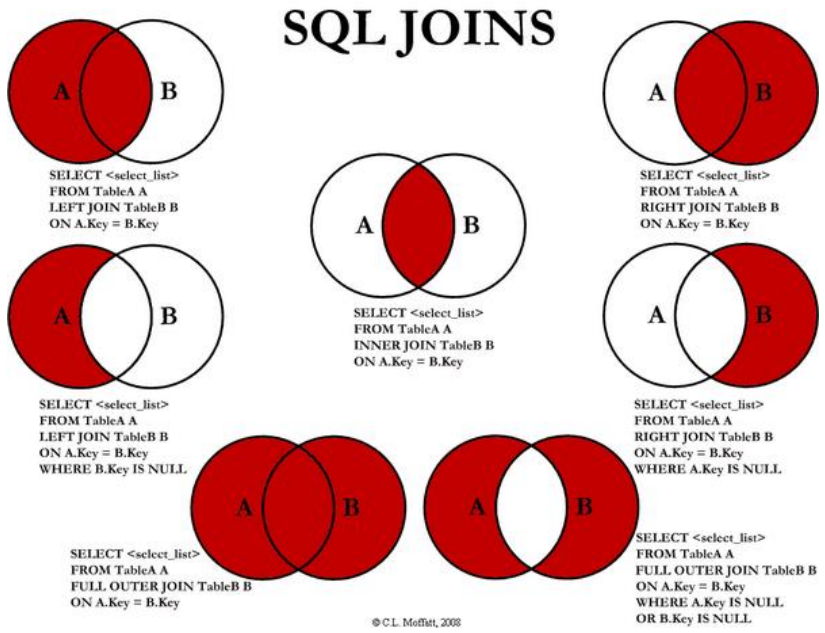
JOINS IN SQL SERVER

LEARNING GOALS

By the end of this lecture students should be able to:

- ✓ Understand about SQL joins in SQL Server
- ✓ Using smoothly SQL joins and apply to project

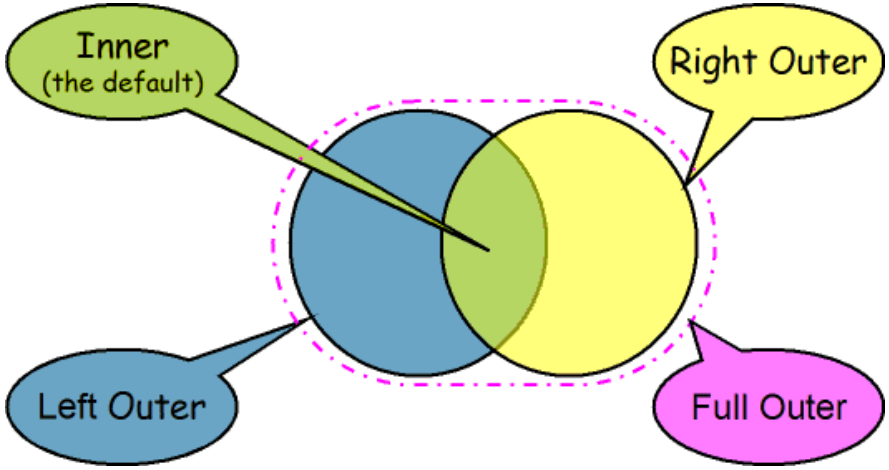


Lesson Agenda

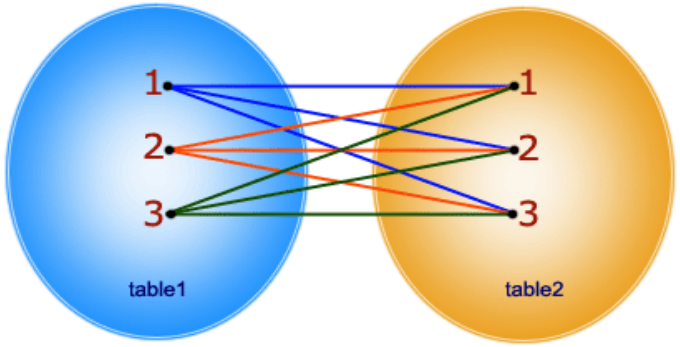
- What's SQL Join?
- Inner Join
- Outer Join
- Cross Join
- Self Join
- Excluding JOIN

Options ?

- SQL Joins are used to combine rows from two or more tables based on logical relationships between the tables.
- Types of Join in SQL:
  - ✓ Inner Join
  - ✓ Outer Join
  - ✓ Cross Join
  - ✓ Self Join




```
SELECT * FROM table1 CROSS JOIN  table2;
```



In CROSS JOIN, each row from 1st table joins with all the rows of another table.
If 1st table contain x rows and y rows in 2nd one the result set will be x * y rows.

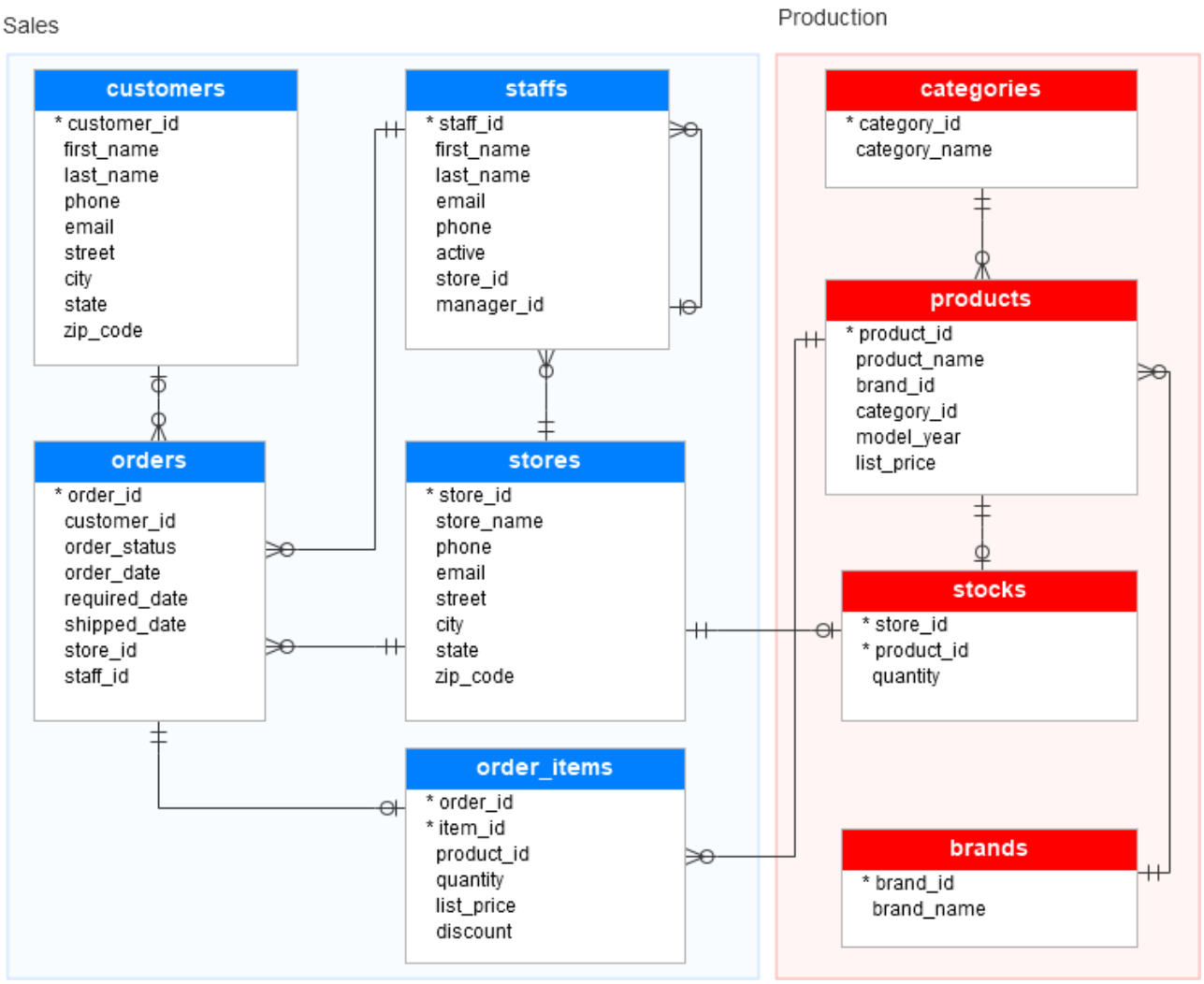
Column Name	Data Type	Nullable	Default	Primary Key
EMP_ID	VARCHAR (5)	No	-	1
EMP_NAME	VARCHAR (20)	Yes	-	-
DT_OF_JOIN	DATE	Yes	-	-
EMP_SUPV	VARCHAR (5)	Yes	-	-
				1 - 4

