The DISTINCT command

This lesson demonstrates how to use the DISTINCT command in a query to return only distinct values.

> Distinct values
>
> Values that are unique and have no duplicates in a given set.

The **DISTINCT** command is a straightforward and versatile SQL command. It can be used during data exploration or to query for unique values. For example, when you're looking through a table of data, you may want to know how many possible distinct values there are in a particular column. You can use **DISTINCT** to do this.

Similarly, if you're executing a query that performs a calculation, you may only want to perform the calculation on unique values and not on any duplicate values.

```
1  SELECT DISTINCT(column_name)
2  FROM table_name;
```

In pgAdmin, take a look at the common table in the fueleconomy database. Executing a SELECT * FROM query shows all the columns in the table. As you can see, the make column contains lots of different car makes.

Data Output | Explain | Messages | Notifications

	make character varying	model character varying	n smallint	years smallint
1	Acura	Integra	42	16
2	Acura	Legend	28	10
3	Acura	MDX 4WD	12	12
4	Acura	NSX	28	14
5	Acura	TSX	27	11
6	Audi	A4	49	19
7	Audi	A4 Avant quattro	49	15
8	Audi	A4 quattro	66	19
9	Audi	A6	20	19
10	Audi	A6 Avant quattro	12	12
11	Audi	A6 quattro	46	20

Say you want to know which distinct car makes are contained in this table. This is a pretty large table of data, so you want to avoid scrolling through the dataset to find all the car makes. Instead of scrolling, you can use the DISTINCT command to return all the distinct values in the make column. Your query should look like this:

Query Editor | Query History

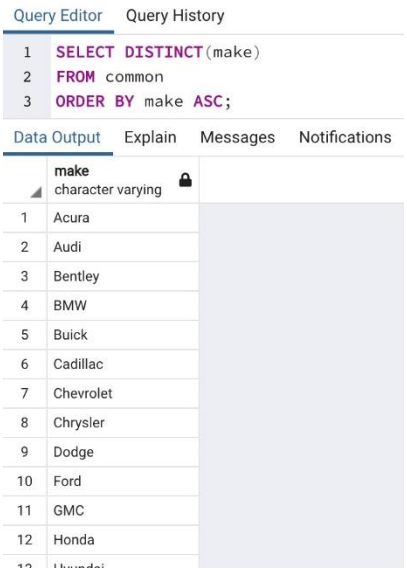
```
1  SELECT DISTINCT(make)
2  FROM common;
```

Data Output | Explain | Messages | Notifications

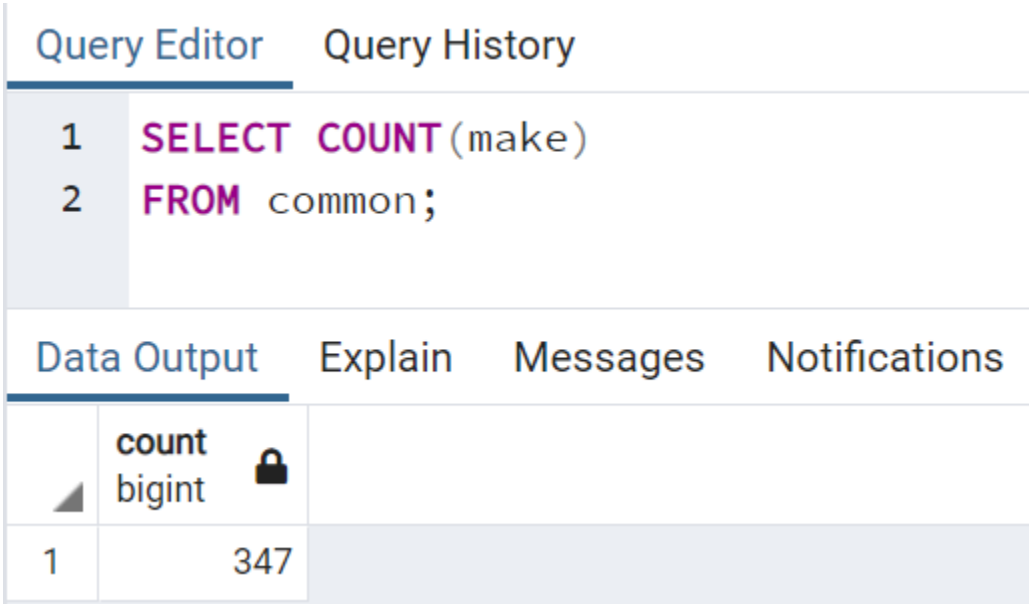
	make character varying
1	Lincoln
2	Honda
3	Ford
4	Scion
5	Maserati
6	Dodge
7	Chevrolet
8	Saturn
9	Infiniti
10	MINI
11	Bentley
12	Pontiac

So, rather than sifting through all the repeated values of car makes to find the unique ones, you can use **DISTINCT** to return this information immediately.

If you like, you can take this one step further by putting the return values in alphabetical order by including the **ORDER BY** clause, like this:



DISTINCT is also useful if you want to perform numeric calculations on the data and you're only interested in performing the calculation on the unique values. For example, say you want to count how many unique car makes there are in the make column. If you use the simple COUNT query below, you get a result of 347.



But from what you saw in the data table initially, there are many rows with repeat car makes. So, this count of 347 overstates the number of distinct car makes because it includes duplicates. In order to see only unique car makes, add the DISTINCT command to your query, as shown below.

Query Editor | Query History

```
1  SELECT COUNT(DISTINCT(make))
2  FROM common;
```

Data Output | Explain | Messages | Notifications

	count bigint
1	42

Using DISTINCT excludes all the duplicate values for car makes, and now the count is 42 distinct car makes.