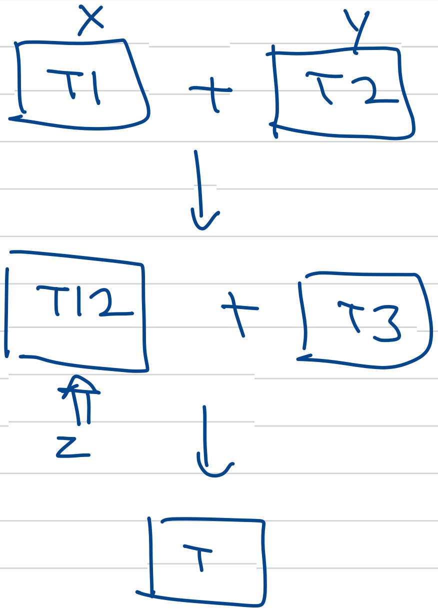
JOINS - 2

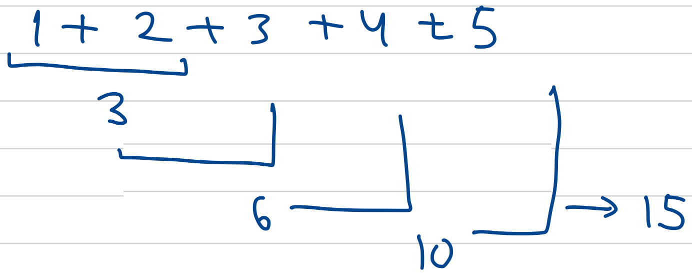
- ✓ Join Multiple Tables
- ✓ Compound Joins
- ✓ Types of Joins - INNER VS OUTER
- ✓ CROSS Join
- ✓ USING
- ✓ NATURAL JOIN
- ✓ IMPICIT JOIN
- ✓ JOIN with WHERE VS ON
- UNION

Ⓘ JOINING MULTIPLE TABLES

```
SELECT f.title, l1.name, l2.name
FROM film f
JOIN language l1
ON f.language_id = l1.language_id
JOIN language l2
ON f.original_language_id = l2.language_id;
```

Pair wise manner





$\boxed{\text{II}}$ COMPOUND JOIN

Any join where we have more than 1 condition. on different cols.

```
SELECT *
FROM   Film  f1
JOIN   Film  f2
ON  (f2.year BETWEEN f1.year -2 AND
                      f1.year +2)
     AND (f2.rental > f1.rental)
```

⇒ Multiple conditions on different cols.

Types of Joins

$$\begin{tikzcd}
 & \text{Types of Joins} \arrow[dl] \arrow[dr] & & \\
\Rightarrow \text{INNER Join (Default)} & & \text{OUTER JOIN} \arrow[dl] \arrow[d] \arrow[dr, dashed] & \\
 & \text{Left outer Join} & \text{Right outer Join} & \text{full outer Join (MySQL doesn't support)}
\end{tikzcd}$$

$$\left.\begin{array}{l}\text{JOIN}\\ \underbrace{\text{INNER}}_{\text{optional}}\ \text{JOIN}\end{array}\right] \rightarrow \text{same}$$

Batches

1	Batch A
2	Batch B
3	Batch C

Student

			batch_id
	1	S1	1
	2	S2	1
✗	3	S3	null
✗	4	S4	null
	5	S5	2

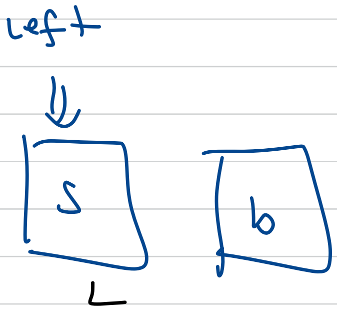
Inner join doesn't match for null values.

```
Select *
 FROM
 student   s
JOIN  Batches b
 ON  s.batch_id = b.batch_id
```

