

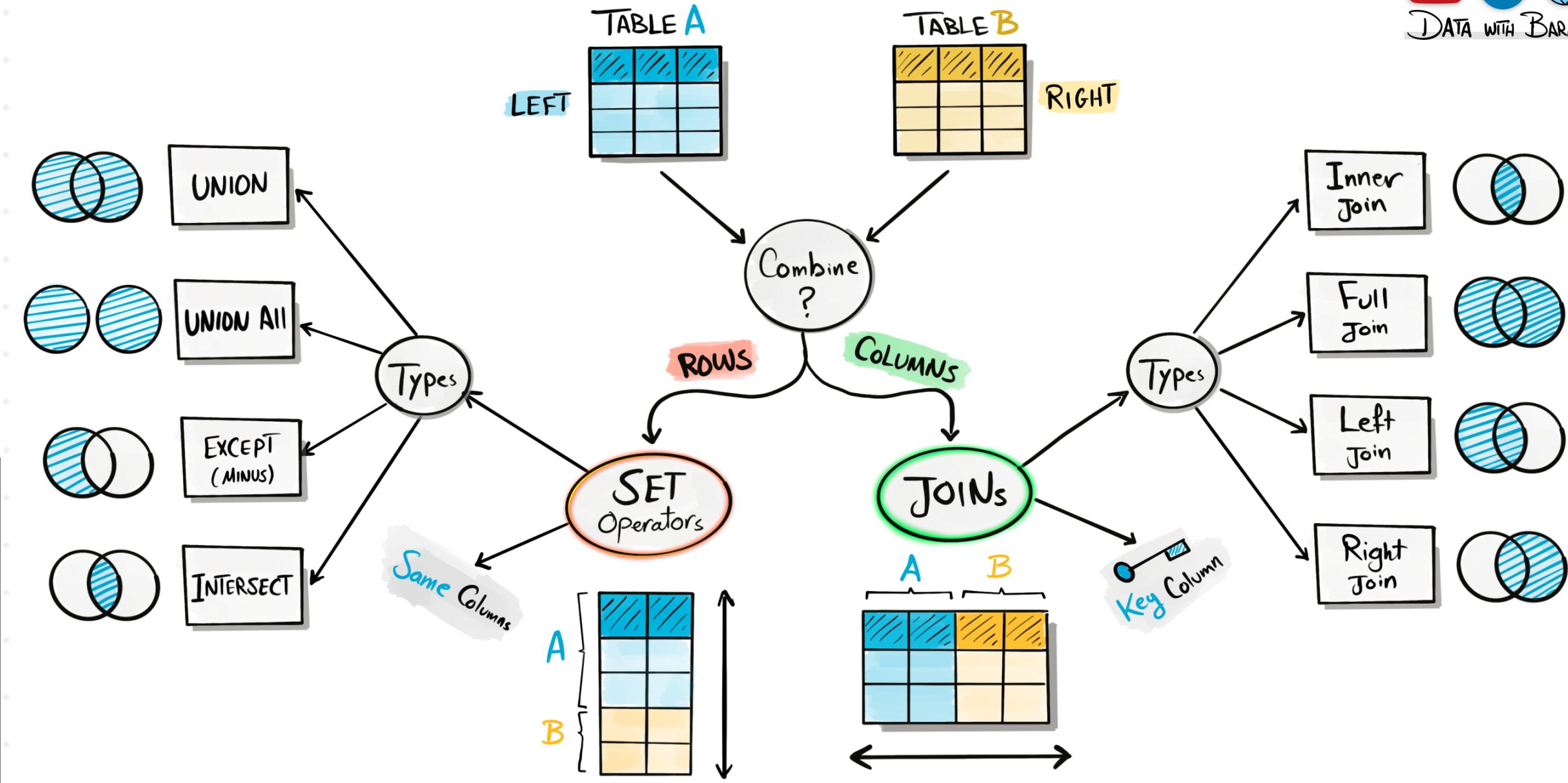


# SQL JOINS

## Combining Data

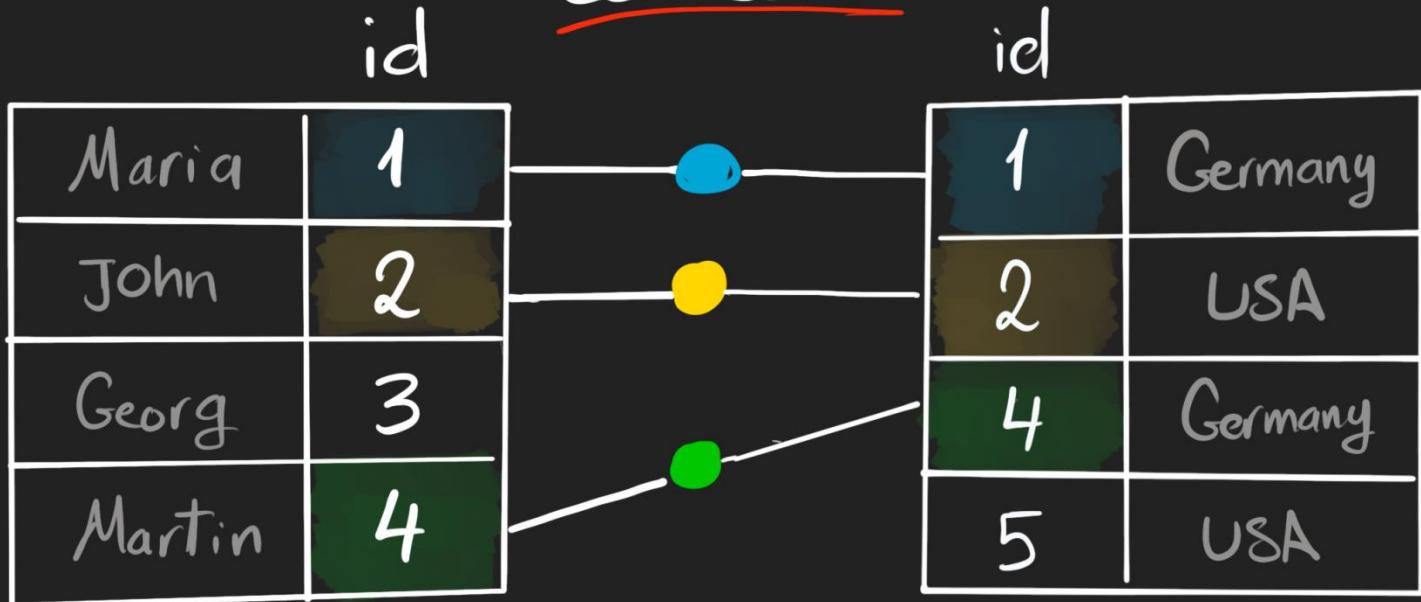
Baraa Khatib Salkini  
YouTube | **DATA WITH BARAA**  
SQL Course | JOINS





# What is SQL JOIN ?

Connected



Combined!

