SQL Joins

Instructor: Maninder Tatla

Email: maninder.tatla@niagaracollegetoronto.ca

Course: WEBD1102 - Web Authoring Fundamentals

Outline

- Introduction to Joins
- Inner Join / Join
- Left Outer Join / Left Join
- Right Outer Join / Right Join

Introduction

- SQL is a special-purpose programming language designed for managing information in a relational database management system (**RDBMS**).
- A JOIN clause is used to combine rows from two or more tables, based on a related column between them.

Create related Tables

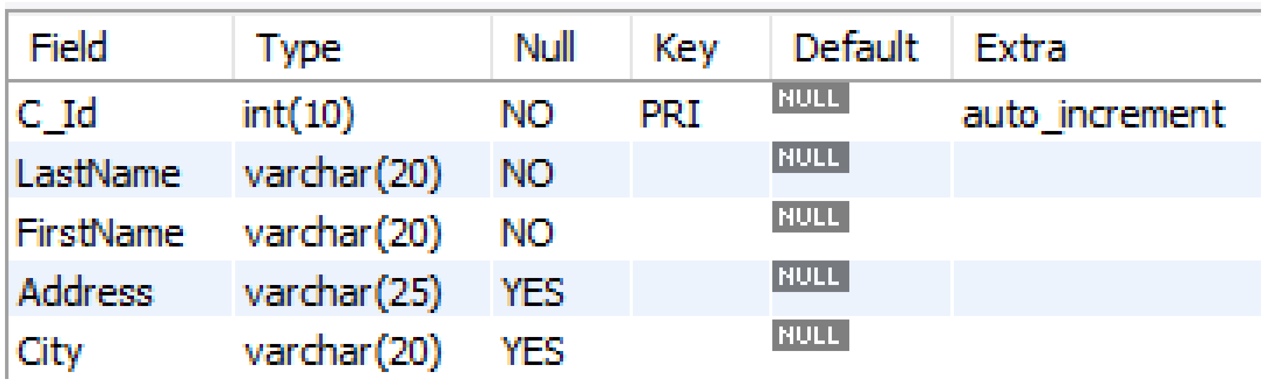
- The following SQL creates a FOREIGN KEY on the “C_Id" column when the "Orders" table is created:

```
CREATE TABLE CUSTOMERS
(
 C_Id INT(10) NOT NULL AUTO_INCREMENT,
 LastName VARCHAR (20) NOT NULL,
 FirstName VARCHAR (20) NOT NULL,
 Address VARCHAR (25),
 City VARCHAR (20),
 PRIMARY KEY (C_Id)
);
```

```
CREATE TABLE Orders
(
O_Id int(10) NOT NULL AUTO_INCREMENT,
OrderNo int(30) NOT NULL,
C_Id int(10),
PRIMARY KEY (O_Id),
FOREIGN KEY (C_Id) REFERENCES CUSTOMERS(C_Id)
);
```

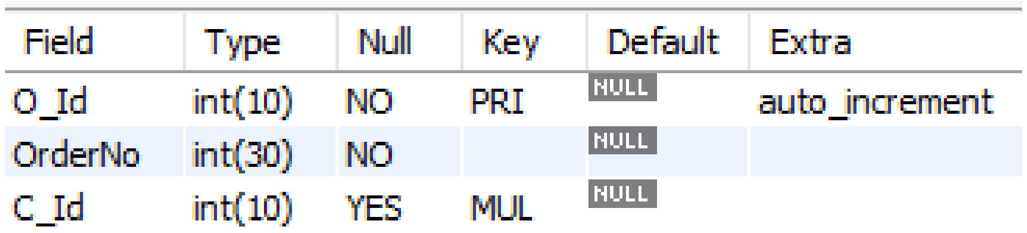
Describe tables

```
DESCRIBE CUSTOMERS;
```



Field	Type	Null	Key	Default	Extra
C_Id	int(10)	NO	PRI	NULL	auto_increment
LastName	varchar(20)	NO		NULL	
FirstName	varchar(20)	NO		NULL	
Address	varchar(25)	YES		NULL	
City	varchar(20)	YES		NULL	

```
DESCRIBE ORDERS;
```



Field	Type	Null	Key	Default	Extra
O_Id	int(10)	NO	PRI	NULL	auto_increment
OrderNo	int(30)	NO		NULL	
C_Id	int(10)	YES	MUL	NULL	

