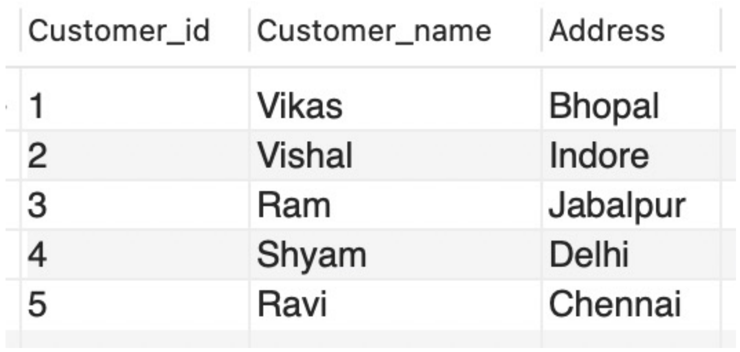
SQL JOINS

IOTA ACADEMY

CLASS OUTLINE

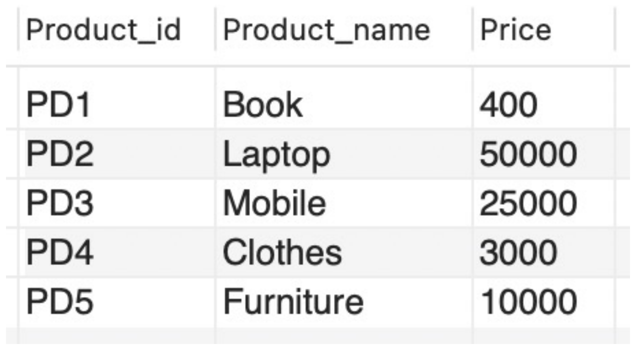
- Union Operator
- Joins and their significance
- Type of Joins in SQL
    - Inner Join
    - Left Outer Join
    - Right Outer Join
    - Full Outer Join
    - Cross Join
    - Self Join
- Examples of Joins

SAMPLE DATABASE TABLE:



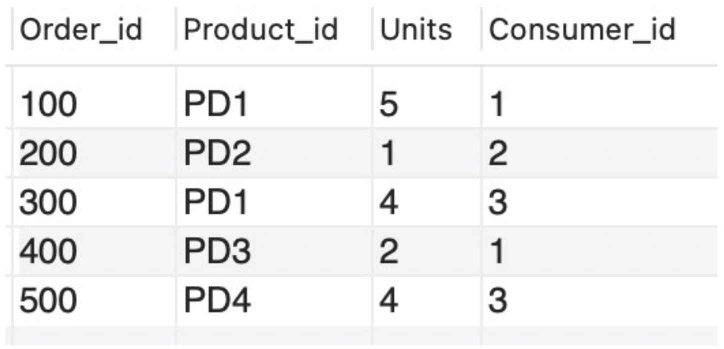
Customer_id	Customer_name	Address
1	Vikas	Bhopal
2	Vishal	Indore
3	Ram	Jabalpur
4	Shyam	Delhi
5	Ravi	Chennai

Customer_table



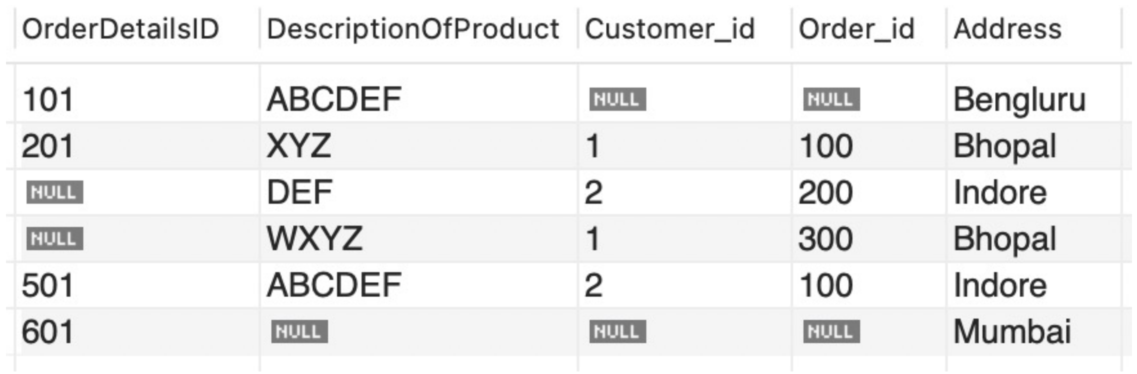
Product_id	Product_name	Price
PD1	Book	400
PD2	Laptop	50000
PD3	Mobile	25000
PD4	Clothes	3000
PD5	Furniture	10000

Product_table



Order_id	Product_id	Units	Consumer_id
100	PD1	5	1
200	PD2	1	2
300	PD1	4	3
400	PD3	2	1
500	PD4	4	3

Order_table



OrderDetailsID	DescriptionOfProduct	Customer_id	Order_id	Address
101	ABCDEF	NULL	NULL	Bengluru
201	XYZ	1	100	Bhopal
NULL	DEF	2	200	Indore
NULL	WXYZ	1	300	Bhopal
501	ABCDEF	2	100	Indore
601	NULL	NULL	NULL	Mumbai

Order_details

UNION OPERATOR

- UNION operator allows you **to combine two or more result sets** of SELECT queries into a **single result set.**

Some basic rules that you must follow:

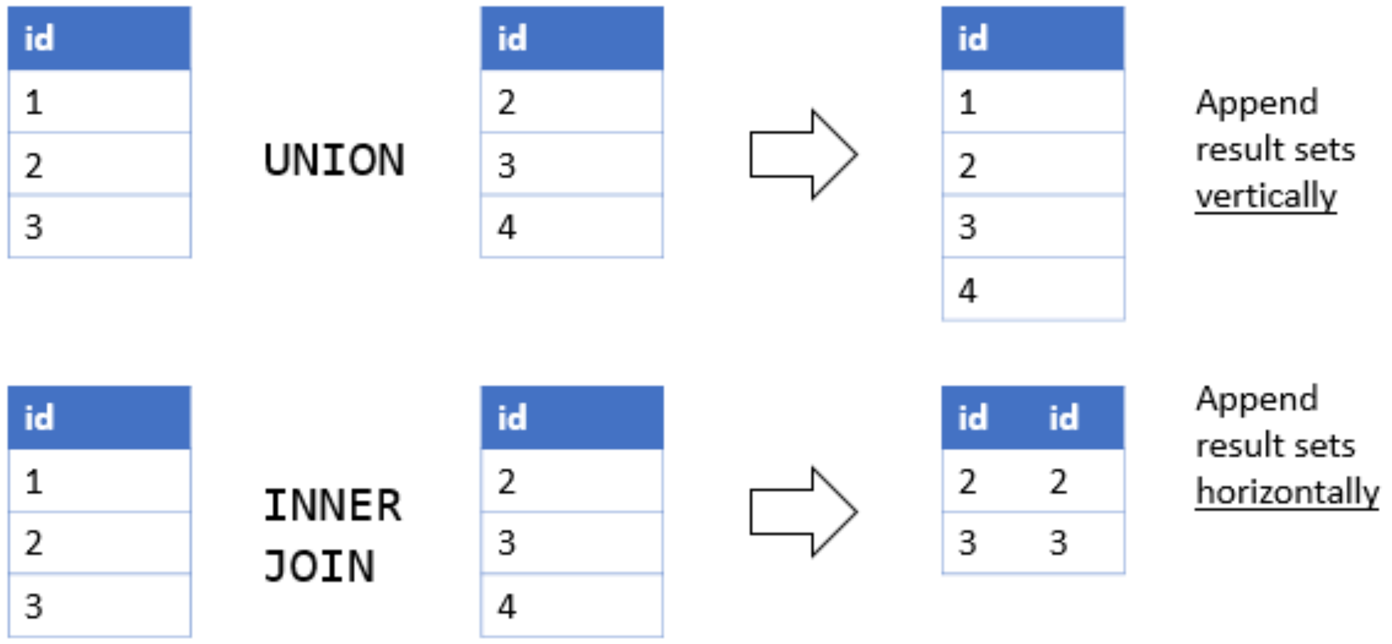
- First, the **number and the order of columns** that appear in all SELECT statements must be the same.
- Second, the data types of columns must be the same or compatible.