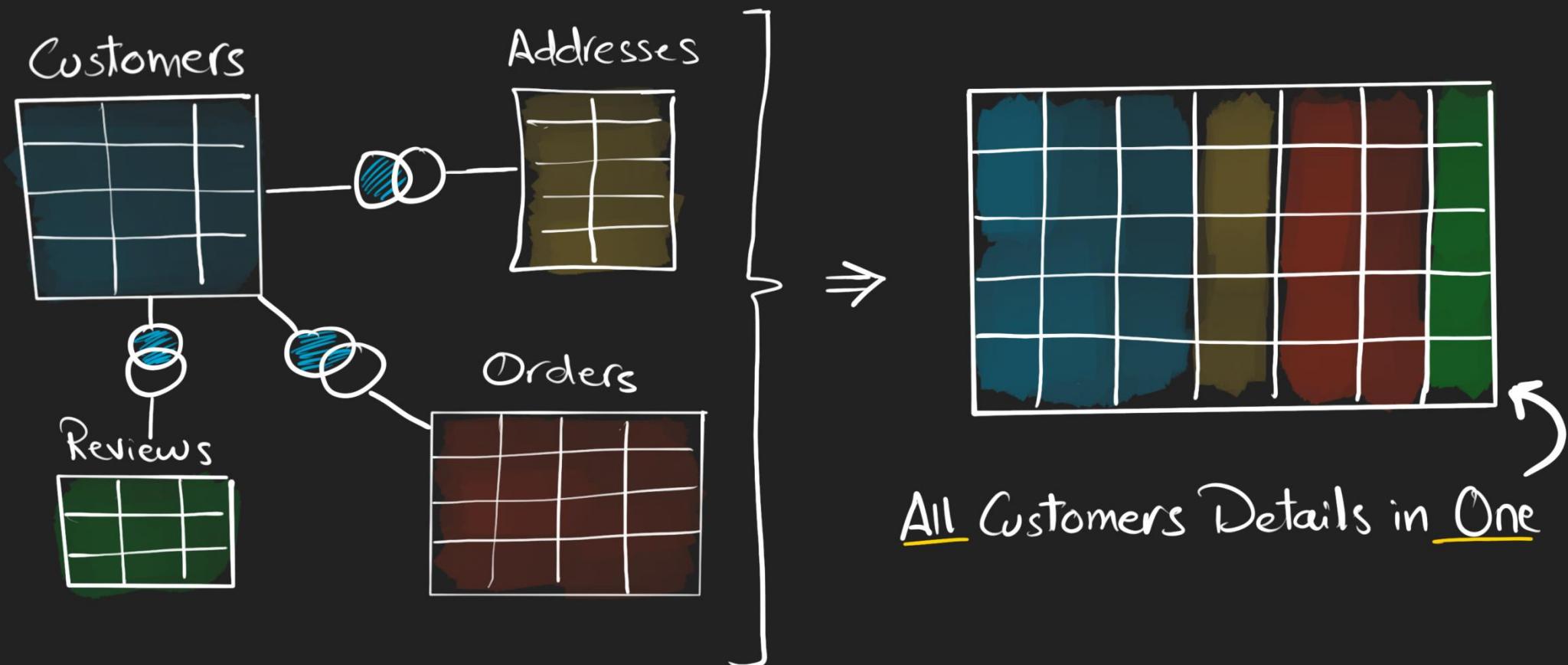


1	Maria	Germany
2	John	USA
4	Martin	Germany

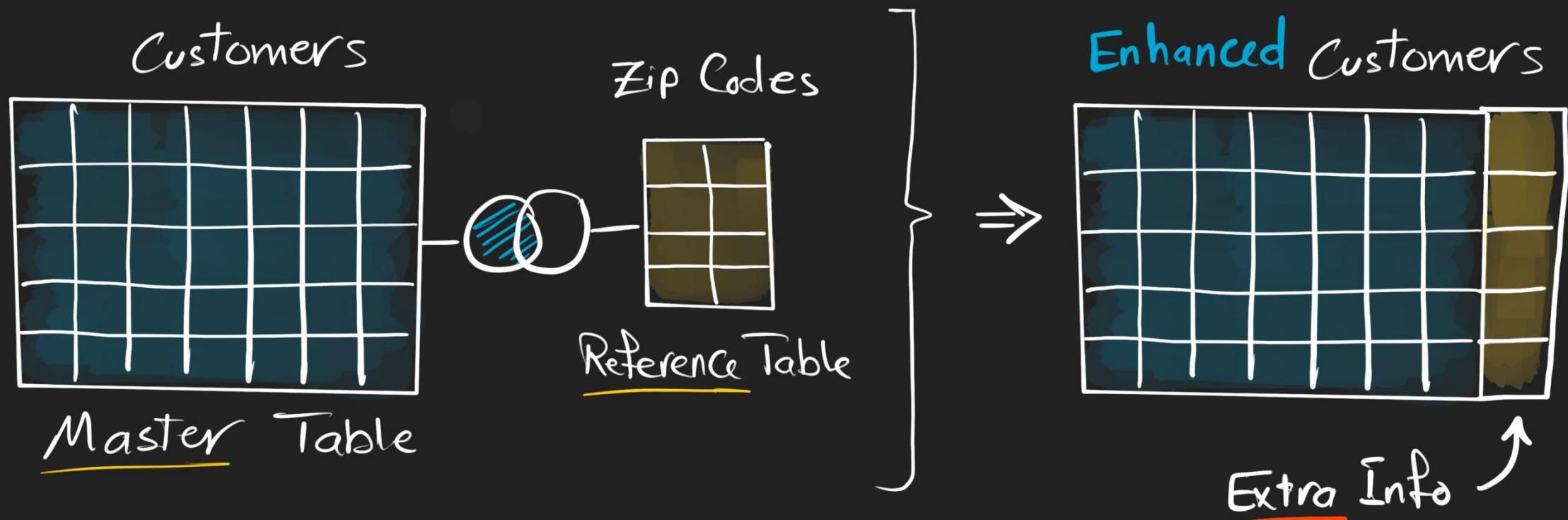
Query  


# 1) ReCombine Data

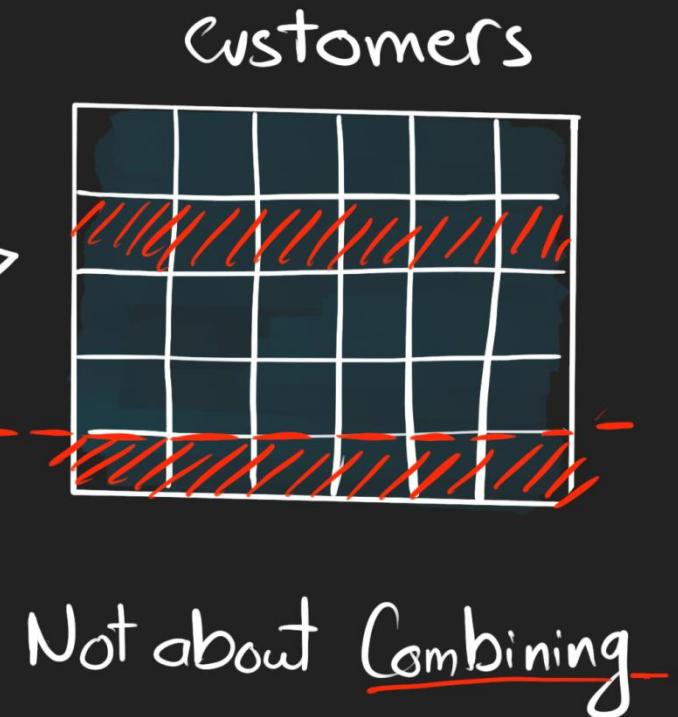
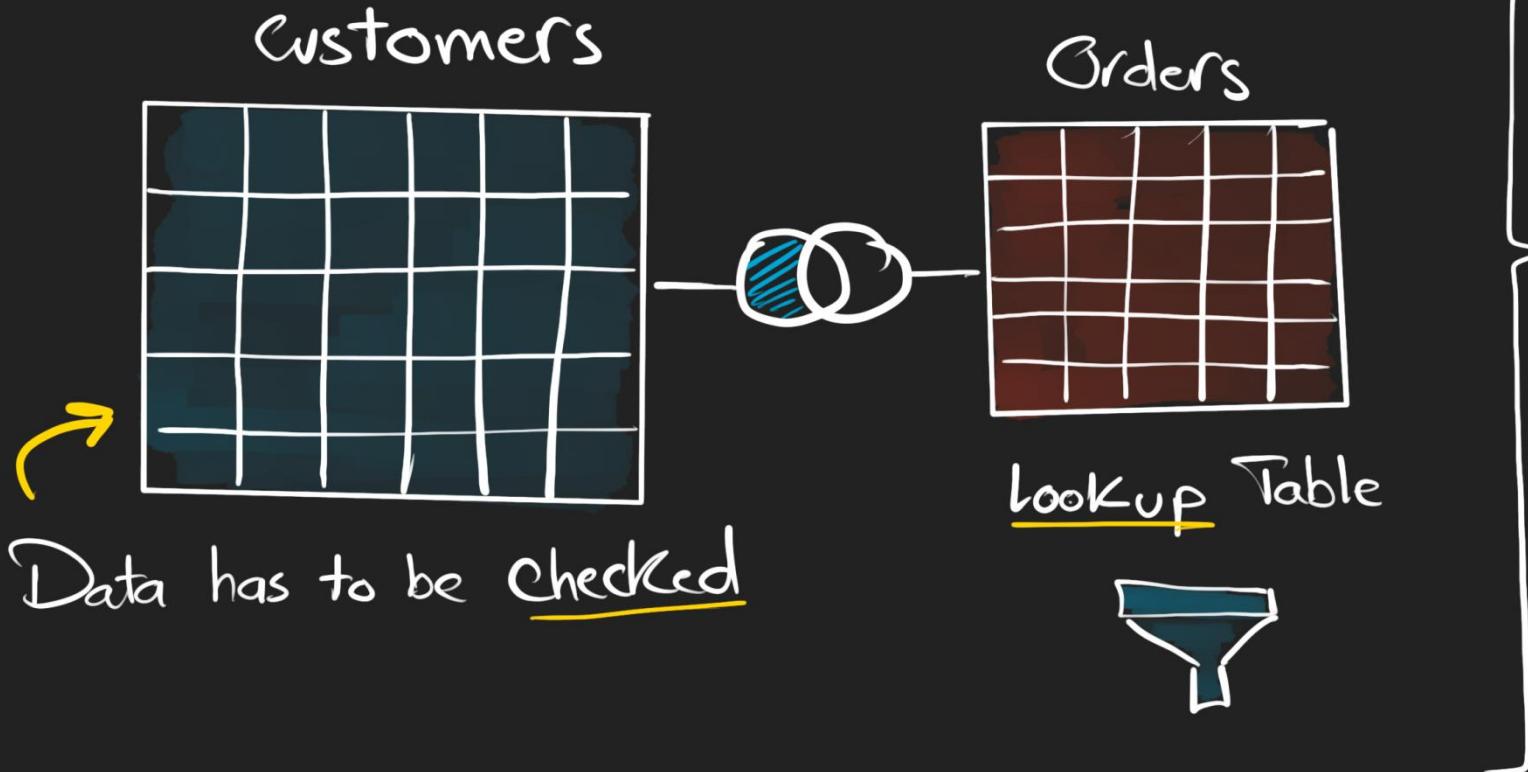
Complete Big Picture!



## 2 Data Enrichment "Getting Extra Data"

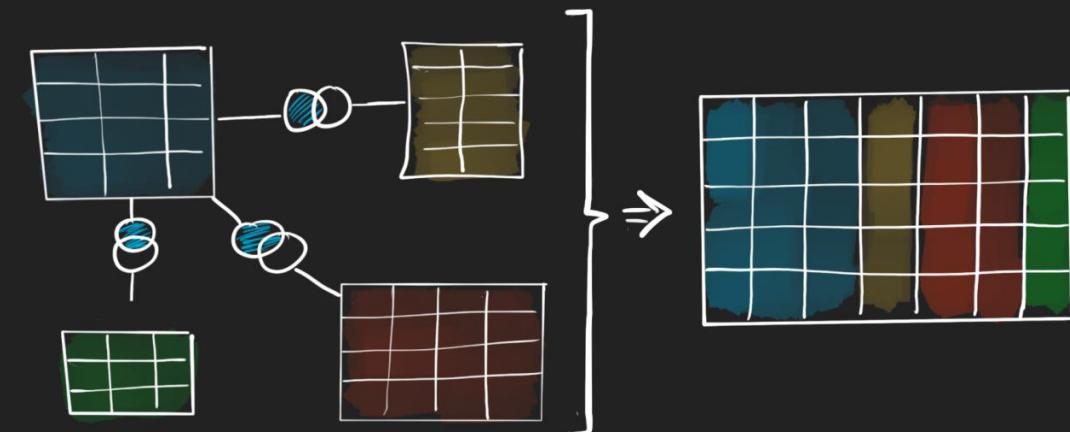


### 3 Check for Existence "Filtering"



## 1 ReCombine Data

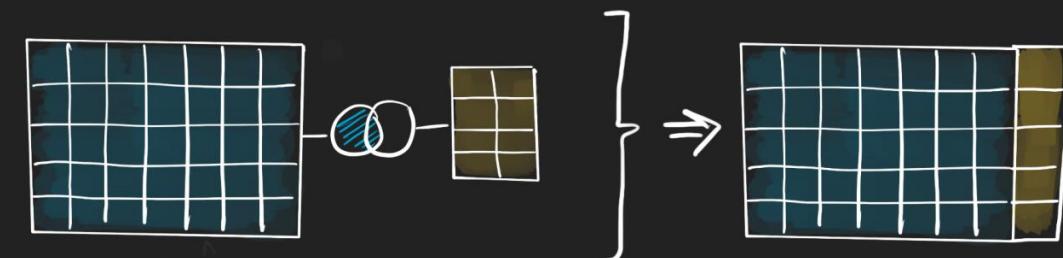
~Big Picture~



INNER  
LEFT  
FULL

## 2 Data Enrichment

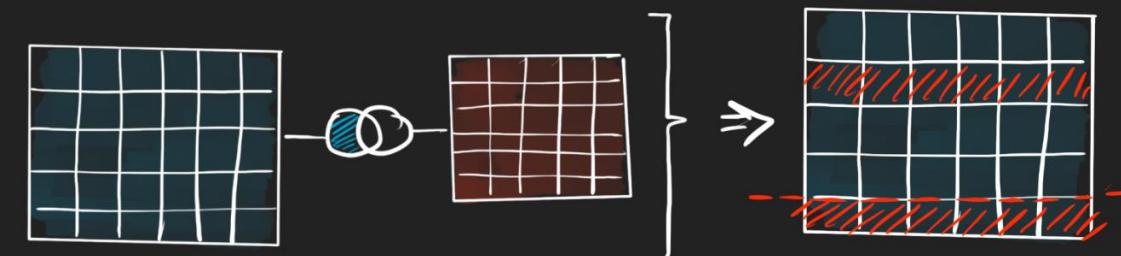
~Extra Info~



LEFT

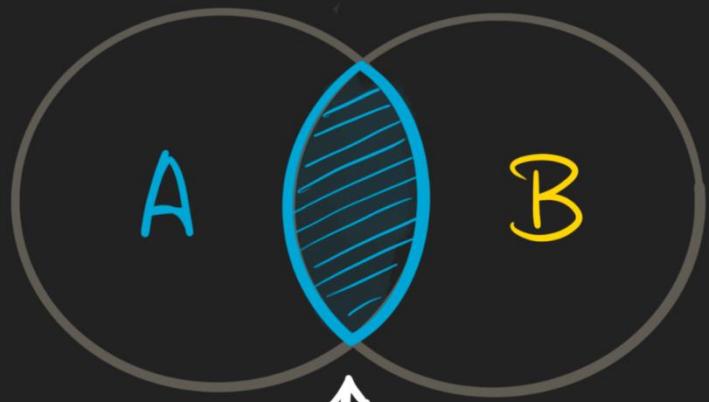
## 3 Check Existence

~Filtering~



INNER  
LEFT + WHERE  
FULL + WHERE

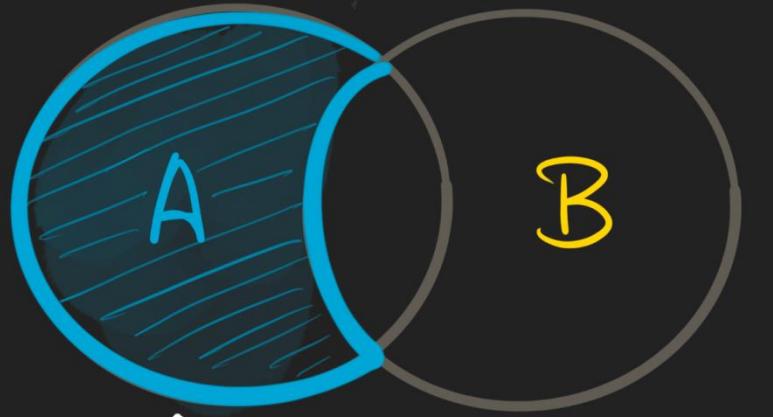
## Joins Possibilities



Matching Data

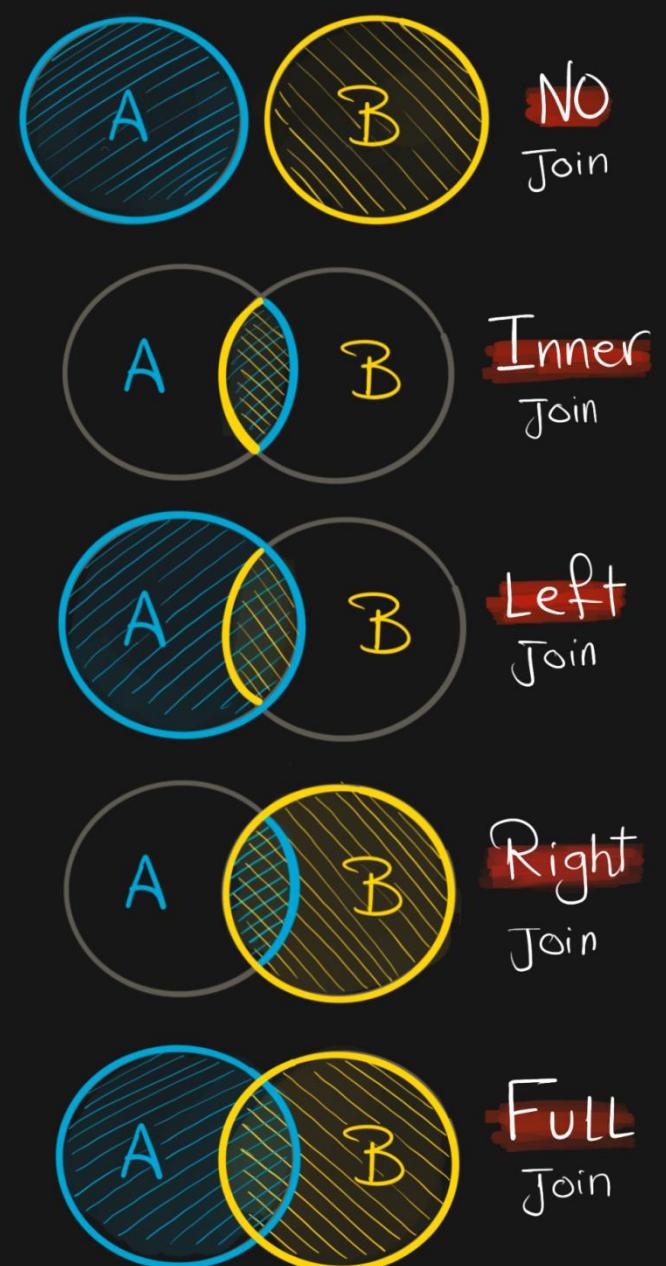


All Data



UnMatching Data

## Basic Join Types



## No JOIN

Returns Data from Tables without Combining Them



SELECT \*  
FROM A;

SELECT \*  
FROM B;