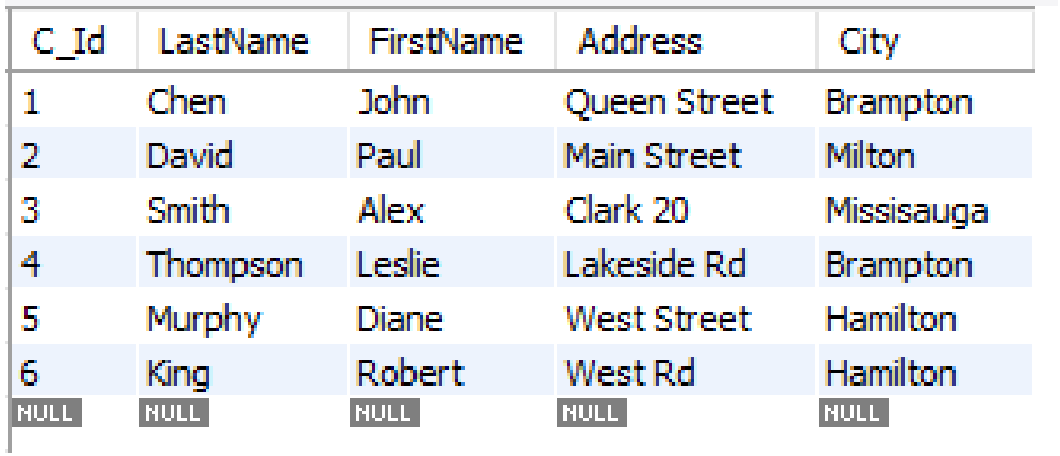
Insert Values

```
INSERT INTO CUSTOMERS(LastName,FirstName,Address,City)VALUES
('Chen','John','Queen Street','Brampton'),
('David','Paul','Main Street','Milton'),
('Smith','Alex','Clark20','Missisauga'),
('Thompson','Leslie','Lakeside Rd','Brampton'),
('Murphy','Diane','West Street','Hamilton'),
('King','Robert','West Rd','Hamilton');
```

```
Insert into Orders(OrderNo,C_ID)values(523,2),(234,1),(451,3),(845,2),(214,5);
```

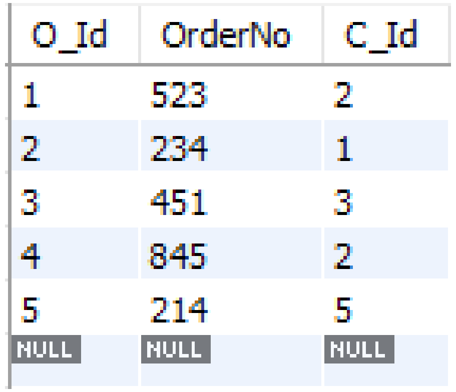
Tables

```
SELECT * from Customers;
```



C_Id	LastName	FirstName	Address	City
1	Chen	John	Queen Street	Brampton
2	David	Paul	Main Street	Milton
3	Smith	Alex	Clark 20	Missisauga
4	Thompson	Leslie	Lakeside Rd	Brampton
5	Murphy	Diane	West Street	Hamilton
6	King	Robert	West Rd	Hamilton
NULL	NULL	NULL	NULL	NULL

```
SELECT * from Orders;
```



O_Id	OrderNo	C_Id
1	523	2
2	234	1
3	451	3
4	845	2
5	214	5
NULL	NULL	NULL

Introduction to JOINS

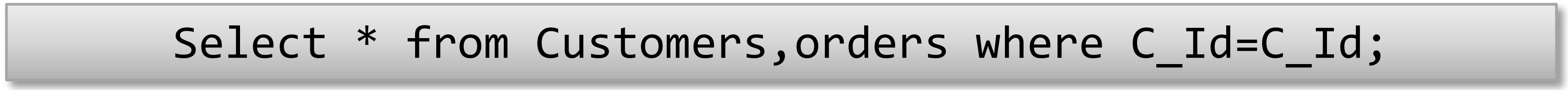
- A JOIN clause is used to combine rows from two or more tables, based on a related column between them.
- If we have a field (meaning column) that is the same in two tables, we can JOIN them together
- This means we can retrieve records where certain criteria from the first table exactly match the criteria from the second table

JOINS – THE EASY WAY

- The easiest way to join two tables:
  - We have a table called **Customer** with a field called C_Id
  - We also have a table called **Order** with a matching field called C_Id
  - We can therefore write a query that gets the results from both tables where the **C_Id** field matches

JOINS – THE EASY WAY

- We might write a query like this:



```
Select * from Customers,orders where C_Id=C_Id;
```

