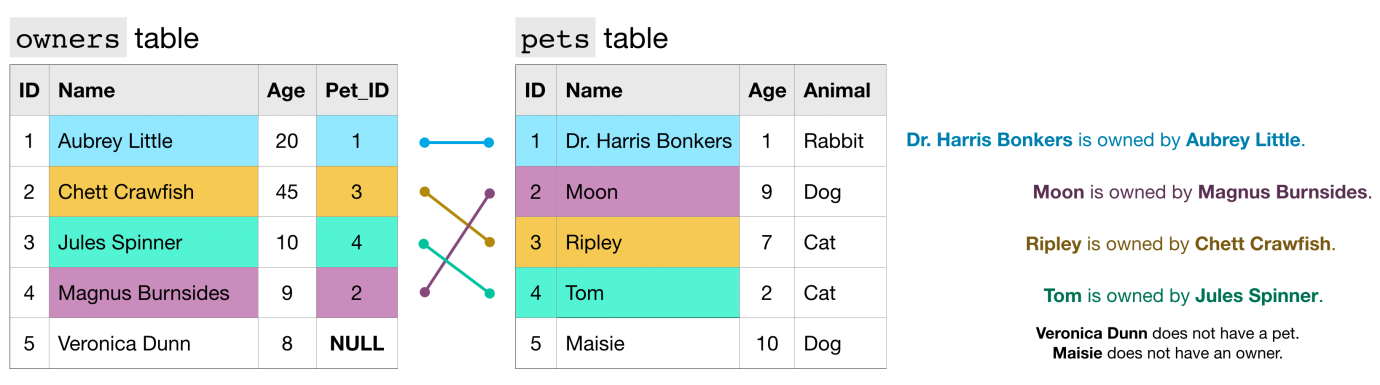
Advanced SQL: 1st lesson – Joins & Unions

In the Intro to SQL micro-course, you learned how to use INNER JOIN to consolidate information from two different tables. Now you'll learn about a few more types of JOIN, along with how to use UNIONs to pull information from multiple tables. Along the way, we'll work with two imaginary tables, called owners and pets.



Each row of the owners table identifies a different pet owner, where the ID column is a unique identifier. The Pet\_ID column (in the owners table) contains the ID for the pet that belongs to the owner (this number matches the ID for the pet from the pets table). For example:

* The pets table shows that Dr. Harris Bonkers is the pet with ID 1.
* The owners table shows that Aubrey Little is the owner of the pet with ID 1.

Putting these two facts together, Dr. Harris Bonkers is owned by Aubrey Little. Likewise, since Veronica Dunn does not have a corresponding Pet\_ID, she does not have a pet. And, since 5 does not appear in the Pet\_ID column, Maisie does not have an owner.

JOIN functions:

Recall that we can use an INNER JOIN to pull rows from both tables where the value in the Pet\_ID column in the owners table has a match in the ID column of the pets table.

query = “””

SELECT o.Name AS Owner\_Name, p.Name AS Pet\_Name

FROM `bigquery-public-data.pet\_records.owners` AS o

INNER JOIN `bigquery-public-data.pet\_records.pets` AS p

ON p.ID = o.Pet\_ID

“””

--------------------- ----------------------

Owner\_Name Pet\_Name

--------------------- ----------------------

Aubrey Little Dr. Harris Bonkers

Magnus Burnsides Moon

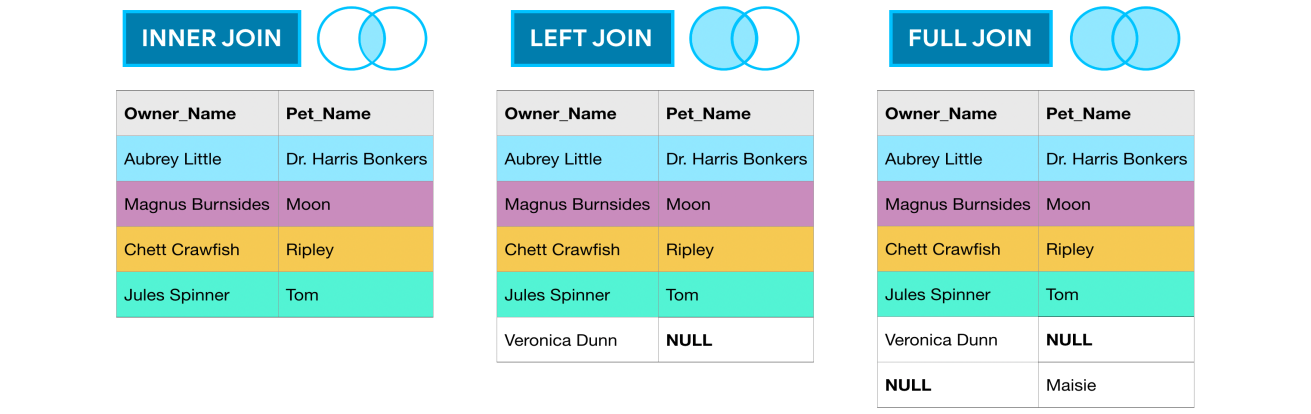
Chett Crawfish Ripley

Jules Spinner Tom

In this case, Veronica Dunn and Maisie are not included in the results. But what if we instead want to create a table containing all pets, regardless of whether they have owners? Or, what if we want to combine all of the rows in both tables? In these cases, we need only use a different type of JOIN. For instance, to create a table containing all rows from the owners table, we use a LEFT JOIN. In this case, "left" refers to the table that appears before the JOIN in the query ("Right" refers to the table that is after the JOIN).



* Replacing INNER JOIN in the query above with LEFT JOIN returns all rows where the two tables have matching entries, along with all of the rows in the left table (whether there is a match or not).
* If we instead use a RIGHT JOIN, we get the matching rows, along with all rows in the right table (whether there is a match or not).
* Finally, a FULL JOIN returns all rows from both tables. Note that in general, any row that does not have a match in both tables will have NULL entries for the missing values. You can see this in the image below.



UNION function:

As you've seen, JOINs horizontally combine results from different tables. If you instead would like to vertically concatenate columns, you can do so with a UNION. The example query below combines the Age columns from both tables.



Note that with a UNION, the data types of both columns must be the same, but the column names can be different (So, for instance, we cannot take the UNION of the Age column from the owners table and the Pet\_Name column from the pets table).

We use UNION ALL to include duplicate values - you'll notice that 9 appears in both the owners table and the pets table, and shows up twice in the concatenated results. If you'd like to drop duplicate values, you need only change UNION ALL in the query to UNION DISTINCT.