Constraint	Type	Table
SYS_C004074	C	EMPLOYEE
EMP_ID	P	EMPLOYEE
EMP_SUPV	R	EMPLOYEE

Primary key

Foreign key
Referencing EMP_ID of this table

Database


Sales
Production
customers
* customer_id
first_name
last_name
phone
email
street
city
state
zip_code
staffs
* staff_id
first_name
last_name
email
phone
active
store_id
manager_id
categories
* category_id
category_name
products
* product_id
product_name
brand_id
category_id
model_year
list_price
orders
* order_id
customer_id
order_status
order_date
required_date
shipped_date
store_id
staff_id
stores
* store_id
store_name
phone
email
street
city
state
zip_code
stocks
* store_id
* product_id
quantity
order_items
* order_id
* item_id
product_id
quantity
list_price
discount
brands
* brand_id
brand_name

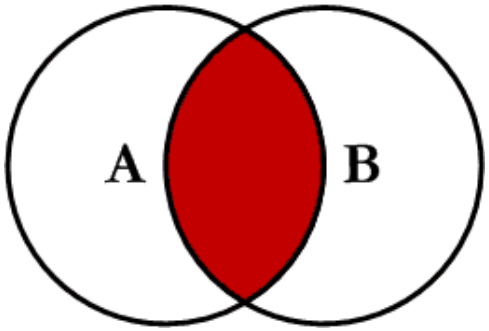
Section 1

INNER JOIN

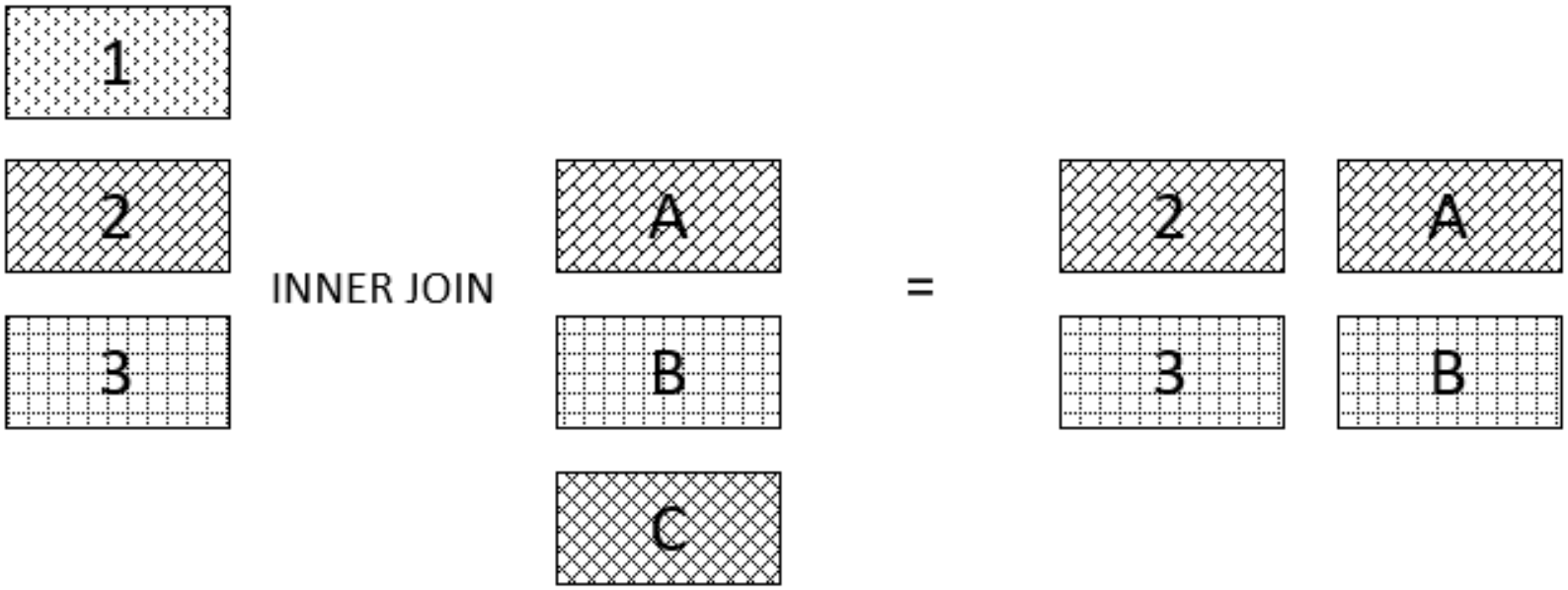
INNER JOIN

- The INNER JOIN selects all rows from both tables as long as there is a match between the columns in both tables.
- Eliminate the rows that do not match with a row from the other table
  - ✓ **Syntax**

```
SELECT col_names
FROM Table_A  A
  INNER JOIN Table_B  B
        ON A.Col1 = B.Col1
```



INNER JOIN (2/2)



INNER JOIN (2/2)

- **Example:**

Customer

CustID	CustName	BirthDate	Country
1	Davolio Nancy	12/8/1968	Germany
2	Fuller Andrew	2/19/1952	Mexico
3	Leverling Janet	8/30/1963	Mexico

[Order]

