Matching Different Data Tables with JOINs



Jon Flanders

@jonflanders www.jonflanders.com

The JOIN Clause

Merges multiple tables into one result set
FROM clause includes all tables
Separates each table with a comma
WHERE clause typically included
Expression with columns from each table
There are different types of JOINs

CROSS JOIN

Simplest JOIN

All rows from both tables

No WHERE clause

Least useful

Inefficient

Cartesian Product

CROSS keyword implied

SELECT p.first_name,
e.email_address
FROM
person p,
email_address e;



■ SELECT CLAUSE

▼ FROM CLAUSE WITHMULTIPLE TABLES



INNER JOIN

Most typical JOIN

Emphasizes relational nature of database

Matches column in first table to second

Primary key to foreign key is most common

SELECT p.first_name, p.last_name, e.email_address FROM person p INNER JOIN email_address e ON p.person_id = e.email_address_person_id;

◆INNER JOIN

◆ON CLAUSE

OUTER JOIN

INNER JOIN doesn't deal with NULL values
OUTER JOIN works even when no match
NULL columns if no match in second table
FULL OUTER JOIN returns all joined rows
NULL when no match in either table

LEFT OUTER JOIN

Another NULL-related JOIN

All rows from the left side will be returned

NULL for non-matching right side table

SELECT p.first_name, p.last_name, e.email_address FROM person p LEFT OUTER JOIN email_address e ON p.person_id = e.email_address_person_id;



▲LEFT OUTER JOIN

| first_name | last_name | email_address |
|------------|-----------|---------------|
| Jon | Flanders | jon@ |
| Fritz | Onion | fritz@ |
| Shannon | Ahern | NULL |

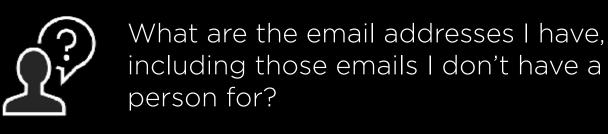
RIGHT OUTER JOIN

Opposite of LEFT OUTER JOIN

All rows from the right side will be returned

NULL for non-matching left side table

```
SELECT
p.first_name, p.last_name,
e.email_address
FROM person p
RIGHT OUTER JOIN
email_address e
ON
p.person_id =
e.email_address_person_id;
```





| first_name | last_name | email_address |
|------------|-----------|---------------|
| Jon | Flanders | jon@ |
| Fritz | Onion | fritz@ |
| NULL | NULL | aaron@ |

```
SELECT
p.first_name, p.last_name,
e.email_address
FROM person p
FULL OUTER JOIN
email_address e
ON p.person_id =
e.email_address_person_id;
```



What are all my contacts and their email addresses, including the ones missing an email address and the ones with an email address but missing a contact name?

▼ FULL OUTER JOIN

| first_name | last_name | email_address |
|------------|-----------|---------------|
| Jon | Flanders | jon@ |
| Fritz | Onion | fritz@ |
| Shannon | Ahern | NULL |
| NULL | NULL | aaron@ |

SELF JOIN

You can JOIN a table on itself

Odd but sometimes useful

No special syntax

Same table on left and right side of JOIN

Useful when table contains hierarchical data

Summary

JOINS make the relational model come to life by associating tables together