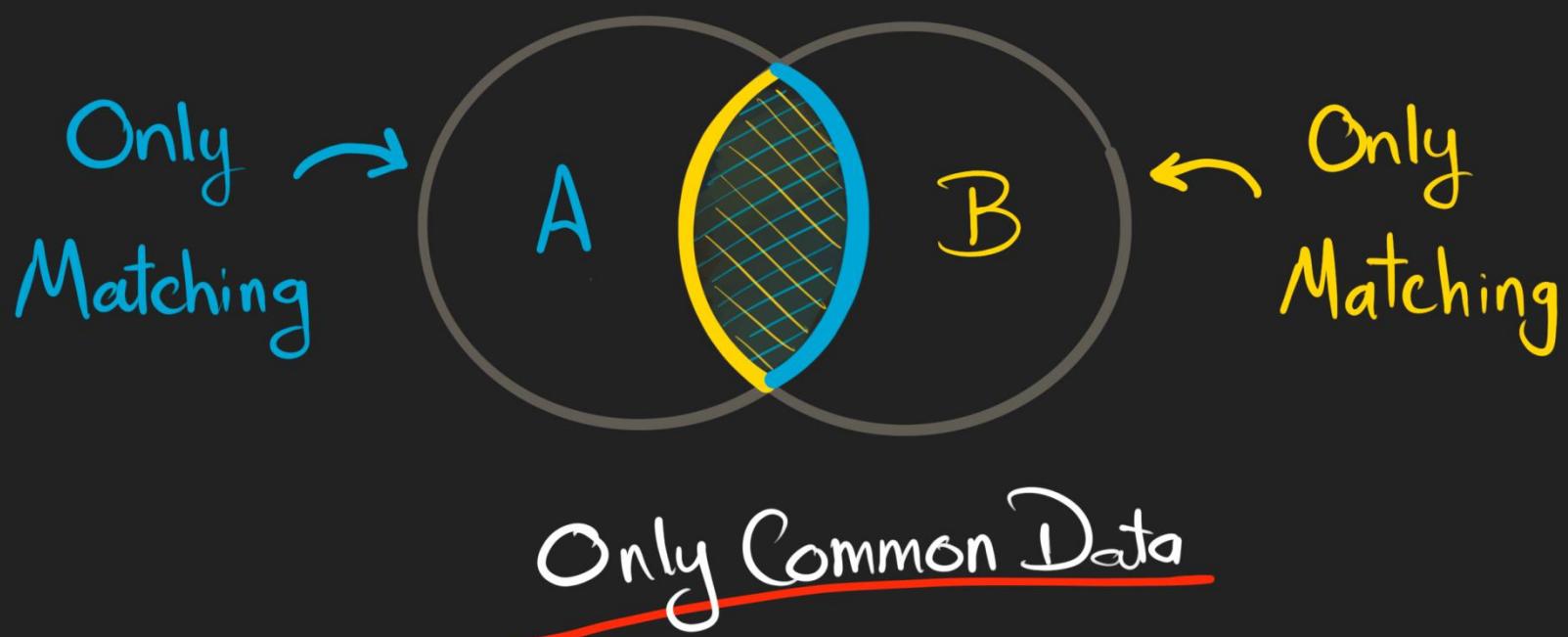
Two Results No Need To Combine

# INNER JOIN

Returns Only Matching Rows from both Tables

default join is inner join.

Use alias of table and in row name use table name also for understanding and clarity.



The Order of Tables  
Doesn't Matters

SELECT \*

FROM A 

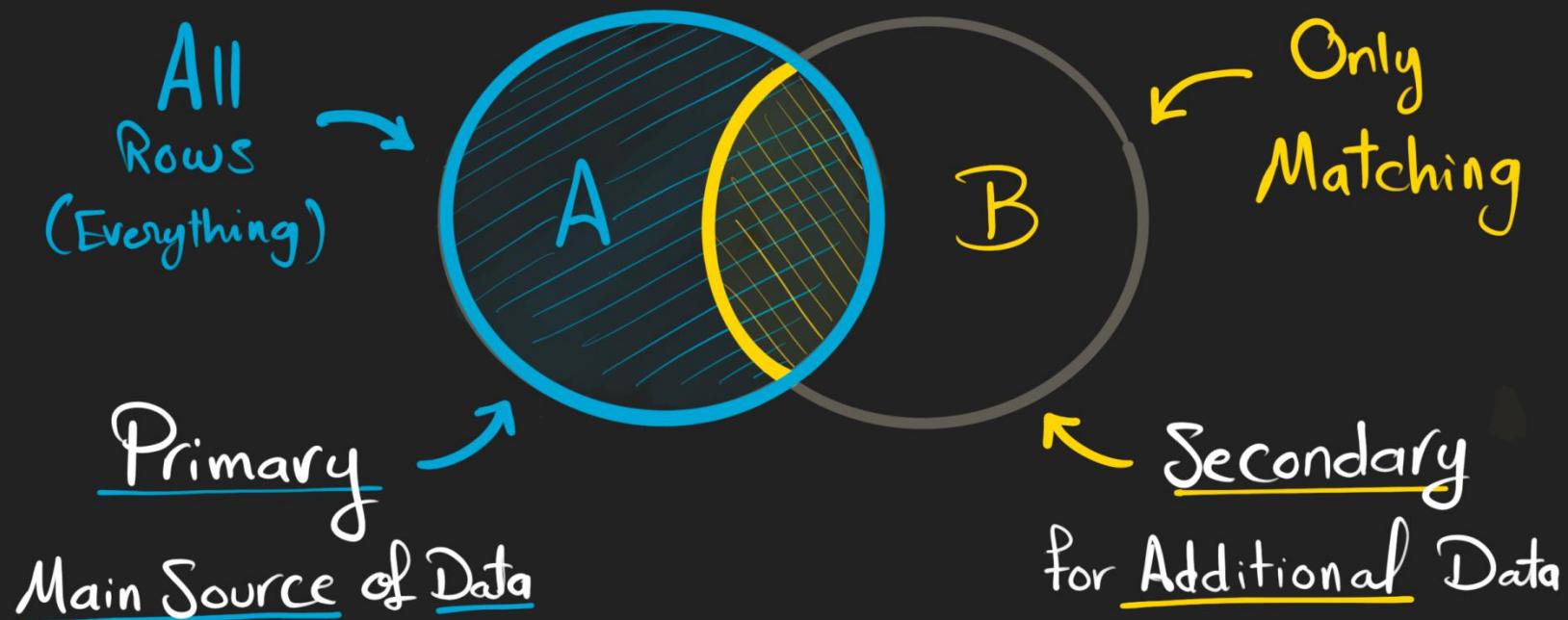
INNER JOIN B

ON A.Key=B.Key

How to Match  
Rows ???

## LEFT JOIN

Returns All rows from Left and Only Matching from Right



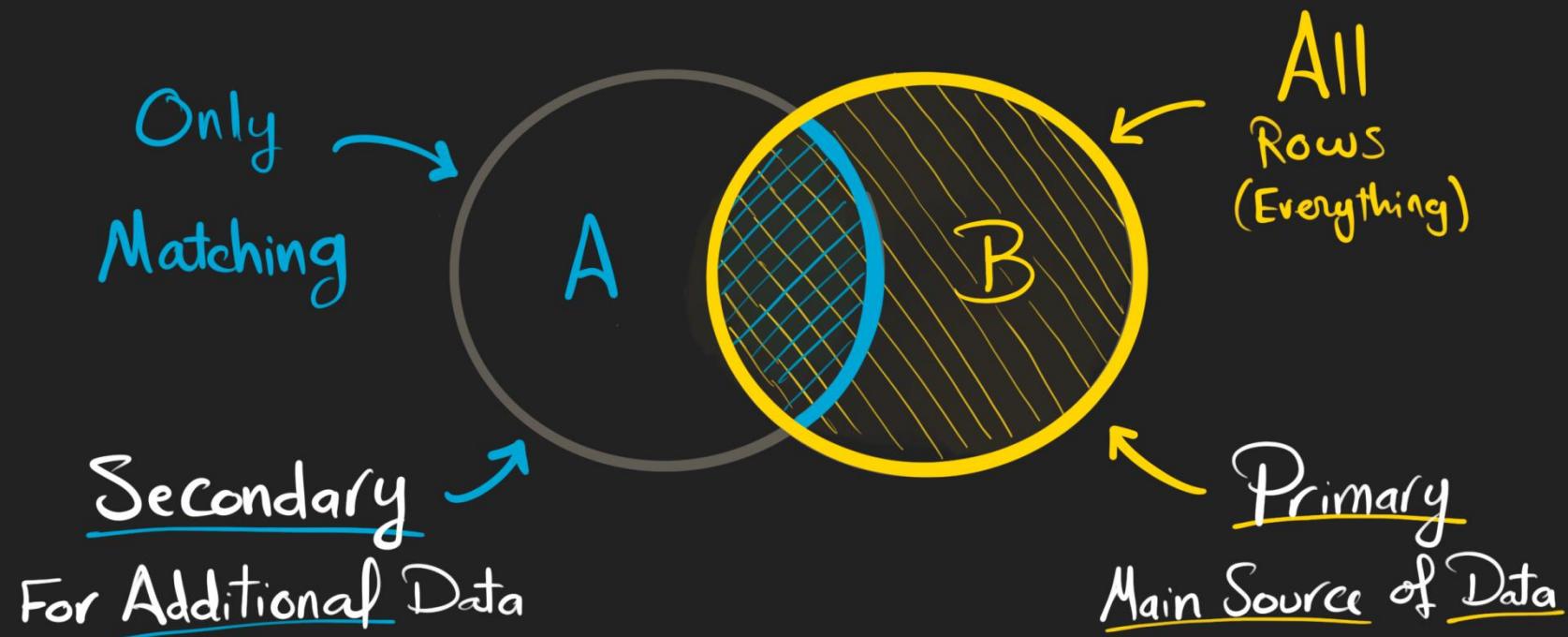
RIGHT/LEFT JOIN can show NULLs even if columns have NOT NULL constraint, because join output is not the physical table.

The Order of Tables is IMPORTANT

`SELECT *` ← Left  
`FROM A` ← Right  
`LEFT JOIN B` ←  
`ON A.Key = B.Key`

## RIGHT JOIN

Returns All Rows from Right and Only Matching from Left



The Order of Tables  
Is IMPORTANT

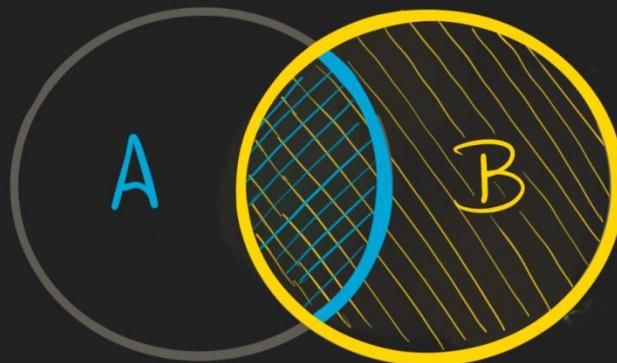
SELECT \*

FROM A

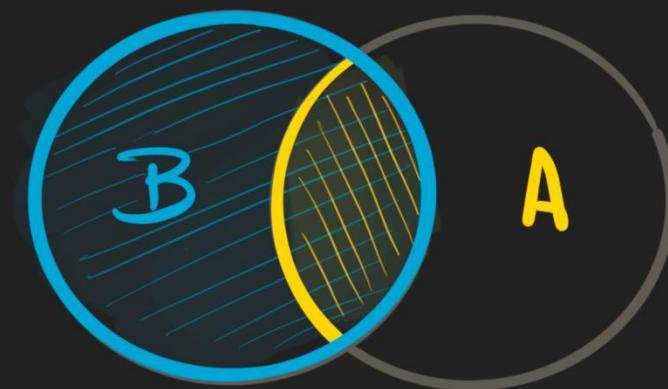
RIGHT JOIN B

ON A.Key = B.Key

## Alternative To RIGHT JOIN



*Same  
Results*



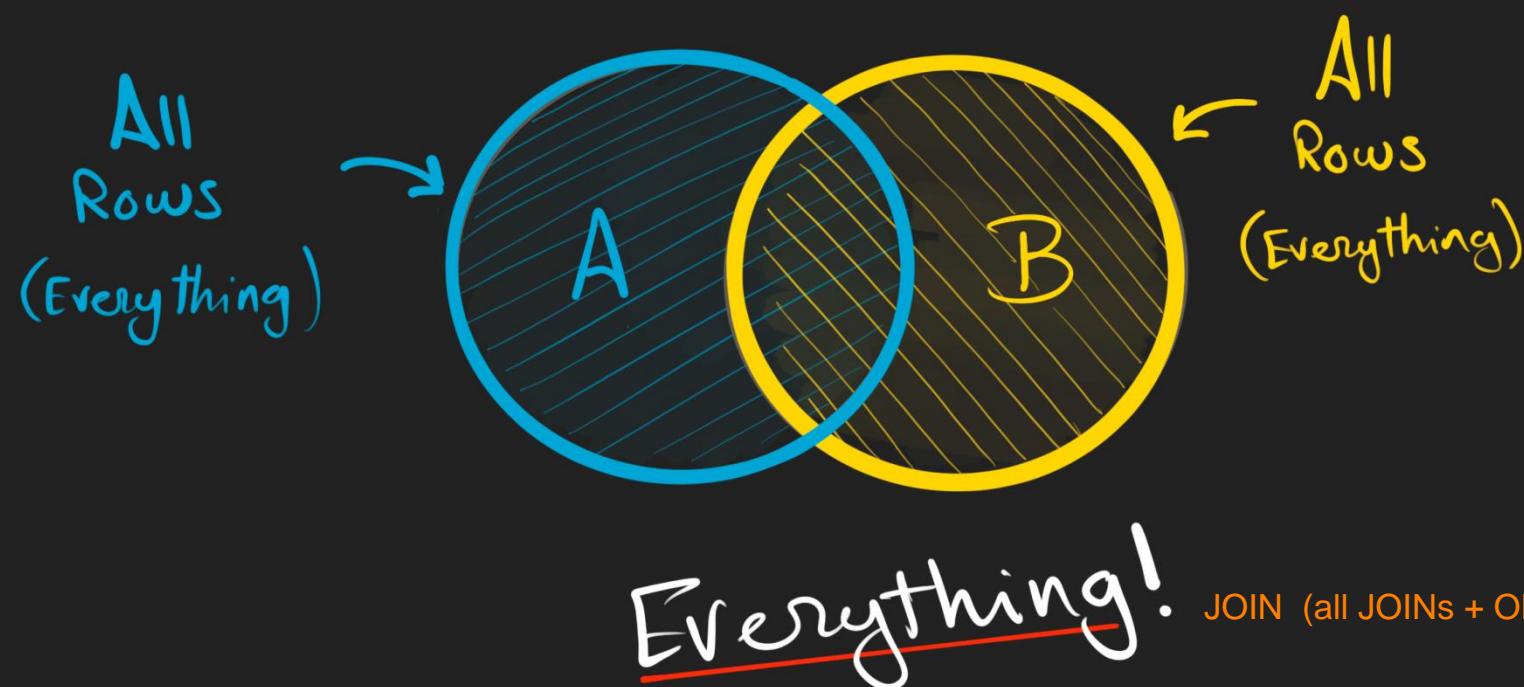
SELECT \*  
FROM A  
RIGHT JOIN B  
ON A.Key = B.Key

