JOIN ON to Link Tables

by Sophia Tutorial

WHAT'S COVERED

This tutorial explores using the JOIN ON to link table data across two tables in two parts:

1. JOIN ON
2. USING vs ON

1. JOIN ON

In the prior tutorial, we joined table data together with the USING clause. This works great for situations where the column names are the same between two tables. However, this is not always the case. For example, if we look at the customer and employee tables, the customer table has a support_rep_id which references the employee_id of the employee table:

customer	
company	VARCHAR (80)
address	VARCHAR (70)
city	VARCHAR (40)
state	VARCHAR (40)
country	VARCHAR (40)
postal_code	VARCHAR (10)
phone	VARCHAR (24)
fax	VARCHAR (24)
email	VARCHAR (60)
support_rep_id	INTEGER
first_name	VARCHAR (40)
customer_id	INTEGER
last_name	VARCHAR (40)

employee	
postal_code	VARCHAR (10)
employee_id	INTEGER
last_name	VARCHAR (20)
first_name	VARCHAR (20)
title	VARCHAR (30)
reports_to	INTEGER
birth_date	TIMESTAMP
hire_date	TIMESTAMP
address	VARCHAR (70)
city	VARCHAR (40)
state	VARCHAR (40)
country	VARCHAR (40)
phone	VARCHAR (24)
fax	VARCHAR (24)
email	VARCHAR (60)

This makes sense, because a support representative would be an employee, but it wouldn't make contextual sense to have employee_id as the name of the column in the customer table. Since the column names are different, we could not use the USING clause. Rather, we must use the ON clause.

The structure of the statement looks like the following:

```
SELECT <columnlist>
FROM <table1>
INNER JOIN <table2> ON <table1column> = <table2column>;
```

We can start the query by filling in the table and column names that we're joining:

```
SELECT <columnlist>
FROM customer
INNER JOIN employee ON support_rep_id = employee_id;
```

We then need to determine what columns we want to return as part of the query. We don't want to return all of the columns. For example, we may want to have all of the customer emails and the respective employee emails so that the employees can send out an email to their customers. Since the email addresses are in an email column in both tables, we need to prefix the column with the table name, like this:

```
SELECT customer.email, employee.email
FROM customer
INNER JOIN employee ON support_rep_id = employee_id;
```

Query Results

Row count: 59

email	email
luisg@embraer.com.br	jane@chinookcorp.com
leonekohler@surfeu.de	steve@chinookcorp.com
ftremblay@gmail.com	jane@chinookcorp.com
bjorn.hansen@yahoo.no	margaret@chinookcorp.com
frantisekw@jetbrains.com	margaret@chinookcorp.com
hholy@gmail.com	steve@chinookcorp.com
astrid.gruber@apple.at	steve@chinookcorp.com
daan_peeters@apple.be	margaret@chinookcorp.com

2. USING vs ON

Although we would use this method when the column names between the tables do not match, it does also work if the columns match (and therefore we could use the USING clause). Let's take a look at an example of the artist and album table with an inner join with the USING clause:

```
SELECT title, name
FROM album
INNER JOIN artist USING (artist_id);
```

We could convert this by removing the USING clause and adding the artist_id prefixed with the table name:

```
SELECT title, name
FROM album
INNER JOIN artist ON album.artist_id = artist.artist_id;
```

In both cases, the result set is the same:

Query Results

Row count: 347

title	name
For Those About To Rock We Salute You	AC/DC
Balls to the Wall	Accept
Restless and Wild	Accept
Let There Be Rock	AC/DC
Big Ones	Aerosmith
Jagged Little Pill	Alanis Morissette
Facelift	Alice In Chains
Warner 25 Anos	Antônio Carlos Jobim

The only case where the result set will be different is if we use the * in the SELECT clause:

SELECT *
FROM album
INNER JOIN artist USING (artist_id);

Query Results

Row count: 347

artist_id	album_id	title	name
1	1	For Those About To Rock We Salute You	AC/DC
2	2	Balls to the Wall	Accept
2	3	Restless and Wild	Accept
1	4	Let There Be Rock	AC/DC
3	5	Big Ones	Aerosmith
4	6	Jagged Little Pill	Alanis Morissette

SELECT *
FROM album
INNER JOIN artist ON album.artist_id = artist.artist_id;

Query Results

Row count: 347

album_id	title	artist_id	artist_id	name
1	For Those About To Rock We Salute You	1	1	AC/DC
2	Balls to the Wall	2	2	Accept
3	Restless and Wild	2	2	Accept
4	Let There Be Rock	1	1	AC/DC
5	Big Ones	3	3	Aerosmith
6	Jagged Little Pill	4	4	Alanis Morissette
7	Facelift	5	5	Alice In Chains

Notice that in the ON clause join, the artist_id appears twice (one for each table) whereas artist_id only appears once in the USING clause. This is because the USING performs an equality join and can only be used when the column names are identical. It is not necessary to include the column twice. Other than that, they function the same.

Video Transcription

[MUSIC PLAYING] Here we're going to explore the use of the inner join utilizing the on clause instead. So in this case here, we have two tables-- the representative table as well as department table. However, there's no consistent name and that's there between the two tables that we can utilize using command in this case here, being that here we have the representative ID. But in the current table, it's not called the representative ID. It's called the manager ID.

So when we have that kind of situation, we have to utilize the inner join utilizing on clause instead. So we can start building this out very similar to the other type of join that we have. We'll do a select star from the first table representative, join department, and then we're going to identify on instead of using. And then we're going to identify the representative table with the representative ID, which should be equal to department dot manager ID.

That will allow us to have the results based on that joint between the two tables when there's a differentiator in this case here, between the two different tables. Here we could also remove the table names being that this unique per table.

[MUSIC PLAYING]

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

The JOIN ON clause allows the joining of tables together on columns that do not have the same names.