left outer join

include all the Rows of the left table. ←

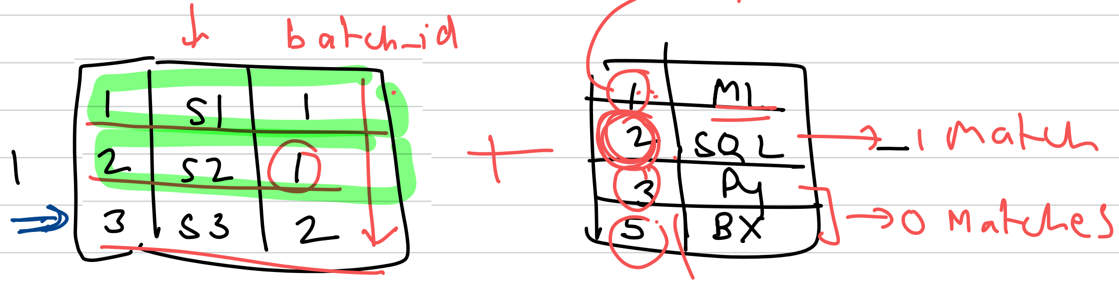
```
SELECT *
FROM students s
LEFT JOIN batches b
ON s.batch_id = b.batch_id.
```



		batch_id	batch_id	batname
1	S1	1	1	BA
2	S2	1	1	RA
3	S3	null	null	null
4	S4	null	null	null
5	S5	2	2	BB

Right JoiN

include all rows of the Right table



1	S1	1
2	S2	1
3	S3	2

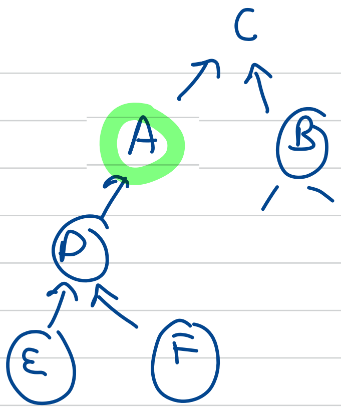
1	Ml
2	SQL
3	Py
5	BX

```
SELECT *
FROM
  students s
 RIGHT JOIN batches b
ON  s.batch_id
        = b.batch_id;
```

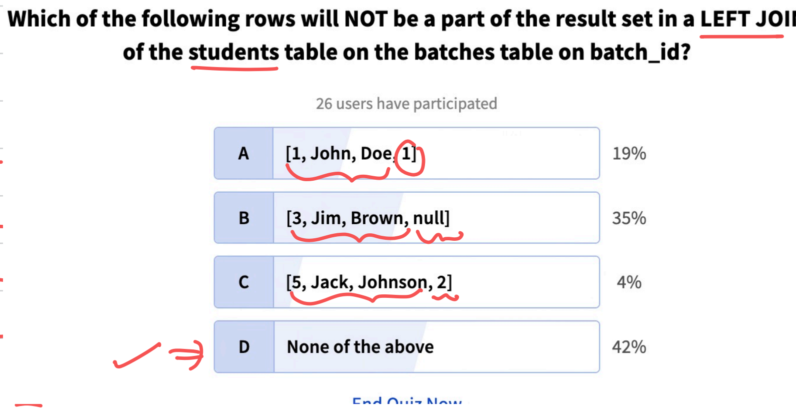
1	ML	1	S1	1
1	ML	2	S2	1
2	SQL	3	S3	2
3	Py	–	–	–
5	BX	–	–	–

first table
↓
left Table

		reporst To
1	A $\rightarrow$	3
2	B $\rightarrow$	3
3	C $\rightarrow$	null
4	D $\rightarrow$	1
5	E $\rightarrow$	4
6	F $\rightarrow$	4



	Mgr
A —	C
B —	C
D —	A
null —	B
E —	D
F —	D
null —	E


Which of the following rows will NOT be a part of the result set in a LEFT JOI
of the students table on the batches table on batch_id?

