JOIN will combine rows from different tables if the join condition is true.

SELECT \*

FROM orders

JOIN subscriptions

ON orders.subscription\_id = subscriptions.subscription\_id;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

LEFT JOIN will return every row in the lefttable, and if the join condition is not met, NULL values are used to fill in the columns from the right table.

A left join will keep all rows from the first table, regardless of whether there is a matching row in the second table.

Join newspaper table and online table on their id columns

SELECT \*

FROM newspaper

JOIN online

ON newspaper.id = online.id;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Primary key is a column that serves a unique identifier for the rows in the table.

Primary keys have a few requirements:

* None of the values can be NULL.
* Each value must be unique (i.e., you can’t have two customers with the same customer\_id in the customers table).
* A table can not have more than one primary key column.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Foreign key is a column that contains the primary key to another table.

When the primary key for one table appears in a different table, it is called a **foreign key**.

(So customer\_id is a primary key when it appears in customers, but a foreign key when it appears in orders.)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

CROSS JOIN lets us combine all rows of one table with all rows of another table.

SELECT shirts.shirt\_color, pants.pants\_color

FROM shirts

CROSS JOIN pants;

SELECT \*

FROM newspaper

CROSS JOIN months

WHERE start\_month <= month AND end\_month >= month;

SELECT month,

COUNT(\*) AS 'subscribers'

FROM newspaper

CROSS JOIN months

WHERE start\_month <= month

AND end\_month >= month

GROUP BY month;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

UNION stacks one dataset on top of another.

* Tables must have the same number of columns.
* The columns must have the same data types in the same order as the first table.

SELECT \* FROM table1 UNION SELECT \* FROM table2;

SELECT \*

FROM newspaper

UNION

SELECT \*

FROM online;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

WITH allows us to define one or more temporary tables that can be used in the final query.

Often times, we want to combine two tables, but one of the tables is the result of another calculation.

WITH previous\_results AS ( SELECT ... ... ... ... )

SELECT \* FROM previous\_results JOIN customers ON \_\_\_\_\_ = \_\_\_\_\_;

*Question:*

inside parentheses (), and give it name previous\_query, Join the temporary table previous\_query with customers table and select the following columns:

* customers.customer\_name
* previous\_query.subscriptions

*Answer:*

WITH previous\_query AS (

SELECT customer\_id,

COUNT(subscription\_id) AS 'subscriptions'

FROM orders

GROUP BY customer\_id

)

SELECT customers.customer\_name,

previous\_query.subscriptions

FROM previous\_query

JOIN customers

ON previous\_query.customer\_id = customers.customer\_id;