UNION OPERATOR…

- A JOIN combines result sets horizontally, a UNION appends result sets vertically.

SYNTAX:

```
SELECT column1, column2
FROM table_A
UNION
SELECT column1, column2
FROM table_B ;
```



id
1
2
3

UNION

id
2
3
4

id
1
2
3
4

Append result sets vertically

id
1
2
3

INNER JOIN

id
2
3
4

id	id
2	2
3	3

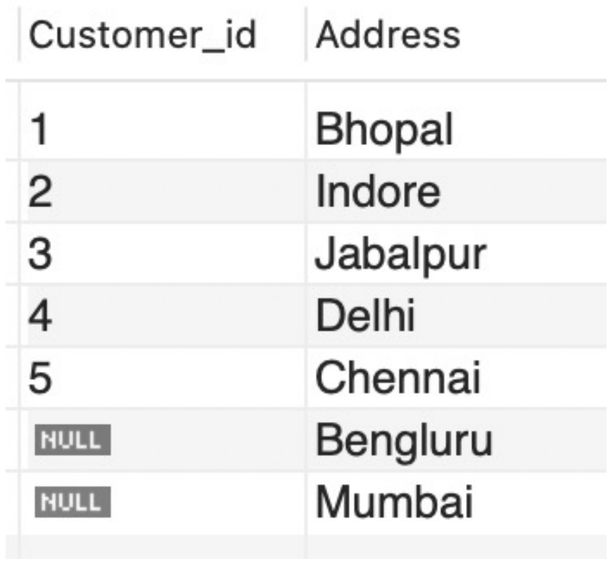
Append result sets horizontally

UNION OPERATOR…

Example:

```
#UNION
SELECT Customer_id, Address
FROM Customer_table
UNION
SELECT Customer_id, Address
FROM Order_details;
```

Output:

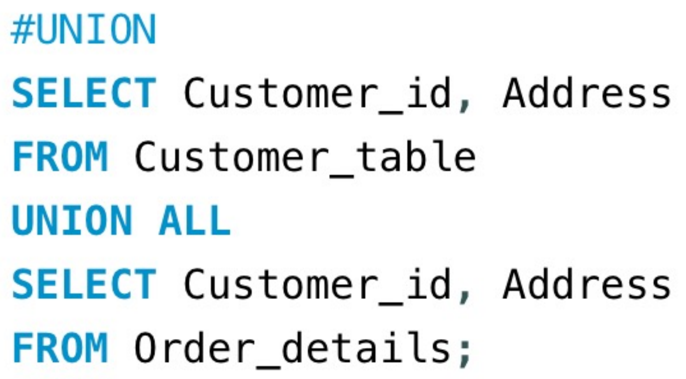


Customer_id	Address
1	Bhopal
2	Indore
3	Jabalpur
4	Delhi
5	Chennai
NULL	Bengluru
NULL	Mumbai

UNION ALL OPERATOR

- By default, the UNION operator only gives unique records even if we don’t specify the DISTINCT clause.
- The **final result set** contains the **distinct values** from separate result sets returned by the queries.
- If you use the UNION ALL explicitly all records will be part of the final result set including duplicate ones.

Example:



```
#UNION
SELECT Customer_id, Address
FROM Customer_table
UNION ALL
SELECT Customer_id, Address
FROM Order_details;
```

JOINS

- A relational database consists of **multiple related tables** linking together using common columns.
- SQL JOIN is used to fetch data from two or more tables, which are joined to appear as a single set of data.
- It is a method **of linking data between one or more tables** based on the values of common columns between the tables.
- In real-world scenarios to get complete information about any object, we need to query data from more than one table, this is where JOINS come to the rescue.

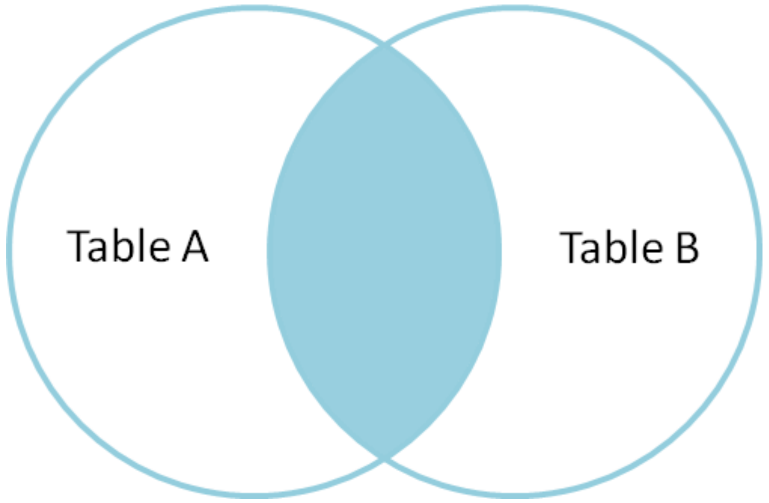
JOINS

Type of Joins in SQL

1. Inner Join
2. Outer Join
   1. Left Outer Join
   2. Right Outer Join
   3. Full Outer Join
3. Cross Join
4. Self Join

INNER JOIN

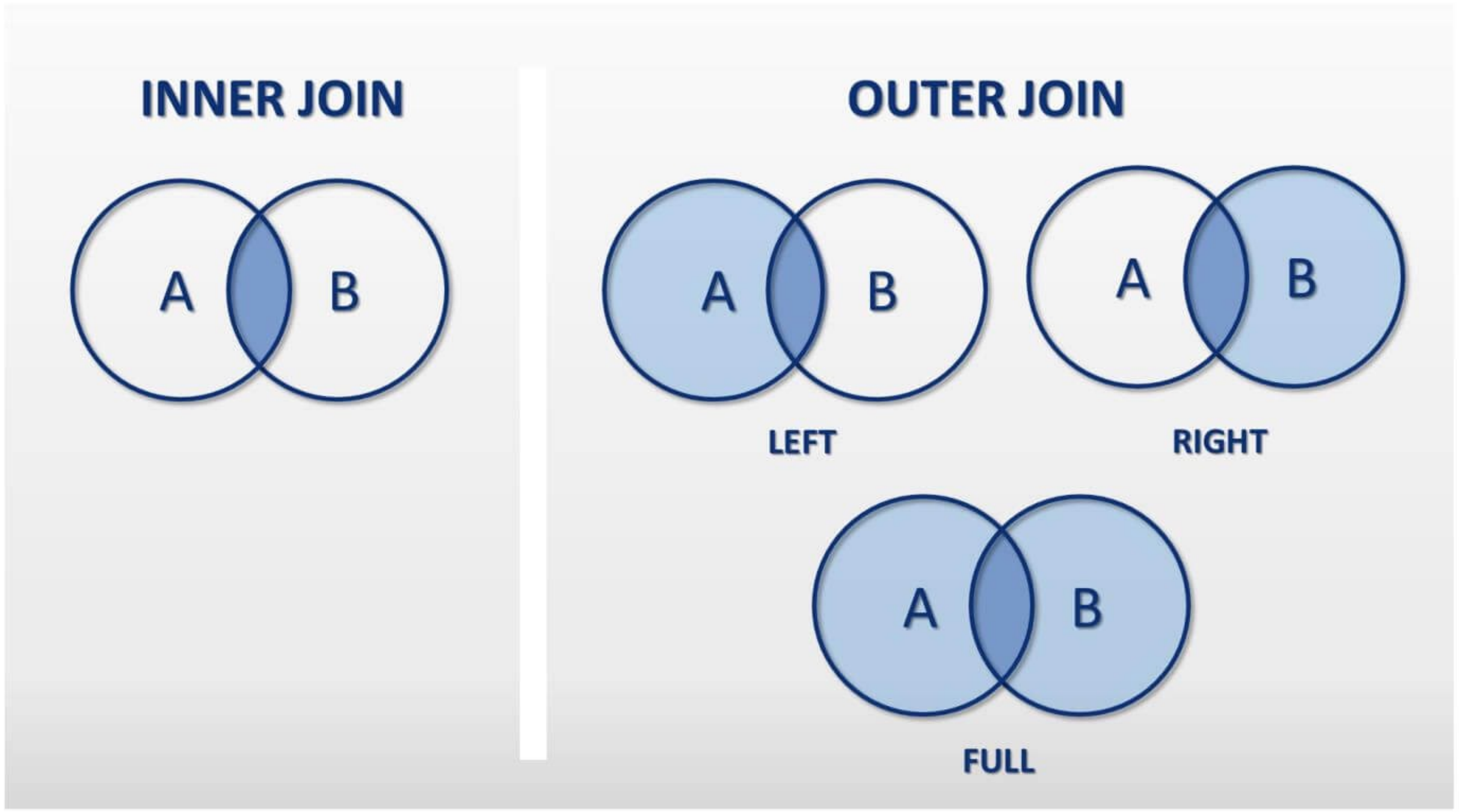
- The INNER JOIN keyword selects all rows from both tables as long as there is a match between the columns.
- This is a simple JOIN in which the **result is based on matched data** as per the equality condition specified in the SQL query.
- It returns only those rows (compared using a comparison operator) that satisfy the join condition.