OrderID	CustID	OrderDate	ShipperID
10308	2	2013-09-18	3
10309	3	2013-09-19	1
10310	77	2013-09-20	2

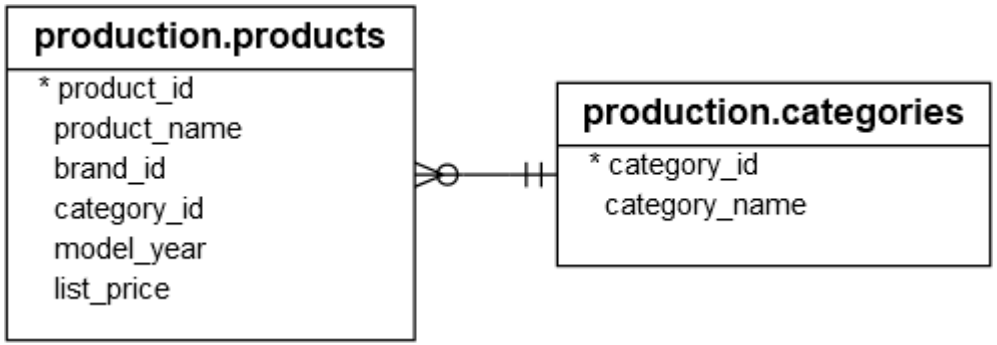
```
SELECT c.CustName, o.OrderID
FROM Customer c
        INNER JOIN [Order] o
                ON c.CustID = o.CustID
ORDER BY c.CustName;
```

- **Result:**

Results | Messages

	CustName	OrderID
1	Fuller Andrew	10308
2	Leverling Janet	10309

Do it your self?



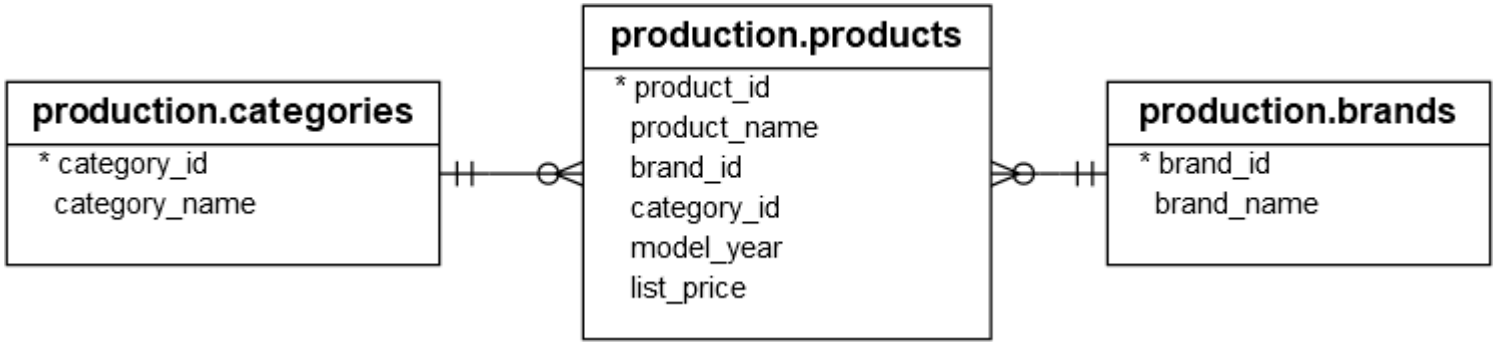
Display information about **product_name, category_name and list_price of all product**

product_name	category_name	list_price
Trek XM700+ Lowstep - 2018	Electric Bikes	3499.99
Trek XM700+ - 2018	Electric Bikes	3499.99
Trek X-Caliber Frameset - 2018	Mountain Bikes	1499.99
Trek X-Caliber 8 - 2018	Mountain Bikes	999.99
Trek X-Caliber 8 - 2017	Mountain Bikes	999.99
Trek X-Caliber 7 - 2018	Mountain Bikes	919.99
Trek Verve+ Lowstep - 2018	Electric Bikes	2299.99
Trek Verve+ - 2018	Electric Bikes	2299.99
Trek Ticket S Frame - 2018	Mountain Bikes	1469.99
Trek Superfly 24 - 2017/2018	Children Bicycles	489.99
Trek Superfly 20 - 2018	Children Bicycles	399.99
Trek Super Commuter+ 8S - 2018	Electric Bikes	4999.99
Trek Super Commuter+ 7 - 2018	Electric Bikes	3599.99
Trek Stache Carbon Frameset - 2018	Mountain Bikes	919.99

Do it your self?

```sql
SELECT
    product_name,
    category_name,
    list_price
FROM
    production.products p
INNER JOIN production.categories c
    ON c.category_id = p.category_id
ORDER BY
    product_name DESC;
```

Do it your self?



Display information about **product_name, category_name, brand_name, list_price** all product

product_name	category_name	brand_name	list_price
Trek XM700+ Lowstep - 2018	Electric Bikes	Trek	3499.99
Trek XM700+ - 2018	Electric Bikes	Trek	3499.99
Trek X-Caliber Frameset - 2018	Mountain Bikes	Trek	1499.99
Trek X-Caliber 8 - 2018	Mountain Bikes	Trek	999.99
Trek X-Caliber 8 - 2017	Mountain Bikes	Trek	999.99
Trek X-Caliber 7 - 2018	Mountain Bikes	Trek	919.99
Trek Verve+ Lowstep - 2018	Electric Bikes	Trek	2299.99
Trek Verve+ - 2018	Electric Bikes	Trek	2299.99
Trek Ticket S Frame - 2018	Mountain Bikes	Trek	1469.99
Trek Superfly 24 - 2017/2018	Children Bicycles	Trek	489.99

Do it your self?

```sql
SELECT
    product_name,
    category_name,
    brand_name,
    list_price
FROM
    production.products p
INNER JOIN production.categories c ON c.category_id = p.category_id
INNER JOIN production.brands b ON b.brand_id = p.brand_id
ORDER BY
    product_name DESC;
```

Section 2

OUTER JOIN

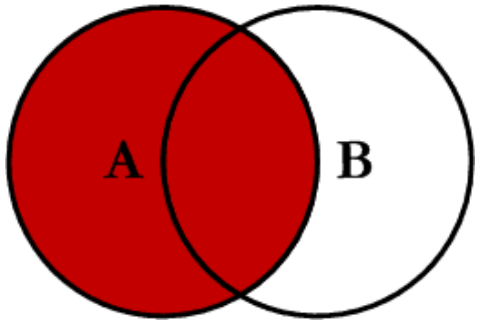
OUTER JOIN

- **Outer Join:** Return all rows from at least one of the tables mentioned in the FROM clause, as long as those rows meet any WHERE or HAVING search conditions:
  - ✓ **LEFT OUTER JOIN (or LEFT JOIN)**
  - ✓ **RIGHT OUTER JOIN (or RIGHT JOIN)**
  - ✓ **FULL OUTER JOIN (or FULL JOIN)**

