JOIN → Two Tables that have **no common keys**

1. **Method 1 → Cross Join**
    a. If there's no relationship at all between the tables, you can use a CROSS JOIN.
    b. It pairs every row in the first place and every row in the second.
    c. ✅ Use this if you truly want all possible combinations.
    d. ⚠️ Be careful — if table1 has 100 rows and table2 has 200 rows, the result will have **20,000 rows**.

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

2. **Method 2 → JOIN using a constant condition**
    a. Force a join with a dummy condition → for eg. using 1=1

```
SELECT *
FROM table1
JOIN table2 ON 1=1;
```

3. **Method 3 → Combine tables vertically (UNION / UNION ALL)**
    a. If the tables have **similar columns** and you want to **stack** them (not join side by side), use `UNION` or `UNION ALL`.
    b. `UNION` removes duplicates.
    c. `UNION ALL` keeps all rows.

```
SELECT id, name, value FROM table1
UNION ALL
SELECT id, name, value FROM table2;
```

4. **Method 4 → Join with artificial or derived keys**
    a. Sometimes you can **create a temporary key** to make a meaningful join.
    b. Example: join the first 10 rows of table1 with the first 10 rows of table2 by adding a row number:
    c. This is useful when you want to pair rows **by position**, not by data relationship.

```
WITH t1 AS (
  SELECT ROW_NUMBER() OVER () AS rn, * FROM table1
),
t2 AS (
  SELECT ROW_NUMBER() OVER () AS rn, * FROM table2
)
```

SELECT *

FROM t1

JOIN t2 ON t1.rn = t2.rn;

Case 1: INNER JOIN

This will produce:

`MIN(count(table1), count(table2))` **rows**

Because only the matching row numbers exist in both tables.

Example:

- table1 → 100 rows
- table2 → 80 rows
  → result = **80 rows**

Case 2: LEFT JOIN

```
SELECT *

FROM t1

LEFT JOIN t2 ON t1.rn = t2.rn;
```

This gives:

`count(table1)` **rows**

Every row from `table1` appears once, and matching `t2` rows fill in when available (otherwise NULLs).

Example:

- table1 → 100 rows
- table2 → 80 rows
  → result = **100 rows**

Case 3: RIGHT JOIN

Opposite of above:

`count(table2)` **rows**

Case 4: FULL OUTER JOIN

```
SELECT *

FROM t1

FULL OUTER JOIN t2 ON t1.rn = t2.rn;
```

Gives:

`MAX(count(table1), count(table2))` **rows**

Example:

- **table1 → 100 rows**
- **table2 → 80 rows**
  → result = 100 rows