MySQL Exercise 7: Joining Tables with Inner Joins

Before completing these exercises, I strongly recommend that you watch the video called "What are Joins?" that describe what joins are, and how different types of joins work.

As one of the last building blocks we need to address our Dognition analysis questions, in this lesson we will learn how to combine tables using inner joins.

1. Inner Joins between 2 tables

To begin, load the sql library, connect to the Dognition database, and set the Dognition database as the default.

In [1]:
```
%load_ext sql
%sql mysql://studentuser:studentpw@mysqlserver/dognitiondb
%sql USE dognitiondb
```

```
0 rows affected.
```

Out[1]: []

Recall that tables in relational databases are linked through primary keys and sometimes other fields that are common to multiple tables (as is the case with our Dognition data set). Our goal when we execute a JOIN or make a joined table is to use those common columns to let the database figure out which rows in one table match up to which rows in another table. Once that mapping is established using at least one common field or column, the database can pull any columns you want out of the mapped, or joined, tables and output the matched data to one common table.

An inner join is a join that outputs only rows that have an exact match in both tables being joined:

To illustrate how this works, let's find out whether dog owners that are particularly surprised by their dog's performance on Dognition tests tend to own similar breeds (or breed types, or breed groups) of dogs. There are many ways to address this question, but let's start by focusing on the dog owners who provided at least 10 ratings for one or more of their dogs in the ratings table. Of these owners, which 200 owners reported the highest average amount of surprise at their dog's performance, and what was the breed, breed_type, and breed_group of each of these owner's dog?

The surprise ratings are stored in the reviews table. The dog breed information is provided in the dogs table. There are two columns that are common to both tables: user_guid and dog_guid. How do we use the common columns to combine information from the two tables?

To join the tables, you can use a WHERE clause and add a couple of details to the FROM clause so that the database knows from what table each field in your SELECT clause comes.

First, start by adding all the columns we want to examine to the SELECT statement:

```
SELECT dog_guid AS DogID, user_guid AS UserID, AVG(rating) AS AvgRating,
       COUNT(rating) AS NumRatings, breed, breed_group, breed_type
```

then list all the tables from which the fields we are interested in come, separated by commas (with no comma at the end of the list):

```
FROM dogs, reviews
```

then add the other restrictions:

```
GROUP BY user_guid, dog_guid, breed, breed_group, breed_type
HAVING NumRatings >= 10
ORDER BY AvgRating DESC
LIMIT 200
```

Try running this query and see what happens:

In [ ]:
```
%%sql
SELECT dogs.dog_guid AS DogID, dogs.user_guid AS UserId, AVG(reviews.rating) A
S AvgRating, COUNT(reviews.rating) AS NumRatings, dogs.breed, dogs.breed_grou
p, dogs.breed_type
FROM dogs, reviews
GROUP BY dogs.user_guid, dogs.dog_guid, dogs.breed, dogs.breed_group, dogs.bre
ed_type
HAVING NumRatings >= 10
ORDER BY AvgRating DESC
LIMIT 200
```

You should receive an error message stating that the identity of dog_guid and user_guid in the field list is ambiguous. The reason is that the column title exists in both tables, and MySQL doesn't know which one we want. We have to specify the table name before stating the field name, and separate the two names by a period (NOTE: read this entire section before deciding whether you want to execute this query)<mark>:

```
SELECT dogs.dog_guid AS DogID, dogs.user_guid AS UserID, AVG(reviews.rating) AS Avg
Rating,
       COUNT(reviews.rating) AS NumRatings, dogs.breed, dogs.breed_group, dogs.bree
d_type
FROM dogs, reviews
GROUP BY dogs.user_guid, dogs.dog_guid, dogs.breed, dogs.breed_group, dogs.breed_ty
pe
HAVING NumRatings >= 10
ORDER BY AvgRating DESC
LIMIT 200
```