Alternative  
⇒

SELECT \*  
FROM B   
LEFT JOIN A  
ON A.Key = B.Key

# FULL JOIN

Returns All Rows from Both Tables



The Order of Tables  
Doesn't Matter

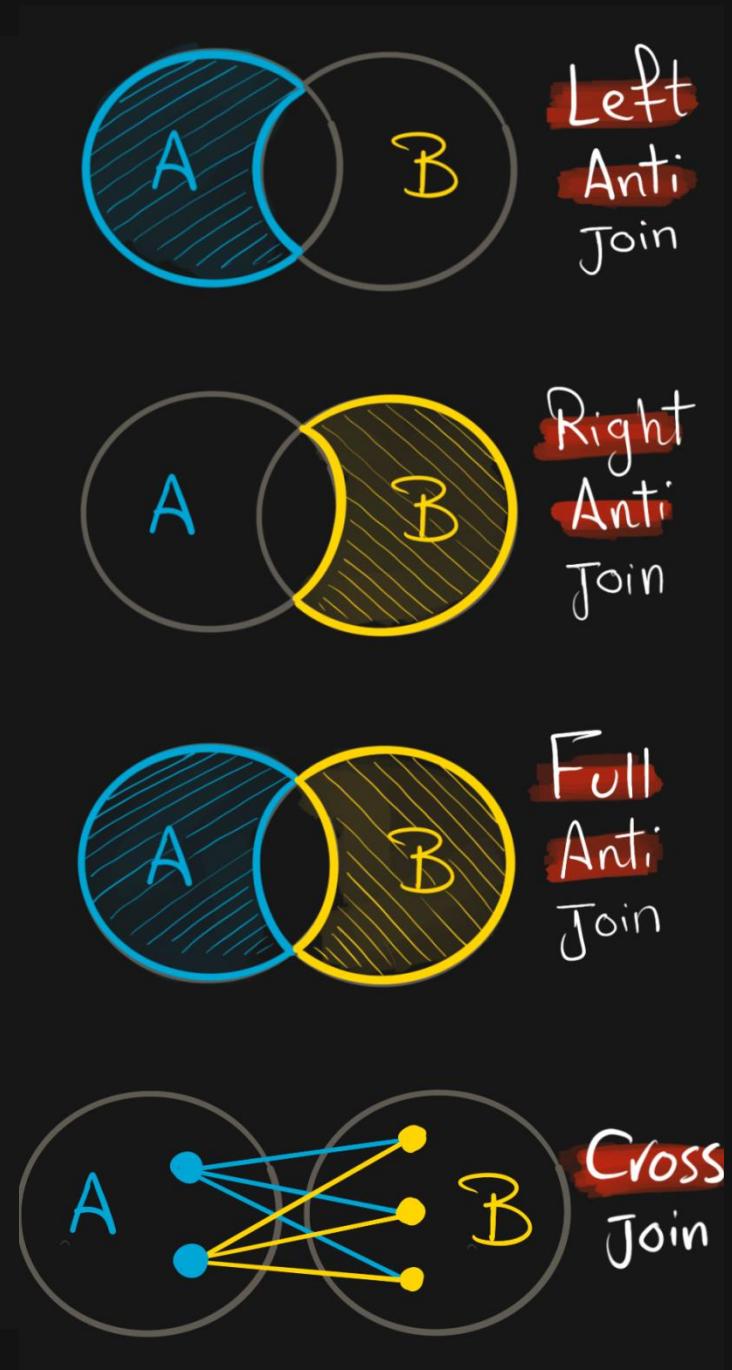
SELECT \*

FROM A

FULL JOIN B

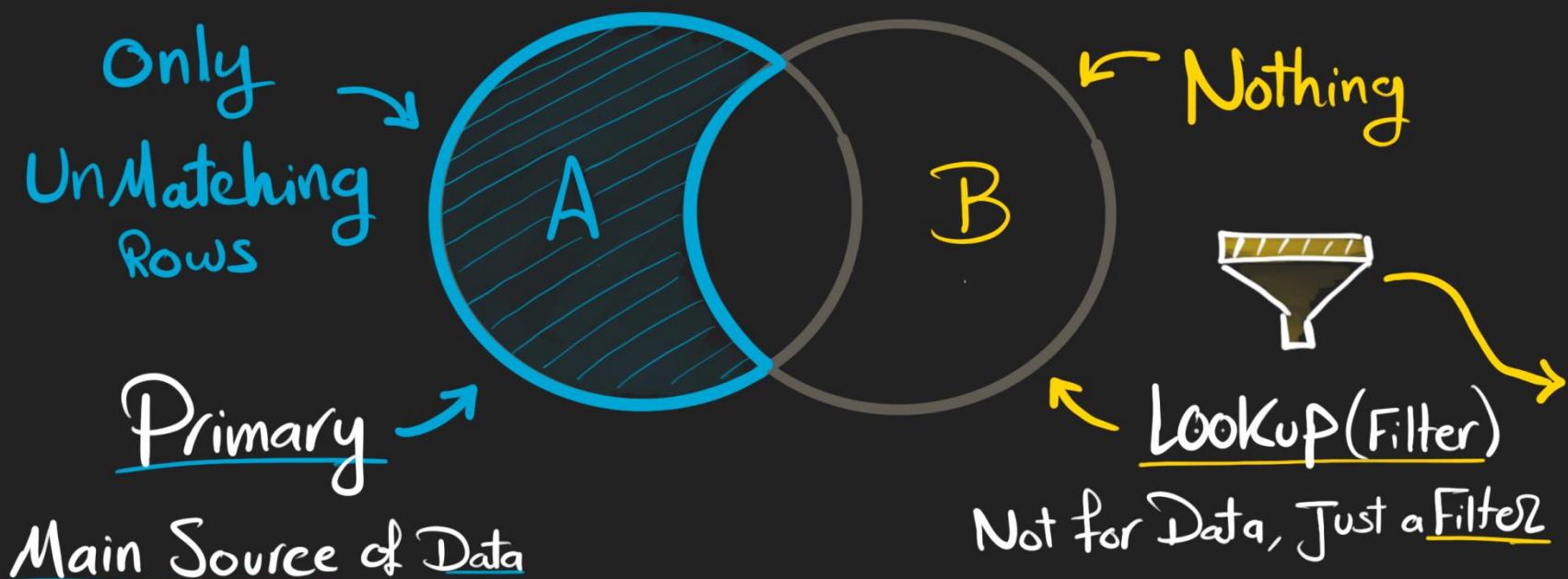
ON A.Key = B.Key

## Advanced Join Types



## LEFT ANTI JOIN

Returns Row from Left that has NO MATCH in Right



The Order of Tables Is IMPORTANT

SELECT \*

FROM A

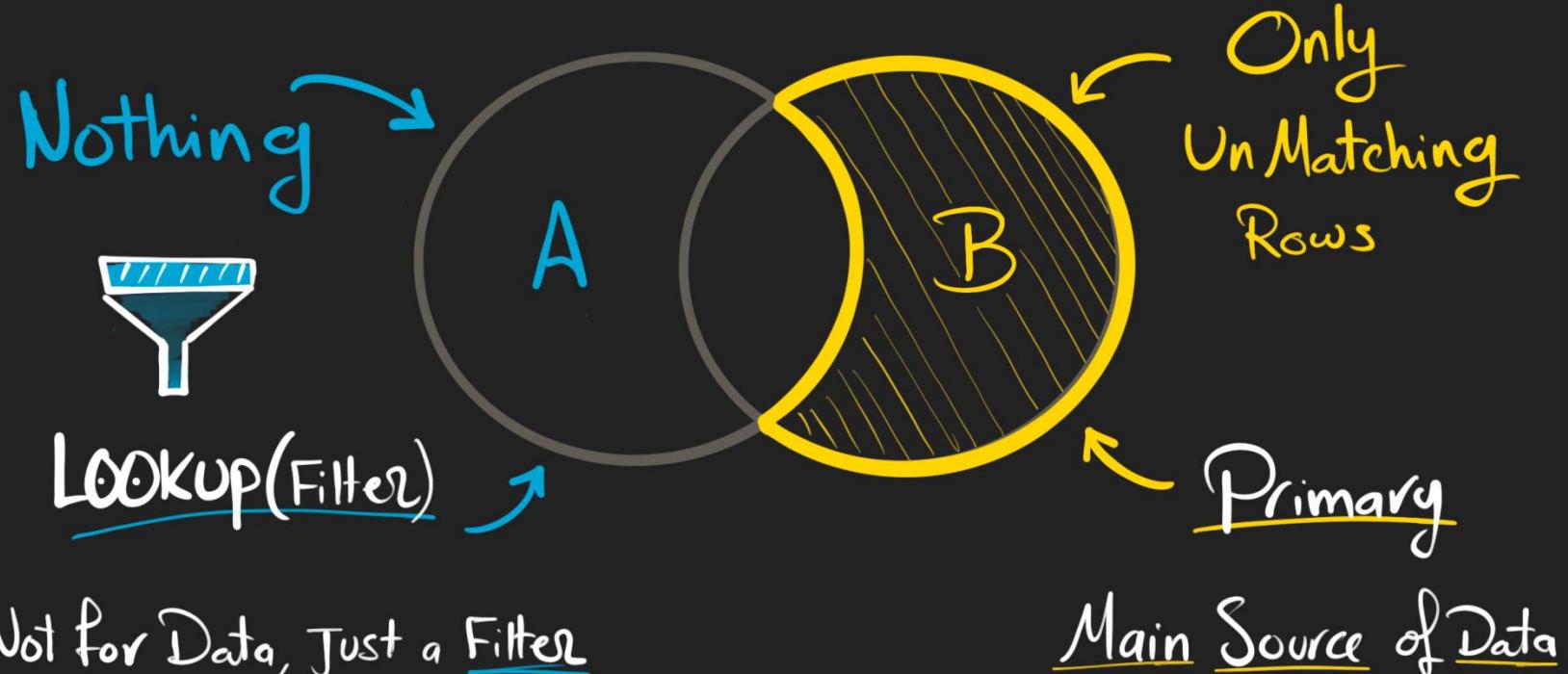
LEFT JOIN B

ON A.Key = B.Key

WHERE B.Key IS NULL

## RIGHT ANTI JOIN

Returns Rows from Right that has No Match in Left



The Order of Tables  
Is IMPORTANT

SELECT \*

FROM A

RIGHT JOIN B

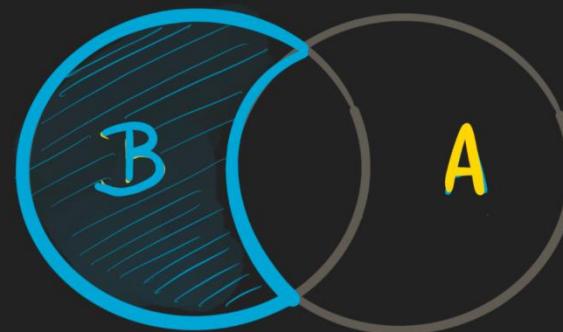
ON A.Key = B.Key

WHERE A.Key IS NULL

## Alternative To RIGHT Anti JOIN



~~Same  
Results~~



SELECT \*  
FROM A  
RIGHT JOIN B  
ON A.Key = B.Key  
WHERE A.Key IS NULL

Alternative  
⇒

SELECT \*  
FROM B   
LEFT JOIN A  
ON A.Key = B.Key  
