GENERAL ASSEMBLY

Join types

LEFT OUTER
RIGHT OUTER
LEFT EXCEPTION
RIGHT EXCEPTION
CROSS
COALESCE

Matthew Morris
Git: Morrisdata
MatthewMorris.DA@gmail.com

GENERAL ASSEMBLY

Previously in Data Analytics

JOIN 2 tables
JOIN multiple tables
UNION

Join Types

SELECT

FROM

JOIN

ON

WHERE

GROUP BY

HAVING

UNION

ORDER BY

LIMIT

Join Types

OUTER, EXCEPTION, AND CARTESIAN JOINS

Employees

id	first_name	last_name
2	Gabe	Moore
3	Doreen	Mandeville
5	Simone	MacDonald
7	Madisen	Flateman
11	Ian	Paasche
13	Mimi	St. Felix

Salaries

id	current_salary
2	50000
3	60000
7	55000
11	75000
13	120000
17	70000

Join Types

OUTER, EXCEPTION, AND CARTESIAN JOINS

- An INNER JOIN (also called a direct join) displays only the rows that have a match in both joined tables.

- An INNER JOIN would yield this table:

```
SELECT * FROM employees INNER JOIN
salaries ON employees.ID = salaries.ID;
```

id	first_name	last_name	id	current_salary
2	Gabe	Moore	2	50000
3	Doreen	Mandeville	3	60000
7	Madisen	Flateman	7	55000
11	Ian	Paasche	11	75000
13	Mimi	St. Felix	13	7000

Join Types

OUTER, EXCEPTION, AND CARTESIAN JOINS

A LEFT OUTER JOIN returns both:

- Data that both tables have in common.
- Data from the **primary** table selected, which does not have matching data to join to in the secondary table.

- A LEFT OUTER JOIN would yield this table:

SELECT * FROM employees LEFT OUTER JOIN salaries ON employees.ID = salaries.ID;

id	first_name	last_name	id	current_salary
2	Gabe	Moore	2	50000
3	Doreen	Mandeville	3	60000
5	Simone	MacDonald	NULL	NULL
7	Madisen	Flateman	7	55000
11	Ian	Paasche	11	75000
13	Mimi	St. Felix	13	120000

Join Types

OUTER, EXCEPTION, AND CARTESIAN JOINS

- A RIGHT OUTER JOIN returns both:
  - Data that two tables have in common.
  - Data from the **secondary** table selected, which does not have matching data to join to in the primary table.
- A RIGHT OUTER JOIN would yield this table:

```
SELECT * FROM employees RIGHT OUTER
JOIN salaries ON employees.ID = salaries.ID;
```

id	first_name	last_name	id	current_salary
2	Gabe	Moore	2	50000
3	Doreen	Mandeville	3	60000
7	Madisen	Flateman	7	55000
11	Ian	Paasche	11	75000
13	Mimi	St. Felix	13	120000
NULL	NULL	NULL	17	70000

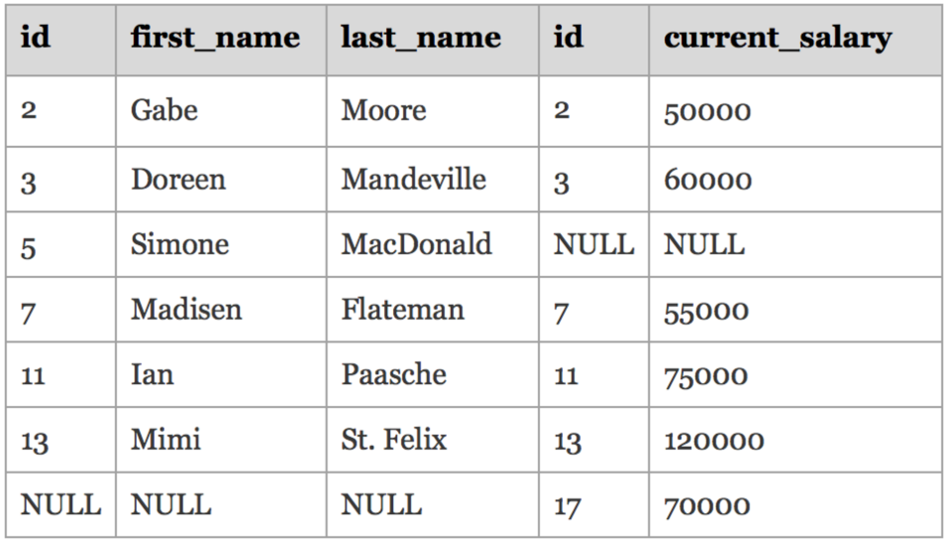
Join Types

OUTER, EXCEPTION, AND CARTESIAN JOINS

- A FULL OUTER JOIN returns all data from each table, regardless of whether it has matching data in the other table.