You can also take advantage of aliases so that you don't have to write out the name of the tables each time. Here I will introduce another syntax for aliases that omits the AS completely. In this syntax, the alias is whatever word (or phrase, if you use quotation marks) follows immediately after the field or table name, separated by a space. Some people find it easier to read queries if you use the syntax without an AS to create table aliases, but retain the syntax with an AS to create field aliases (but both syntaxes work for field aliases and table aliases). So we could write:

```
SELECT d.dog_guid AS DogID, d.user_guid AS UserID, AVG(r.rating) AS AvgRating,
       COUNT(r.rating) AS NumRatings, d.breed, d.breed_group, d.breed_type
FROM dogs d, reviews r
GROUP BY DogID, UserID, d.breed, d.breed_group, d.breed_type
HAVING NumRatings >= 10
ORDER BY AvgRating DESC
LIMIT 200
```

I am tempted to tell you to run this query so that you will see what happens, but instead, I will explain what will happen and let you decide if you want to see what the output looks...and feels...like.

There is nothing built into the database table definitions that can instruct the server how to combine the tables on its own (remember, this is how relational databases save space and remain flexible). Further, the query as written does not tell the database how the two tables are related. As a consequence, rather than match up the two tables according to the values in the user_id and/or dog_id column, the database will do the only thing it knows how to do which is output every single combination of the records in the dogs table with the records in the reviews table. In other words, every single row of the dogs table will get paired with every single row of the reviews table. This is known as a Cartesian product. Not only will it be a heavy burden on the database to output a table that has the full length of one table multiplied times the full length of another (and frustrating to you, because the query would take a very long time to run), the output would be close to useless.

To prevent this from happening, tell the database how to relate the tables in the WHERE clause:

```
SELECT d.dog_guid AS DogID, d.user_guid AS UserID, AVG(r.rating) AS AvgRating,
       COUNT(r.rating) AS NumRatings, d.breed, d.breed_group, d.breed_type
FROM dogs d, reviews r
WHERE d.dog_guid=r.dog_guid
GROUP BY UserID, DogID, d.breed, d.breed_group, d.breed_type
HAVING NumRatings >= 10
ORDER BY AvgRating DESC
LIMIT 200
```

To be very careful and exclude any incorrect dog_guid or user_guid entries, you can include both shared columns in the WHERE clause:

```
SELECT d.dog_guid AS DogID, d.user_guid AS UserID, AVG(r.rating) AS AvgRating,
       COUNT(r.rating) AS NumRatings, d.breed, d.breed_group, d.breed_type
FROM dogs d, reviews r
WHERE d.dog_guid=r.dog_guid AND d.user_guid=r.user_guid
GROUP BY UserID, DogID, d.breed, d.breed_group, d.breed_type
HAVING NumRatings >= 10
ORDER BY AvgRating DESC
LIMIT 200
```

Try running this query now:

In [2]:
```
%%sql
SELECT d.dog_guid AS DogID, d.user_guid AS UserID, AVG(r.rating) AS AvgRating,
 COUNT(r.rating) AS NumRatings, d.breed, d.breed_group, d.breed_group
FROM dogs d, reviews r
WHERE d.dog_guid = r.dog_guid AND d.user_guid = r.user_guid
GROUP BY UserID, DogId, d.breed, d.breed_group, d.breed_type
HAVING NumRatings >= 10
ORDER BY AvgRating DESC
LIMIT 200
```

```
38 rows affected.
```

Out[2]:

DogID	UserID	AvgRating	NumRatings	breed	breed_group	breed_g
fdbf39f8-7144-11e5-ba71-058fbc01cf0b	ce987914-7144-11e5-ba71-058fbc01cf0b	8.0000	12	Canaan Dog	Herding	Herding
fdc09a82-7144-11e5-ba71-058fbc01cf0b	ce99bb12-7144-11e5-ba71-058fbc01cf0b	5.0000	10	Golden Doodle	None	None
fdbef330-7144-11e5-ba71-058fbc01cf0b	ce984d2c-7144-11e5-ba71-058fbc01cf0b	4.5385	13	Havanese	Toy	Toy
fdc0518a-7144-11e5-ba71-058fbc01cf0b	ce99661c-7144-11e5-ba71-058fbc01cf0b	3.7333	15	Mixed	None	None
fd684cf6-7144-11e5-ba71-058fbc01cf0b	ce473856-7144-11e5-ba71-058fbc01cf0b	3.7222	18	Labrador Retriever	Sporting	Sporting
fdbf66e4-7144-11e5-ba71-058fbc01cf0b	ce98b9b0-7144-11e5-ba71-058fbc01cf0b	3.5000	14	Golden Retriever	Sporting	Sporting
fdc09b0e-7144-11e5-ba71-058fbc01cf0b	ce99bb76-7144-11e5-ba71-058fbc01cf0b	3.2308	13	Golden Doodle	None	None
fdbeedc2-7144-11e5-ba71-058fbc01cf0b	ce9847dc-7144-11e5-ba71-058fbc01cf0b	3.1538	13	Eurasier	None	None
fdc180be-7144-11e5-ba71-058fbc01cf0b	ce9a7110-7144-11e5-ba71-058fbc01cf0b	3.0625	16	Chihuahua-Dachshund Mix	None	None
fd6aabd6-7144-11e5-ba71-058fbc01cf0b	ce6cd804-7144-11e5-ba71-058fbc01cf0b	3.0000	13	American Pit Bull Terrier	None	None

fdbfc0b2-7144-11e5-ba71-058fbc01cf0b	ce98f3bc-7144-11e5-ba71-058fbc01cf0b	2.6923	13	Mixed	None	None
fdbedef4-7144-11e5-ba71-058fbc01cf0b	ce982e46-7144-11e5-ba71-058fbc01cf0b	2.4615	13	Mixed	None	None
fdbec8ce-7144-11e5-ba71-058fbc01cf0b	ce980e3e-7144-11e5-ba71-058fbc01cf0b	2.4545	11	Basset Hound	Hound	Hound
fdbeff06-7144-11e5-ba71-058fbc01cf0b	ce9856f0-7144-11e5-ba71-058fbc01cf0b	2.4000	10	Bearded Collie	Herding	Herding
fdbfe22c-7144-11e5-ba71-058fbc01cf0b	ce99049c-7144-11e5-ba71-058fbc01cf0b	2.4000	10	Irish Water Spaniel	Sporting	Sporting
fdbed7ce-7144-11e5-ba71-058fbc01cf0b	ce981dc0-7144-11e5-ba71-058fbc01cf0b	2.3333	18	Havanese	Toy	Toy
fdc1c31c-7144-11e5-ba71-058fbc01cf0b	ce9ac674-7144-11e5-ba71-058fbc01cf0b	2.3077	13	Labrador Retriever	Sporting	Sporting
fdc0c49e-7144-11e5-ba71-058fbc01cf0b	ce99d674-7144-11e5-ba71-058fbc01cf0b	2.2778	18	Other	None	None
fdc19b30-7144-11e5-ba71-058fbc01cf0b	ce9aab30-7144-11e5-ba71-058fbc01cf0b	2.2222	18	Brittany	Sporting	Sporting
fdbedf94-7144-11e5-ba71-058fbc01cf0b	ce982ed2-7144-11e5-ba71-058fbc01cf0b	2.0000	11	Mixed	None	None