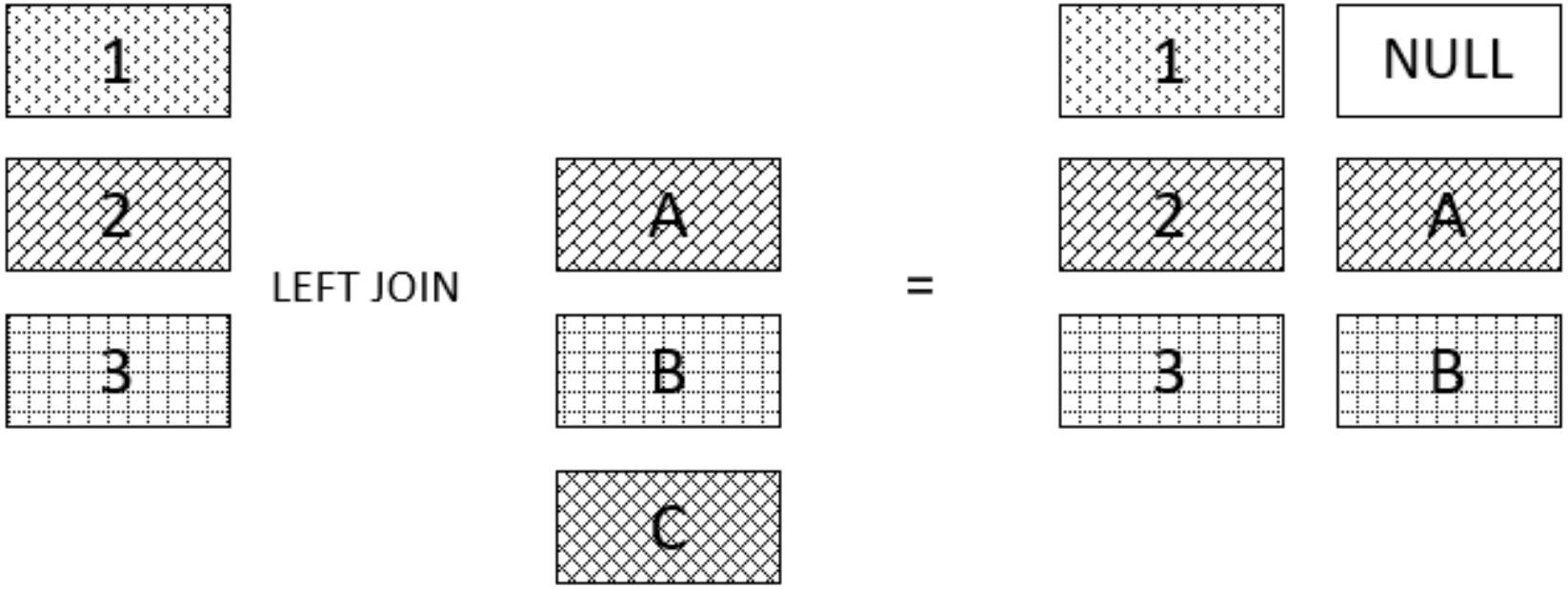
LEFT OUTER JOIN

- Return all of the records in the left table (table A) regardless if any of those records has a match in the right table (table B)
  - ✓ In the results where there is no matching condition, the row contains NULL values for the right table's columns.
- **Syntax**

```
SELECT col_names
FROM Table_A  A  LEFT JOIN Table_B  B
ON A.Col1 = B.Col1
```



INNER JOIN (2/2)



LEFT OUTER JOIN

- **Example:**

Customer

CustID	CustName	BirthDate	Country
1	Davolio Nancy	12/8/1968	Germany
2	Fuller Andrew	2/19/1952	Mexico
3	Leverling Janet	8/30/1963	Mexico

[Order]

OrderID	CustID	OrderDate	ShipperID
10308	2	2013-09-18	3
10309	3	2013-09-19	1
10310	77	2013-09-20	2

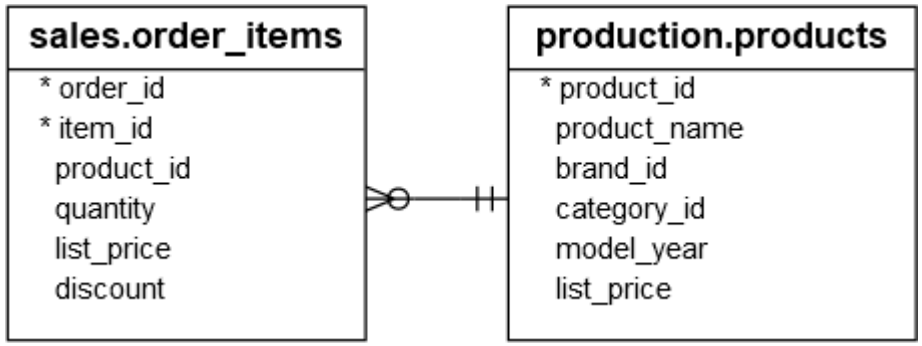
```
SELECT c.CustName, o.OrderID
FROM Customer c LEFT JOIN [Order] o
ON c.CustID = o.CustID
ORDER BY c.CustName;
```

- **Result:**

Results | Messages

	CustName	OrderID
1	Davolio Nancy	NULL
2	Fuller Andrew	10308
3	Leverling Janet	10309

Do it your self?



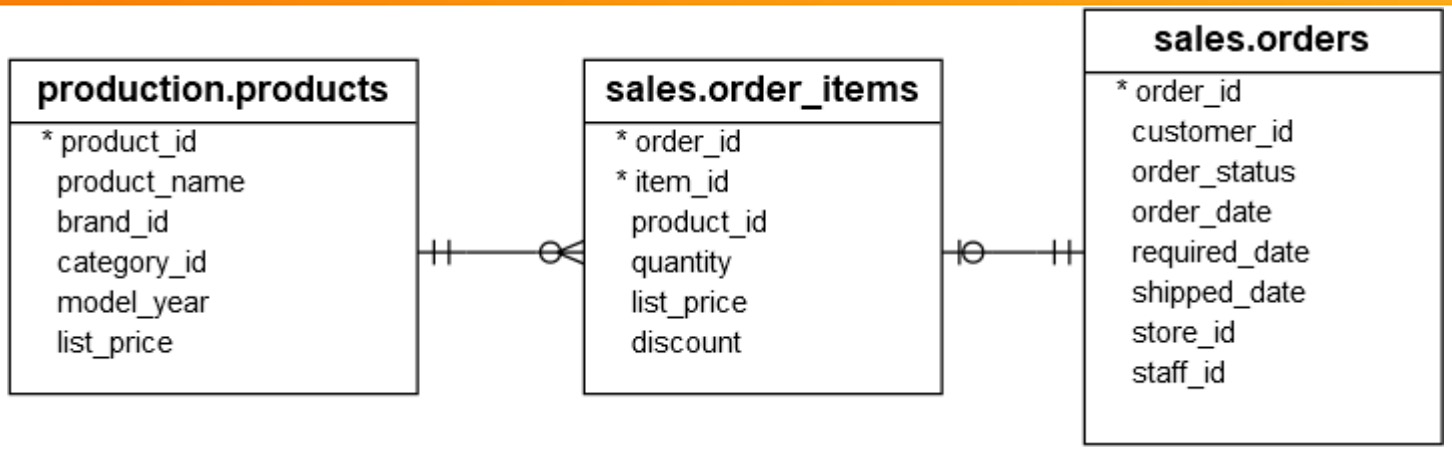
Display information about **product_name and Order ID of** all product