WHERE A.Key IS NULL

## FULL ANTI JOIN

Returns Only Rows that Don't Match in either Tables



Only UnMatching Rows



Only UnMatching Rows



Only Unmatching Data

The Order of Tables  
Doesn't Matter

SELECT \*  
FROM A

FULL JOIN B

ON A.Key = B.Key

WHERE

B.Key IS NULL

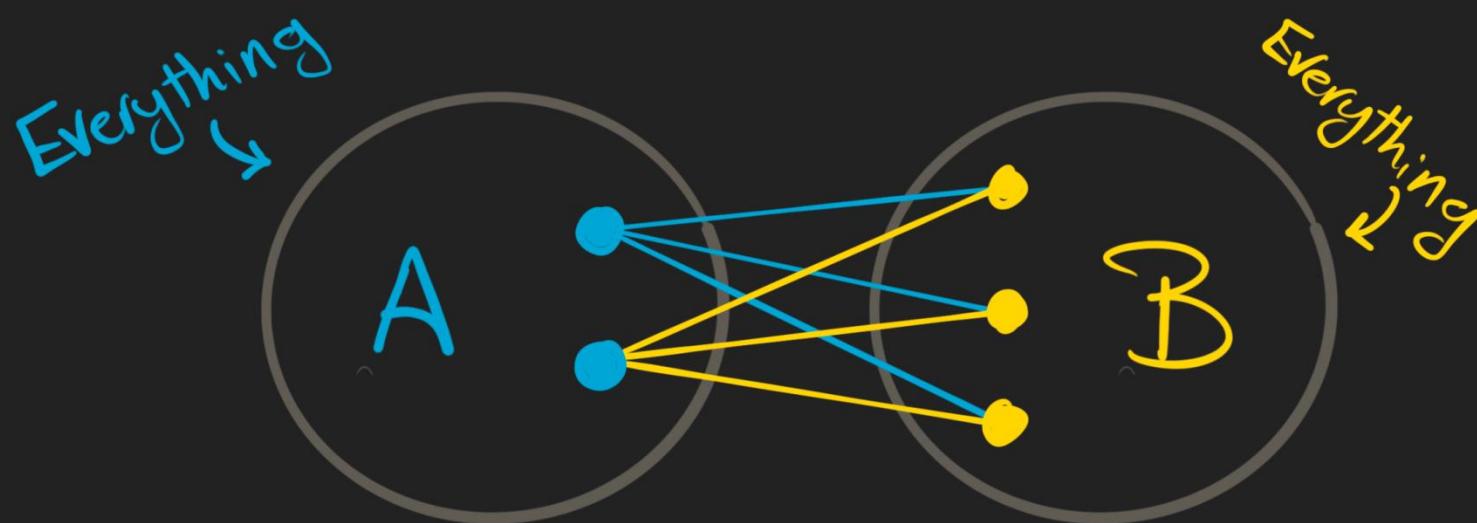
OR

A.Key IS NULL

## CROSS JOIN

Combines Every Row from Left with Every Row from Right

All Possible Combinations - Cartesian Join -



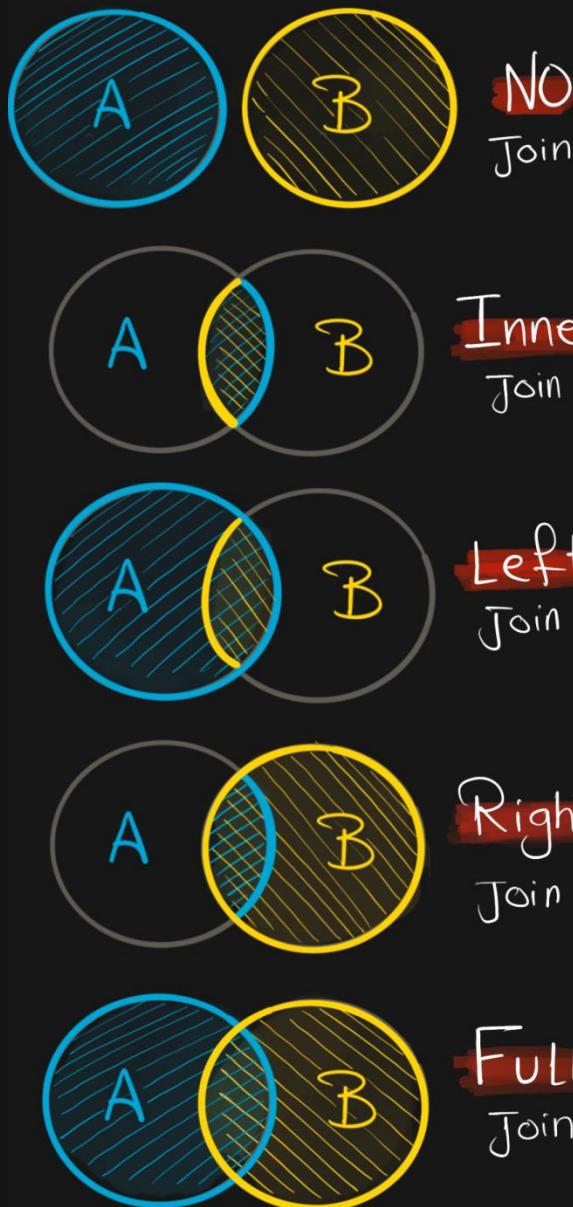
$$2 \times 3 = 6 \leftarrow \text{Total Rows}$$

The Order of Table Doesn't Matter

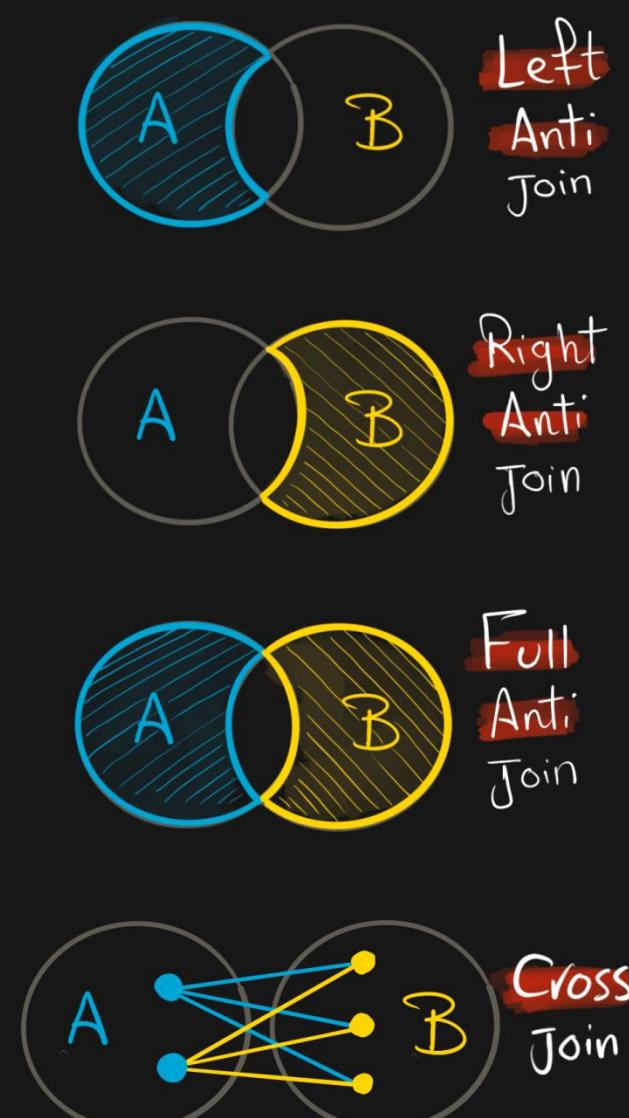
`SELECT *`  
`FROM A`   
`CROSS JOIN B`

 No Condition  
Needed

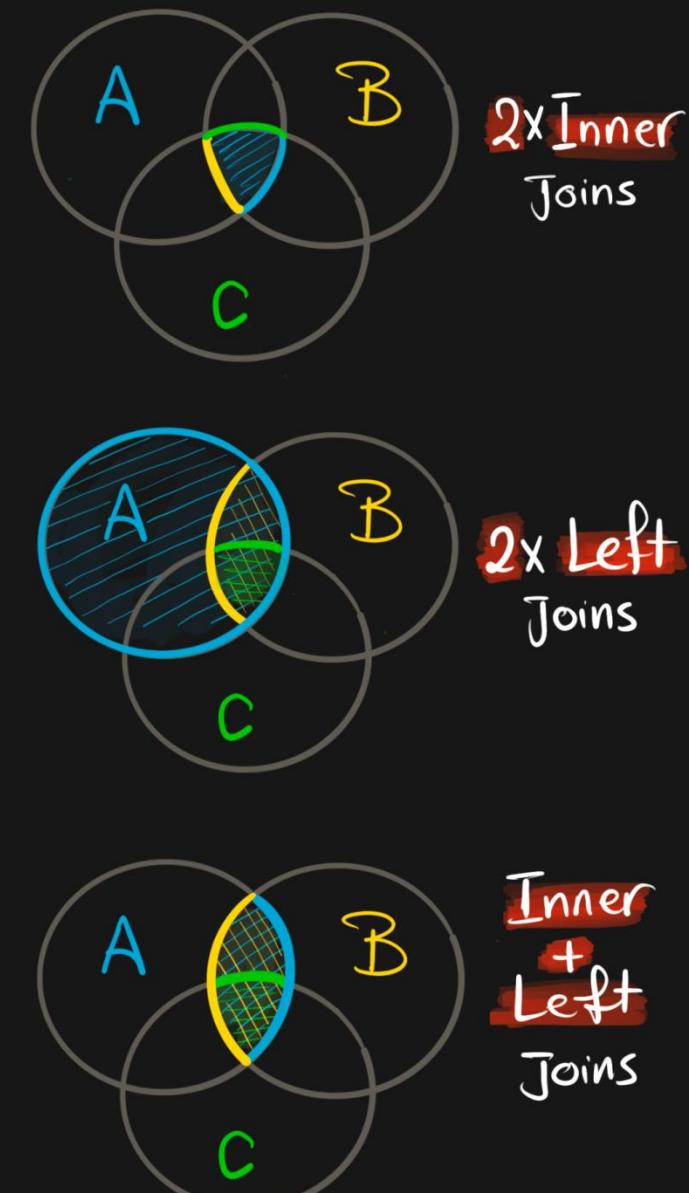
## Basics



## Advanced



## Multi - Tables



## How I Join Multiple Tables

SELECT \*

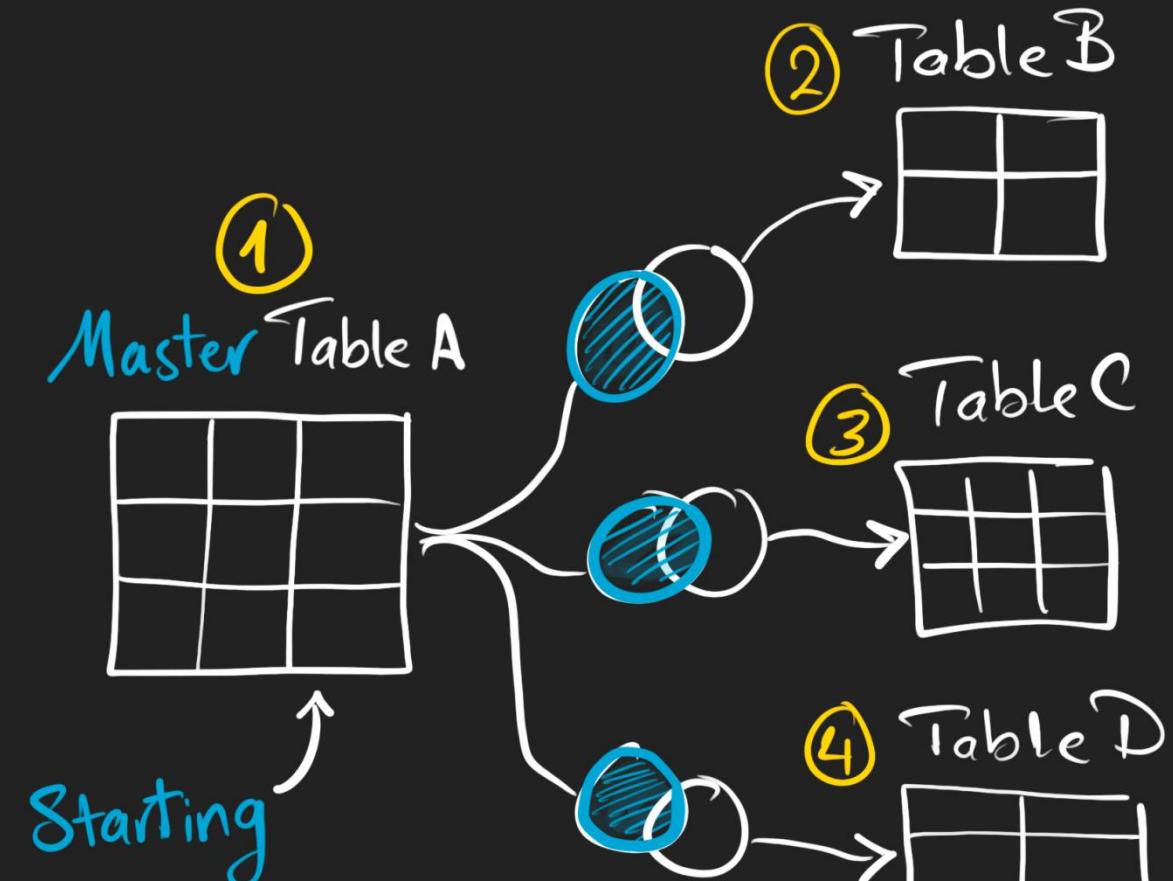
FROM A

LEFT B ON...

LEFT C ON...

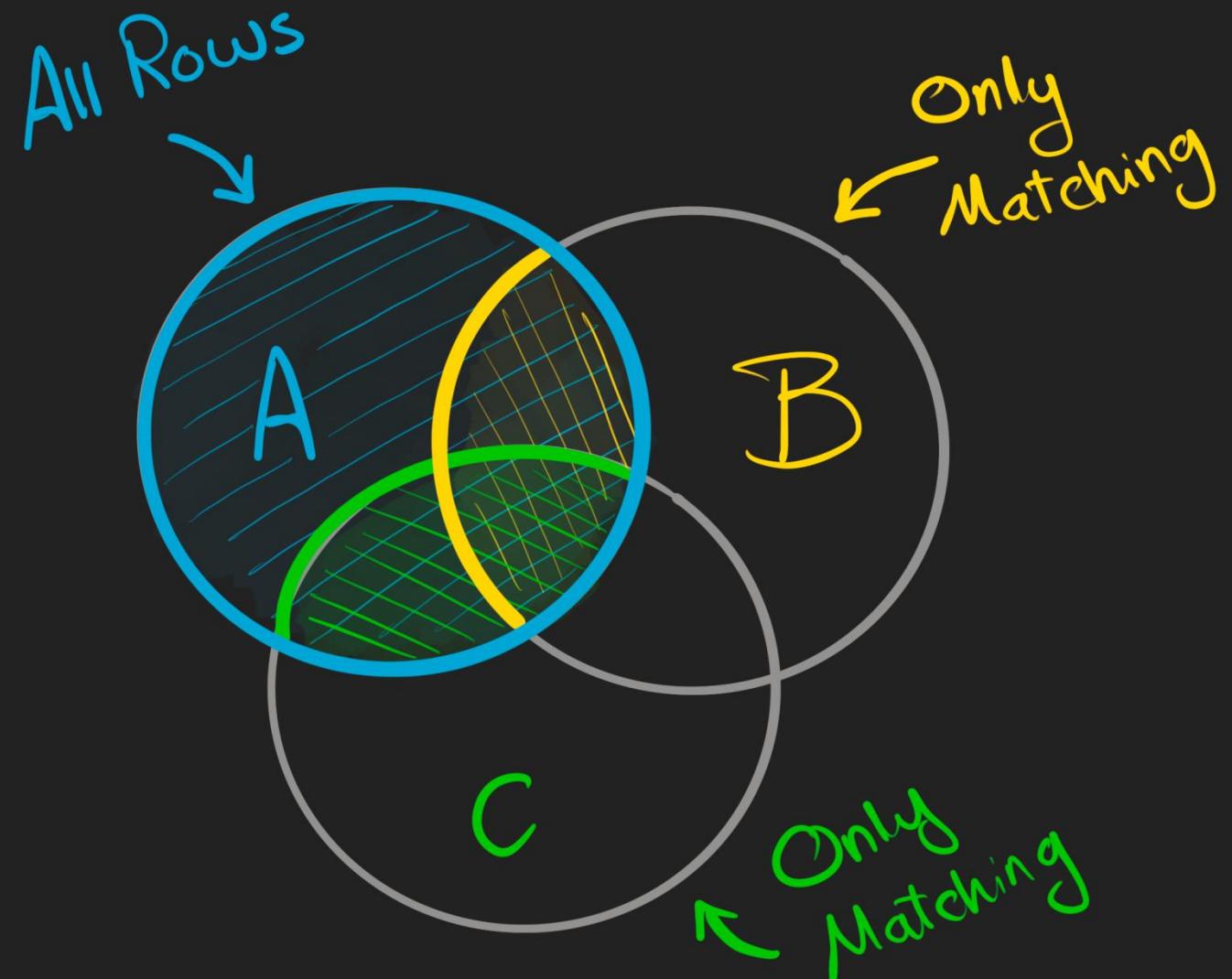
LEFT D ON...