- A FULL OUTER JOIN would yield this table:

SELECT * FROM employees FULL OUTER JOIN salaries ON employees.ID = salaries.ID;



id	first_name	last_name	id	current_salary
2	Gabe	Moore	2	50000
3	Doreen	Mandeville	3	60000
5	Simone	MacDonald	NULL	NULL
7	Madisen	Flateman	7	55000
11	Ian	Paasche	11	75000
13	Mimi	St. Felix	13	120000
NULL	NULL	NULL	17	70000

Join Types

OUTER, EXCEPTION, AND CARTESIAN JOINS

- An EXCEPTION JOIN returns only the data from the primary, or first table selected, which does not have matching data to join to in the secondary table.
- An EXCEPTION JOIN would yield this table:

```
SELECT * FROM employees LEFT OUTER
JOIN salaries ON employees.ID = salaries.ID
WHERE salaries.ID IS NULL;
```

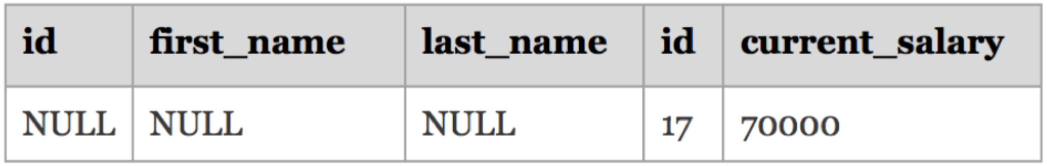
id	first_name	last_name	id	current_salary
5	Simone	MacDonald	NULL	NULL

Join Types

OUTER, EXCEPTION, AND CARTESIAN JOINS

- A RIGHT EXCEPTION JOIN returns only the data from the secondary, or second table selected, which does not have matching data to join to in the primary table.
- A RIGHT EXCEPTION JOIN would yield this table:

```
SELECT * FROM employees RIGHT OUTER
JOIN salaries ON employees.ID = salaries.ID
WHERE employees.ID IS NULL;
```



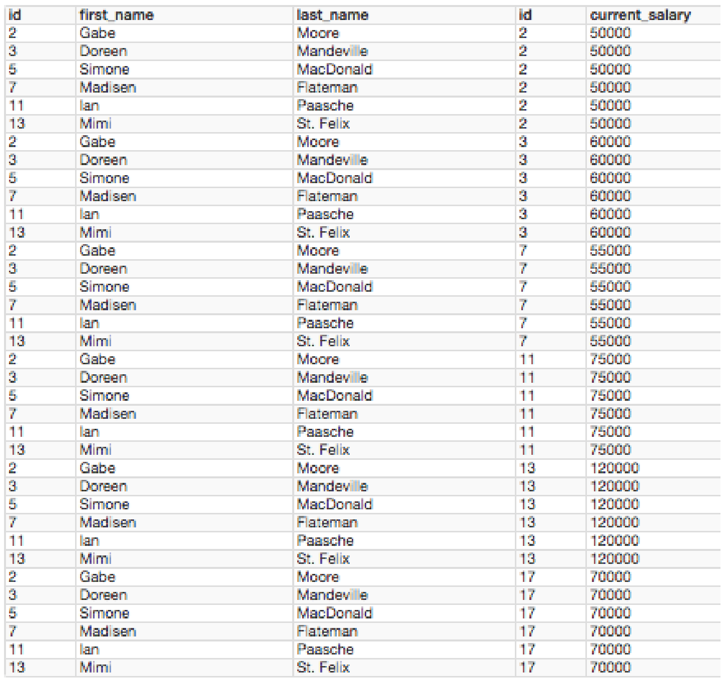
id	first_name	last_name	id	current_salary
NULL	NULL	NULL	17	70000

Join Types

OUTER, EXCEPTION, AND CARTESIAN JOINS

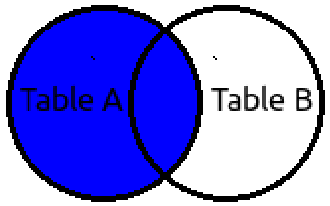
- A CROSS JOIN matches every row of the primary table with every row of the secondary table.
- This type of join results in a Cartesian product of the tables, is generally detrimental to slow performance, and is not desired.
- A CROSS JOIN would yield this table:

```
SELECT * FROM employees CROSS JOIN salaries;
```