product_name	order_id
Electra Savannah 1 (20-inch) - Girl's - 2018	NULL
Electra Townie Go! 8i Ladies' - 2018	NULL
Trek Checkpoint ALR 5 Women's - 2019	NULL
Trek Checkpoint ALR Frameset - 2019	NULL
Trek Precaliber 12 Girl's - 2018	NULL
Surly Krampus Frameset - 2018	NULL
Trek Checkpoint SL 5 Women's - 2019	NULL
Trek 820 - 2016	NULL
Trek Checkpoint ALR 4 Women's - 2019	NULL
Trek Kids' Dual Sport - 2018	NULL
Trek Checkpoint ALR 5 - 2019	NULL
Electra Sweet Ride 1 (20-inch) - Girl's - 2018	NULL
Trek Domane SLR 6 Disc Women's - 2018	NULL
Trek Checkpoint SL 6 - 2019	NULL
Electra Townie Original 7D EQ - Women's - 2016	1
Trek Remedy 29 Carbon Frameset - 2016	1
Surly Straggler - 2016	1
Electra Townie Original 7D EQ - 2016	1
Trek Fuel EX 8 29 - 2016	1
Electra Townie Original 7D EQ - Women's - 2016	2

Do it your self?

```sql
SELECT
    product_name,
    order_id
FROM
    production.products p
LEFT JOIN sales.order_items o ON o.product_id = p.product_id
ORDER BY
    order_id;
```

Do it your self?



Display information about **p.product_name, o.order_id, i.item_id, o.order_date** all product

product_name	order_id	item_id	order_date
Electra Savannah 1 (20-inch) - Girl's - 2018	NULL	NULL	NULL
Electra Townie Go! 8i Ladies' - 2018	NULL	NULL	NULL
Trek Checkpoint ALR 5 Women's - 2019	NULL	NULL	NULL
Trek Checkpoint ALR Frameset - 2019	NULL	NULL	NULL
Trek Precaliber 12 Girl's - 2018	NULL	NULL	NULL
Surly Krampus Frameset - 2018	NULL	NULL	NULL
Trek Checkpoint SL 5 Women's - 2019	NULL	NULL	NULL
Trek 820 - 2016	NULL	NULL	NULL
Trek Checkpoint ALR 4 Women's - 2019	NULL	NULL	NULL
Trek Kids' Dual Sport - 2018	NULL	NULL	NULL
Trek Checkpoint ALR 5 - 2019	NULL	NULL	NULL
Electra Sweet Ride 1 (20-inch) - Girl's - 2018	NULL	NULL	NULL
Trek Domane SLR 6 Disc Women's - 2018	NULL	NULL	NULL
Trek Checkpoint SL 6 - 2019	NULL	NULL	NULL
Electra Townie Original 7D EQ - Women's - 2016	1	1	2016-01-01
Trek Remedy 29 Carbon Frameset - 2016	1	2	2016-01-01
Surly Straggler - 2016	1	3	2016-01-01
Electra Townie Original 7D EQ - 2016	1	4	2016-01-01
Trek Fuel EX 8 29 - 2016	1	5	2016-01-01

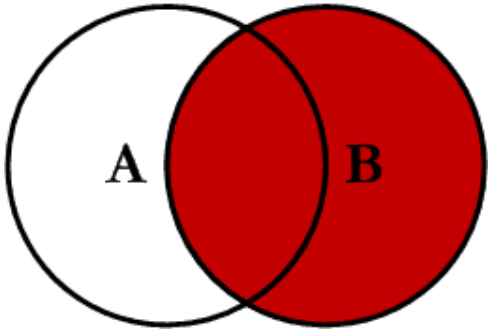
Do it your self?

```
SELECT
    p.product_name,
    o.order_id,
    i.item_id,
    o.order_date
FROM
    production.products p
        LEFT JOIN sales.order_items i
                ON i.product_id = p.product_id
        LEFT JOIN sales.orders o
                ON o.order_id = i.order_id
ORDER BY
    order_id;
```

RIGHT OUTER JOIN

- Return all of the records in the right table (table B) regardless if any of those records have a match in the left table (table A)
  - ✓ In the results where there is no matching condition, the row contains NULL values for the left table's columns.
- **Syntax**

```
SELECT col_names
FROM Table_A  A  RIGHT JOIN Table_B  B
ON A.Col1 = B.Col1
```



RIGHT OUTER JOIN

- **Example:**

Customer

CustID	CustName	BirthDate	Country
1	Davolio Nancy	12/8/1968	Germany
2	Fuller Andrew	2/19/1952	Mexico
3	Leverling Janet	8/30/1963	Mexico

[Order]

OrderID	CustID	OrderDate	ShipperID
10308	2	2013-09-18	3
10309	3	2013-09-19	1
10310	77	2013-09-20	2

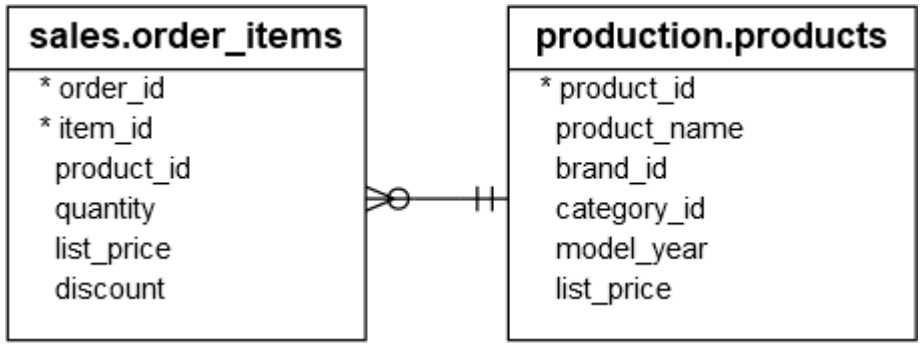
```
SELECT c.CustName, o.OrderID
FROM Customer c RIGHT JOIN [Order] o
ON c.CustID = o.CustID
ORDER BY c.CustName;
```

- **Result:**

Results | Messages

	CustName	OrderID
1	NULL	10310
2	Fuller Andrew	10308
3	Leverling Janet	10309

Do it your self?