WHERE Control what  
to keep



## How I Join Multiple Tables

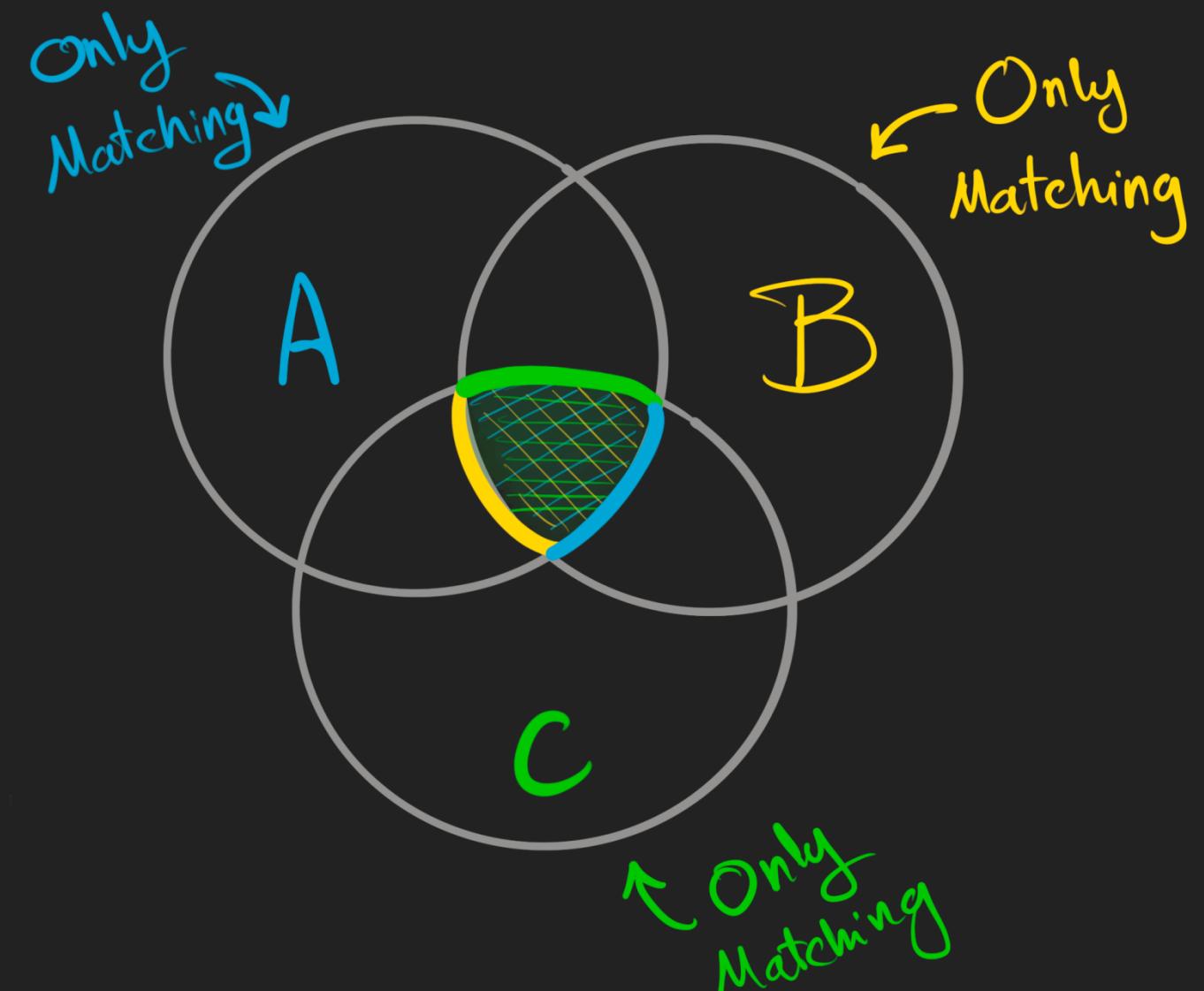
```
SELECT *  
FROM A  
LEFT B ON...  
LEFT C ON...  
LEFT D ON...  
WHERE Control what  
      to keep
```



## Inner Join Multiple Tables

```
SELECT *  
FROM A  
INNER B ON...  
INNER C ON...
```

:





# SET Operators

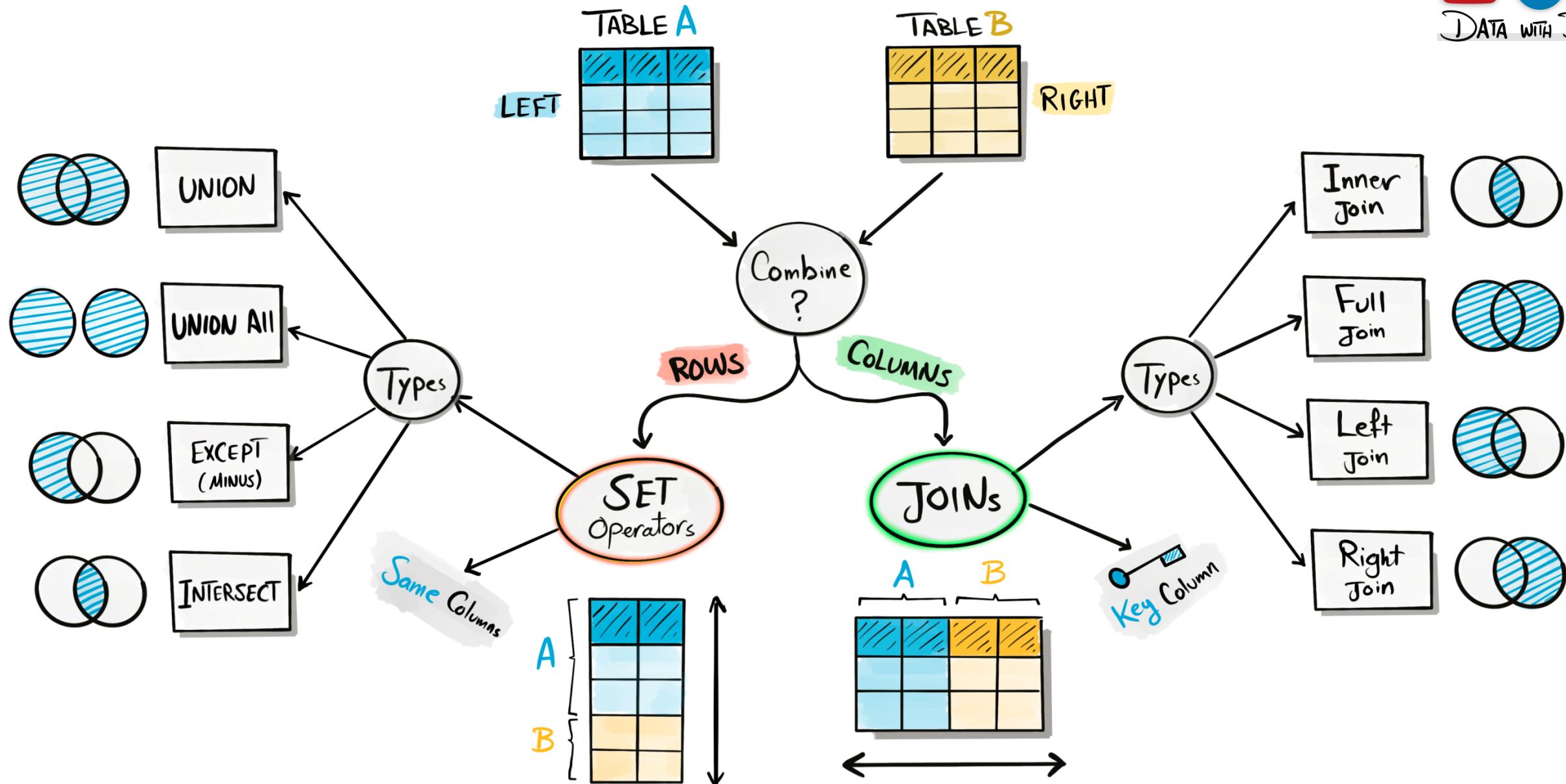
## Combining Data

Baraa Khatib Salkini  
YouTube | **DATA WITH BARAA**  
SQL Course | SET Operators



# SET OPERATORS

Combine the results of multiple queries into a single result set.



# CTE SYNTAX

```
SELECT
  FirstName
  LastName
FROM Customers
JOIN Clause
WHERE Clause
GROUP BY Clause
```

1st SELECT Statement

SET Operator

UNION

```
SELECT
  FirstName
  LastName
FROM Employees
JOIN Clause
WHERE Clause
GROUP BY Clause
```

2nd SELECT Statement

ORDER BY

can be used only at the  
end to sort the final Result

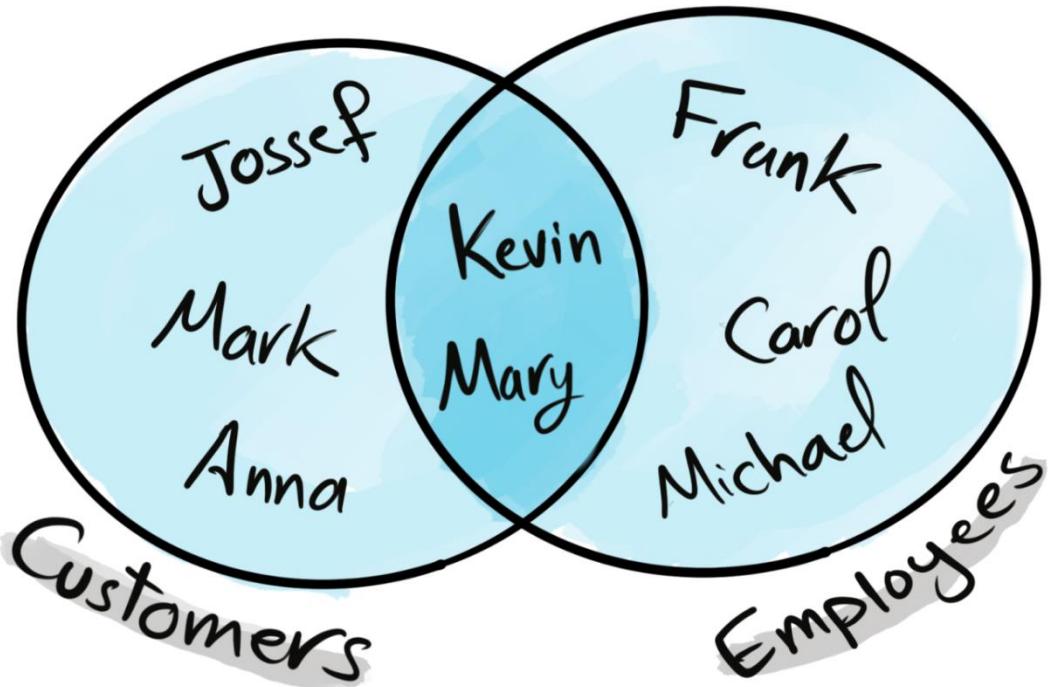
ORDER BY FirstName

# SET RULES

1. SET operators can be used in any clause.
2. ORDER BY is allowed only once—at the end of the query.
3. Each query must have the same number of columns.
4. Column data types must be compatible across queries.
5. The result set takes column names from the first query.

