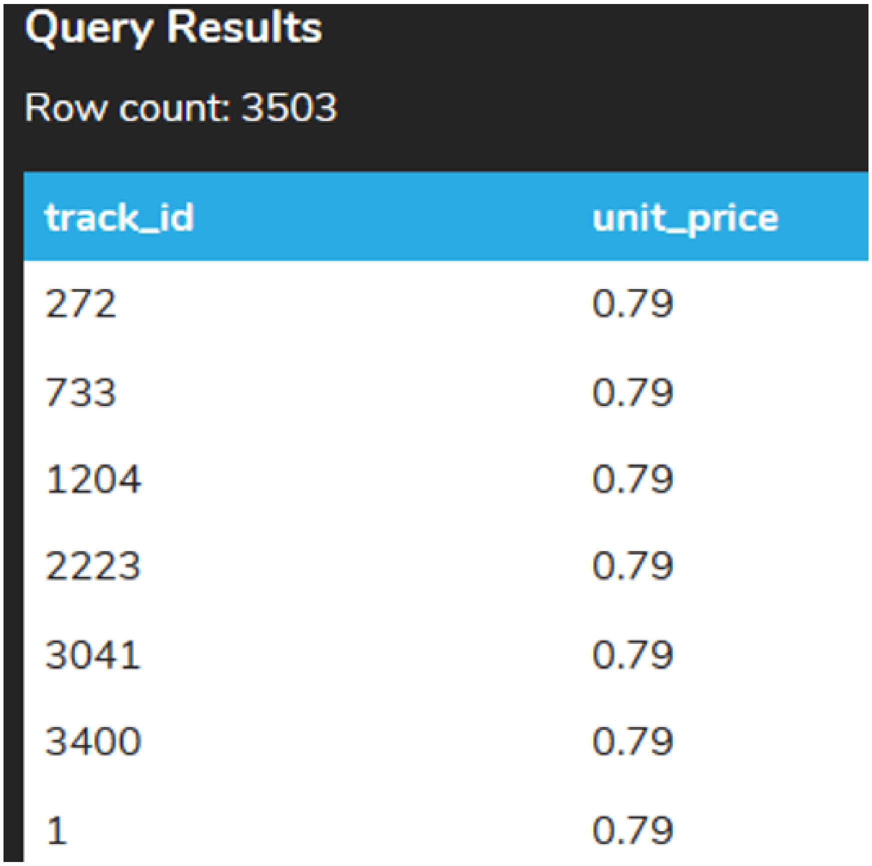
1. Using UPDATE to Edit Multiple Rows

There are times when we may want to update the entire table. For example, if we may have a permanent sale on the tracks to give a 20% discount, we can update the entire table like this:

```
UPDATE track
SET unit_price = unit_price * .8;
```

The above example multiplies the existing value of unit_price by .8 (or discounts it by 20%) and then sets that value to be the new unit price. Since we do not have a WHERE clause, this will update every single row.

We will be able to see that the unit_price has been decreased from $0.99 for most tracks to $0.79:



Query Results

Row count: 3503

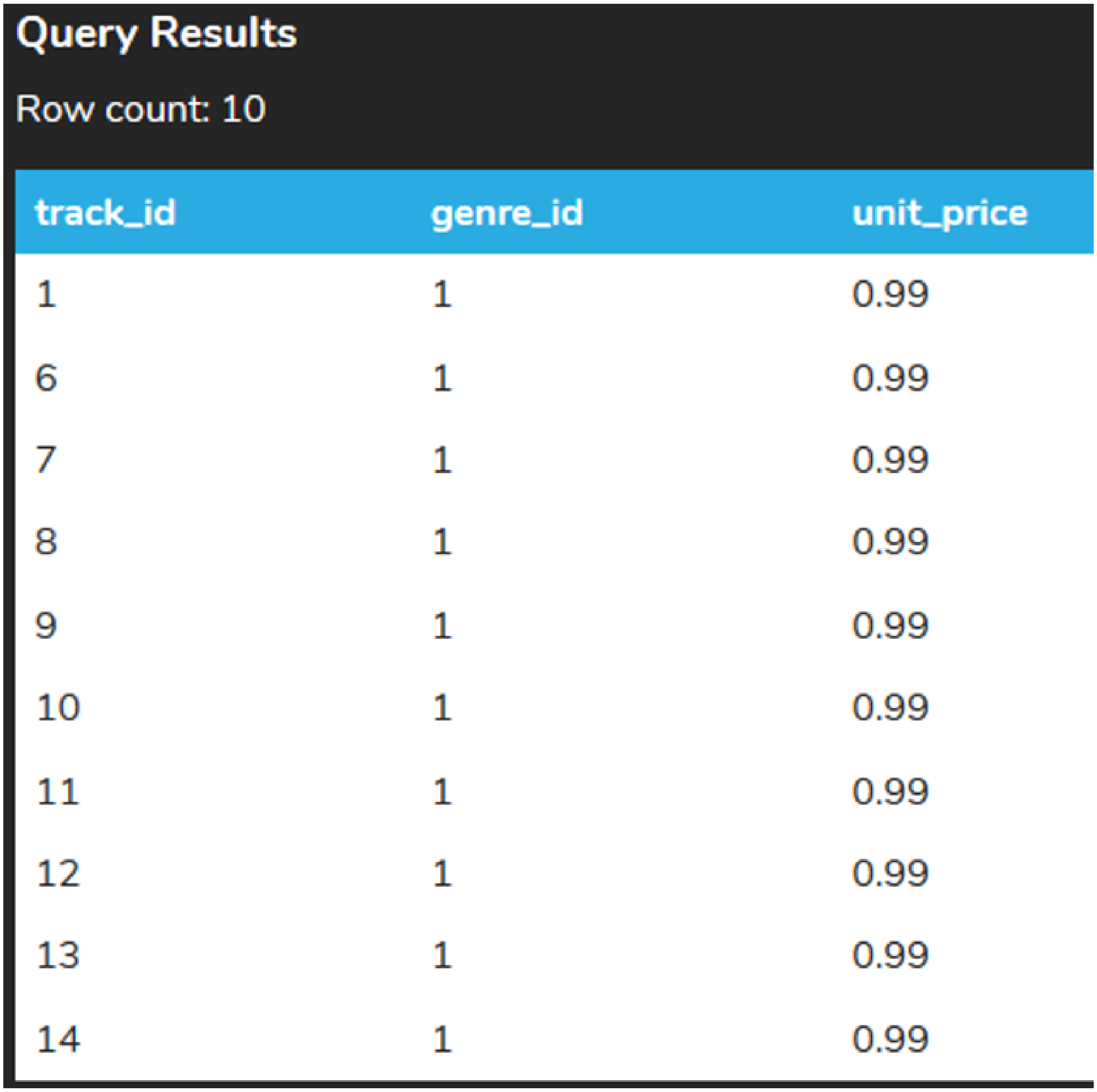
track_id	unit_price
272	0.79
733	0.79
1204	0.79
2223	0.79
3041	0.79
3400	0.79
1	0.79

If you are going to update only certain records based on some criteria, it can be helpful to first run a SELECT statement with the WHERE clause to verify what rows will be included in the operation.

For example, we can give a discount on the tracks for a specific album. We can first start by selecting it:

```
SELECT track_id, genre_id, unit_price
FROM track
WHERE album_id = 1;
```

This returns us with 10 rows:



Query Results

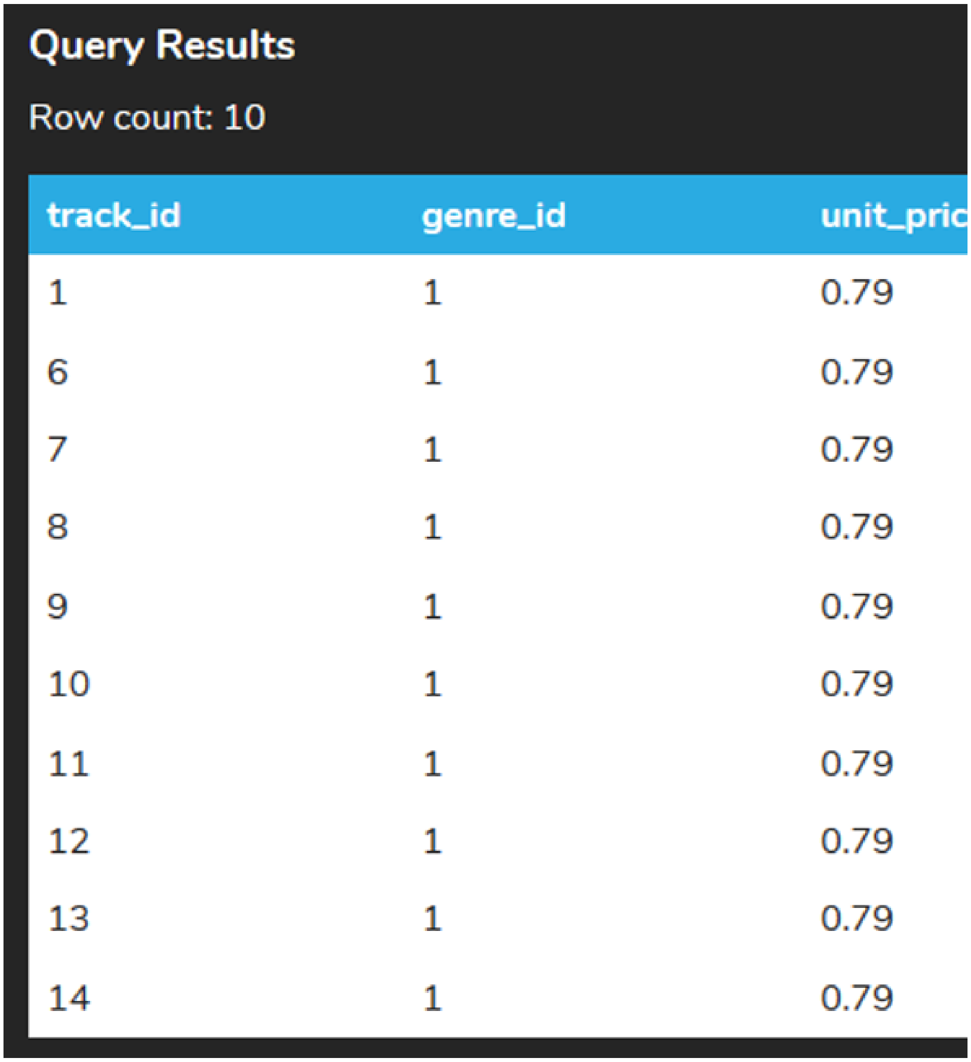
Row count: 10

track_id	genre_id	unit_price
1	1	0.99
6	1	0.99
7	1	0.99
8	1	0.99
9	1	0.99
10	1	0.99
11	1	0.99
12	1	0.99
13	1	0.99
14	1	0.99

Using the same WHERE clause, we can then apply the update:

```
UPDATE track
SET unit_price = unit_price * .8
WHERE album_id = 1;
```

If we now query the table, we should see the unit_price being updated.



Query Results

Row count: 10

track_id	genre_id	unit_pric
1	1	0.79
6	1	0.79
7	1	0.79
8	1	0.79
9	1	0.79
10	1	0.79
11	1	0.79
12	1	0.79
13	1	0.79
14	1	0.79

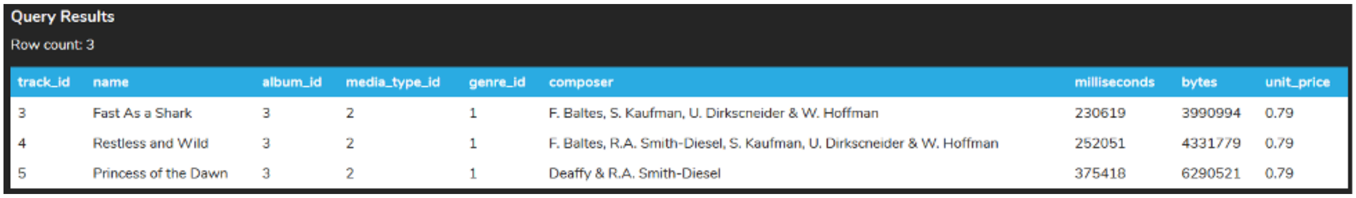
2. Using RETURNING

In the previous lesson, you learned about using RETURNING to see the results of your update. You learned that using RETURNING by itself returns only the values of the affected columns in the updated rows.