26 users have participated

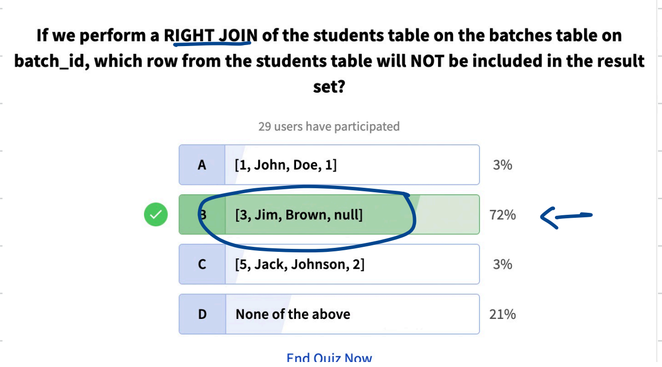
A [1, John, Doe, 1] 19%

B [3, Jim, Brown, null] 35%

C [5, Jack, Johnson, 2] 4%

[✓ D] None of the above 42%

null — F —



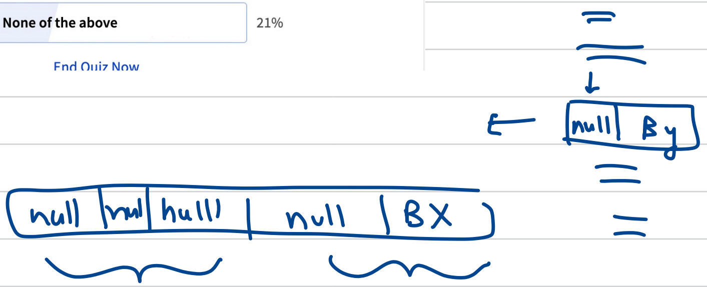
If we perform a RIGHT JOIN of the students table on the batches table on batch_id, which row from the students table will NOT be included in the result set?

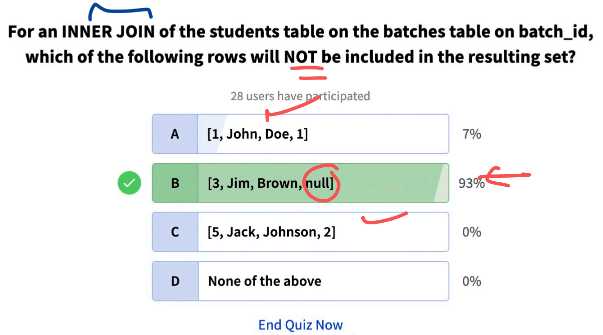
29 users have participated

A	[1, John, Doe, 1]	3%
[✓ B]	[3, Jim, Brown, null]	72%
C	[5, Jack, Johnson, 2]	3%
D	None of the above	21%

End Quiz Now

Students
RIGHT JOIN batches



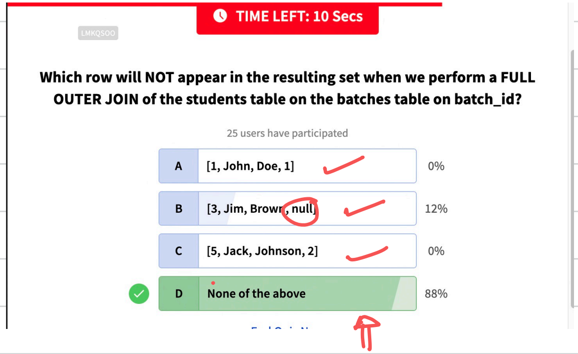


For an INNER JOIN of the students table on the batches table on batch_id, which of the following rows will NOT be included in the resulting set?

28 users have participated

A	[1, John, Doe, 1]	7%
✓ B	[3, Jim, Brown, null]	93%
C	[5, Jack, Johnson, 2]	0%
D	None of the above	0%

End Quiz Now



TIME LEFT: 10 Secs

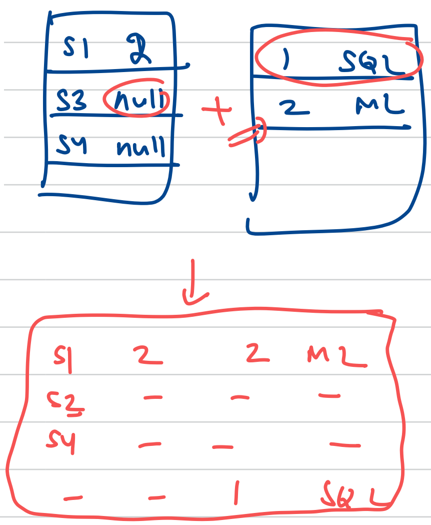
LMKQSOO

Which row will NOT appear in the resulting set when we perform a FULL OUTER JOIN of the students table on the batches table on batch_id?