fdc0c502-7144-11e5-ba71-058fbc01cf0b	ce99d732-7144-11e5-ba71-058fbc01cf0b	2.0000	15	Golden Retriever	Sporting	Sporting
fdbfddfe-7144-11e5-ba71-058fbc01cf0b	ce99049c-7144-11e5-ba71-058fbc01cf0b	1.9375	16	Irish Water Spaniel	Sporting	Sporting
fd7104ae-7144-11e5-ba71-058fbc01cf0b	ce7212a6-7144-11e5-ba71-058fbc01cf0b	1.9000	10	Mixed	None	None
fdc198d8-7144-11e5-ba71-058fbc01cf0b	ce9a9500-7144-11e5-ba71-058fbc01cf0b	1.8000	10	German Shepherd Dog	Herding	Herding
fdbe923c-7144-11e5-ba71-058fbc01cf0b	ce97f926-7144-11e5-ba71-058fbc01cf0b	1.7647	17	Beagle	Hound	Hound
fd6a7490-7144-11e5-ba71-058fbc01cf0b	ce6cf3fc-7144-11e5-ba71-058fbc01cf0b	1.7500	20	Golden Retriever	Sporting	Sporting
fdbef5e2-7144-11e5-ba71-058fbc01cf0b	ce984f70-7144-11e5-ba71-058fbc01cf0b	1.7000	10	Havanese	Toy	Toy
fdbea952-7144-11e5-ba71-058fbc01cf0b	ce9805ce-7144-11e5-ba71-058fbc01cf0b	1.6667	15	Golden Doodle	None	None
fdbf9f10-7144-11e5-ba71-058fbc01cf0b	ce98d6de-7144-11e5-ba71-058fbc01cf0b	1.6667	18	Australian Cattle Dog-Border Collie Mix	None	None
fdbf686a-7144-11e5-ba71-058fbc01cf0b	ce98bca8-7144-11e5-ba71-058fbc01cf0b	1.6000	10	Mixed	None	None

fdbf6144-7144-11e5-ba71-058fbc01cf0b	ce98b776-7144-11e5-ba71-058fbc01cf0b	1.5455	11	American Pit Bull Terrier-Labrador Retriever Mix	None	None
fdc1edec-7144-11e5-ba71-058fbc01cf0b	ce9ae4b0-7144-11e5-ba71-058fbc01cf0b	1.5333	15	Poodle	Non-Sporting	Non-Spo
fdbe7cac-7144-11e5-ba71-058fbc01cf0b	ce97e10c-7144-11e5-ba71-058fbc01cf0b	1.3333	12	Labrador Retriever	Sporting	Sporting
fdbe8a9e-7144-11e5-ba71-058fbc01cf0b	ce97f1f6-7144-11e5-ba71-058fbc01cf0b	1.2500	16	Mixed	None	None
fdbfe56a-7144-11e5-ba71-058fbc01cf0b	ce99049c-7144-11e5-ba71-058fbc01cf0b	1.2143	14	Irish Water Spaniel	Sporting	Sporting
fdbf0456-7144-11e5-ba71-058fbc01cf0b	ce985d9e-7144-11e5-ba71-058fbc01cf0b	0.7500	16	English Springer Spaniel	Sporting	Sporting
fdc23ba8-7144-11e5-ba71-058fbc01cf0b	ce9b306e-7144-11e5-ba71-058fbc01cf0b	0.4211	19	Cockapoo	None	None
fdbec040-7144-11e5-ba71-058fbc01cf0b	ce98098e-7144-11e5-ba71-058fbc01cf0b	0.4000	10	Mixed	None	None

The query should execute quickly. This would NOT have been the case if you did not include the WHERE clause to combine the two tables. If you accidentally request a Cartesian product from datasets with billions of rows, you could be waiting for your query output for days (and will probably get in trouble with your database administrator). So always remember to tell the database how to join your tables!

Let's examine our joined table a bit further. The joined table outputted by the query above should have 38 rows, despite the fact that we set our LIMIT at 200. The reason for this is that it turns out that a relatively small number of customers provided 10 or more reviews. If you remove the HAVING and LIMIT BY clause from the query, you should end up with 389 rows. **Go ahead and try it:**

In [6]:
```
%%sql
SELECT d.dog_guid AS DogID, d.user_guid AS UserID, AVG(r.rating) AS AvgRating,
 COUNT(r.rating) AS NumRatings, d.breed, d.breed_group, d.breed_type
FROM dogs d, reviews r
WHERE d.dog_guid=r.dog_guid AND d.user_guid=r.user_guid
GROUP BY UserID, DogID, d.breed, d.breed_group, d.breed_type
ORDER BY AvgRating DESC
LIMIT 20
```

```
20 rows affected.
```

Out[6]:

DogID	UserID	AvgRating	NumRatings	breed	breed_group	breed_ty
fdbedd14-7144-11e5-ba71-058fbc01cf0b	ce975994-7144-11e5-ba71-058fbc01cf0b	9.0000	1	Mixed	None	Mixed Breed/ Other/ I Don't Kno
fdbf94de-7144-11e5-ba71-058fbc01cf0b	ce98cc34-7144-11e5-ba71-058fbc01cf0b	9.0000	2	Pembroke Welsh Corgi	Herding	Pure Bree
fdbf178e-7144-11e5-ba71-058fbc01cf0b	ce986546-7144-11e5-ba71-058fbc01cf0b	9.0000	1	Airedale Terrier	Terrier	Pure Bree
fdc057ac-7144-11e5-ba71-058fbc01cf0b	ce996b9e-7144-11e5-ba71-058fbc01cf0b	9.0000	1	Mixed	None	Mixed Breed/ Other/ I Don't Kno
fdc121dc-7144-11e5-ba71-058fbc01cf0b	ce9a310a-7144-11e5-ba71-058fbc01cf0b	9.0000	1	Mixed	None	Mixed Breed/ Other/ I Don't Kno
fdc19554-7144-11e5-ba71-058fbc01cf0b	ce9a8a24-7144-11e5-ba71-058fbc01cf0b	9.0000	1	Miniature Pinscher	Toy	Pure Bree
fdc147ac-7144-11e5-ba71-058fbc01cf0b	ce9a545a-7144-11e5-ba71-058fbc01cf0b	9.0000	1	Tibetan Terrier	Non-Sporting	Pure Bree
fdc1c880-7144-11e5-ba71-058fbc01cf0b	ce96c768-7144-11e5-ba71-058fbc01cf0b	9.0000	1	Maltese	Toy	Pure Bree
fdc059fa-7144-11e5-ba71-058fbc01cf0b	ce996dc4-7144-11e5-ba71-058fbc01cf0b	9.0000	1	Irish Water Spaniel	Sporting	Pure Bree
fdc019c2-7144-11e5-ba71-058fbc01cf0b	ce99489e-7144-11e5-ba71-058fbc01cf0b	9.0000	1	Mixed	None	Mixed Breed/ Other/ I Don't Kno

fd41bca8-7144-11e5-ba71-058fbc01cf0b	ce7b055a-7144-11e5-ba71-058fbc01cf0b	8.0000	1	Labrador Retriever	Sporting	Pure Bree
fdbf39f8-7144-11e5-ba71-058fbc01cf0b	ce987914-7144-11e5-ba71-058fbc01cf0b	8.0000	12	Canaan Dog	Herding	Pure Bree
fdc1e1a8-7144-11e5-ba71-058fbc01cf0b	ce9ad934-7144-11e5-ba71-058fbc01cf0b	7.5000	2	Mixed	None	Mixed Breed/ Other/ I Don't Kno
fdbe7a40-7144-11e5-ba71-058fbc01cf0b	ce97df9a-7144-11e5-ba71-058fbc01cf0b	7.0000	6	Mixed	None	Mixed Breed/ Other/ I Don't Kno
fdbe8774-7144-11e5-ba71-058fbc01cf0b	ce97ebfc-7144-11e5-ba71-058fbc01cf0b	7.0000	4	Labrador Retriever	Sporting	Pure Bree
fdbf940c-7144-11e5-ba71-058fbc01cf0b	ce98cb80-7144-11e5-ba71-058fbc01cf0b	7.0000	1	Soft Coated Wheaten Terrier	Terrier	Pure Bree
fdc1baca-7144-11e5-ba71-058fbc01cf0b	ce9ac2fa-7144-11e5-ba71-058fbc01cf0b	7.0000	1	Poodle-Miniature Schnauzer Mix	None	Cross Breed
fdc2417a-7144-11e5-ba71-058fbc01cf0b	ce9b35e6-7144-11e5-ba71-058fbc01cf0b	7.0000	1	Bichon Frise	Non-Sporting	Pure Bree
fdc1706a-7144-11e5-ba71-058fbc01cf0b	ce9a6846-7144-11e5-ba71-058fbc01cf0b	6.8000	5	Mixed	None	Mixed Breed/ Other/ I Don't Kno
fdc0c2aa-7144-11e5-ba71-058fbc01cf0b	ce99d322-7144-11e5-ba71-058fbc01cf0b	6.5000	2	Border Collie	Herding	Pure Bree

It's clear from looking at this output that (A) not many customers provided ratings, and (B) when they did, they usually were not very surprised by their dog's performance. Therefore, these ratings are probably not going to provide a lot of instructive insight into how to improve Dognition's completion rate. However, the ratings table still provides a great opportunity to illustrate the results of different types of joins.