OUTER JOINS

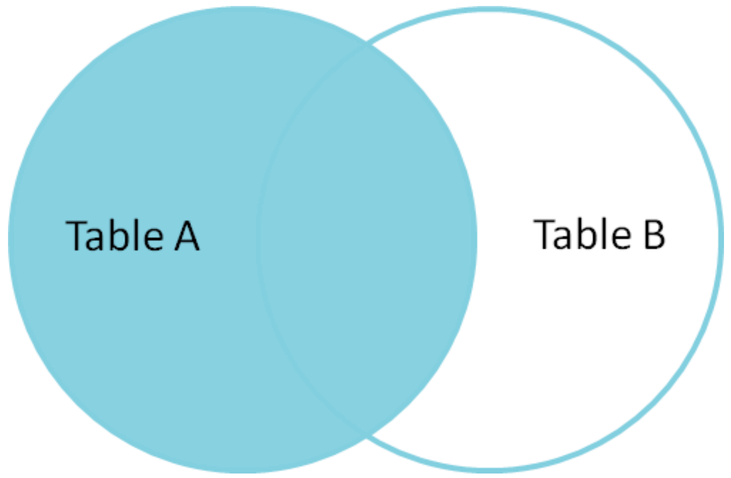
- An **OUTER JOIN**, not only outputs the data records of both tables that fulfil the selection condition like INNER JOIN (for example, the equality of the values of two columns), **but also all other tuples of one table or the other.**
- Here the **result is based on both matched data and unmatched data**.
- Types of Outer Join
  - Left Join
  - Right Join
  - Full Join

INNER AND OUTER JOINS



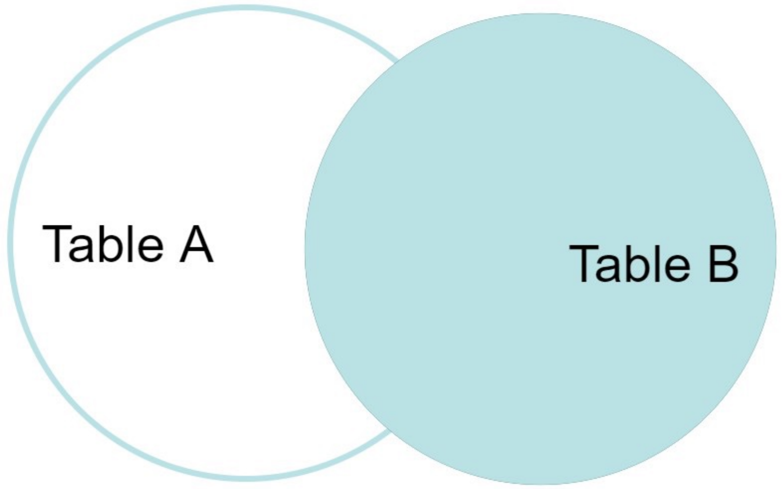
LEFT JOIN

- The LEFT JOIN keyword returns all records from the left table (Table A), and the matching records from the right table (Table B).
- The LEFT JOIN keyword returns all records from the left table, even if there are no matches in the right table.
- Missing values are set to NULL.



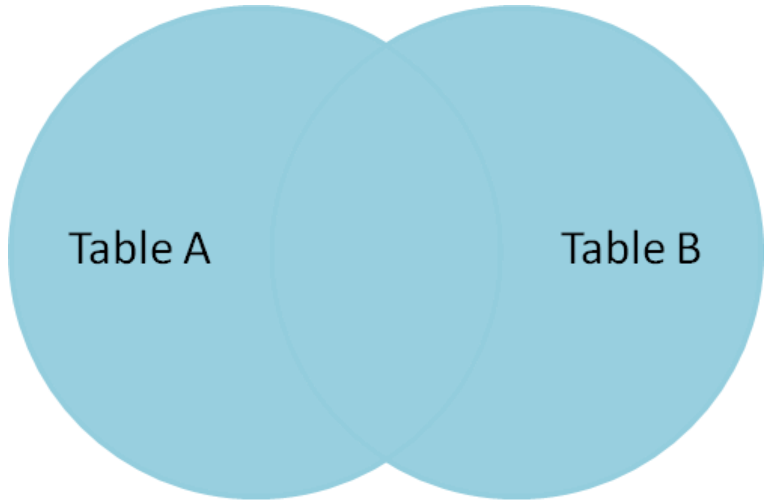
RIGHT JOIN

- It is similar to LEFT JOIN.
- The RIGHT JOIN keyword returns all records from the right table (Table B), and the matching records from the left table (Table A).
- The RIGHT JOIN keyword returns all records from the right table, even if there are no matches in the left table.



FULL OUTER JOIN

- A **FULL OUTER JOIN** is a combination of LEFT OUTER JOIN and RIGHT OUTER JOIN.
- It returns all the matched values as well as unmatched values from both tables.
- Missing values are set to NULL.



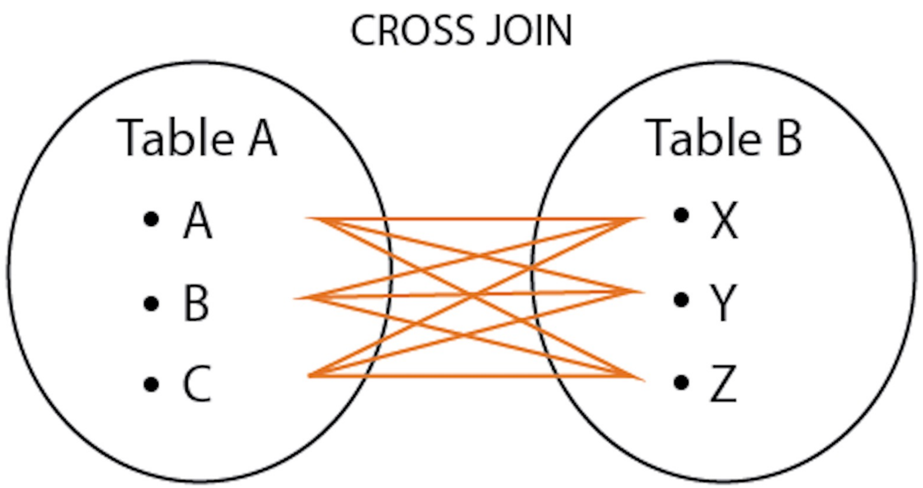
FULL OUTER JOIN

Note: MySQL does not support full outer join out of the box, unlike other databases such as PostgreSQL, and SQL Server.