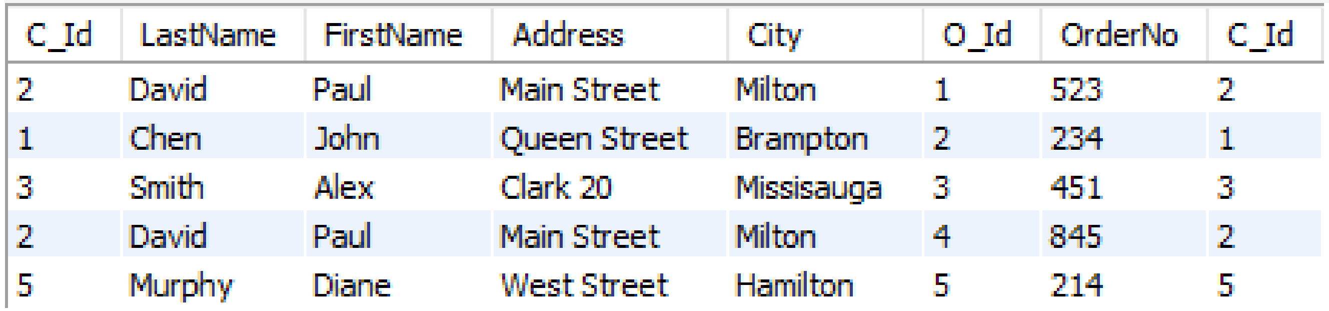
… but there may be some confusion over which **C_Id** belongs to which table - the above query won’t work yet

- As such, we’re going to add the table name into the query on the next slide so that MySQL knows which columns we want

JOINS – THE EASY WAY

- Here’s a working query to match up the **C_Id** in the Customers table with the **C_Id** in the Orders table

```
Select * from Customers,orders where Customers.C_Id=Orders.C_Id;
```

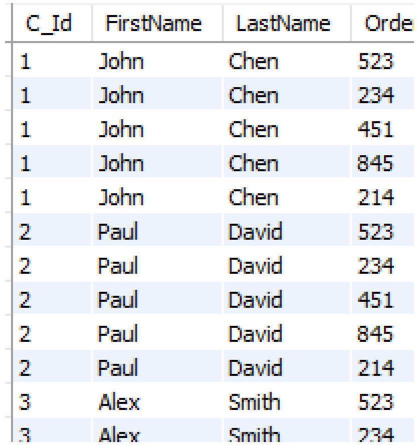


C_Id	LastName	FirstName	Address	City	O_Id	OrderNo	C_Id
2	David	Paul	Main Street	Milton	1	523	2
1	Chen	John	Queen Street	Brampton	2	234	1
3	Smith	Alex	Clark 20	Missisauga	3	451	3
2	David	Paul	Main Street	Milton	4	845	2
5	Murphy	Diane	West Street	Hamilton	5	214	5

- Notice the query does not return the cross-product of both tables - it is very selective and only returns rows where the C_Id matches in both tables

JOINS – THE EASY WAY

```
Select Customers.C_Id, Customers.FirstName, Customers.LastName, Orders.OrderNo
from Customers,orders;
```

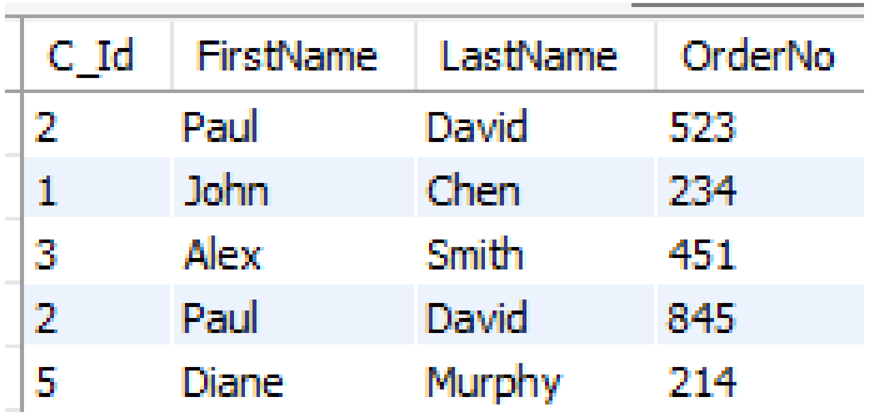


C_Id	FirstName	LastName	Orde
1	John	Chen	523
1	John	Chen	234
1	John	Chen	451
1	John	Chen	845
1	John	Chen	214
2	Paul	David	523
2	Paul	David	234
2	Paul	David	451
2	Paul	David	845
2	Paul	David	214
3	Alex	Smith	523
3	Alex	Smith	234

- There are duplicates fields in the query that matches on C_Id – all the data from both table rows is present wherever the C_Id line up

JOINS – THE EASY WAY

```
Select Customers.C_Id, Customers.FirstName, Customers.LastName, Orders.OrderNo
from Customers,orders where Customers.C_Id=Orders.C_Id;
```