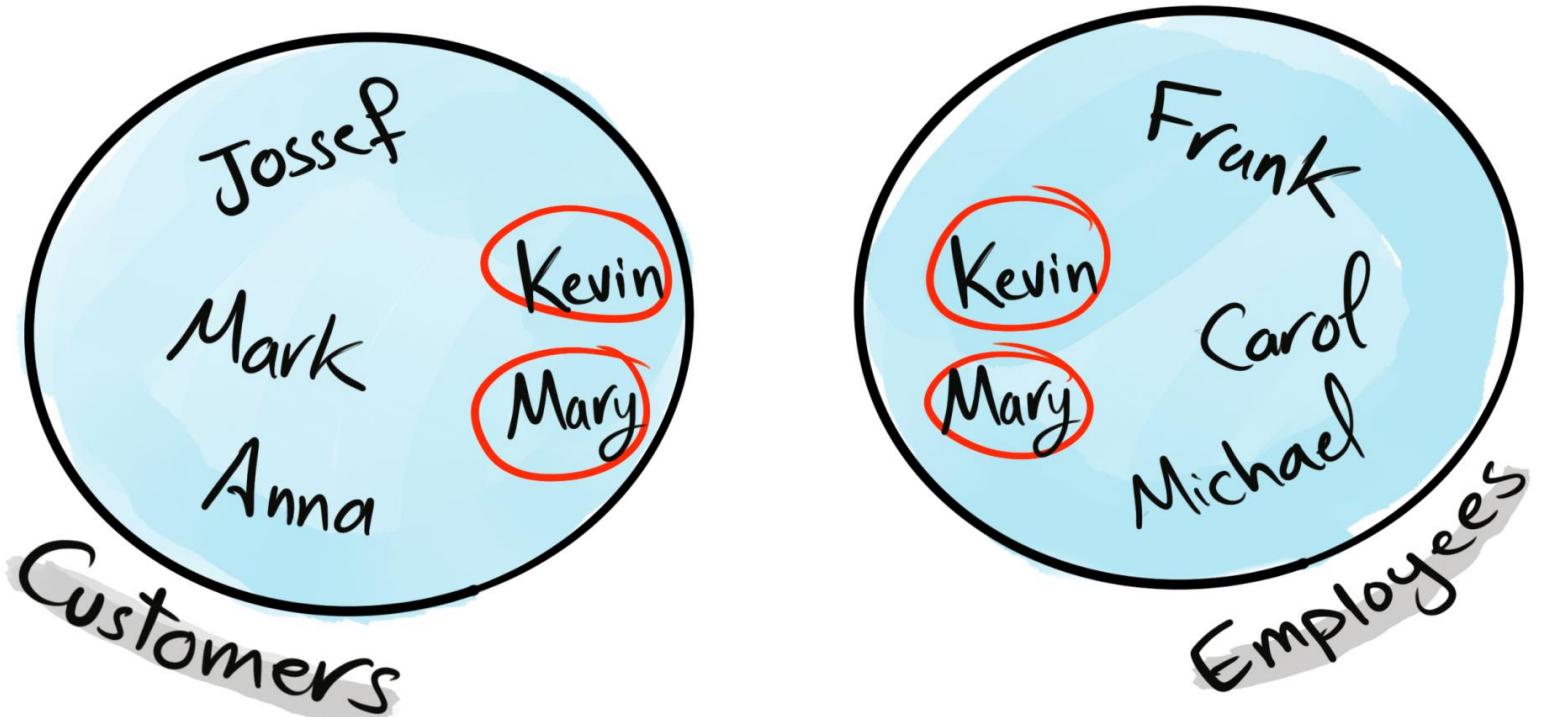
It just care about number of columns and data type not meaning of content inside it.  
Order of column matters for matching.

# UNION



Returns All **distinct** rows from **both** tables

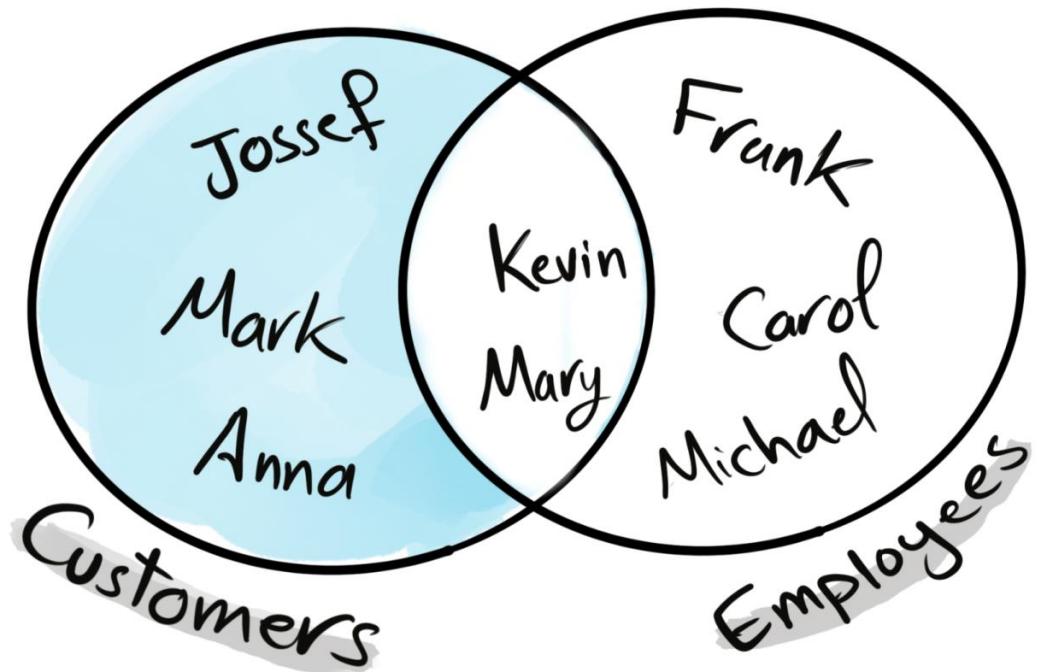
# UNION ALL



Returns All rows, including **duplicates**

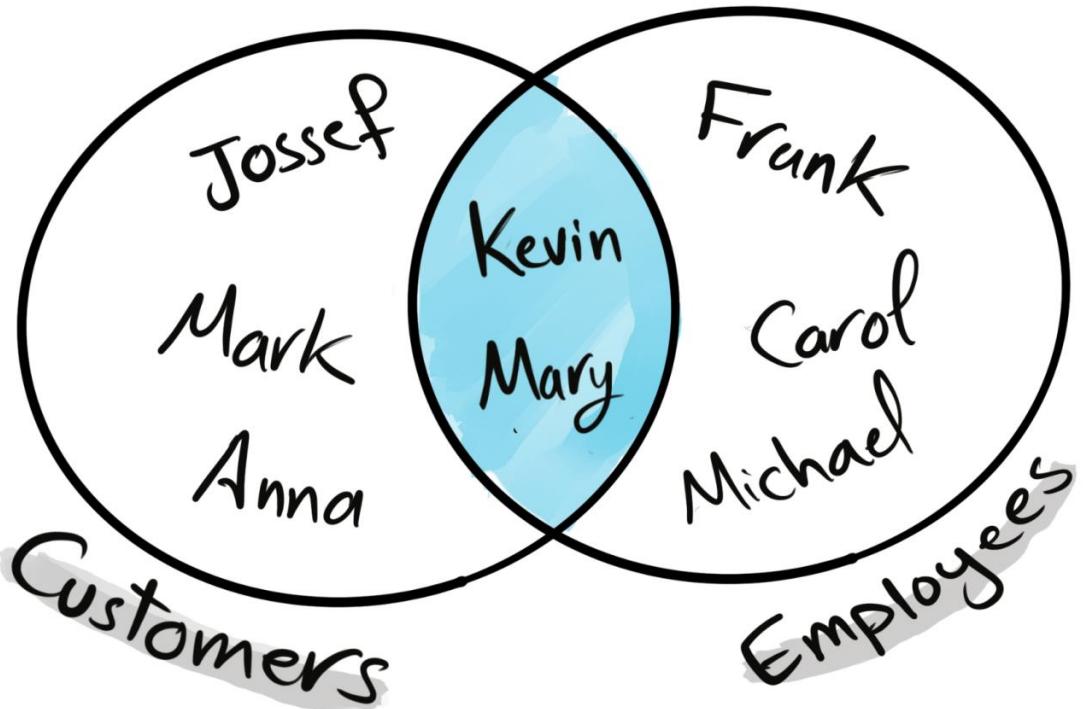
EXCEPT

(MINUS)

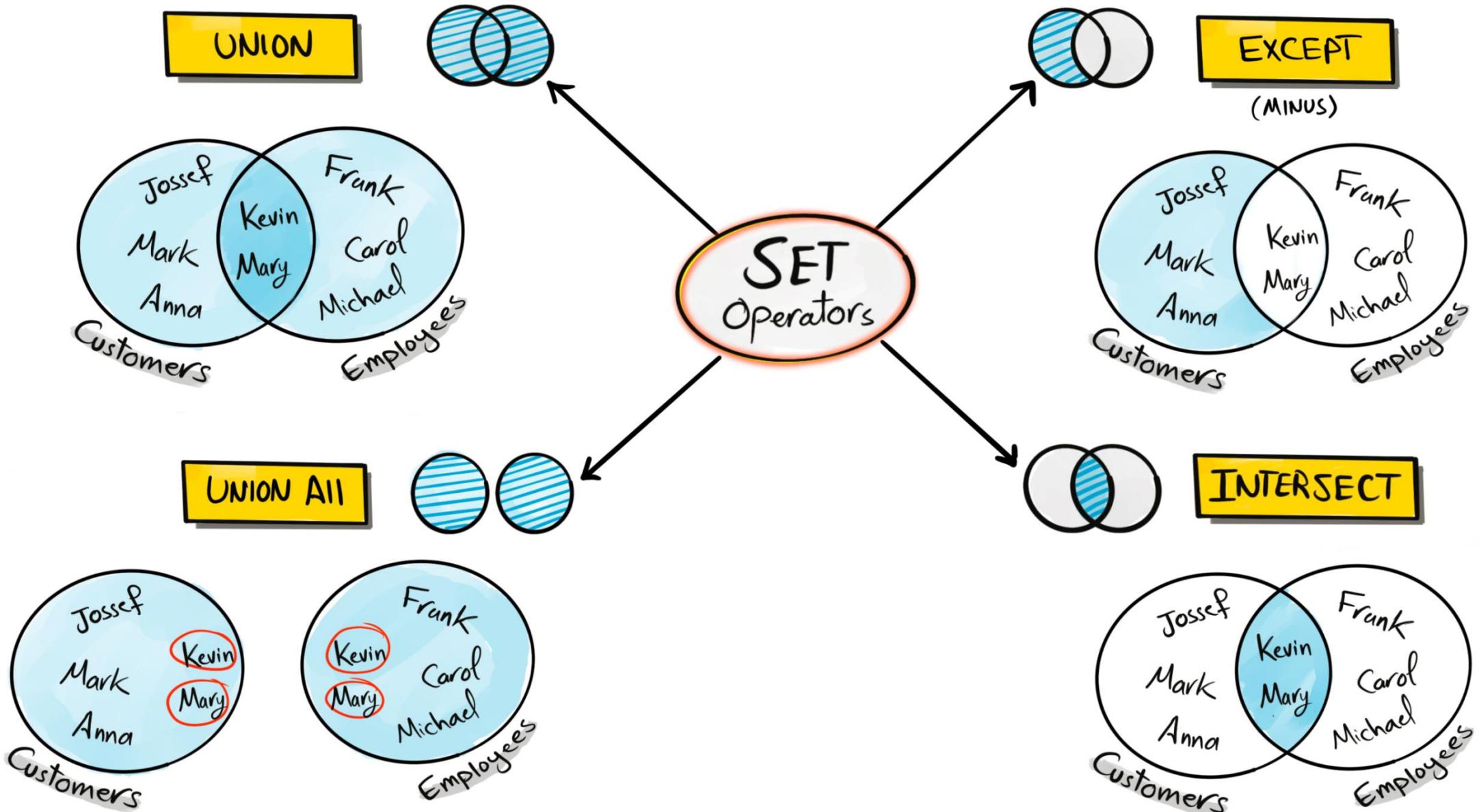


Returns **unique** rows in **1st Table** that are **not** in **2nd Table**

# INTERSECT



Returns **Common rows** between two Tables

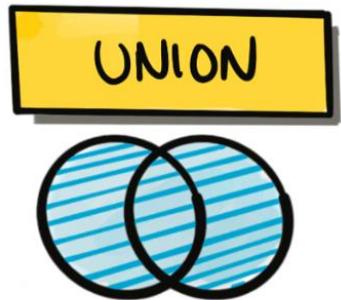


# UNION ALL vs UNION

- **UNION ALL** is faster than **UNION** because it doesn't remove duplicates.
- Use **UNION ALL** if you are sure there are no duplicates.
- Use **UNION ALL** to identify duplicates and data quality issues

# How UNION Works

Employees	
FirstName	LastName
Frank	Lee
Kevin	Brown
Mary	NULL
Michael	Ray
Carol	Baker



Customers	
FirstName	LastName
Jossef	Goldberg
Kevin	Brown
Mary	NULL
Mark	Schwarz
Anna	Adams

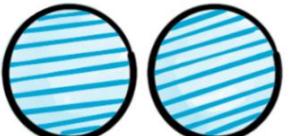
Result

FirstName	LastName
Frank	Lee
Kevin	Brown
Mary	NULL
Michael	Ray
Carol	Baker
Jossef	Goldberg
Mark	Schwarz
Anna	Adams

# How UNION ALL Works

Employees	
FirstName	LastName
Frank	Lee
Kevin	Brown
Mary	NULL
Michael	Ray
Carol	Baker

UNION ALL



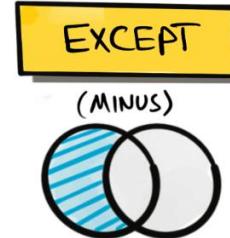
Customers	
FirstName	LastName
Jossef	Goldberg
Kevin	Brown
Mary	NULL
Mark	Schwarz
Anna	Adams

Result

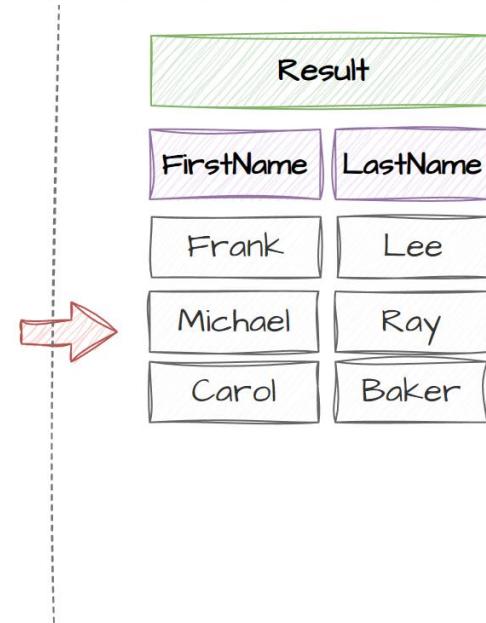
FirstName	LastName
Frank	Lee
Kevin	Brown
Mary	NULL
Michael	Ray
Carol	Baker
Jossef	Goldberg
Kevin	Brown
Mary	NULL
Mark	Schwarz
Anna	Adams

# How EXCEPT Works

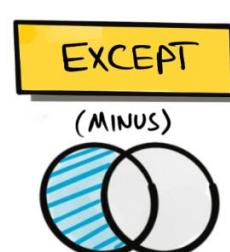
Employees	
FirstName	LastName
Frank	Lee
Kevin	Brown
Mary	NULL
Michael	Ray
Carol	Baker