- So we will need to do a full outer join using a combination of other join types such as LEFT JOIN ad RIGHT JOIN.

CROSS JOIN

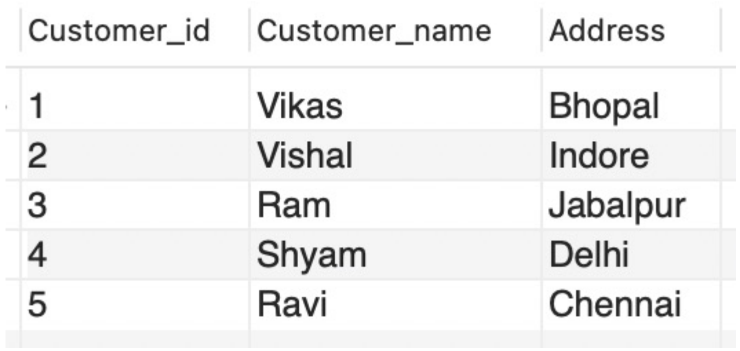
- Cross join is a cartesian product of two tables. It will connect all rows of the left table to each row of the right table.
- So, the query result of a cross-join is a number of rows in the left table multiplied by the number of rows in the right table.



SELF JOIN

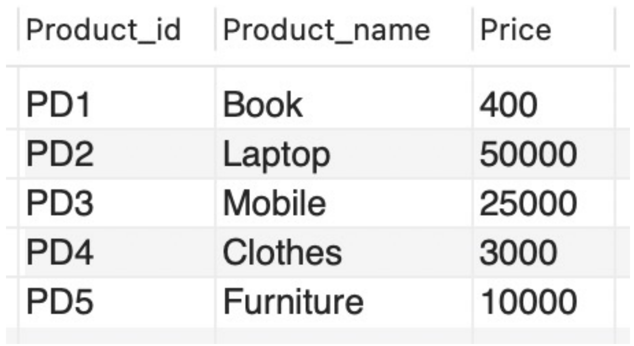
- A SELF JOIN is a join that is used to join a table with **itself**.
- The syntax of self-join is the same as the syntax of joining two different tables.
- One can apply all type of joins on a single table too by self-joining it.

SAMPLE DATABASE TABLE:



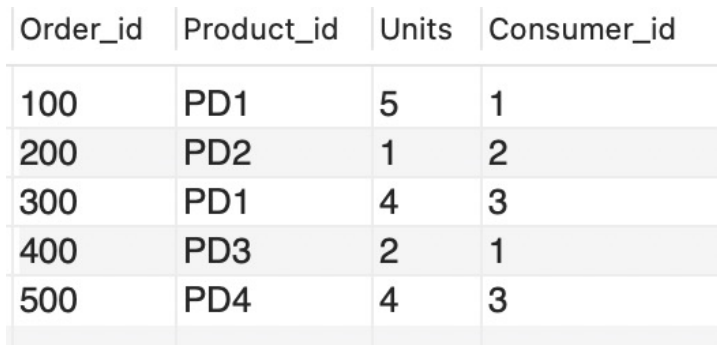
Customer_id	Customer_name	Address
1	Vikas	Bhopal
2	Vishal	Indore
3	Ram	Jabalpur
4	Shyam	Delhi
5	Ravi	Chennai

Customer_table



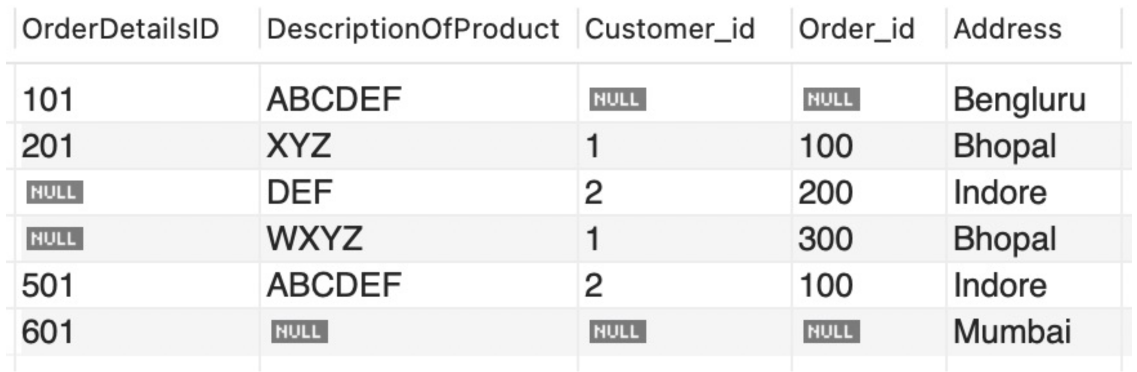
Product_id	Product_name	Price
PD1	Book	400
PD2	Laptop	50000
PD3	Mobile	25000
PD4	Clothes	3000
PD5	Furniture	10000

Product_table



Order_id	Product_id	Units	Consumer_id
100	PD1	5	1
200	PD2	1	2
300	PD1	4	3
400	PD3	2	1
500	PD4	4	3

Order_table



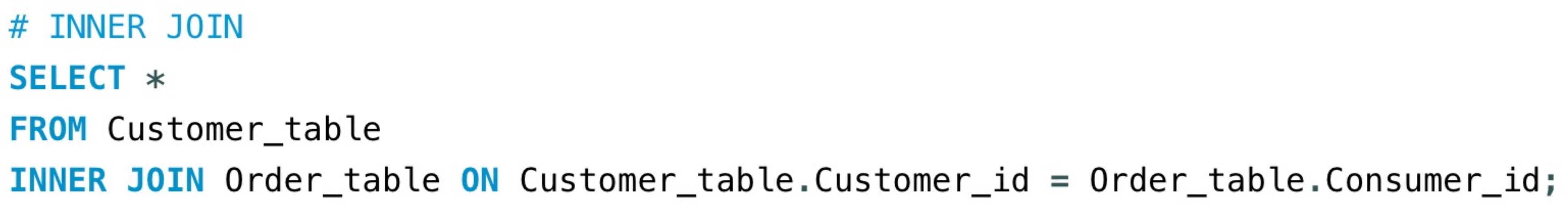
OrderDetailsID	DescriptionOfProduct	Customer_id	Order_id	Address
101	ABCDEF	NULL	NULL	Bengluru
201	XYZ	1	100	Bhopal
NULL	DEF	2	200	Indore
NULL	WXYZ	1	300	Bhopal
501	ABCDEF	2	100	Indore
601	NULL	NULL	NULL	Mumbai

Order_details

SYNTAX: INNER JOIN

SYNTAX:

SELECT *
FROM table_A
INNER JOIN table_B **ON** table_A.column1 **=** table_B.column1;

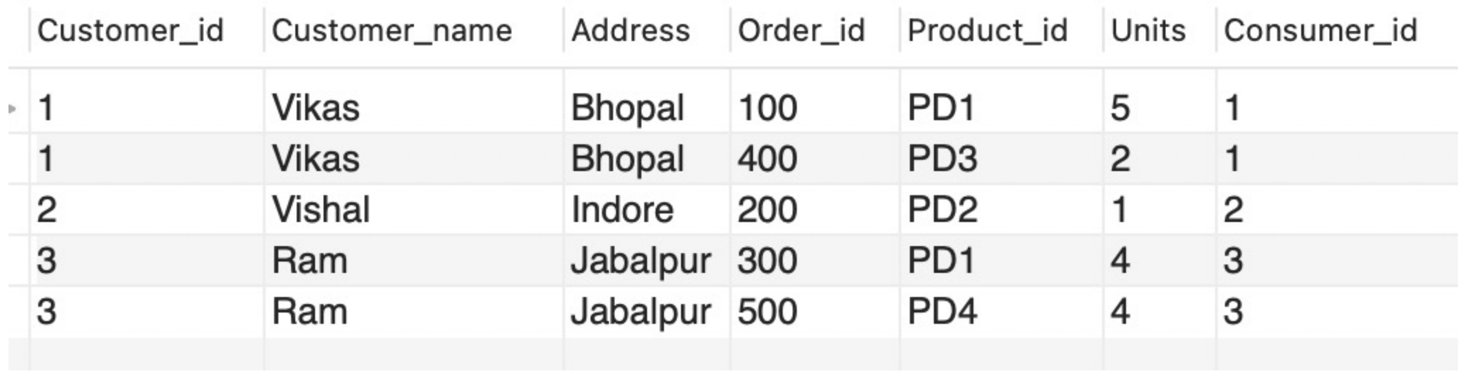


```
# INNER JOIN
SELECT *
FROM Customer_table
INNER JOIN Order_table ON Customer_table.Customer_id = Order_table.Consumer_id;
```