C_Id	FirstName	LastName	OrderNo
2	Paul	David	523
1	John	Chen	234
3	Alex	Smith	451
2	Paul	David	845
5	Diane	Murphy	214

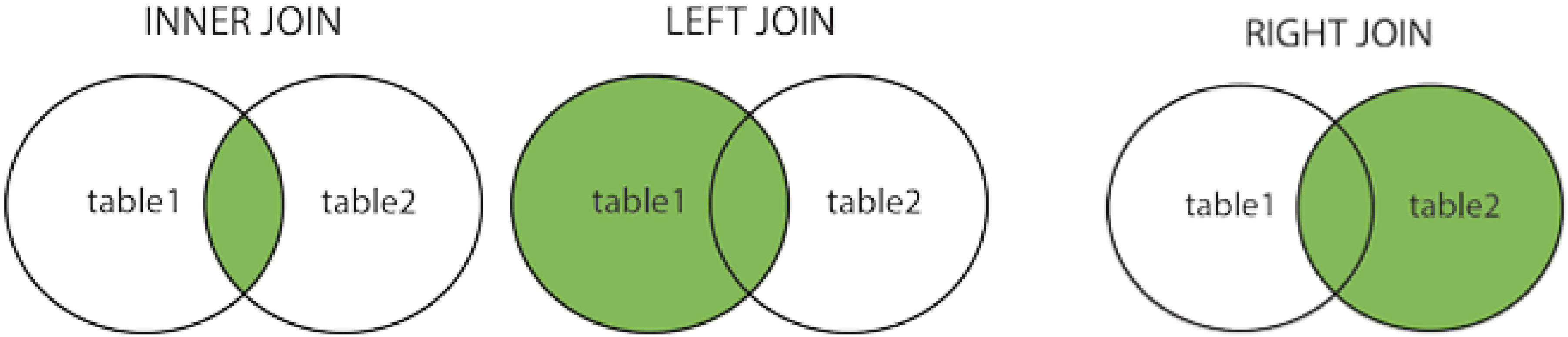
- To avoid duplicates fields, we can use where clause:

```
where Customers.C_Id=Orders.C_Id;
```

Types of SQL JOINs

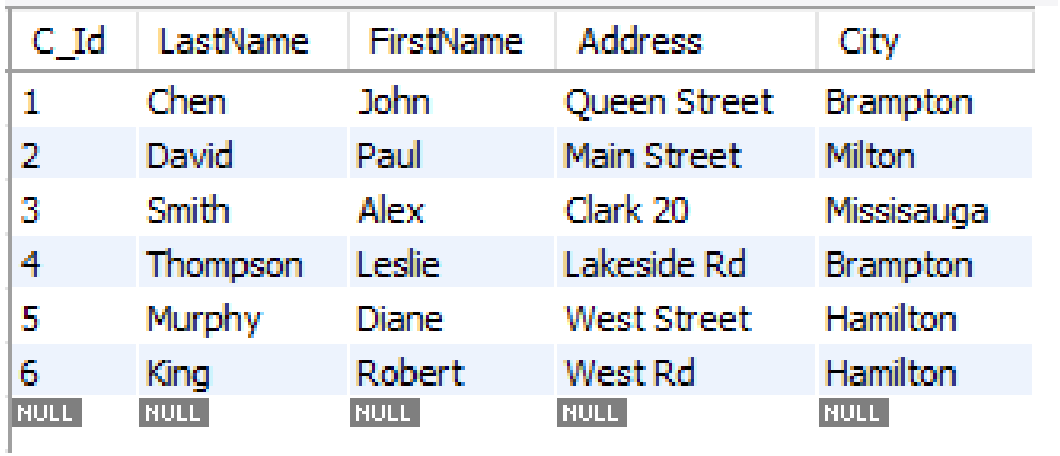
- Here are the different types of the JOINs in SQL:
- **(INNER) JOIN**: Returns records that have matching values in both tables
- **LEFT (OUTER) JOIN:** Returns all records from the left table, and the matched records from the right table
- **RIGHT (OUTER) JOIN:** Returns all records from the right table, and the matched records from the left table

Types of SQL JOINs



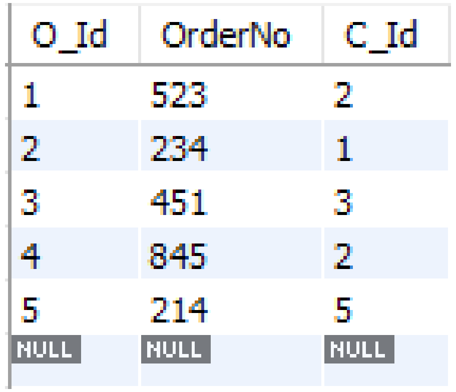
Tables

```
SELECT * FROM Customers;
```