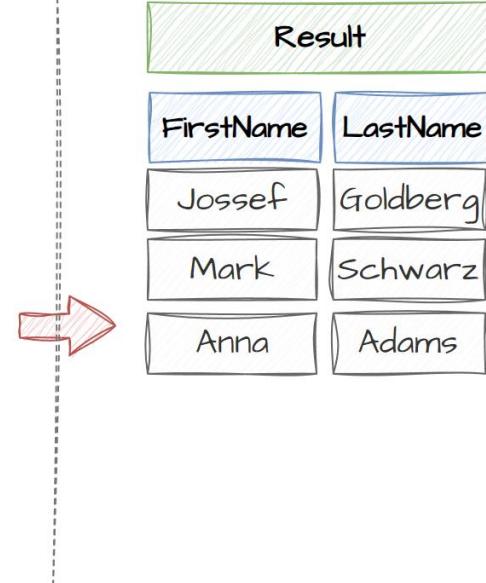
Customers	
FirstName	LastName
Jossef	Goldberg
Kevin	Brown
Mary	NULL
Mark	Schwarz
Anna	Adams



Customers	
FirstName	LastName
Jossef	Goldberg
Kevin	Brown
Mary	NULL
Mark	Schwarz
Anna	Adams



Employees	
FirstName	LastName
Frank	Lee
Kevin	Brown
Mary	NULL
Michael	Ray
Carol	Baker



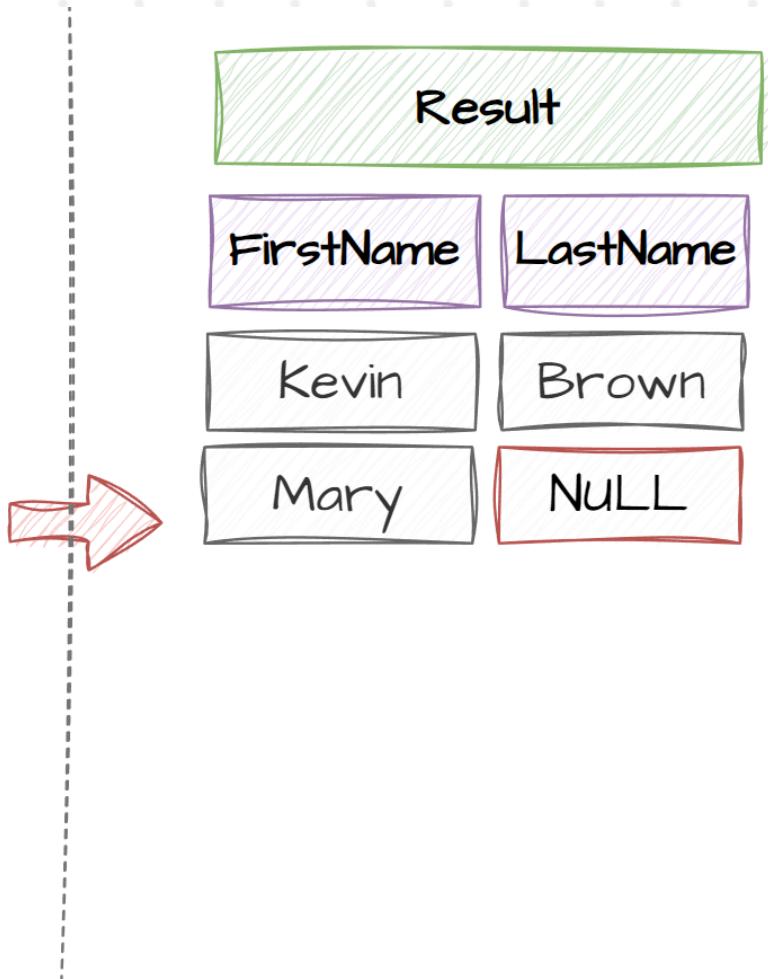
# How INTERSECT Works

Employees	
FirstName	LastName
Frank	Lee
Kevin	Brown
Mary	NULL
Michael	Ray
Carol	Baker

INTERSECT



Customers	
FirstName	LastName
Jossef	Goldberg
Kevin	Brown
Mary	NULL
Mark	Schwarz
Anna	Adams



# SET OPERATORS

## UNION

Returns All rows from both sets, **elimination duplicates**

```
SELECT FirstName, LastName  
FROM Customers  
  
UNION  
  
SELECT FirstName, LastName  
FROM Employees
```

## UNION ALL

Returns All rows from both sets, **including duplicates**

```
SELECT FirstName, LastName  
FROM Customers  
  
UNION ALL  
  
SELECT FirstName, LastName  
FROM Employees
```

## EXCEPT/MINUS

Return unique rows in **first set** that are not in **second table**

```
SELECT FirstName, LastName  
FROM Customers  
  
EXCEPT  
  
SELECT FirstName, LastName  
FROM Employees
```

## INTERSECT

Return only the **common rows** between two sets

```
SELECT FirstName, LastName  
FROM Customers  
  
INTERSECT  
  
SELECT FirstName, LastName  
FROM Employees
```

## SET USE CASE Combine Information

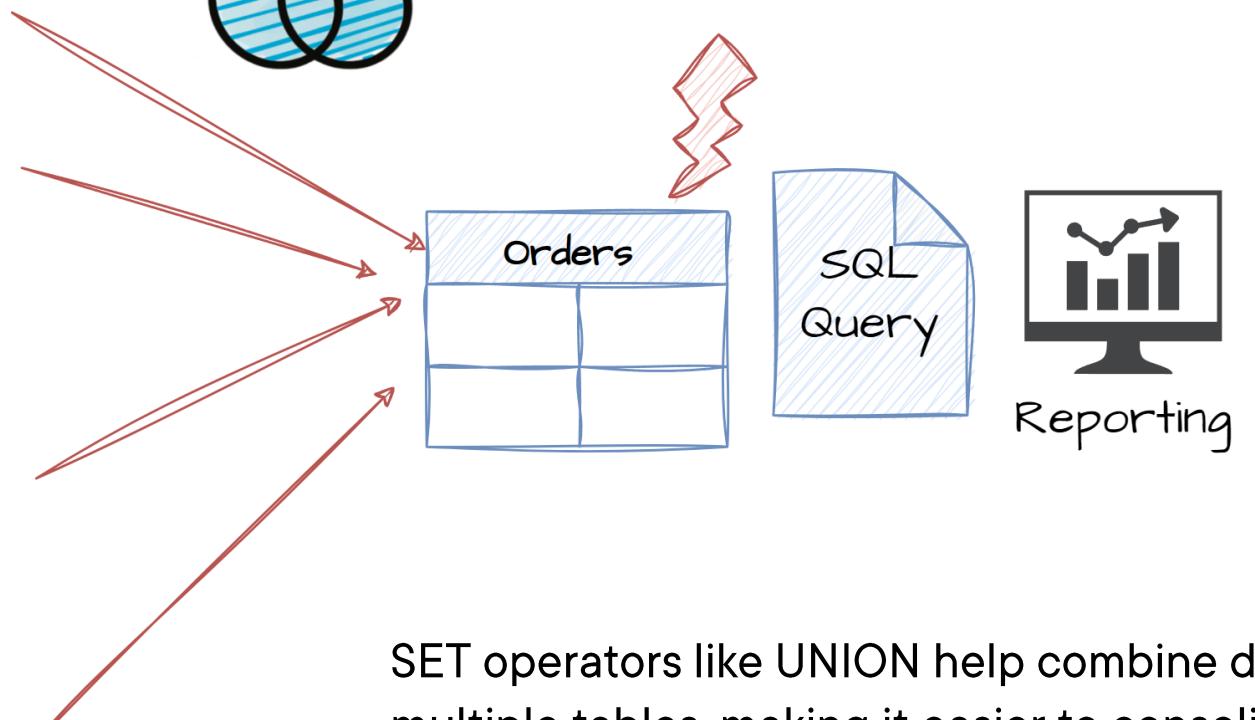
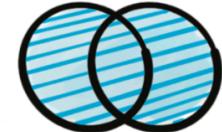
Orders 2025	

Orders 2024	

Orders 2023	

Orders 2022	

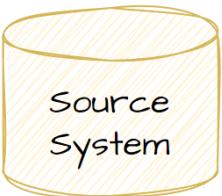
UNION



SET operators like UNION help combine data from multiple tables, making it easier to consolidate information for reporting and analysis

## SET USE CASE Delta Detection

SET operators like EXCEPT help detect changes between datasets, making it easier to identify new, updated, or missing records during data integration.

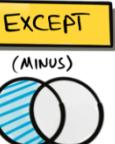


day 1

customer_id	name	email	order_Date
1	John Doe	john@gmail.com	2024-09-17
2	Jan Doe	jan@outlook.com	2024-09-18

day 2

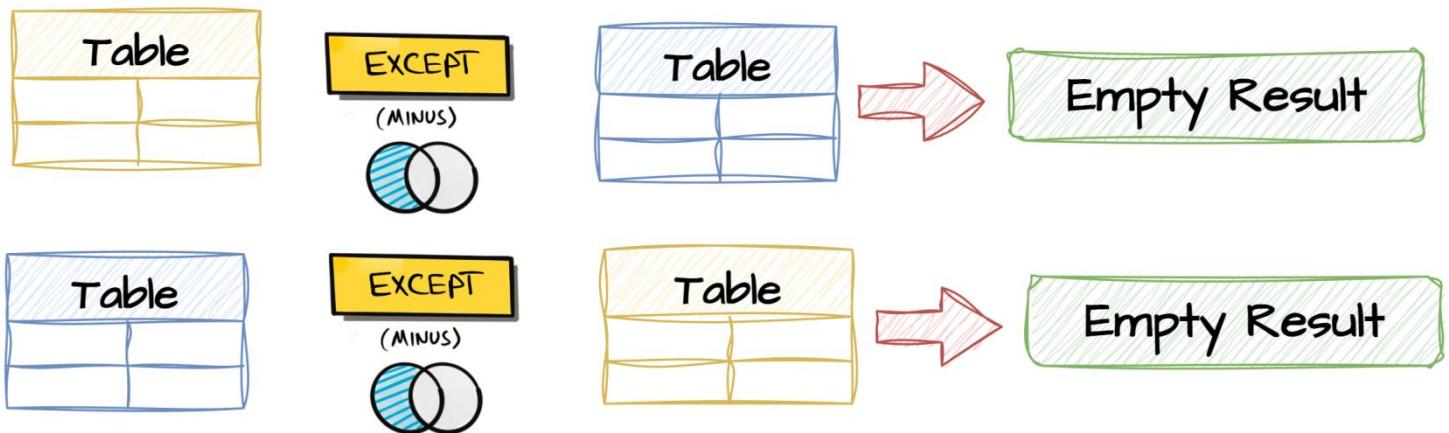
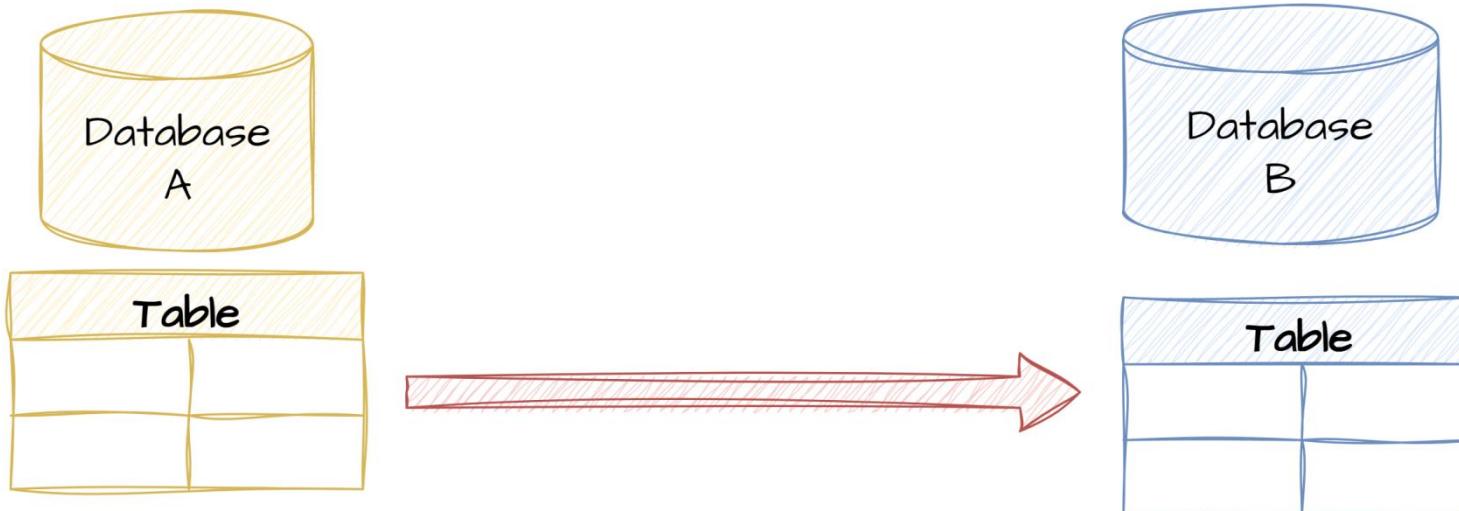
customer_id	name	email	order_Date
1	John Doe	john@gmail.com	2024-09-17
3	Alice	Alice@outlook.com	2024-09-19



customer_id	name	email	order_Date
1	John Doe	john@gmail.com	2024-09-17
2	Jan Doe	jan@outlook.com	2024-09-18
3	Alice	Alice@outlook.com	2024-09-19

## SET USE CASE Data completeness Check

SET operators like EXCEPT help verify data completeness by comparing tables across databases, ensuring no records are missing or mismatched.



# SET OPERATORS

Combine the results of multiple queries into a single result set

Types



UNION



UNION ALL



EXCEPT



INTERSECT

RULES

- Same Nr. of Columns, Data Types, order of Columns.

- 1st Query Controls Column names.

USE CASES

- Combine Information (UNION + UNION ALL)

- Delta Detection (EXCEPT)

- Data Completeness Check (EXCEPT)