Display information about **product_name, order_id** all product

product_name	order_id
Electra Savannah 1 (20-inch) - Girl's - 2018	NULL
Electra Townie Go! 8i Ladies' - 2018	NULL
Trek Checkpoint ALR 5 Women's - 2019	NULL
Trek Checkpoint ALR Frameset - 2019	NULL
Trek Precaliber 12 Girl's - 2018	NULL
Surly Krampus Frameset - 2018	NULL
Trek Checkpoint SL 5 Women's - 2019	NULL
Trek 820 - 2016	NULL
Trek Checkpoint ALR 4 Women's - 2019	NULL
Trek Kids' Dual Sport - 2018	NULL
Trek Checkpoint ALR 5 - 2019	NULL
Electra Sweet Ride 1 (20-inch) - Girl's - 2018	NULL
Trek Domane SLR 6 Disc Women's - 2018	NULL
Trek Checkpoint SL 6 - 2019	NULL
Electra Townie Original 7D EQ - Women's - 2016	1
Trek Remedy 29 Carbon Frameset - 2016	1
Surly Straggler - 2016	1
Electra Townie Original 7D EQ - 2016	1
Trek Fuel EX 8 29 - 2016	1

Do it your self?

```
SELECT
    product_name,
    order_id
FROM
    sales.order_items o
    RIGHT JOIN production.products p
        ON o.product_id = p.product_id
ORDER BY
    order_id;
```

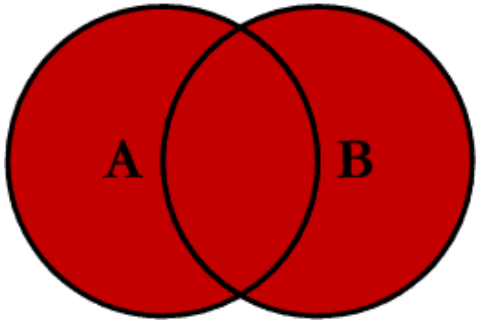
FULL OUTER JOIN

- Return all of the records from both tables, joining records from the left table (table A) that match records from the right table (table B)
- **Syntax**

```
SELECT col_names

FROM Table_A  A  FULL JOIN Table_B  B

ON A.Col1 = B.Col1
```



FULL OUTER JOIN (2/2)

- **Example:**

Customer

CustID	CustName	BirthDate	Country
1	Davolio Nancy	12/8/1968	Germany
2	Fuller Andrew	2/19/1952	Mexico
3	Leverling Janet	8/30/1963	Mexico

[Order]

OrderID	CustID	OrderDate	ShipperID
10308	2	2013-09-18	3
10309	3	2013-09-19	1
10310	77	2013-09-20	2

```
SELECT c.CustName, o.OrderID
FROM Customer c FULL JOIN [Order] o
ON c.CustID = o.CustID
ORDER BY c.CustName;
```

- **Result:**

Results | Messages

	CustName	OrderID
1	NULL	10310
2	Davolio Nancy	NULL
3	Fuller Andrew	10308
4	Leverling Janet	10309

Do it your self?

```
CREATE SCHEMA pm;
GO
```

```
CREATE TABLE pm.projects(
    id INT PRIMARY KEY IDENTITY,
    title VARCHAR(255) NOT NULL
);

CREATE TABLE pm.members(
    id INT PRIMARY KEY IDENTITY,
    name VARCHAR(120) NOT NULL,
    project_id INT,
    FOREIGN KEY (project_id)
        REFERENCES pm.projects(id)
);
```

Do it your self?

```
INSERT INTO
    pm.projects(title)
VALUES
    ('New CRM for Project Sales'),
    ('ERP Implementation'),
    ('Develop Mobile Sales Platform');


INSERT INTO
    pm.members(name, project_id)
VALUES
    ('John Doe', 1),
    ('Lily Bush', 1),
    ('Jane Doe', 2),
    ('Jack Daniel', null);
```

Do it your self?

id	title
1	New CRM for Project Sales
2	ERP Implementation
3	Develop Mobile Sales Platform

id	name	project_id
1	John Doe	1
2	Lily Bush	1
3	Jane Doe	2
4	Jack Daniel	NULL

Display information about **member and project working**

member	project
John Doe	New CRM for Project Sales
Lily Bush	New CRM for Project Sales
Jane Doe	ERP Implementation
Jack Daniel	NULL
NULL	Develop Mobile Sales Platform

Do it your self?

```
SELECT
    m.name member,
    p.title project
FROM
    pm.members m
    FULL OUTER JOIN pm.projects p
        ON p.id = m.project_id;
```

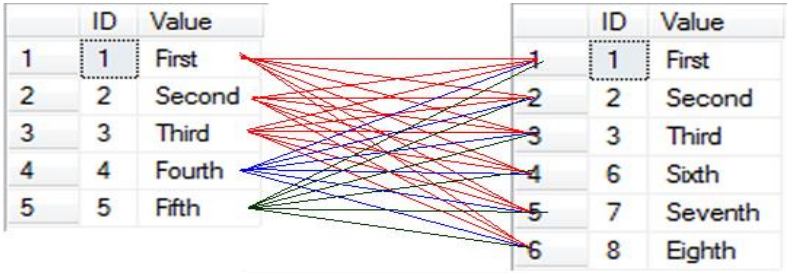
Section 3

CROSS JOIN

CROSS JOIN

- Return records that are multiplication of record number from both the tables
  - ✓ No need any condition to join
- **Syntax**

```
SELECT col_names
FROM Table_A  A  CROSS JOIN Table_B  B
```