25 users have participated

A	[1, John, Doe, 1]	0%
B	[3, Jim, Brown, null]	12%
C	[5, Jack, Johnson, 2]	0%
✓ D	None of the above	88%

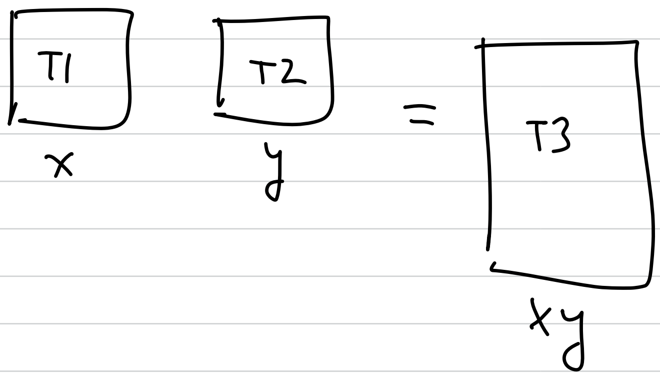
Full outer join

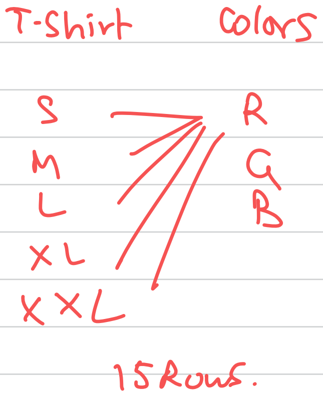


CROSS JOIN

↳ don't Specify any condition

↳ all combinations from T1 & T2





```
SELECT *
FROM   T1
JOIN   T2 ;
```

↑ CROSS JOIN

USING

(Synatic Sugar)

```
SELECT *
FROM   students  s
JOIN   batches   b
ON     s.batch-id  = b.batch-id
```

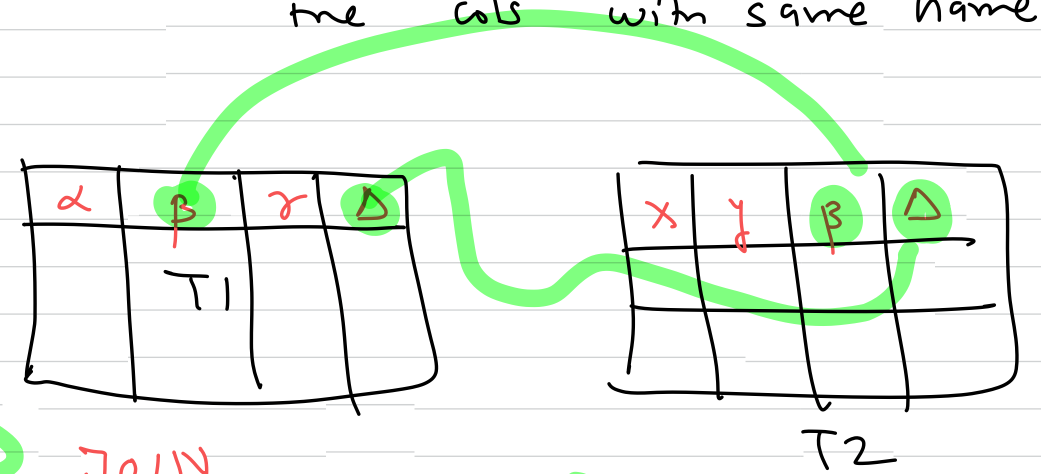
equivalent

```
SELECT *
FROM   students  s
JOIN   batches   b
USING  (batch-id);
```

must be present in both tables.

Natural Join

Joining 2 Tables, they are mostly on the cols with same name



JOIN
ON (T1.β = T2.β) AND T1.Δ = T2.Δ

easy Syntax
```
SELECT *
  FROM  T1
  NATURAL JOIN  T2,
```
→ will compare all cols that have same