C_Id	LastName	FirstName	Address	City
1	Chen	John	Queen Street	Brampton
2	David	Paul	Main Street	Milton
3	Smith	Alex	Clark 20	Missisauga
4	Thompson	Leslie	Lakeside Rd	Brampton
5	Murphy	Diane	West Street	Hamilton
6	King	Robert	West Rd	Hamilton
NULL	NULL	NULL	NULL	NULL

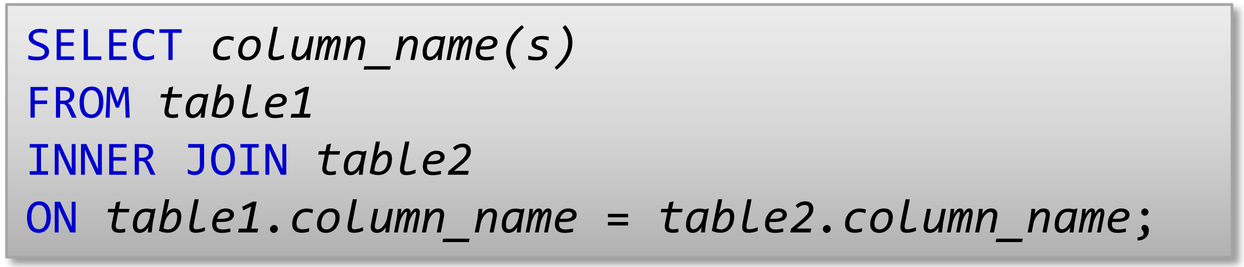
```
SELECT * FROM Orders;
```



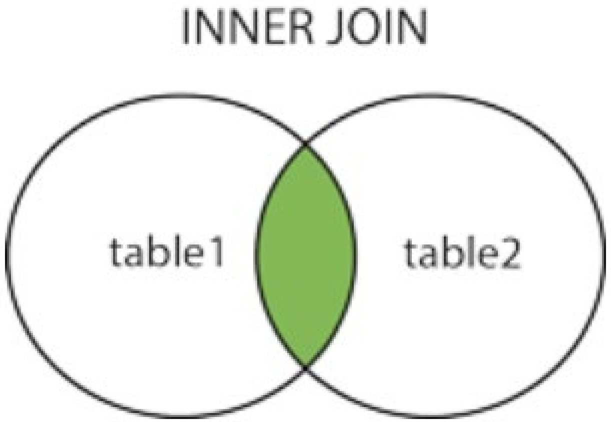
O_Id	OrderNo	C_Id
1	523	2
2	234	1
3	451	3
4	845	2
5	214	5
NULL	NULL	NULL

SQL INNER JOIN

- The INNER JOIN keyword selects records that have matching values in both tables.



```
SELECT column_name(s)
FROM table1
INNER JOIN table2
ON table1.column_name = table2.column_name;
```



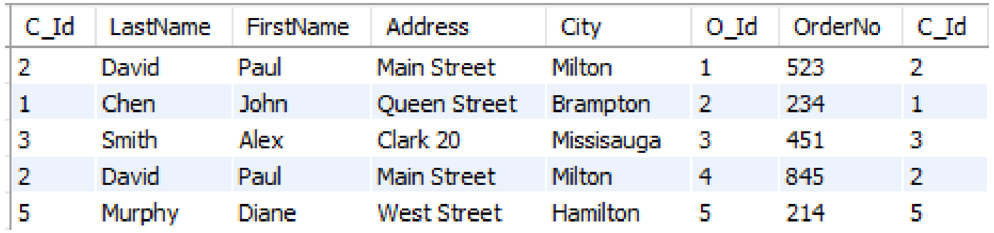
- **Note:** The INNER JOIN keyword selects all rows from both tables as long as there is a match between the columns.

- If there are records in the "Orders" table that do not have matches in "Customers", these orders will not be shown!

INNER JOIN

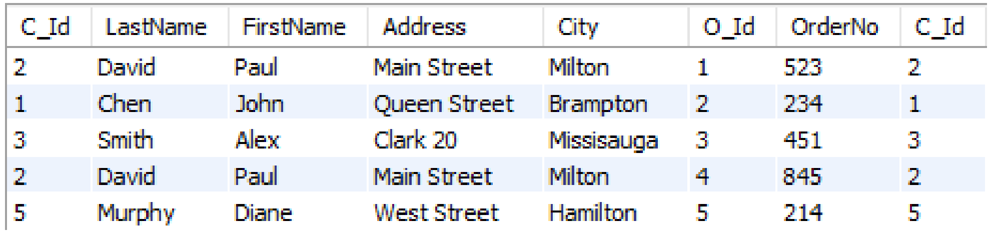
- **INNER JOIN:** Returns records that have matching values in both tables

```
SELECT * FROM Customers,Orders WHERE Customers.C_Id=Orders.C_Id;
```



