VIEW to Simplify Queries

by Sophia

WHAT'S COVERED

This lesson explores using CREATE VIEW to join multiple table data to simplify queries, in two parts. Specifically, this lesson will cover:

1. Using CREATE VIEW to Combine Tables
2. Creating a View With More Than Two Tables

1. Using CREATE VIEW to Combine Tables

One helpful use for CREATE VIEW is to combine data from multiple tables. This enables us to make data available in a single result set that would normally not be located in the same place. For example, seeing the support_rep_id may not be extremely useful in an organization unless you know who that value belongs to. Instead, you could include the name of the support rep, similar to the following:

```
CREATE VIEW customer_contact
AS
SELECT customer.*, employee.first_name as support_first_name, employee.last_name as support_last_name
FROM customer, employee
WHERE customer.support_rep_id = employee.employee_id;
```

If we queried the customer_contact view, it would look like the following:

```
SELECT *
FROM customer_contact;
```


Query Results
Row count: 59

customer_id	first_name	last_name	company	address	city	state	country	postal_code	phone	fax	email	support_rep_id	support_first_name	support_last_name
1	Luís	Gonçalves	Embraer - Empresa Brasileira de Aeronáutica S.A.	Av. Brigadeiro Faria Lima, 2170	São José dos Campos	SP	Brazil	12227-000	+55 (12) 3923-5555	+55 (12) 3923-5566	luisg@embraer.com.br	3	Jane	Peacock
2	Leonie	Köhler		Theodor-Heuss-Straße 34	Stuttgart		Germany	70174	+49 0711 2842222		leonekohler@surfeu.de	5	Steve	Johnson
3	François	Tremblay		1498 rue Bélanger	Montréal	QC	Canada	H2G 1A7	+1 (514) 721-4711		ftremblay@gmail.com	3	Jane	Peacock

We could also specify the column names desired rather than displaying all columns:

```
SELECT first_name, last_name, support_first_name, support_last_name
FROM customer_contact;
```

Query Results

Row count: 59

first_name	last_name	support_first_name	support_last_name
Luís	Gonçalves	Jane	Peacock
Leonie	Köhler	Steve	Johnson
François	Tremblay	Jane	Peacock

Using CREATE VIEW can be a great time-saver because it helps you avoid having to re-type a query each time you want to run it. If not for VIEW, we would have to type and run the following longish query every time we wanted its information:

```
SELECT customer.first_name, customer.last_name, employee.first_name as support_first_name, employee.last_name as support_last_name
```

```
FROM customer, employee
WHERE customer.support_rep_id = employee.employee_id;
```

2. Creating a View With More Than Two Tables

In most of the query examples in this course so far that have involved multiple tables, we have only included the primary and foreign key columns. For example, when we query the track table, we have been focused on track_id. However, sometimes we might also want to look at the artist's name, album title, and track name, all at the same time. Creating a view for this purpose can simplify that process:

```
CREATE VIEW artist_album_track
AS
SELECT artist.name as artist_name, album.title as album_title, track.name as track_name
FROM artist
INNER JOIN album ON artist.artist_id = album.artist_id
INNER JOIN track ON album.album_id = track.album_id;
```

Then, rather than querying the tables each time we want that list, as shown below:

```
SELECT artist.name as artist_name, album.title as album_title, track.name as track_name
FROM artist
INNER JOIN album ON artist.artist_id = album.artist_id
INNER JOIN track ON album.album_id = track.album_id;
```

We can simply query the view directly, like this:

```
SELECT *
FROM artist_album_track;
```

Query Results

Row count: 3503

artist_name	album_title	track_name
AC/DC	For Those About To Rock We Salute You	For Those About To Rock (We Salute You)
Accept	Balls to the Wall	Balls to the Wall
Accept	Restless and Wild	Fast As a Shark
Accept	Restless and Wild	Restless and Wild
Accept	Restless and Wild	Princess of the Dawn

Consider if we wanted to add some filters into our SELECT statement, such as only listing the rows that belong to AC/DC. Instead of doing this:

```
SELECT artist.name as artist_name, album.title as album_title, track.name as track_name
FROM artist
INNER JOIN album ON artist.artist_id = album.artist_id
INNER JOIN track ON album.album_id = track.album_id
WHERE artist.name = 'AC/DC';
```

We would query the view like this:

```
SELECT *
FROM artist_album_track
WHERE artist_name = 'AC/DC';
```

Query Results

Row count: 18

artist_name	album_title	track_name
AC/DC	For Those About To Rock We Salute You	For Those About To Rock (We Salute You)
AC/DC	For Those About To Rock We Salute You	Put The Finger On You
AC/DC	For Those About To Rock We Salute You	Let's Get It Up
AC/DC	For Those About To Rock We Salute You	Inject The Venom

The second option greatly simplifies the query without having to join each of the tables together.

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 combine data from multiple tables using CREATE VIEW**. The view can then be used for other queries, making the complex data relationships easier for them to understand. Depending on the desired relationship between the tables, the query may involve various types of joins and **creating a view with more than two (i.e., any number) of tables**.

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