SYNTAX: INNER JOIN

```
# INNER JOIN
SELECT *
FROM Customer_table
INNER JOIN Order_table ON Customer_table.Customer_id = Order_table.Consumer_id;
```

Output:

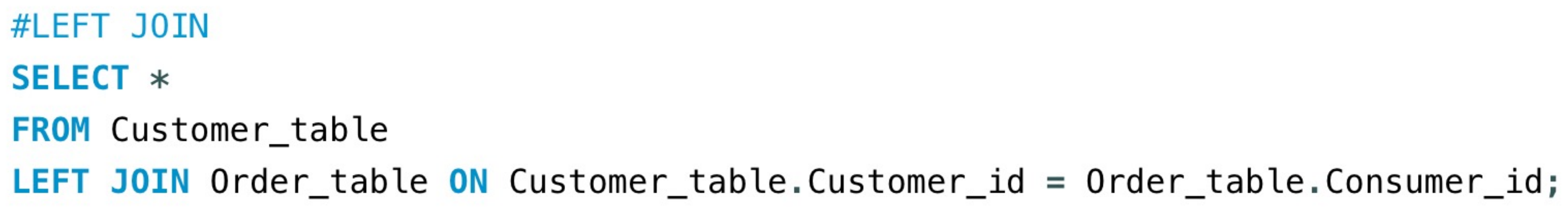


Customer_id	Customer_name	Address	Order_id	Product_id	Units	Consumer_id
1	Vikas	Bhopal	100	PD1	5	1
1	Vikas	Bhopal	400	PD3	2	1
2	Vishal	Indore	200	PD2	1	2
3	Ram	Jabalpur	300	PD1	4	3
3	Ram	Jabalpur	500	PD4	4	3

SYNTAX: LEFT JOIN

SYNTAX:

```
SELECT *
FROM table_A
LEFT JOIN table_B ON table_A.column1 = table_B.column1;
```

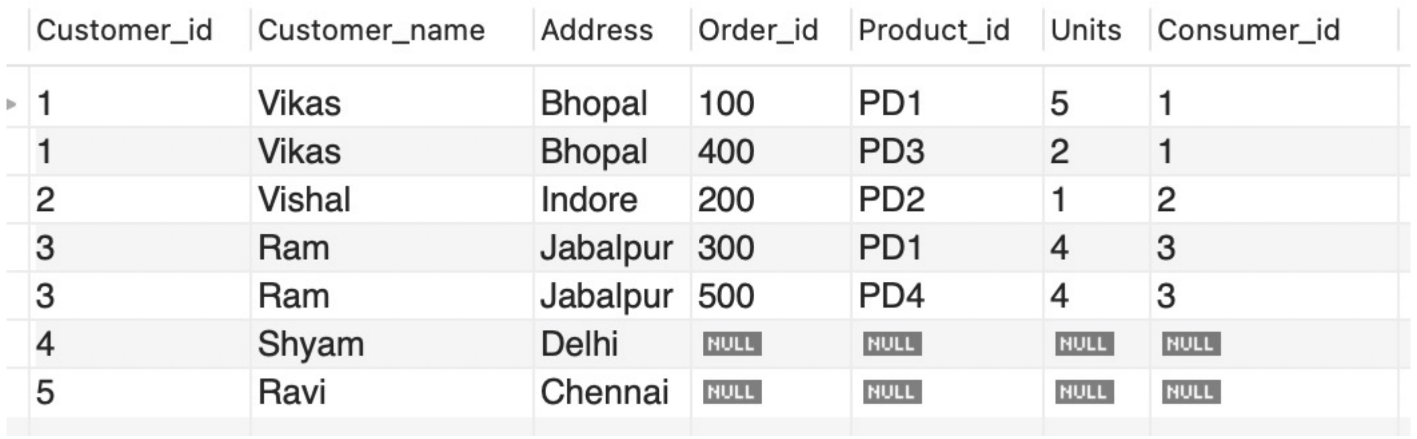


```
#LEFT JOIN
SELECT *
FROM Customer_table
LEFT JOIN Order_table ON Customer_table.Customer_id = Order_table.Consumer_id;
```

SYNTAX: LEFT JOIN

```
#LEFT JOIN
SELECT *
FROM Customer_table
LEFT JOIN Order_table ON Customer_table.Customer_id = Order_table.Consumer_id;
```

Output:

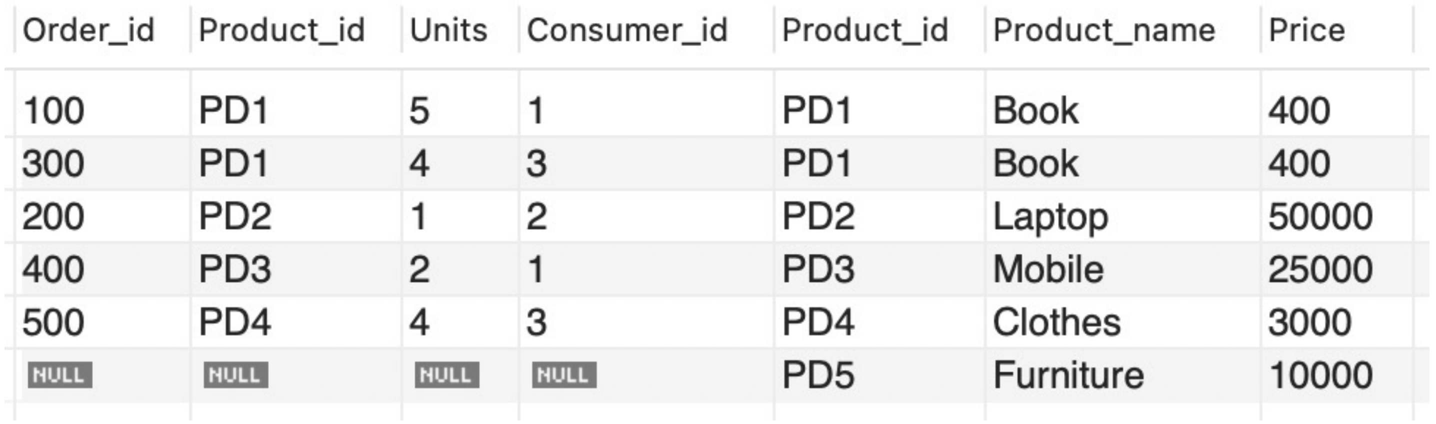


Customer_id	Customer_name	Address	Order_id	Product_id	Units	Consumer_id
1	Vikas	Bhopal	100	PD1	5	1
1	Vikas	Bhopal	400	PD3	2	1
2	Vishal	Indore	200	PD2	1	2
3	Ram	Jabalpur	300	PD1	4	3
3	Ram	Jabalpur	500	PD4	4	3
4	Shyam	Delhi	NULL	NULL	NULL	NULL
5	Ravi	Chennai	NULL	NULL	NULL	NULL

SYNTAX: RIGHT JOIN

```
#RIGHT JOIN
SELECT *
FROM Order_table
RIGHT JOIN Product_table ON Product_table.Product_id = Order_table.Product_id;
```

Output:

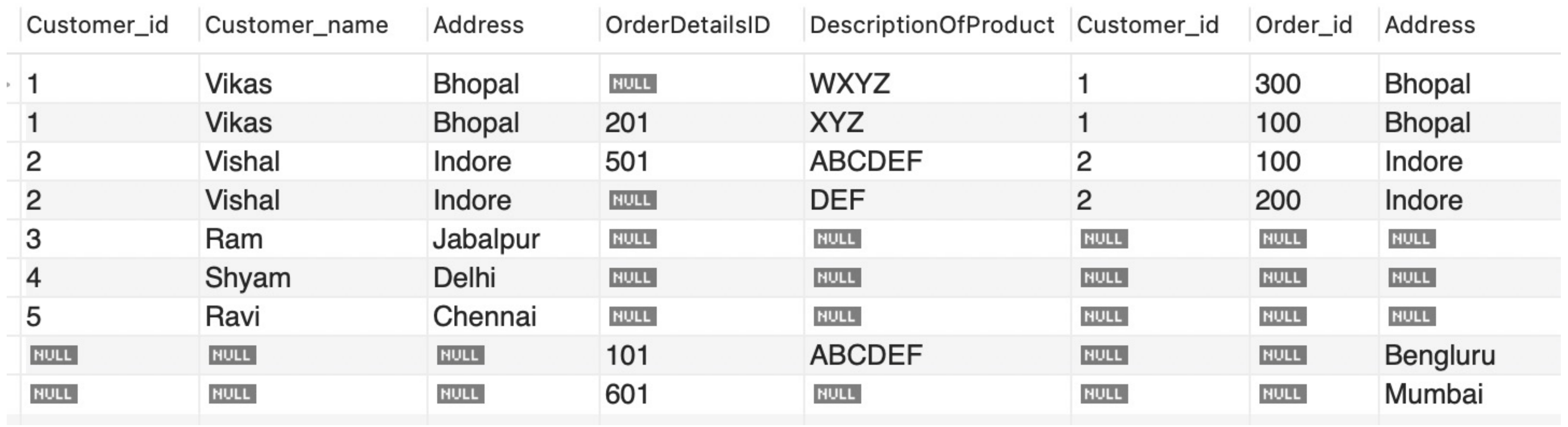


Order_id	Product_id	Units	Consumer_id	Product_id	Product_name	Price
100	PD1	5	1	PD1	Book	400
300	PD1	4	3	PD1	Book	400
200	PD2	1	2	PD2	Laptop	50000
400	PD3	2	1	PD3	Mobile	25000
500	PD4	4	3	PD4	Clothes	3000
NULL	NULL	NULL	NULL	PD5	Furniture	10000

SYNTAX: FULL JOIN

```
#FULL JOIN or FULL OUTER JOIN
SELECT *
FROM Customer_table
LEFT JOIN Order_details ON Customer_table.Customer_id= Order_details.Customer_id
UNION
SELECT *
FROM Customer_table
RIGHT JOIN Order_details ON Customer_table.Customer_id = Order_details.Customer_id;
```

Output:

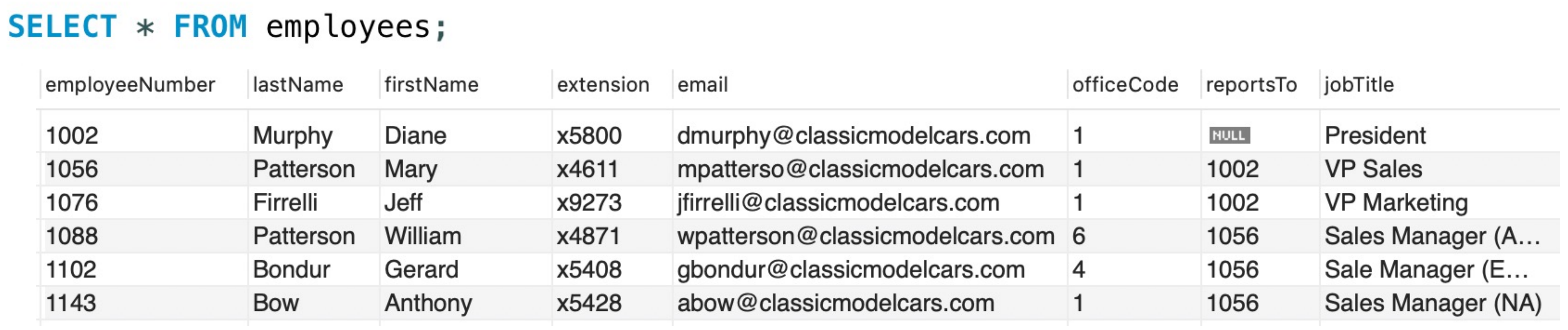


Customer_id	Customer_name	Address	OrderDetailsID	DescriptionOfProduct	Customer_id	Order_id	Address
1	Vikas	Bhopal	NULL	WXYZ	1	300	Bhopal
1	Vikas	Bhopal	201	XYZ	1	100	Bhopal
2	Vishal	Indore	501	ABCDEF	2	100	Indore
2	Vishal	Indore	NULL	DEF	2	200	Indore
3	Ram	Jabalpur	NULL	NULL	NULL	NULL	NULL
4	Shyam	Delhi	NULL	NULL	NULL	NULL	NULL
5	Ravi	Chennai	NULL	NULL	NULL	NULL	NULL
NULL	NULL	NULL	101	ABCDEF	NULL	NULL	Bengluru
NULL	NULL	NULL	601	NULL	NULL	NULL	Mumbai

SELF JOIN

- Joining a table with itself is called Self-Join.
- The self-join is often used to query **hierarchical data** or to **compare** a row with other rows within the same table.
- To perform a self-join, you **must use table aliases** to not repeat the same table name twice in a single query.

Example: Using famous 'Classic Models' Database



```sql
SELECT * FROM employees;
```

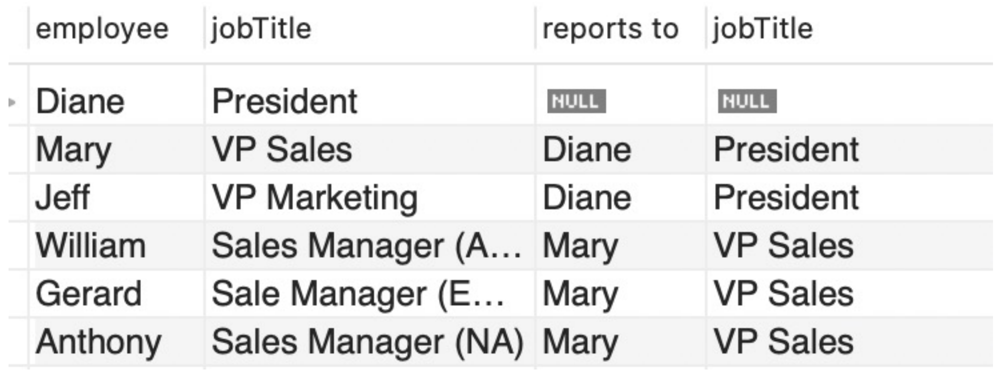
employeeNumber	lastName	firstName	extension	email	officeCode	reportsTo	jobTitle
1002	Murphy	Diane	x5800	dmurphy@classicmodelcars.com	1	NULL	President
1056	Patterson	Mary	x4611	mpatterso@classicmodelcars.com	1	1002	VP Sales
1076	Firrelli	Jeff	x9273	jfirrelli@classicmodelcars.com	1	1002	VP Marketing
1088	Patterson	William	x4871	wpatterson@classicmodelcars.com	6	1056	Sales Manager (A…
1102	Bondur	Gerard	x5408	gbondur@classicmodelcars.com	4	1056	Sale Manager (E…
1143	Bow	Anthony	x5428	abow@classicmodelcars.com	1	1056	Sales Manager (NA)

SYNTAX: SELF JOIN

- One way to self-join a table is why using INNER JOIN or LEFT JOIN syntax.
- Another way can be through equating common KEY using a WHERE clause.

