Introduction to SQL: 3rd lesson – Group By, Having & Count

Now that you can select raw data, you're ready to learn how to group your data and count things within those groups. This can help you answer questions like:

* How many of each kind of fruit has our store sold?
* How many species of animal has the vet office treated?

To do this, you'll learn about three new techniques: GROUP BY, HAVING and COUNT().

COUNT() function:

COUNT(), as you may have guessed from the name, returns a count of things. If you pass it the name of a column, it will return the number of entries in that column. For instance, if we SELECT the COUNT() of the ID column in the pets table, it will return 4, because there are 4 ID's in the table.

query = “””

SELECT COUNT(ID)

FROM `bigquery-public-data.pet\_records.pets`

“””

-----

f0\_

-----

4

COUNT() is an example of an aggregate function, which takes many values and returns one. (Other examples of aggregate functions include SUM(), AVG(), MIN(), and MAX().) As you'll notice in the picture above, aggregate functions introduce strange column names (like f0\_\_). Later in this tutorial, you'll learn how to change the name to something more descriptive.

GROUP BY clause:

GROUP BY takes the name of one or more columns, and treats all rows with the same value in that column as a single group when you apply aggregate functions like COUNT(). For example, say we want to know how many of each type of animal we have in the pets table. We can use GROUP BY to group together rows that have the same value in the Animal column, while using COUNT() to find out how many ID's we have in each group.

query = “””

SELECT Animal, COUNT(ID)

FROM `bigquery-public-data.pet\_records.pets`

GROUP BY Animal

“””

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

Animal f0\_

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

Rabbit 1

Dog 1

Cat 2

It returns a table with three rows (one for each distinct animal). We can see that the pets table contains 1 rabbit, 1 dog, and 2 cats.

GROUP BY … HAVING clause:

HAVING is used in combination with GROUP BY to ignore groups that don't meet certain criteria. So this query, for example, will only include groups that have more than one ID in them.

query = “””

SELECT Animal, COUNT(ID)

FROM `bigquery-public-data.pet\_records.pets`

GROUP BY Animal

HAVING COUNT(ID) > 1

“””

Since only one group meets the specified criterion, the query will return a table with only one row.

Note on using GROUP BY:

Note that because it tells SQL how to apply aggregate functions (like COUNT()), it doesn't make sense to use GROUP BY without an aggregate function. Similarly, if you have any GROUP BY clause, then all variables must be passed to either a GROUP BY command, or an aggregation function. Consider the query below.

query\_good = “””

SELECT parent, COUNT(id)

FROM `bigquery-public-data.hacker\_news.full`

GROUP BY parent

“””

Note that there are two variables: parent and id

* parent was passed to a GROUP BY command (in GROUP BY parent), and
* id was passed to an aggregate function (in COUNT(id)).

And this query won't work, because the author column isn't passed to an aggregate function or a GROUP BY clause.

query\_bad = “””

SELECT `by` AS author, parent, COUNT(id)

FROM `bigquery-public-data.hacker\_news.full`

GROUP BY parent

“””

If make this error, you'll get the error message:

SELECT list expression references column (column's name) which is neither grouped nor aggregated at.

You may notice the `by` column in this query is surrounded by backticks. This is because BY is a reserved keyword used in clauses including GROUP BY. In BigQuery reserved keywords used as identifiers must be quoted in backticks to avoid an error. We also make subsequent references to this column more readable by adding an alias to rename it to author.