C_Id	LastName	FirstName	Address	City	O_Id	OrderNo	C_Id
2	David	Paul	Main Street	Milton	1	523	2
1	Chen	John	Queen Street	Brampton	2	234	1
3	Smith	Alex	Clark 20	Missisauga	3	451	3
2	David	Paul	Main Street	Milton	4	845	2
5	Murphy	Diane	West Street	Hamilton	5	214	5

```
SELECT * FROM Customers INNER JOIN Orders ON Customers.C_Id=Orders.C_Id;
```

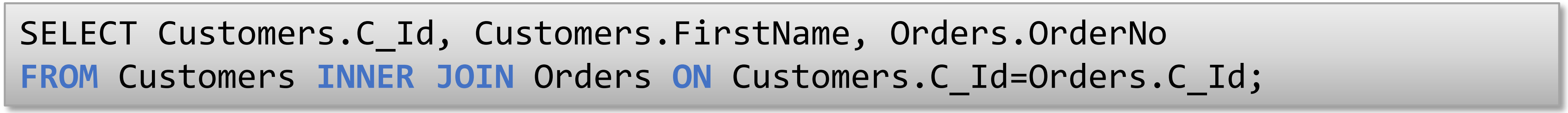


C_Id	LastName	FirstName	Address	City	O_Id	OrderNo	C_Id
2	David	Paul	Main Street	Milton	1	523	2
1	Chen	John	Queen Street	Brampton	2	234	1
3	Smith	Alex	Clark 20	Missisauga	3	451	3
2	David	Paul	Main Street	Milton	4	845	2
5	Murphy	Diane	West Street	Hamilton	5	214	5

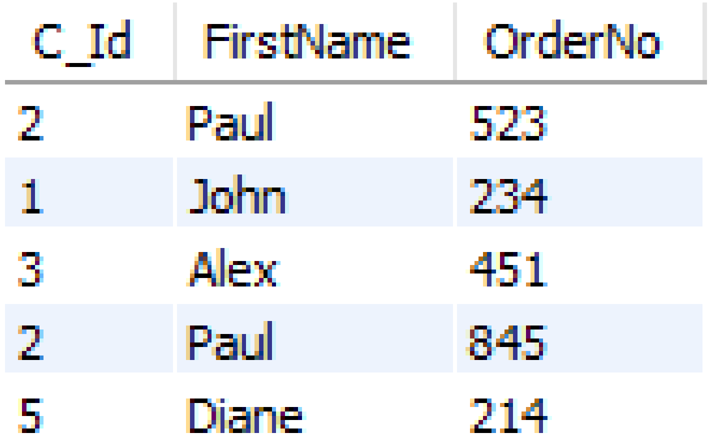
If the query says **INNER JOIN**, we know we're **JOIN**ing them, and we know **ON** what

INNER JOIN

- **INNER JOIN:** Returns records that have matching values in both tables



```
SELECT Customers.C_Id, Customers.FirstName, Orders.OrderNo
FROM Customers INNER JOIN Orders ON Customers.C_Id=Orders.C_Id;
```

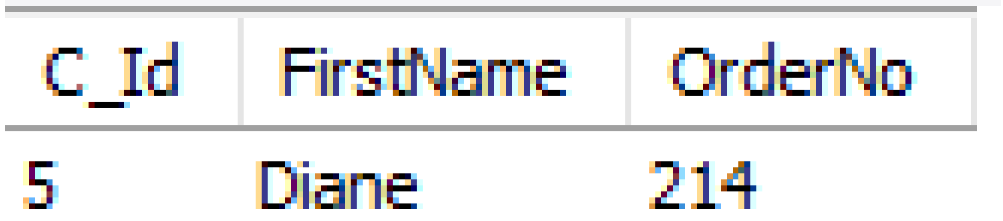


C_Id	FirstName	OrderNo
2	Paul	523
1	John	234
3	Alex	451
2	Paul	845
5	Diane	214

INNER JOIN

- We can always add in a WHERE clause after the JOIN if we need to be more specific anyway

```
SELECT Customers.C_Id, Customers.FirstName, Orders.OrderNo
FROM Customers INNER JOIN Orders ON Customers.C_Id=Orders.C_Id
WHERE FirstName like 'D%';
```

