PostgreSQL: Joins :

PostgreSQL **JOINS** (inner and outer)

PostgreSQL JOINS are used to retrieve data from multiple tables. A PostgreSQL JOIN is performed whenever two or more tables are joined in a SQL statement.

There are different types of PostgreSQL joins:

- **Inner Join**
- **Left (Outer) Join.**
- **Left (Outer) Join Excluding Inner Join.**
- **Right (Outer) Join.**
- **Right (Outer) Join Excluding Inner Join.**
- **(Full) Outer Join.**
- **(Full) Outer Join Excluding Inner Join.**
- **CROSS JOIN.**
- **UNION & UNION ALL.**
- **INTERSECT & EXCEPT**

So let's discuss PostgreSQL JOIN syntax, look at visual illustrations of PostgreSQL JOINS, and explore JOIN examples.

INNER JOIN (simple join)

You've already written a statement that uses a PostgreSQL INNER JOIN. It is the most common type of join. PostgreSQL INNER JOINS return all rows from multiple tables where the join condition is met.

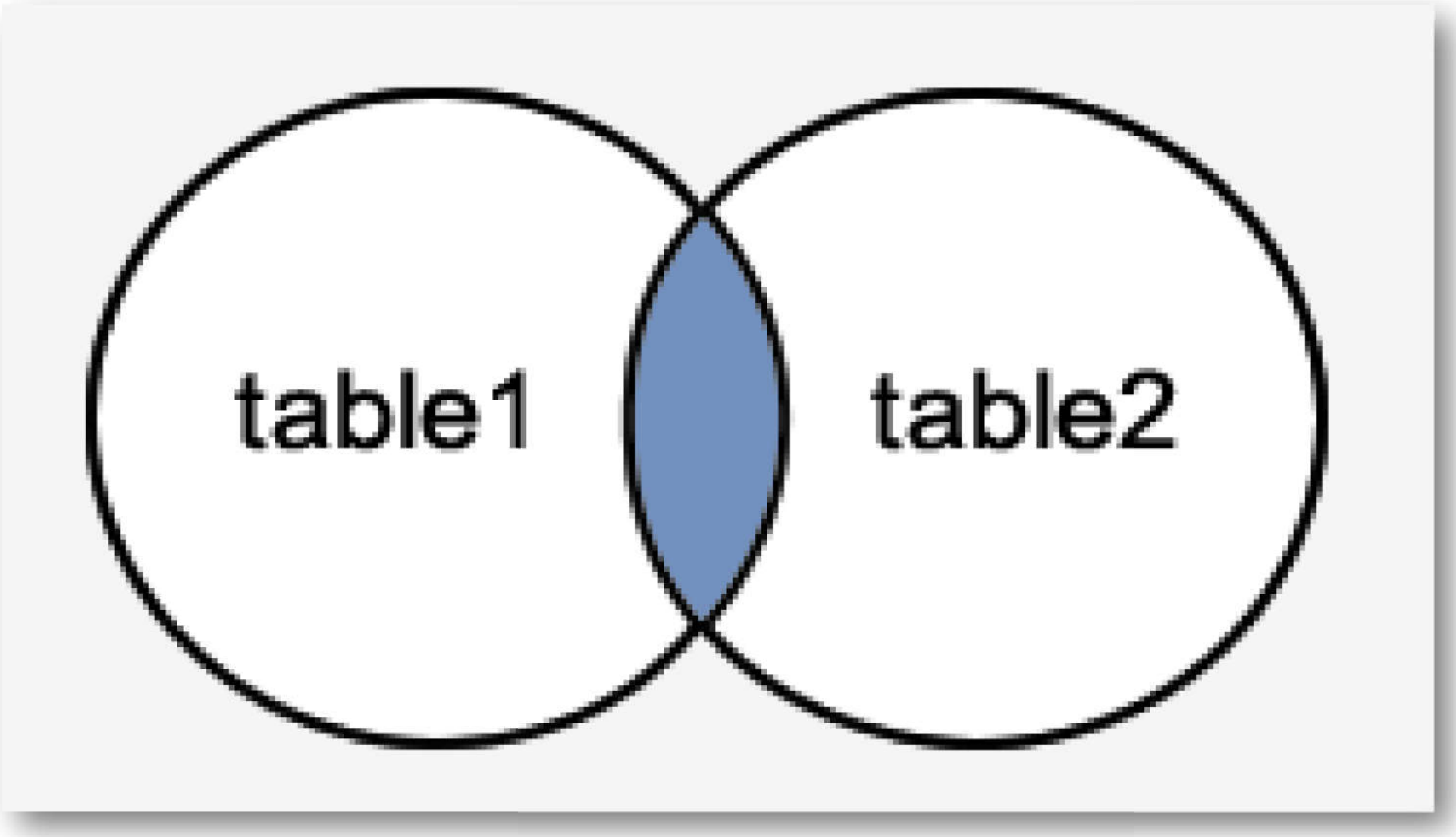
Inner join produces only the set of records that match in both Table A and Table B Most commonly used, best understood join.

Inner Joins do not have to use equality to join the fields and we Can use <, >, <>

The syntax for the INNER JOIN in PostgreSQL is:

```
SELECT columns
FROM table1
INNER JOIN table2 (or JOIN table2)
ON table1.column = table2.column;
```

The PostgreSQL INNER JOIN returns the shaded area:



The PostgreSQL INNER JOIN would return the records where *table1* and *table2* intersect.