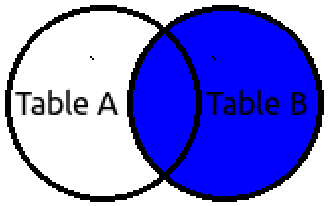


id	first_name	last_name	id	current_salary
2	Gabe	Moore	2	50000
3	Doreen	Mandeville	2	50000
5	Simone	MacDonald	2	50000
7	Madisen	Flateman	2	50000
11	Ian	Paasche	2	50000
13	Mimi	St. Felix	2	50000
2	Gabe	Moore	3	60000
3	Doreen	Mandeville	3	60000
5	Simone	MacDonald	3	60000
7	Madisen	Flateman	3	60000
11	Ian	Paasche	3	60000
13	Mimi	St. Felix	3	60000
2	Gabe	Moore	7	55000
3	Doreen	Mandeville	7	55000
5	Simone	MacDonald	7	55000
7	Madisen	Flateman	7	55000
11	Ian	Paasche	7	55000
13	Mimi	St. Felix	7	55000
2	Gabe	Moore	11	75000
3	Doreen	Mandeville	11	75000
5	Simone	MacDonald	11	75000
7	Madisen	Flateman	11	75000
11	Ian	Paasche	11	75000
13	Mimi	St. Felix	11	75000
2	Gabe	Moore	13	120000
3	Doreen	Mandeville	13	120000
5	Simone	MacDonald	13	120000
7	Madisen	Flateman	13	120000
11	Ian	Paasche	13	120000
13	Mimi	St. Felix	13	120000
2	Gabe	Moore	17	70000
3	Doreen	Mandeville	17	70000
5	Simone	MacDonald	17	70000
7	Madisen	Flateman	17	70000
11	Ian	Paasche	17	70000
13	Mimi	St. Felix	17	70000

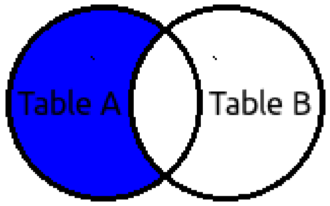
Join Types



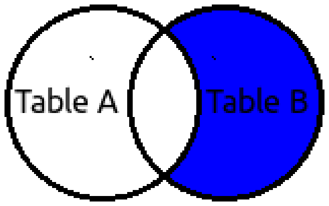
```
SELECT [list] FROM
    [Table A] A
LEFT JOIN
    [Table B] B
ON A.Value = B.Value
```



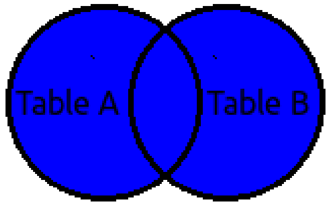
```
SELECT [list] FROM
    [Table A] A
RIGHT JOIN
    [Table B] B
ON A.Value = B.Value
```



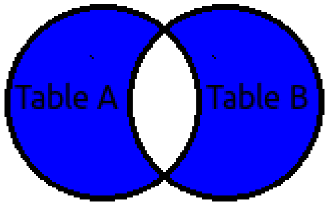
```
SELECT [list] FROM
    [Table A] A
LEFT JOIN
    [Table B] B
ON A.Value = B.Value
WHERE B.Value IS NULL
```



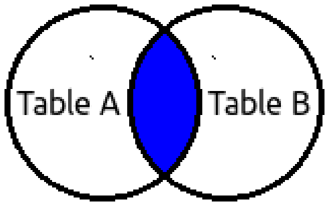
```
SELECT [list] FROM
    [Table A] A
RIGHT JOIN
    [Table B] B
ON A.Value = B.Value
WHERE A.Value IS NULL
```



```
SELECT [list] FROM
    [Table A] A
FULL OUTER JOIN
    [Table B] B
ON A.Value = B.Value
```



```
SELECT [list] FROM
    [Table A] A
FULL OUTER JOIN
    [Table B] B
ON A.Value = B.Value
WHERE A.Value IS NULL
OR B.Value IS NULL
```



```
SELECT [list] FROM
    [Table A] A
INNER JOIN
    [Table B] B
ON A.Value = B.Value
```

Join Types

“I want to see all of the information we can get on inactive stores for sales, if there are any, and their addresses.”

Process to pick the right join

Join Types

ACTIVITY: OUTER AND FULL OUTER JOINS

EXERCISE

DIRECTIONS

Your Deloitte boss won't let up with the questioning. Please write queries to answer the following questions:

- Were there any sales in the database completed at an inactive store?
- Which sales included tequila products?
- Which tequila products were not sold?
- Which distinct products were sold in Mason City, IA?
- Which Scotch whiskies were sold in Mason City, IA?
- Which unique types of products, other than whiskies, were sold in Mason City, IA?
- As a check for data consistency, were there any sales of products that are not listed in the product table?
- As another check for data consistency, were there any sales at a store that does not exist?

Workshop

Q & A

I never guess. It is a capital mistake to theorize before one has data.
Insensibly one begins to twist facts to suit theories,
instead of theories to suit facts.
-Sir Arthur Conan Doyle, Author

Conclusion

We looked at the various types of Table joins and when you would use them.

We reviewed the syntax for LEFT (and RIGHT) Joins as well as FULL OUTER anc
EXCEPTION.

Finally, we took a look at optimizing your queries to run efficiently.

EXIT TICKET

CLASS : Querying a Relational Database

QUESTION:

Why do you need an alias when using Joins?