In contrast, RETURNING * means you want the values of all columns of the affected rows.

In the following example, we are applying the same 20% discount to tracks with an album_id of 3. This time, however, we are using RETURNING *, and the query results therefore show all columns from the affected rows.

```
UPDATE track
SET unit_price = unit_price * .8
WHERE album_id = 3
RETURNING *;
```



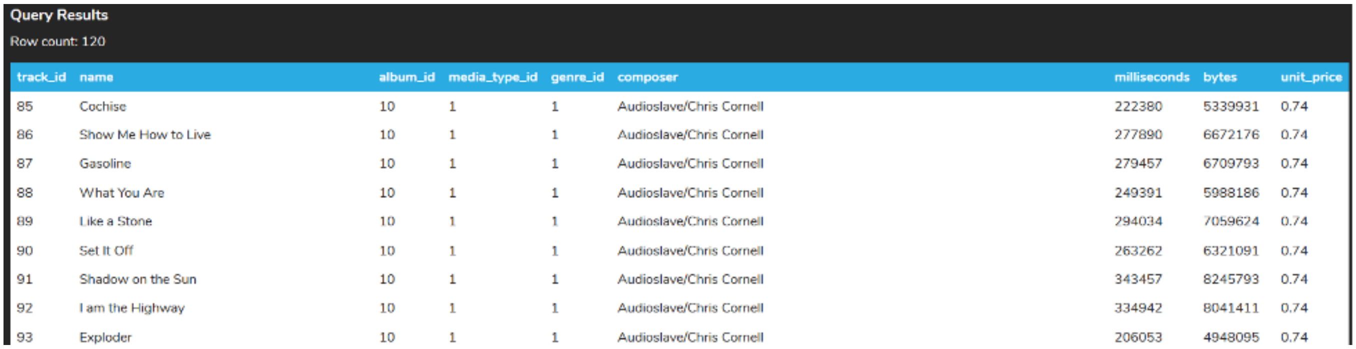
Query Results

Row count: 3

track_id	name	album_id	media_type_id	genre_id	composer	milliseconds	bytes	unit_price
3	Fast As a Shark	3	2	1	F. Baltes, S. Kaufman, U. Dirkscneider & W. Hoffman	230619	3990994	0.79
4	Restless and Wild	3	2	1	F. Baltes, R.A. Smith-Diesel, S. Kaufman, U. Dirkscneider & W. Hoffman	252051	4331779	0.79
5	Princess of the Dawn	3	2	1	Deaffy & R.A. Smith-Diesel	375418	6290521	0.79

We can also use ranges, such as applying a 25% discount to those tracks with album_id values between 10–20:

```
UPDATE track
SET unit_price = unit_price * .75
WHERE album_id BETWEEN 10 AND 20
RETURNING *;
```



Query Results

Row count: 120

track_id	name	album_id	media_type_id	genre_id	composer	milliseconds	bytes	unit_price
85	Cochise	10	1	1	Audioslave/Chris Cornell	222380	5339931	0.74
86	Show Me How to Live	10	1	1	Audioslave/Chris Cornell	277890	6672176	0.74
87	Gasoline	10	1	1	Audioslave/Chris Cornell	279457	6709793	0.74
88	What You Are	10	1	1	Audioslave/Chris Cornell	249391	5988186	0.74
89	Like a Stone	10	1	1	Audioslave/Chris Cornell	294034	7059624	0.74
90	Set It Off	10	1	1	Audioslave/Chris Cornell	263262	6321091	0.74
91	Shadow on the Sun	10	1	1	Audioslave/Chris Cornell	343457	8245793	0.74
92	I am the Highway	10	1	1	Audioslave/Chris Cornell	334942	8041411	0.74
93	Exploder	10	1	1	Audioslave/Chris Cornell	206053	4948095	0.74

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 UPDATE** without a WHERE clause to update every record in a table, and to use math operators to change values. You also learned the difference between **using the RETURNING clause** by itself and with an asterisk. By itself, it returns only the affected rows and columns. With the asterisk, it returns all the columns for the affected rows.

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