LEFT OUTER JOIN

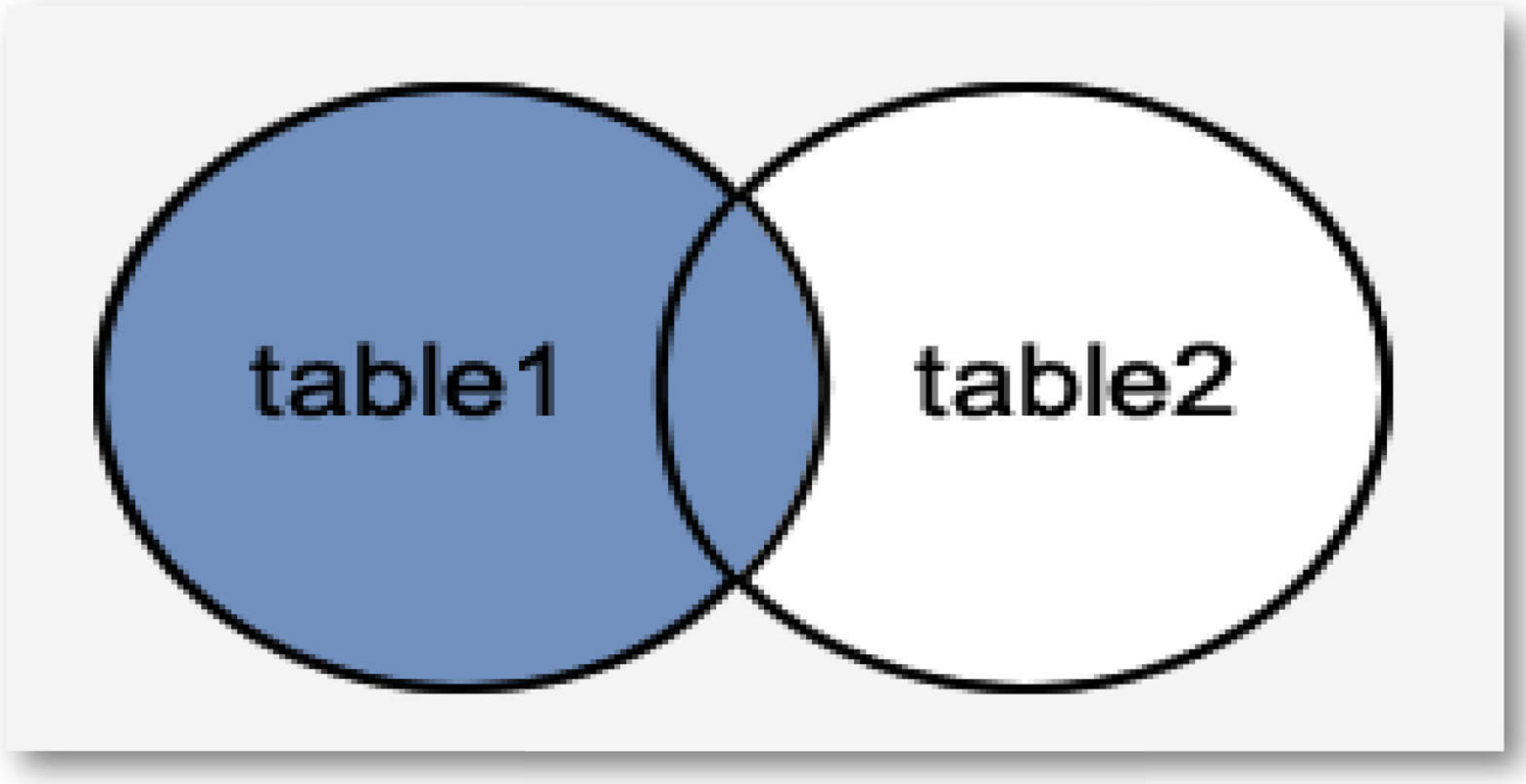
Another type of join is called a PostgreSQL LEFT OUTER JOIN. This type of join returns all rows from the LEFT-hand table specified in the ON condition and **only** those rows from the other table where the joined fields are equal (join condition is met).

Syntax:

The syntax for the PostgreSQL LEFT OUTER JOIN is:

```
SELECT columns
FROM table1
LEFT OUTER JOIN table2
ON table1.column = table2.column;
```

The PostgreSQL LEFT OUTER JOIN returns the shaded area:



The PostgreSQL LEFT OUTER JOIN would return the all records from *table1* and only those records from *table2* that intersect with *table1*.