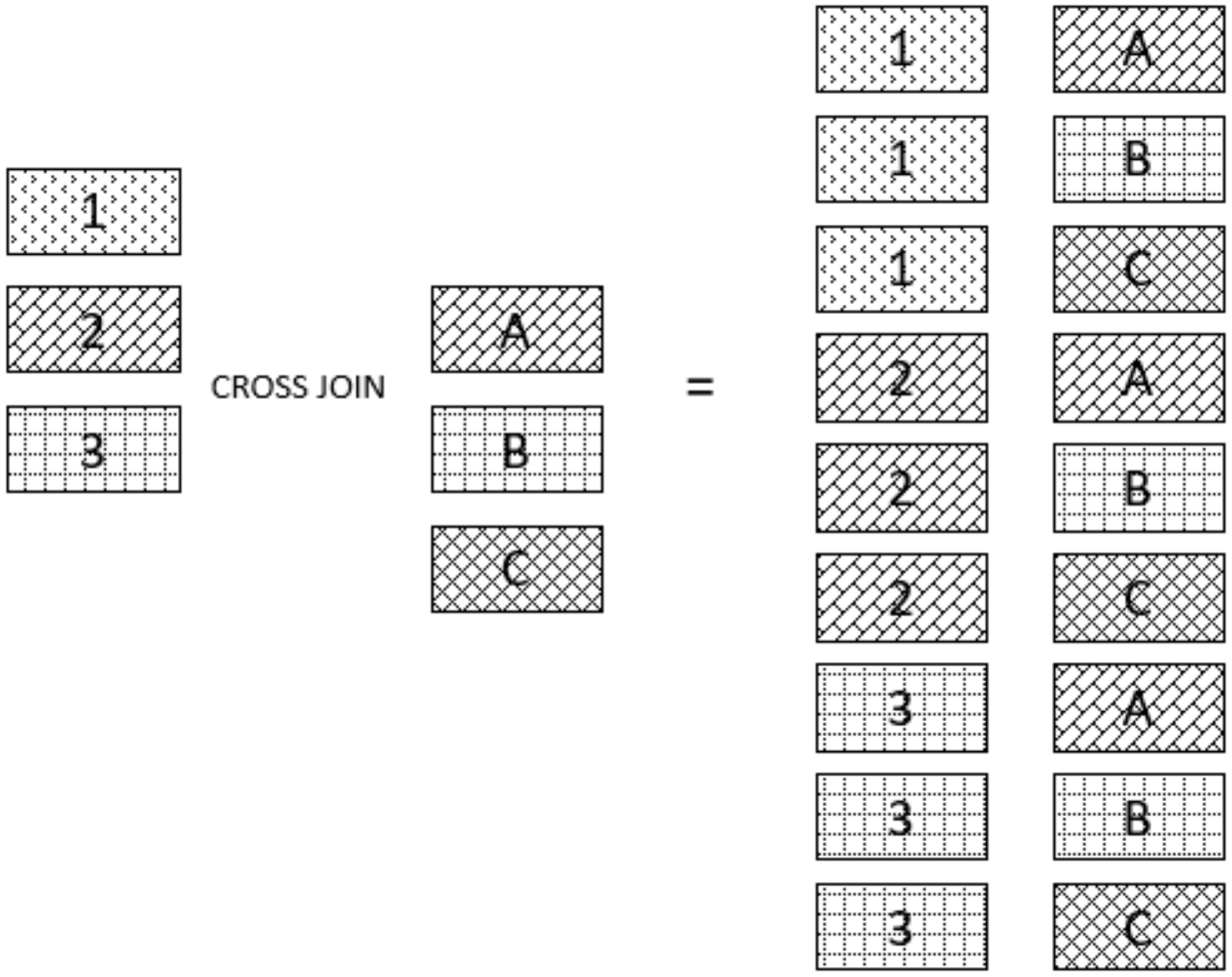


	ID	Value
1	1	First
2	2	Second
3	3	Third
4	4	Fourth
5	5	Fifth

	ID	Value
1	1	First
2	2	Second
3	3	Third
4	6	Sixth
5	7	Seventh
6	8	Eighth

	ID	Value	ID	Value
1	1	First	1	First
2	1	First	2	Second
3	1	First	3	Third
4	1	First	6	Sixth
5	1	First	7	Seventh
6	1	First	8	Eighth
7	2	Second	1	First
8	2	Second	2	Second
9	2	Second	3	Third
10	2	Second	6	Sixth
11	2	Second	7	Seventh
12	2	Second	8	Eighth

CROSS JOIN



CROSS JOIN

Example:

Customer

CustID	CustName	BirthDate	Country
1	Davolio Nancy	12/8/1968	Germany
2	Fuller Andrew	2/19/1952	Mexico
3	Leverling Janet	8/30/1963	Mexico

[Order]

OrderID	CustID	OrderDate	ShipperID
10308	2	2013-09-18	3
10309	3	2013-09-19	1
10310	77	2013-09-20	2

```
SELECT c.CustName, o.OrderID
FROM Customer c CROSS JOIN [Order] o
```

Result:

Results | Messages

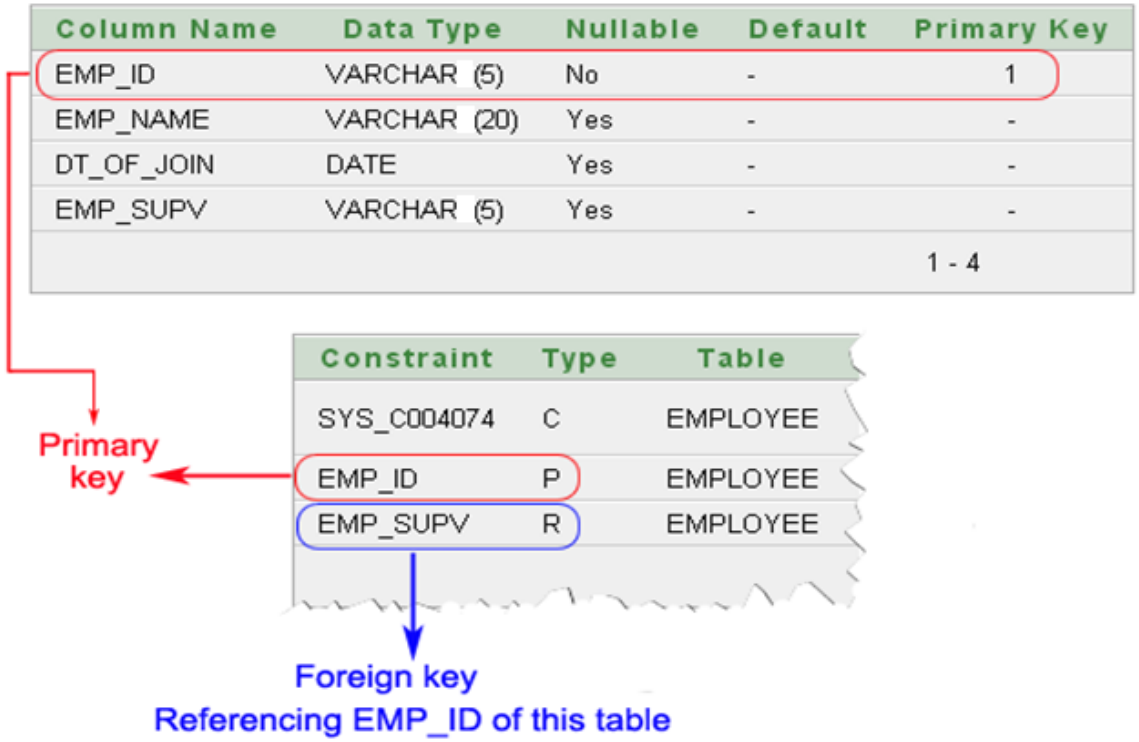
	CustName	OrderID
1	Davolio Nancy	10308
2	Fuller Andrew	10308
3	Leverling Janet	10308
4	Davolio Nancy	10309
5	Fuller Andrew	10309
6	Leverling Janet	10309
7	Davolio Nancy	10310
8	Fuller Andrew	10310
9	Leverling Janet	10310

Section 4

SELF JOIN

Self JOIN

- A SELF JOIN is a join of a table to itself. In SELF JOIN, we can use:
  - ✓ **INNER JOIN**
  - ✓ **OUTER JOIN**
  - ✓ **CROSS JOIN**



Self JOIN

❖ **Example:**

Results | Messages

	EMP_ID	EMP_NAME	DT_OF_JOIN	EMP_SUPV
1	10120	Hansen Ola	2013-01-01	NULL
2	10121	Svendson Tove	2013-02-01	10120
3	10122	Pettersen Kari	2013-03-01	10120
4	10123	Alfreds Futterkiste	2013-04-01	10121

```
SELECT   e1.EMP_NAME AS Employee_Name,
         e2.EMP_NAME AS Manager_Name
FROM Employee e1 LEFT JOIN Employee e2
ON e1.EMP_SUPV = e2.EMP_ID
```