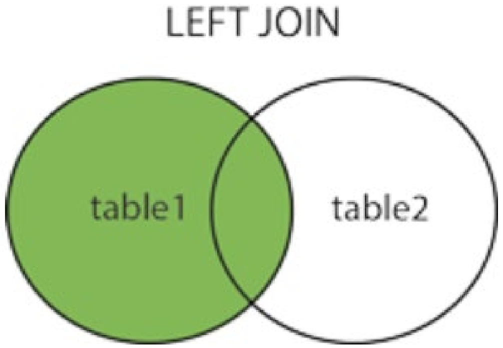


C_Id	FirstName	OrderNo
5	Diane	214

LEFT JOIN

- The LEFT JOIN keyword returns all records from the left table (table1), and the matching records from the right table (table2).
- The result is 0 records from the right side, if there is no match.
- Note: In some databases **LEFT JOIN** is called **LEFT OUTER JOIN**.

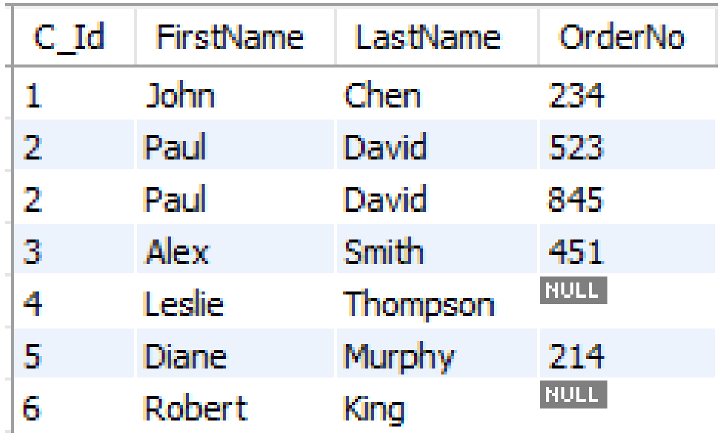
```
SELECT column_name(s)
FROM table1
LEFT JOIN table2
ON table1.column_name = table2.column_name;
```



LEFT JOIN

- **LEFT JOIN:** The LEFT JOIN keyword returns all records from the left table (table1 - **Customers**), and the matched records from the right table (table2 - **Orders**). The result is NULL from the right side, if there is no match.

```
SELECT Customers.C_Id, Customers.FirstName, Customers.LastName, Orders.OrderNo
FROM Customers LEFT JOIN Orders ON Customers.C_Id=Orders.C_Id;
```

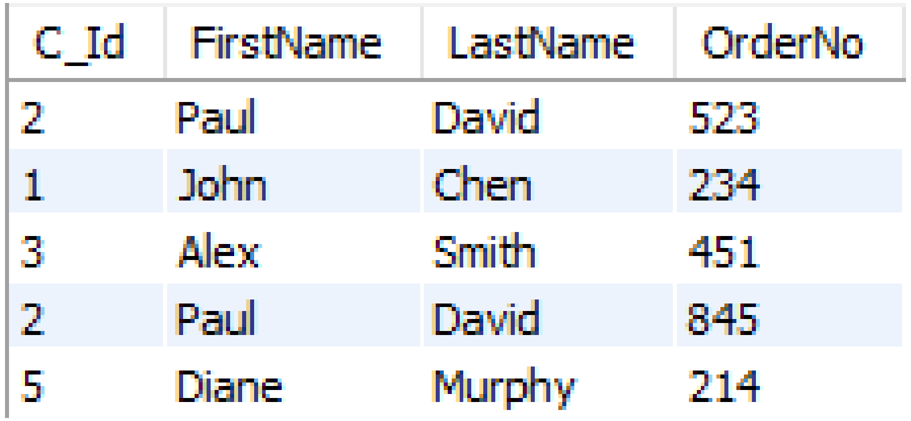


C_Id	FirstName	LastName	OrderNo
1	John	Chen	234
2	Paul	David	523
2	Paul	David	845
3	Alex	Smith	451
4	Leslie	Thompson	NULL
5	Diane	Murphy	214
6	Robert	King	NULL

LEFT JOIN

- **LEFT JOIN:** The LEFT JOIN keyword returns all records from the left table (table1 - **Orders**), and the matched records from the right table (table2 - **Customers**). The result is NULL from the right side, if there is no match.

```
SELECT Customers.C_Id, Customers.FirstName, Customers.LastName, Orders.OrderNo
FROM Orders LEFT JOIN Customers ON Customers.C_Id=Orders.C_Id;
```