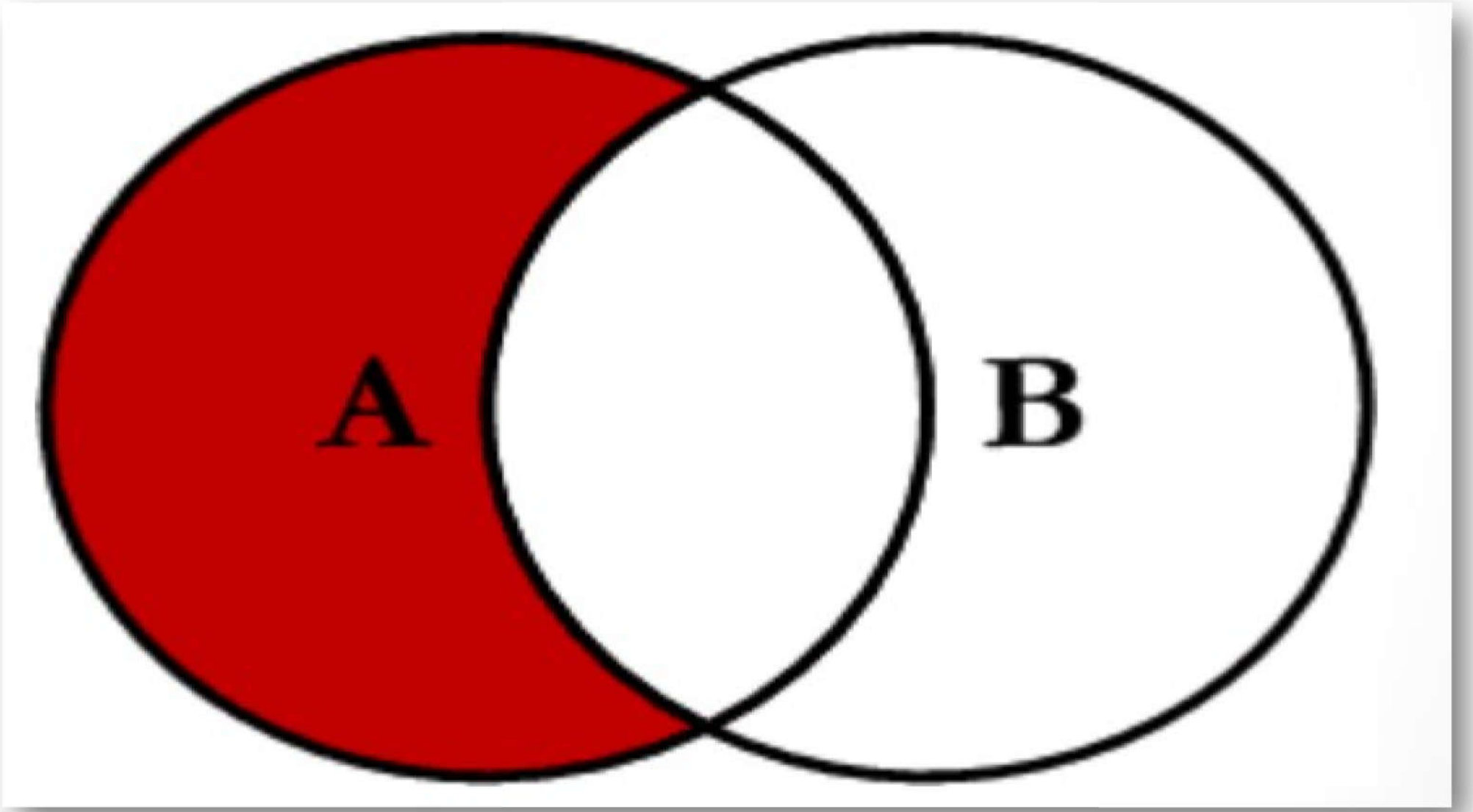
Left Join Excluding Inner Join

This query will 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 *
FROM Table A
LEFT JOIN Table B
ON TableA.PK = TableB.PK
WHERE TableB.PK IS NULL;
```

Perform left outer join, and then exclude the records we don't want from the right side via a WHERE clause



RIGHT OUTER JOIN

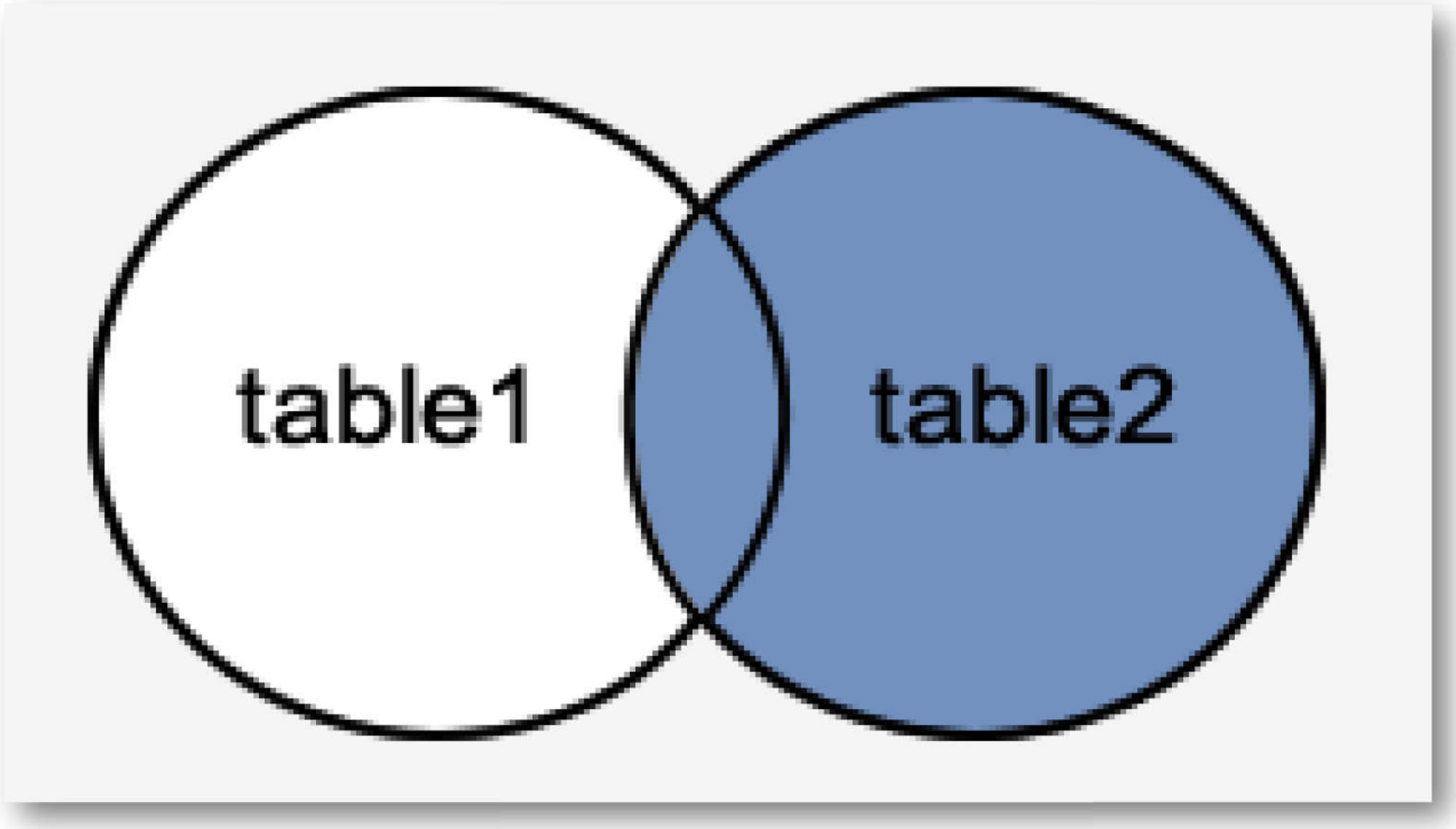
Another type of join is called a PostgreSQL RIGHT OUTER JOIN. This type of join returns all rows from the RIGHT-hand table specified in the ON condition and **only** those rows from the other table where the joined fields are equal (join condition is met).

Syntax:

The syntax for the PostgreSQL RIGHT OUTER JOIN is:

```
SELECT columns
FROM table1
RIGHT OUTER JOIN table2
ON table1.column = table2.column;
```

The PostgreSQL RIGHT OUTER JOIN returns the shaded area:



The PostgreSQL RIGHT OUTER JOIN would return the all records from *table2* and only those records from *table1* that intersect with *table2*.

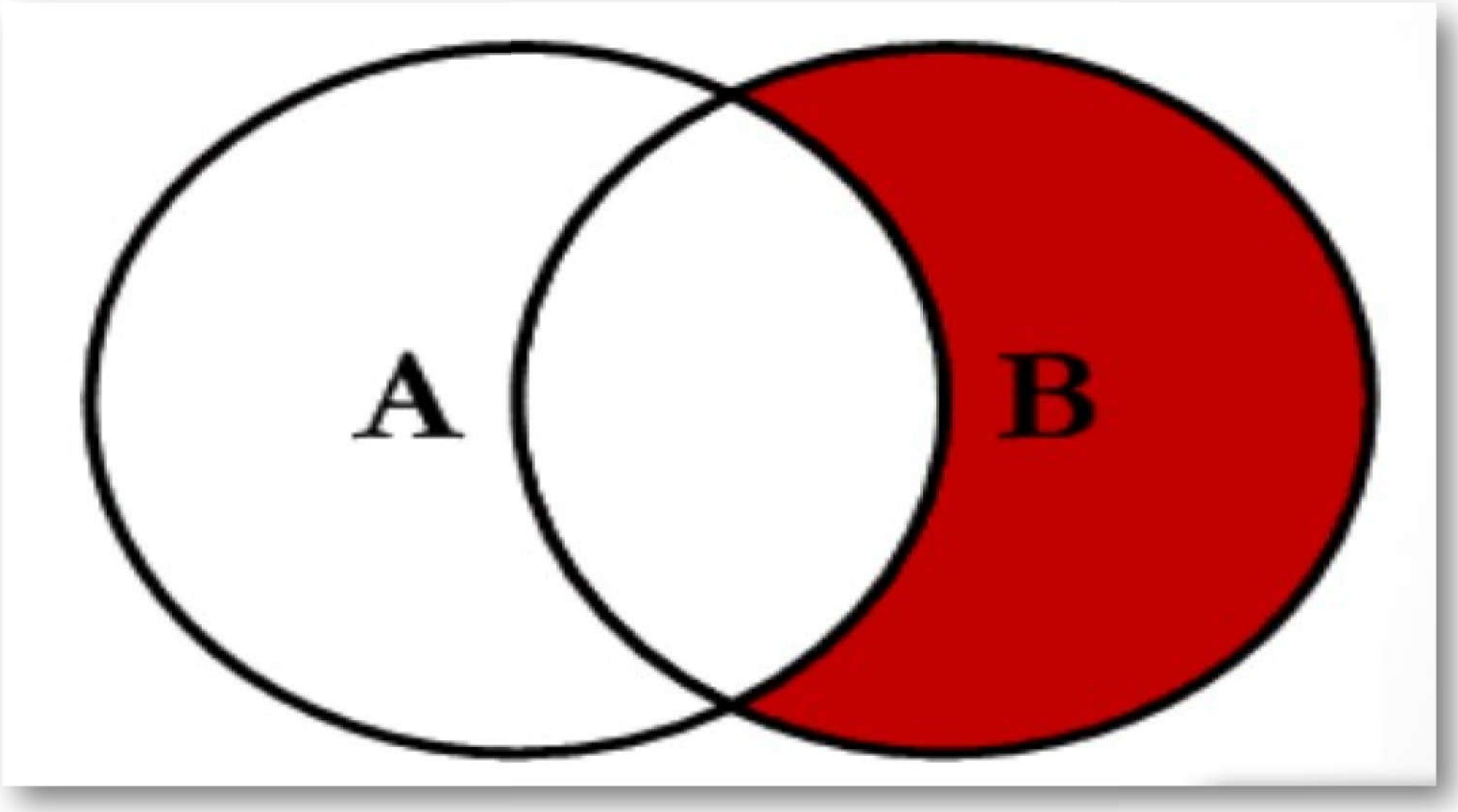
Right Join Excluding Inner Join

This query will return all of the records in the right table (table B) that do not match any records in the left table (table A).

Syntax:

```
SELECT *
FROM TableA
RIGHT JOIN TableB
ON TableA.PK = TableB.PK
WHERE TableA.PK IS NULL
```

Perform right outer join, then exclude the records we don't want from the left side via a **WHERE** clause.



FULL OUTER JOIN:

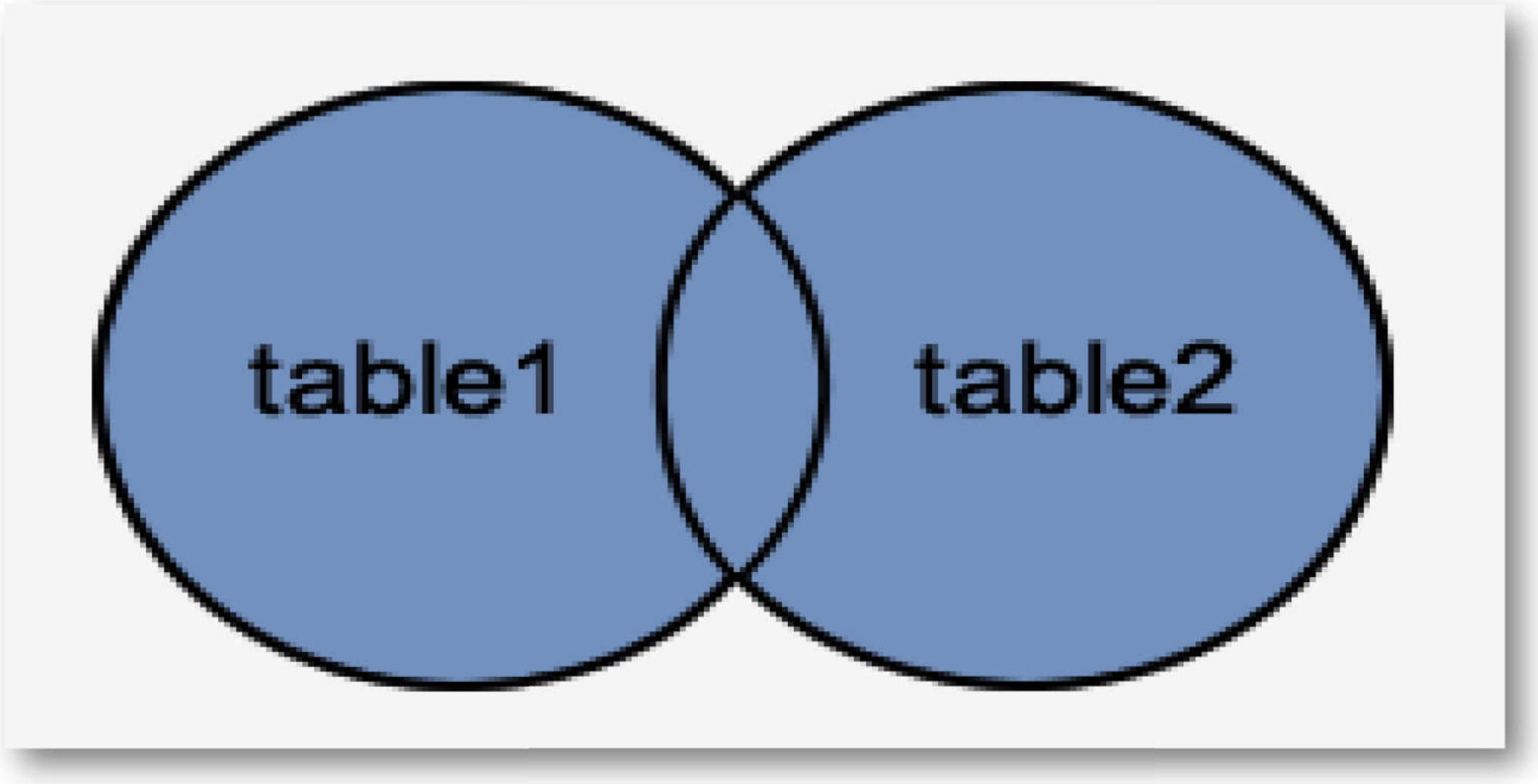
Another type of join is called a PostgreSQL FULL OUTER JOIN. This type of join returns all rows from the LEFT-hand table and RIGHT-hand table with nulls in place where the join condition is not met.

Syntax:

The syntax for the PostgreSQL FULL OUTER JOIN is:

```
SELECT columns
FROM table1
FULL OUTER JOIN table2
ON table1.column = table2.column;
```

The PostgreSQL FULL OUTER JOIN returns the shaded area:



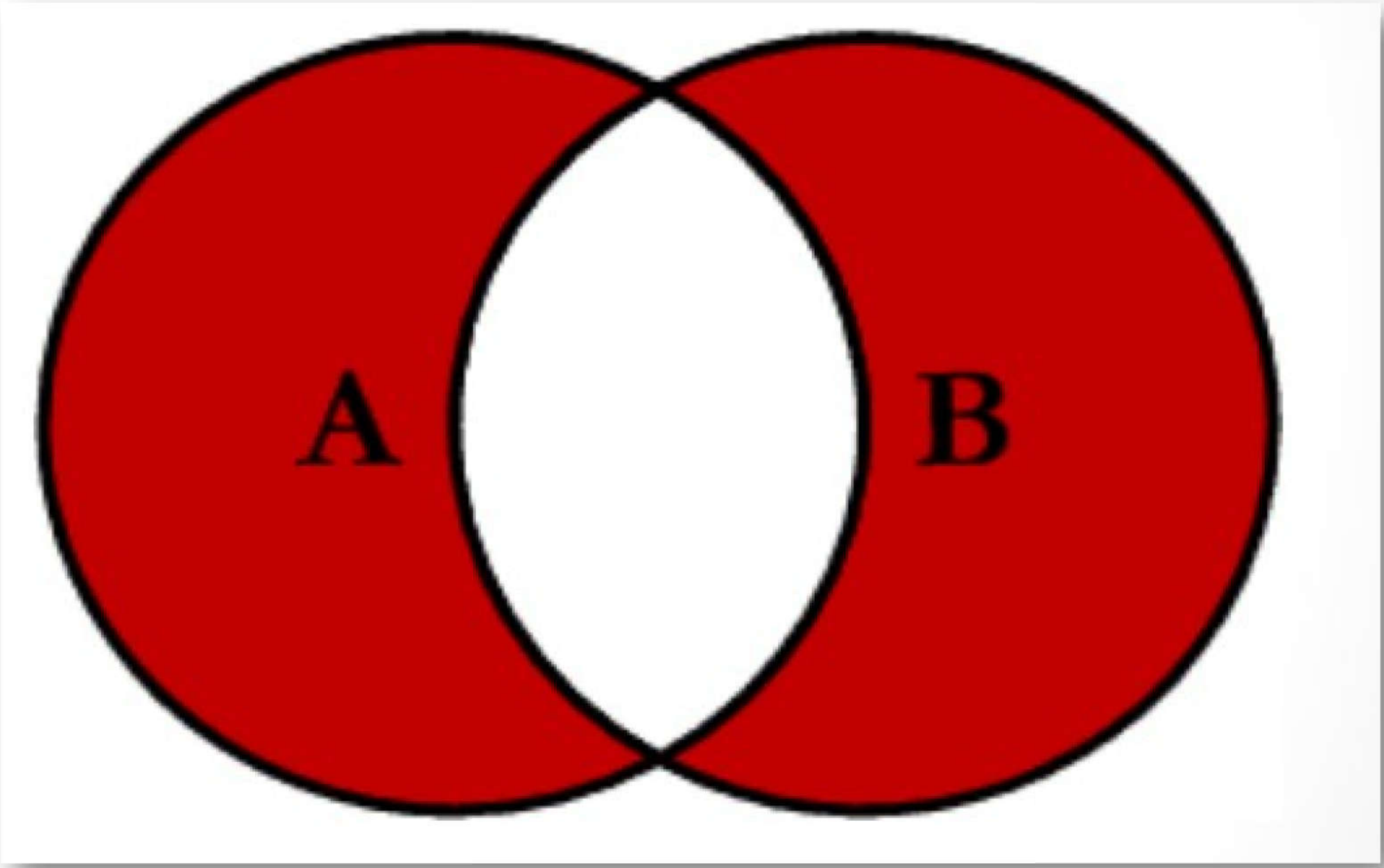
The PostgreSQL FULL OUTER JOIN would return the all records from both *table1* and *table2*.

Full Outer Excluding Inner Join

This query will return all of the records in Table A and Table B that do not have a matching record in the other table.

Syntax:

```
SELECT *
FROM TableA
FULL OUTER JOIN TableB
ON TableA.PK = TableB.PK
WHERE TableA.PK IS NULL OR TableB.PK IS NULL
```


A
B

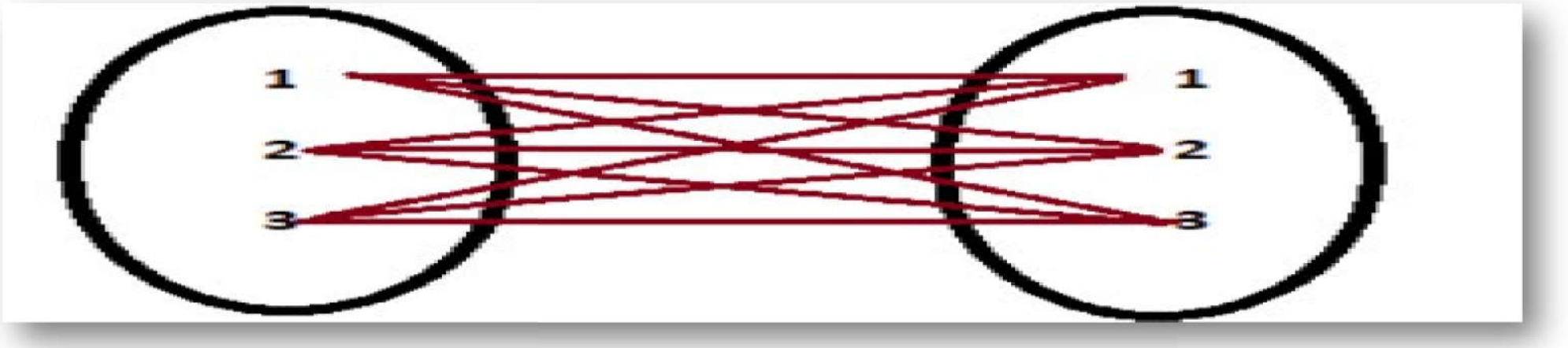
Cross Join

The cross join is returning every record in Table A combined with every record in Table B.

It gives the same results as not using a WHERE clause

Syntax:

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



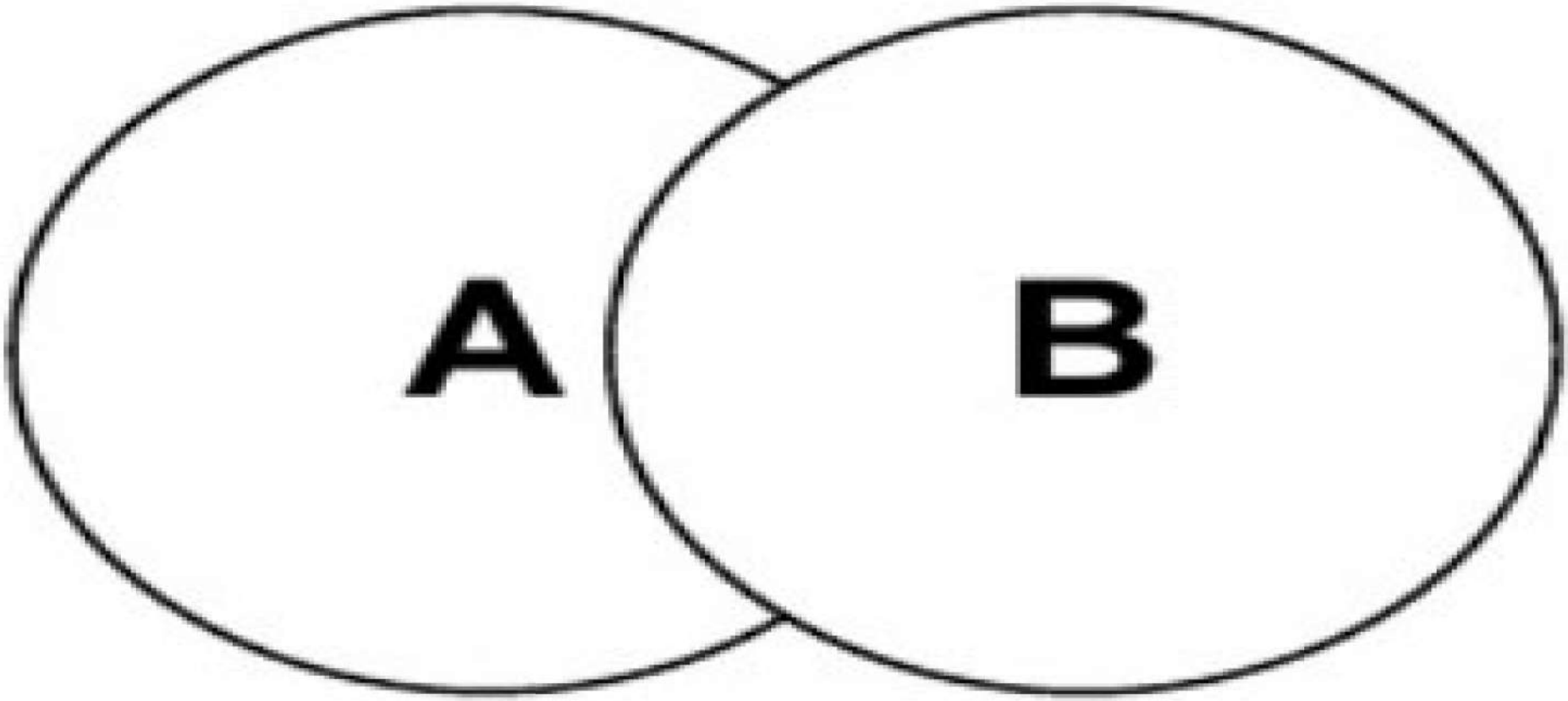
UNION

The Union operator returns rows from both tables. If used by itself, the UNION returns a distinct list of rows.

A **UNION** is useful when you want to sort results from two separate queries as one combined result.

Syntax:

```
SELECT *
FROM A
UNION
SELECT *
FROM B
```



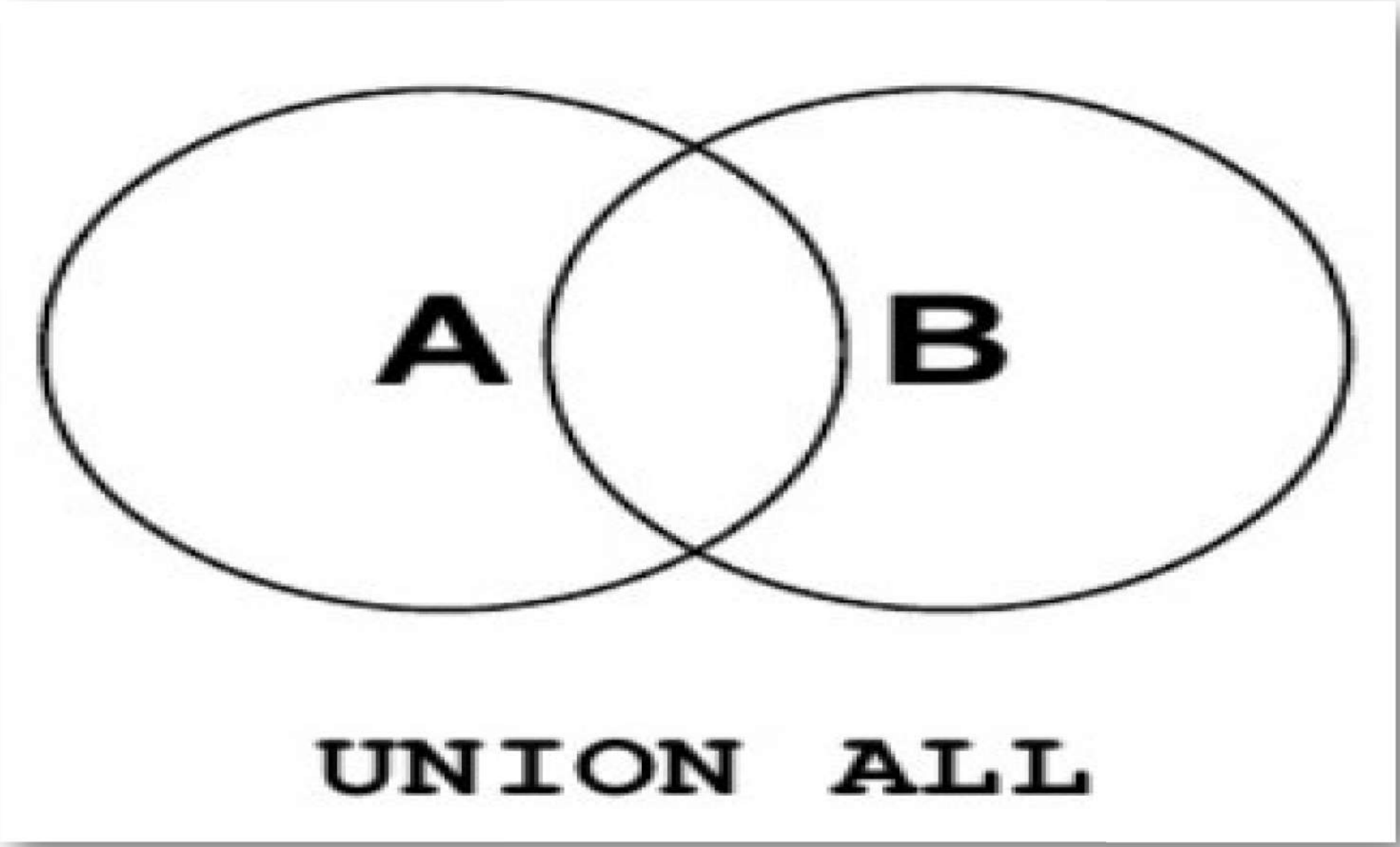
UNION

UNION ALL

Using UNION ALL returns all rows from both tables.

Syntax:

```
SELECT *
FROM A
UNION ALL
SELECT *
FROM B
```


A
B
UNION ALL

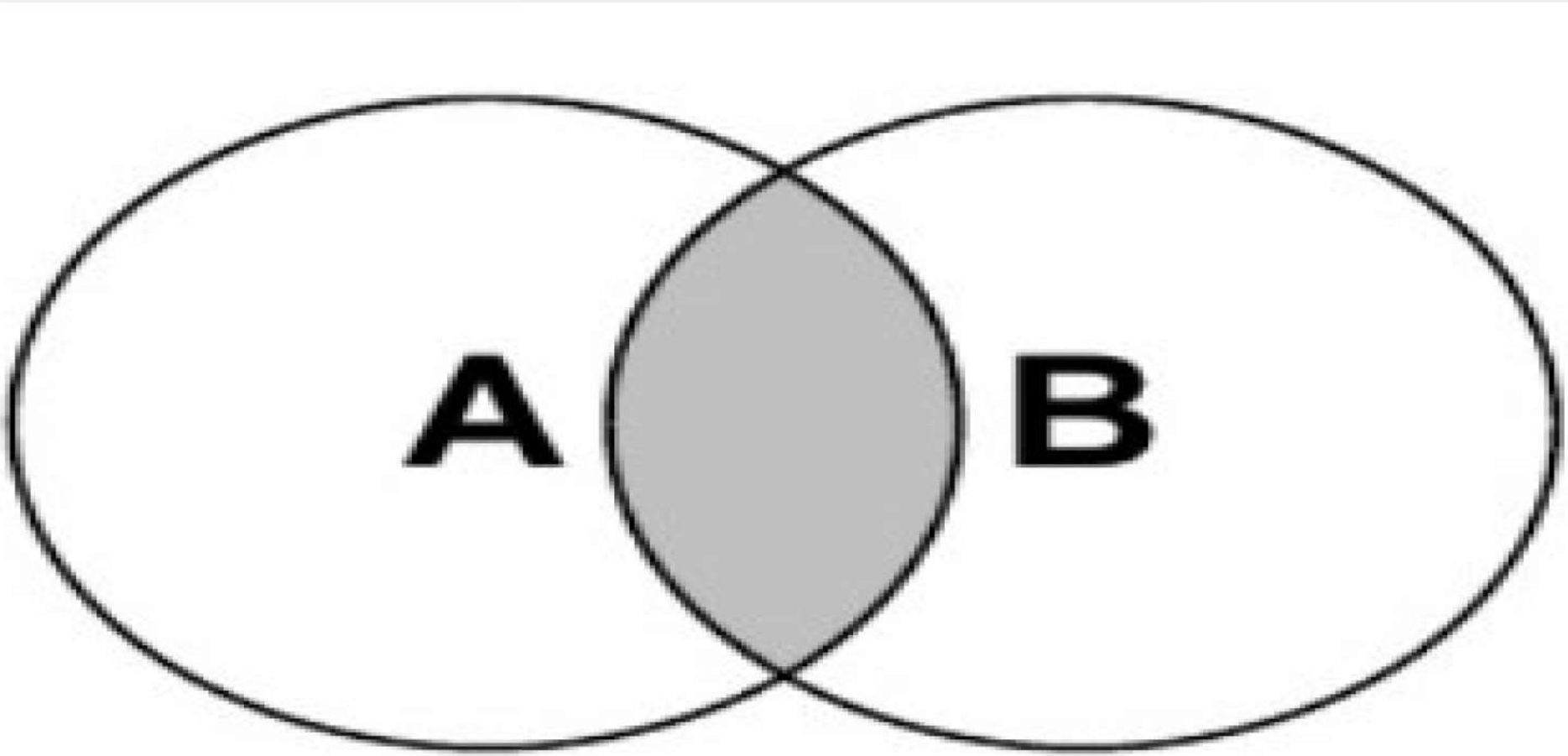
INTERSECT

Use an intersect operator to returns rows that are in common between two tables.

It returns unique rows from both the left and right queries. This query is useful when you want to find results that are in common between two queries.

Syntax:

```
SELECT *
FROM A
INTERSECT
SELECT *
FROM B
```



INTERSECT

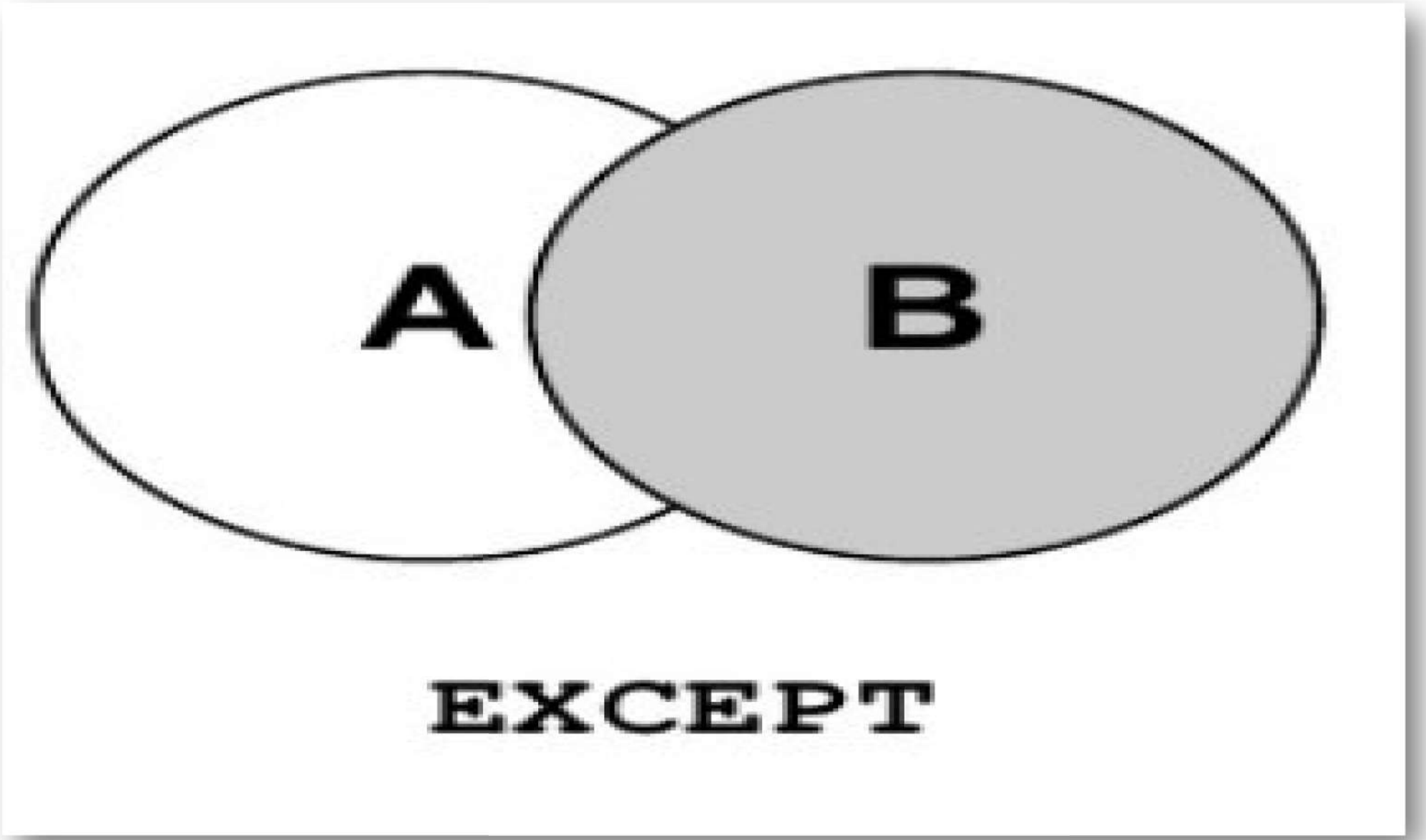
EXCEPT

Use the EXCEPT Operator to return only rows found in the left query. It returns unique rows from the left query that isn't in the right query's results.

This query is useful when you're looking to find rows that are in one set but not another.

Syntax:

```
SELECT *
FROM A
EXCEPT
SELECT *
FROM B
```


A
B
EXCEPT