- Implicit Join

```
SELECT *
FROM students s, batches b;
```

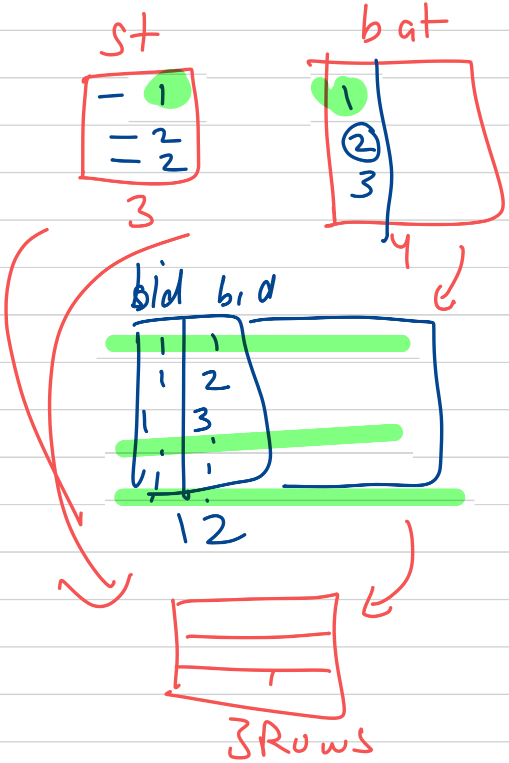
↳ same as doing a CROSS-JOIN

JOIN with WHERE vs ON

```
SELECT  *
FROM    students  s
JOIN    batches   b
WHERE   s.bid = b.bid ;
```

vs

```
SELECT  *
FROM    students  s
JOIN    batches   b
   ON   s.bid = b.bid ,
```

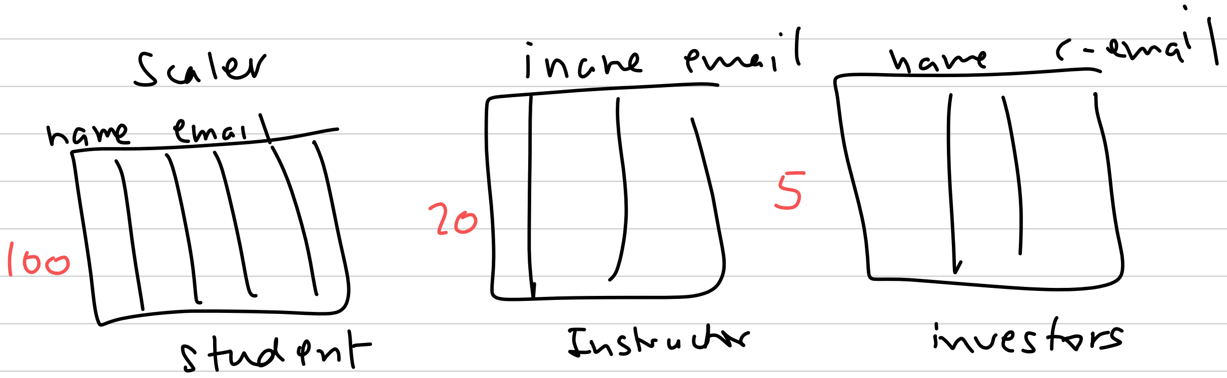


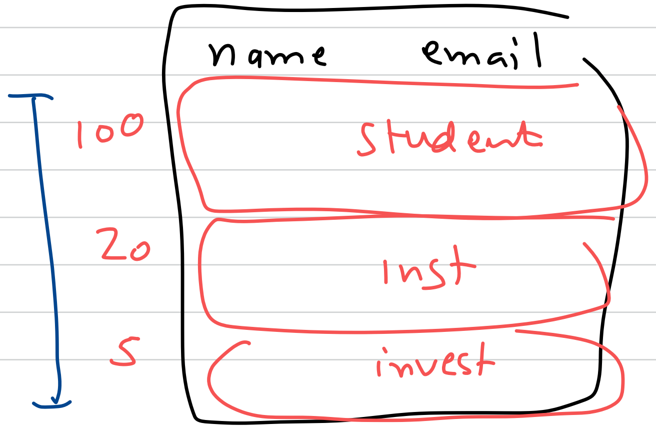
THE ON condition is applied during the creation of intermediate table,
Resulting in lower memory usage and better performance.

The WHERE is applied before the printing stage, and it results in a additional memory and slower performance.

Unless you have to create all possible pairs, avoid using CROSS JOINS.

UNION





= 125 Rows

vertically stacking

```
SELECT name, email
  FROM students
UNION
SELECT iname, iemail
      FROM instructors
UNION
SELECT name, cemail
  FROM investors,
```

```
-- UNION
SELECT firstName,email AS "Contact Info"
FROM employees
UNION
SELECT customerName,phone
FROM customers
UNION
SELECT first_name,last_name
FROM scalerDB.students;
```