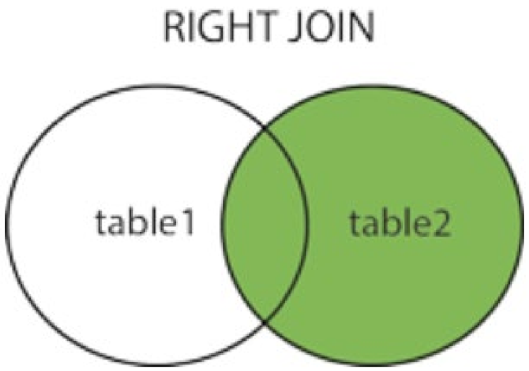


C_Id	FirstName	LastName	OrderNo
2	Paul	David	523
1	John	Chen	234
3	Alex	Smith	451
2	Paul	David	845
5	Diane	Murphy	214

RIGHT JOIN

- The **RIGHT JOIN** keyword returns all records from the right table (table2), and the matching records from the left table (table1).
- The result is 0 records from the left side, if there is no match.
- Note: In some databases **RIGHT JOIN** is called **RIGHT OUTER JOIN**.

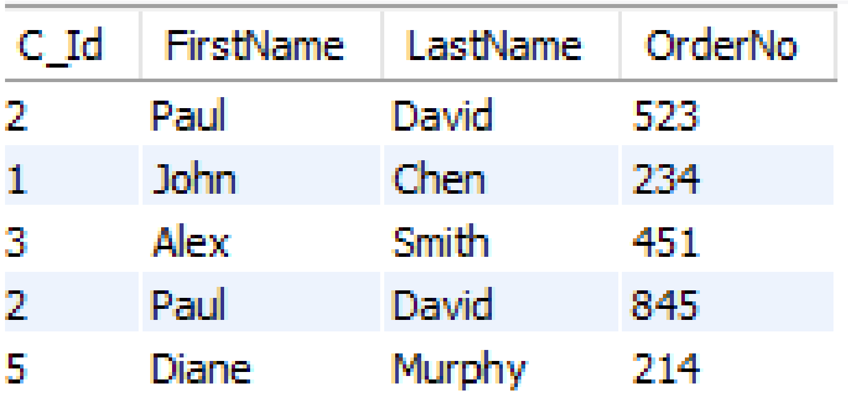
```
SELECT column_name(s)
FROM table1
LEFT JOIN table2
ON table1.column_name = table2.column_name;
```



RIGHT JOIN

- **RIGHT JOIN:** The RIGHT JOIN keyword returns all records from the right table (table2 - **Orders**), and the matched records from the left table (table1 - **Customers**). The result is NULL from the left side, when there is no match.

```
SELECT Customers.C_Id, Customers.FirstName, Customers.LastName, Orders.OrderNo
FROM Customers RIGHT JOIN Orders ON Customers.C_Id=Orders.C_Id;
```

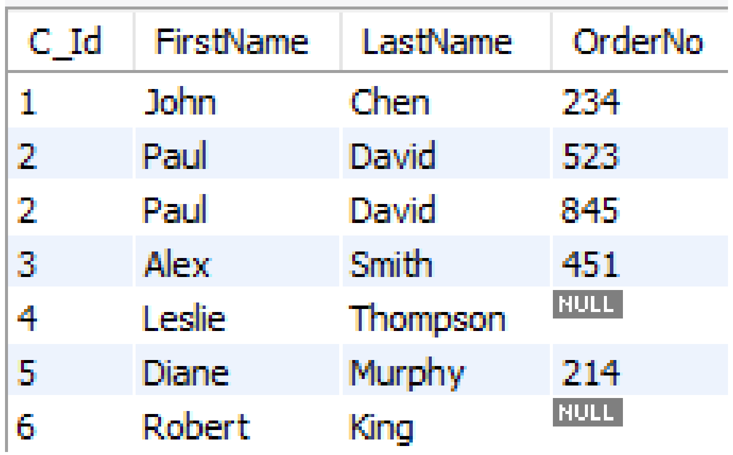


C_Id	FirstName	LastName	OrderNo
2	Paul	David	523
1	John	Chen	234
3	Alex	Smith	451
2	Paul	David	845
5	Diane	Murphy	214

RIGHT JOIN

- **RIGHT JOIN:** The RIGHT JOIN keyword returns all records from the right table (table2 - **Customers**), and the matched records from the left table (table1 - **Orders**). The result is NULL from the left side, when there is no match.

```sql
SELECT Customers.C_Id, Customers.FirstName, Customers.LastName, Orders.OrderNo
FROM Orders RIGHT JOIN Customers ON Customers.C_Id=Orders.C_Id;
```



C_Id	FirstName	LastName	OrderNo
1	John	Chen	234
2	Paul	David	523
2	Paul	David	845
3	Alex	Smith	451
4	Leslie	Thompson	NULL
5	Diane	Murphy	214
6	Robert	King	NULL

Any questions please?