# SQL Joins

Page :
Date :

**Q** What is a join?

**A:-** A join is a feature of sql that allows us to get data from two different tables where the records are related.

They are used in all queries that involve multiple tables, which is a lot of queries that we will write as a developer.

**Q.** Why do we need joins?

**A:-** It is because a well-designed database has its data split into several tables.

But we use joins to get data from related tables into a single query and a single set of results.

**Ex:-**

person

| person_id | first_name | company_id |
|---|---|---|
| 1 | Susan | (null) |
| 2 | John | 2 |
| 3 | Mark | 2 |
| 4 | Sarah | 1 |
| 5 | Robert | 4 |

company

| id | company_name |
|---|---|
| 1 | ABC Pvt Ltd |
| 2 | XYZ Services |
| 3 | CB Construction |
| 4 | Plumbing Solutions |
| 5 | Magic Printing |

We can select from these tables individually.

```
SELECT
    person_id,
    first_name,
    company_id
FROM person;
```

```
SELECT
    id,
    company_name
FROM company;
```

But, in this, we will have two different queries and two different results

Page :
Date :

But, what if we want the data in one result? what if we wanted to see a list of people's names and the company names they are related to.?

We can use JOIN.

- We start by writing the select query, then some columns from the person table

We specify the person table, then we need to specify the second table. Before the second table, we specify a join.

- There are few different types of joins we can use in SQL. The differences are in how they handle data found in one table but not another.

## Inner Join

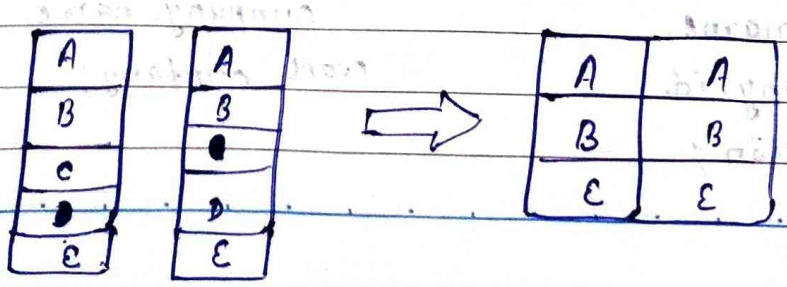
A inner join lets us relate records between two tables, and returns records where there is a match in both tables.

If there is a record in one table without a match in the other, it is not shown in the results.

for ex:- If there is a person that has no company id record, the person won't be shown if we use an inner join

(or)

If there is a company with no person records, the company won't be shown



Page :
Date :

```
SELECT
person_id,                          INNER JOIN = JOIN
first_name,                         (Inner is optional)
company_id
company_name
FROM person ──────────→ first table
INNER JOIN company ──→ second table
  ON person.company_id = company.id
```

or we can also specify the fields are from which table

```
SELECT
person.person_id,
person.first_name,
person.company_id,
company.company_name
FROM person
INNER JOIN company
ON person.company_id = company.id
```

result

| person_id | first_name | company_id | company_name |
|---|---|---|---|
| 2 | John | 2 | XYZ Services |
| 3 | Mark | 2 | XYZ Services |
| 4 | Sarah | 1 | ABC Pvt. Ltd. |
| 5 | Robert | 4 | Plumbing solutions |

**Left Outer join**

This join type will show all records from the table on the left of the join keyword which is the table specified first.

Then if there are any records matching the right table, then they are also included, if there are no matches null is shown.

| A |
|---|
| B |
| C |
| |
| E |
| |

| A |
|---|
| B |
| |
| D |
| E |
| |

| A | A |
|---|---|
| B | B |
| C | |
| E | E |

```
SELECT
    person.person_id,
    person.first_name,
    person.company_id,
    company.company_name
FROM person
LEFT OUTER JOIN company
    ON person.company_id = company.id
```

LEFT OUTER JOIN
= LEFT JOIN
(OUTER is optional)

Result

| person_id | first_name | company_id | company_name |
|---|---|---|---|
| 1 | Susan | (null) | (null) |
| 2 | John | 2 | XYZ Services |
| 3 | Mark | 2 | XYZ Services |
| 4 | Sarah | 1 | ABC Pty Ltd |
| 5 | Robert | 4 | Plumbing Solutions |

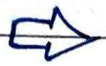
So, this join is helpful if we want all records in a table even if there is no matches in another table.

## Right Outer Join

This join type will show all records from table on the right of the join keyword.

Then all the matching records from the left table will also be shown, if no matching records then null will be shown

| | |
|---|---|
| A | A |
| B | B |
| C | |
| | D |
| E | E |



| | |
|---|---|
| A | A |
| B | B |
| | D |
| E | E |

```
SELECT
person.person_id,
person.first_name,
person.company_id,
company.company_name
FROM person
RIGHT OUTER JOIN company
ON person.company_id = company.id
```

RIGHT OUTER JOIN
= RIGHT JOIN
(OUTER IS OPTIONAL)

result

| person_id | first_name | company_id | company_name |
|---|---|---|---|
| 2 | John | 2 | XYZ services |
| 3 | Mark | 2 | XYZ services |
| 4 | Sarah | 1 | ABC pty Ltd |
| 5 | Robert | 4 | Plumbing Solutions |
| (null) | (null) | 3 | CB construction |
| (null) | (null) | 5 | Magic Printing |

Left = Right
with opposite tables

```
FROM person                    =    FROM company
LEFT OUTER JOIN company             RIGHT OUTER JOIN person
```

## Full outer join

Combination of both left and right outer join.

- Order of the tables doesn't matter.

| person_id | first_name | company_id | company_name |
|---|---|---|---|
| 1 | Susan | (null) | (null) |
| 2 | John | 2 | XYZ services |
| 3 | Mark | 2 | XYZ services |
| 4 | Sarah | 1 | ABC pty Ltd. |
| 5 | Robert | 4 | Plumbing solutions |
| (null) | (null) | 3 | CB Construction |
| (null) | (null) | 5 | Magic Printing |

↳ query for this

```
SELECT
    person.person_id,
    person.first_name,
    person.company_id,
    company.company_name
FROM person
FULL OUTER JOIN company
    ON person.company_id = company.id
```

Result