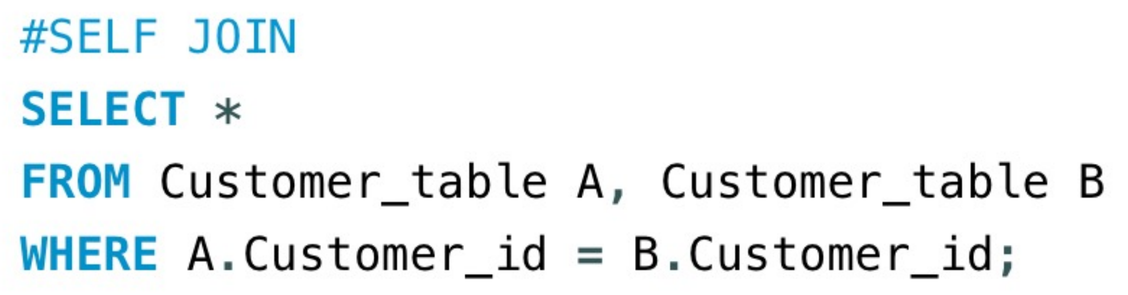
Question: Find out who reports to whom in an organization using the employees' table.

```
SELECT e.firstName AS 'employee', e.jobTitle , m.firstName AS 'reports to', m.jobTitle
FROM employees e
LEFT JOIN employees m ON m.employeeNumber = e.reportsto ;
```



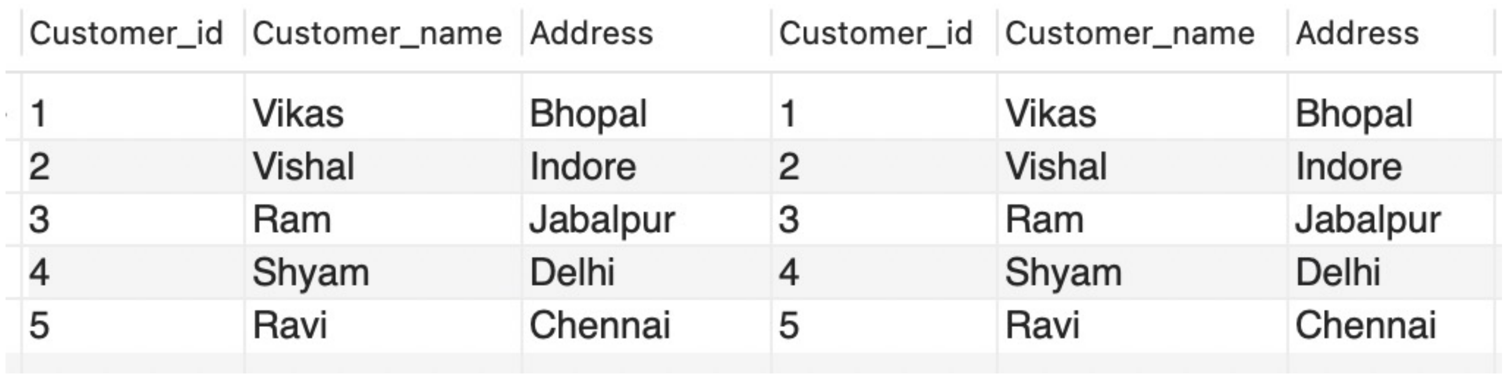
employee	jobTitle	reports to	jobTitle
Diane	President	NULL	NULL
Mary	VP Sales	Diane	President
Jeff	VP Marketing	Diane	President
William	Sales Manager (A…	Mary	VP Sales
Gerard	Sale Manager (E…	Mary	VP Sales
Anthony	Sales Manager (NA)	Mary	VP Sales

SYNTAX: SELF JOIN



```
#SELF JOIN
SELECT *
FROM Customer_table A, Customer_table B
WHERE A.Customer_id = B.Customer_id;
```

Output:

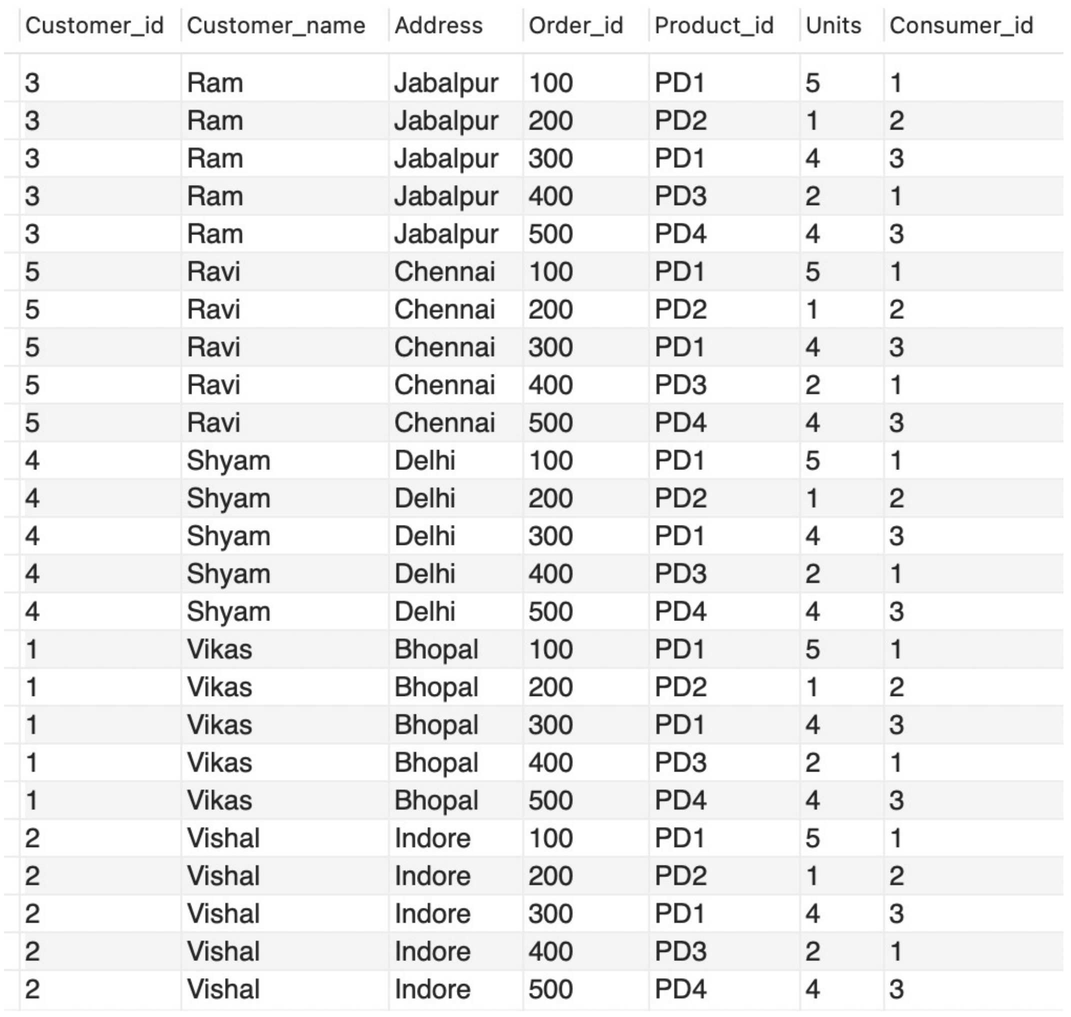


Customer_id	Customer_name	Address	Customer_id	Customer_name	Address
1	Vikas	Bhopal	1	Vikas	Bhopal
2	Vishal	Indore	2	Vishal	Indore
3	Ram	Jabalpur	3	Ram	Jabalpur
4	Shyam	Delhi	4	Shyam	Delhi
5	Ravi	Chennai	5	Ravi	Chennai

SYNTAX: CROSS JOIN

```
#CROSS JOIN
SELECT *
FROM Customer_table
CROSS JOIN Order_table
ORDER BY Customer_name;
```