❖ **Result:**

Results | Messages

	Employee_Name	Manager_Name
1	Hansen Ola	NULL
2	Svendson Tove	Hansen Ola
3	Pettersen Kari	Hansen Ola
4	Alfreds Futterkiste	Svendson Tove

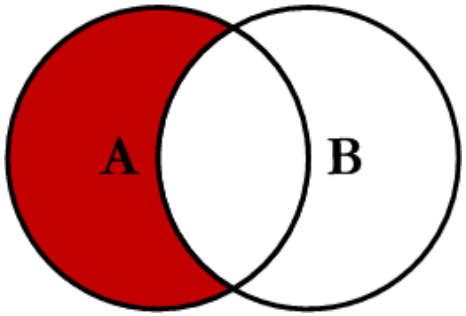
Section 5

EXCLUDING JOIN

LEFT Excluding JOIN

- Return all of the records in the left table (table A) that do not match any records in the right table (table B)
- **Syntax**

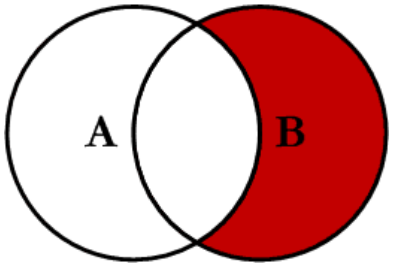
```
SELECT col_names
FROM Table_A  A  LEFT JOIN Table_B  B
ON A.Col1 = B.Col1
WHERE B.Col1 IS NULL
```



RIGHT Excluding JOIN

- Returns records in the right table (table B) that do not match any records in the left table (table A)
  - ✓ In the results where there is no matching condition, the row contains NULL values for the right table's columns.
- **Syntax**

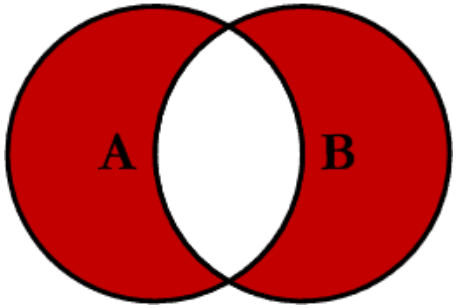
```
SELECT col_names
FROM Table_A  A  RIGHT JOIN Table_B  B
ON A.Col1 = B.Col1
WHERE A.Col1 IS NULL
```



OUTER JOIN EXCLUDING JOIN

- Return all of the records in the left table (table A) and all of the records in the right table (table B) that do not match
- **Syntax**

```
SELECT col_names
FROM Table_A  A
FULL OUTER JOIN Table_B  B
ON A.Col1 = B.Col1
WHERE A.Col1 IS NULL OR B.Col1 IS NULL
```

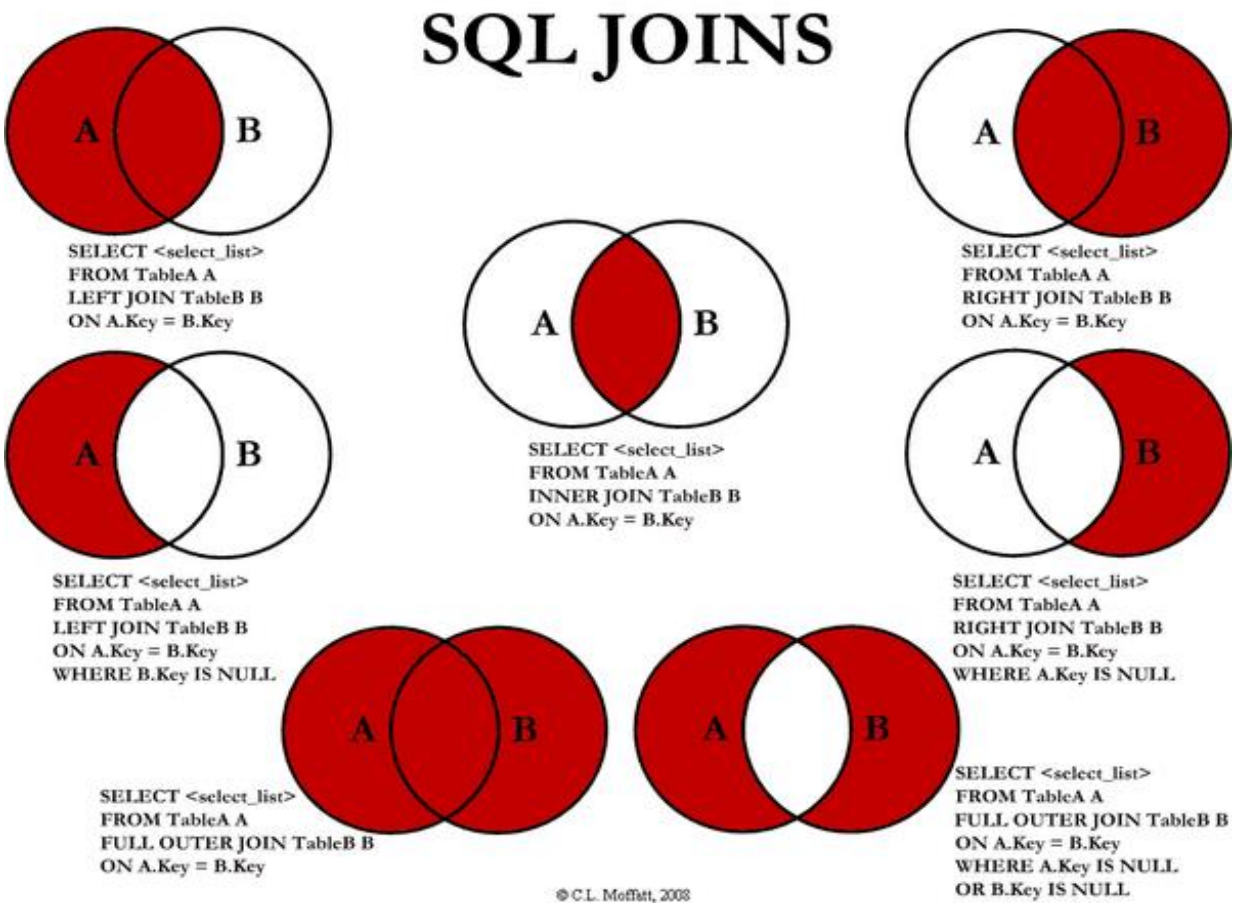


Joining Three or More Tables

- Since FROM clauses can contain multiple join specifications, this allows many tables to be joined for a single query.
- **Syntax**

```
SELECT col_names
FROM Table_A  A  JOIN Table_B  B
ON A.Col1 = B.Col1 LEFT JOIN Table_C C
ON B.Col2 = C.Col2
....
```

SUMMARY


SQL JOINS
A
B
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
SELECT <select_list>
FROM TableA A
INNER JOIN TableB B
ON A.Key = B.Key
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
WHERE B.Key IS NULL
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
OR B.Key IS NULL
© C.L. Moffatt, 2008

Q&A
Thank you