Output:



Customer_id	Customer_name	Address	Order_id	Product_id	Units	Consumer_id
3	Ram	Jabalpur	100	PD1	5	1
3	Ram	Jabalpur	200	PD2	1	2
3	Ram	Jabalpur	300	PD1	4	3
3	Ram	Jabalpur	400	PD3	2	1
3	Ram	Jabalpur	500	PD4	4	3
5	Ravi	Chennai	100	PD1	5	1
5	Ravi	Chennai	200	PD2	1	2
5	Ravi	Chennai	300	PD1	4	3
5	Ravi	Chennai	400	PD3	2	1
5	Ravi	Chennai	500	PD4	4	3
4	Shyam	Delhi	100	PD1	5	1
4	Shyam	Delhi	200	PD2	1	2
4	Shyam	Delhi	300	PD1	4	3
4	Shyam	Delhi	400	PD3	2	1
4	Shyam	Delhi	500	PD4	4	3
1	Vikas	Bhopal	100	PD1	5	1
1	Vikas	Bhopal	200	PD2	1	2
1	Vikas	Bhopal	300	PD1	4	3
1	Vikas	Bhopal	400	PD3	2	1
1	Vikas	Bhopal	500	PD4	4	3
2	Vishal	Indore	100	PD1	5	1
2	Vishal	Indore	200	PD2	1	2
2	Vishal	Indore	300	PD1	4	3
2	Vishal	Indore	400	PD3	2	1
2	Vishal	Indore	500	PD4	4	3

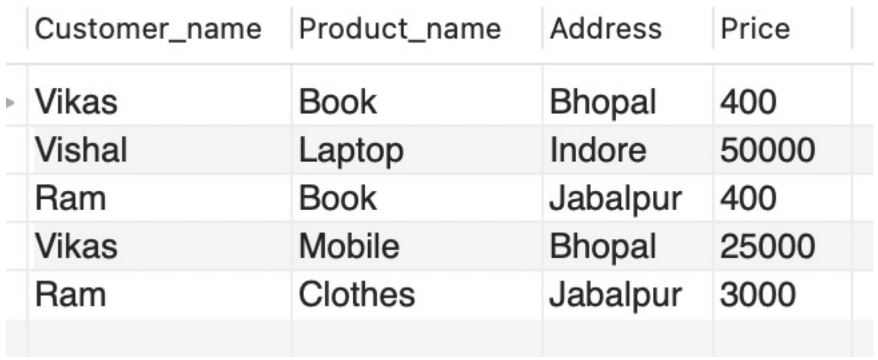
NOTE: Total records = records(tableA) * records(tableB)

EXAMPLES: JOINS

Multiple JOINS example:

```
SELECT Customer_table.Customer_name, Product_table.Product_name, Customer_table.Address, Product_table.Price
FROM Customer_table
INNER JOIN Order_table ON Customer_table.Customer_id = Order_table.Consumer_id
INNER JOIN Product_table ON Order_table.Product_id = Product_table.Product_id;
```

Output:



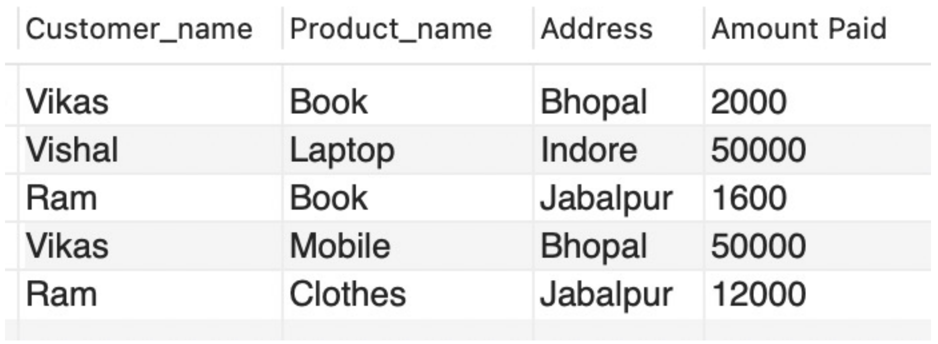
Customer_name	Product_name	Address	Price
Vikas	Book	Bhopal	400
Vishal	Laptop	Indore	50000
Ram	Book	Jabalpur	400
Vikas	Mobile	Bhopal	25000
Ram	Clothes	Jabalpur	3000

EXAMPLES: JOINS…

Multiple JOINS example: Creating a new column calculating the amount paid

```
SELECT Customer_table.Customer_name, Product_table.Product_name, Customer_table.Address,
Product_table.Price*Order_table.Units  AS `Amount Paid`
FROM Customer_table
INNER JOIN Order_table ON Customer_table.Customer_id = Order_table.Consumer_id
INNER JOIN Product_table ON Order_table.Product_id = Product_table.Product_id;
```

Output:



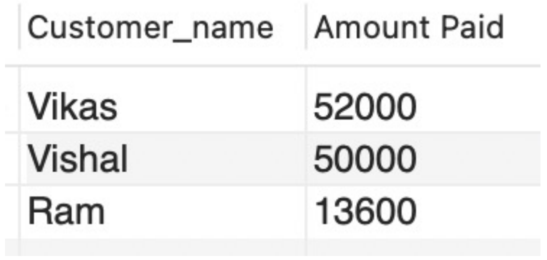
Customer_name	Product_name	Address	Amount Paid
Vikas	Book	Bhopal	2000
Vishal	Laptop	Indore	50000
Ram	Book	Jabalpur	1600
Vikas	Mobile	Bhopal	50000
Ram	Clothes	Jabalpur	12000

EXAMPLES: JOINS…

Multiple JOINS example: GROUP BY Clause

```sql
SELECT Customer_table.Customer_name, SUM(Product_table.Price*Order_table.Units) AS `Amount Paid`
FROM Customer_table
INNER JOIN Order_table ON Customer_table.Customer_id = Order_table.Consumer_id
INNER JOIN Product_table ON Order_table.Product_id = Product_table.Product_id
GROUP BY Customer_table.Customer_name;
```

Output:



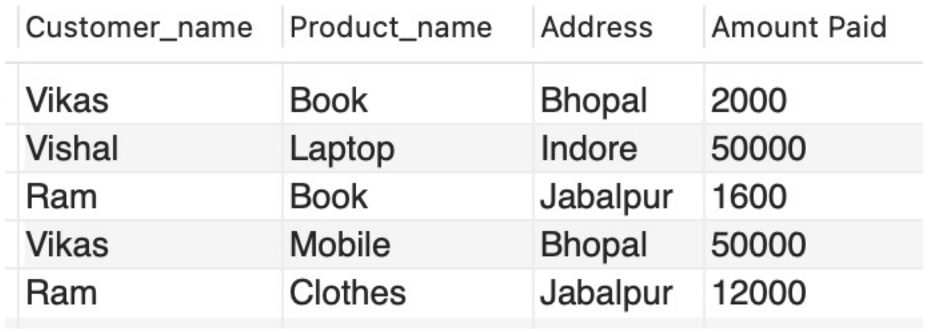
Customer_name	Amount Paid
Vikas	52000
Vishal	50000
Ram	13600

EXAMPLES: JOINS…

Multiple JOINS example: Using Aliases to shorten & simplify query

```sql
SELECT C.Customer_name, P.Product_name, C.Address, P.Price*O.Units AS `Amount Paid`
FROM (Customer_table AS C)
INNER JOIN (Order_table AS O) ON C.Customer_id = O.Consumer_id
INNER JOIN (Product_table AS P) ON O.Product_id = P.Product_id;
```

Output:



Customer_name	Product_name	Address	Amount Paid
Vikas	Book	Bhopal	2000
Vishal	Laptop	Indore	50000
Ram	Book	Jabalpur	1600
Vikas	Mobile	Bhopal	50000
Ram	Clothes	Jabalpur	12000

THANK YOU