To help prepare us for this:

Questions 1-4: How many unique dog_guids and user_guids are there in the reviews and dogs table independently?

In [8]:
```
%%sql
SELECT COUNT(DISTINCT dog_guid)
FROM dogs
```

```
1 rows affected.
```

Out[8]:

COUNT(DISTINCT dog_guid)
35050

In [9]:
```
%%sql
SELECT COUNT(DISTINCT user_guid)
FROM dogs
```

```
1 rows affected.
```

Out[9]:

COUNT(DISTINCT user_guid)
30967

In [10]:
```
%%sql
SELECT COUNT(DISTINCT dog_guid)
FROM reviews
```

```
1 rows affected.
```

Out[10]:

COUNT(DISTINCT dog_guid)
5991

In [11]:
```
%%sql
SELECT COUNT(DISTINCT user_guid)
FROM reviews
```

```
1 rows affected.
```

Out[11]:

COUNT(DISTINCT user_guid)
5586

These counts indicate some important things:

- Many customers in both the reviews and the dogs table have multiple dogs
- There are many more unique dog_guids and user_guids in the dogs table than the reviews table
- There are many more unique dog_guids and user_guids in the reviews table than in the output of our inner join

Let's test one more thing.

Try the inner join query once with just the dog_guid or once with just the user_guid clause in the WHERE statement:

In [16]:
```
%%sql
SELECT COUNT(DISTINCT d.dog_guid)
FROM dogs d
JOIN reviews r
WHERE d.dog_guid = r.dog_guid
```

```
1 rows affected.
```

Out[16]:

COUNT(DISTINCT d.dog_guid)
395

In [17]:
```
%%sql
SELECT COUNT(DISTINCT d.user_guid)
FROM dogs d
JOIN reviews r
WHERE d.user_guid = r.user_guid
```

```
1 rows affected.
```

Out[17]:

COUNT(DISTINCT d.user_guid)
5586

When you run the query by joining on the dog_guid only, you still get 389 rows in your output. When you run the query by joining on the user_guid only, you get 5586 rows in your output. This means that:

- All of the user_guids in the reviews table are in the dogs table
- Only 389 of the over 5000 dog_guids in the reviews table are in the dogs table

Perhaps most importantly for our current purposes, these COUNT queries show you that *inner joins only output the data from rows that have equivalent values in both tables being joined*. If you wanted to include all the dog_guids or user_guids in one or both of the tables, you would have to use an outer join, which we will practice in the next lesson.

Try an inner join on your own.

Question 5: How would you extract the user_guid, dog_guid, breed, breed_type, and breed_group for all animals who completed the "Yawn Warm-up" game (you should get 20,845 rows if you join on dog_guid only)?

In [22]:
```
%%sql
SELECT d.dog_guid AS DogID, d.user_guid AS UserID, d.breed, d.breed_group, d.b
reed_type
FROM dogs d, complete_tests c
WHERE d.dog_guid=c.dog_guid AND test_name = "Yawn Warm-up"
LIMIT 20
```

```
20 rows affected.
```

Out[22]:

DogID	UserID	breed	breed_group	breed_type
fd27b272-7144-11e5-ba71-058fbc01cf0b	ce134e42-7144-11e5-ba71-058fbc01cf0b	Labrador Retriever	Sporting	Pure Breed
fd27b5ba-7144-11e5-ba71-058fbc01cf0b	ce1353d8-7144-11e5-ba71-058fbc01cf0b	Shetland Sheepdog	Herding	Pure Breed
fd27b6b4-7144-11e5-ba71-058fbc01cf0b	ce135ab8-7144-11e5-ba71-058fbc01cf0b	Golden Retriever	Sporting	Pure Breed
fd27b79a-7144-11e5-ba71-058fbc01cf0b	ce13507c-7144-11e5-ba71-058fbc01cf0b	Golden Retriever	Sporting	Pure Breed
fd27b86c-7144-11e5-ba71-058fbc01cf0b	ce135e14-7144-11e5-ba71-058fbc01cf0b	Shih Tzu	Toy	Pure Breed
fd27b948-7144-11e5-ba71-058fbc01cf0b	ce13615c-7144-11e5-ba71-058fbc01cf0b	Siberian Husky	Working	Pure Breed
fd27ba1a-7144-11e5-ba71-058fbc01cf0b	ce135e14-7144-11e5-ba71-058fbc01cf0b	Shih Tzu	Toy	Pure Breed
fd27bbbe-7144-11e5-ba71-058fbc01cf0b	ce135f2c-7144-11e5-ba71-058fbc01cf0b	Mixed	None	Mixed Breed/ Other/ I Don't Know
fd27c1c2-7144-11e5-ba71-058fbc01cf0b	ce136a1c-7144-11e5-ba71-058fbc01cf0b	Labrador Retriever	Sporting	Pure Breed
fd27c5be-7144-11e5-ba71-058fbc01cf0b	ce136ac6-7144-11e5-ba71-058fbc01cf0b	Shih Tzu-Poodle Mix	None	Cross Breed
fd27c74e-7144-11e5-ba71-058fbc01cf0b	ce136c24-7144-11e5-ba71-058fbc01cf0b	German Shepherd Dog-Pembroke Welsh Corgi Mix	None	Cross Breed
fd27c7d0-7144-11e5-ba71-058fbc01cf0b	ce136e36-7144-11e5-ba71-058fbc01cf0b	Vizsla	Sporting	Pure Breed
fd27c852-7144-11e5-ba71-058fbc01cf0b	ce136ee0-7144-11e5-ba71-058fbc01cf0b	Pug	Toy	Pure Breed

fd27c8d4-7144-11e5-ba71-058fbc01cf0b	ce136f94-7144-11e5-ba71-058fbc01cf0b	Boxer	Working	Pure Breed
fd27c956-7144-11e5-ba71-058fbc01cf0b	ce134be0-7144-11e5-ba71-058fbc01cf0b	German Shepherd Dog-Nova Scotia Duck Tolling Retriever Mix	None	Cross Breed
fd27cb72-7144-11e5-ba71-058fbc01cf0b	ce1371a6-7144-11e5-ba71-058fbc01cf0b	Beagle	Hound	Pure Breed
fd27cd98-7144-11e5-ba71-058fbc01cf0b	ce136f94-7144-11e5-ba71-058fbc01cf0b	Beagle	Hound	Pure Breed
fd27ce1a-7144-11e5-ba71-058fbc01cf0b	ce136f94-7144-11e5-ba71-058fbc01cf0b	Beagle	Hound	Pure Breed
fd27cea6-7144-11e5-ba71-058fbc01cf0b	ce1373ae-7144-11e5-ba71-058fbc01cf0b	Mixed	None	Mixed Breed/ Other/ I Don't Know
fd27cf28-7144-11e5-ba71-058fbc01cf0b	ce13750c-7144-11e5-ba71-058fbc01cf0b	Chesapeake Bay Retriever	Sporting	Pure Breed

2. Joining More than 2 Tables

In theory, you can join as many tables together as you want or need. To join multiple tables you take the same approach as we took when we were joining two tables together: list all the fields you want to extract in the SELECT statement, specify which table they came from in the SELECT statement, list all the tables from which you will need to extract the fields in the FROM statement, and then tell the database how to connect the tables in the WHERE statement.

To extract the user_guid, user's state of residence, user's zip code, dog_guid, breed, breed_type, and breed_group for all animals who completed the "Yawn Warm-up" game, you might be tempted to query:

```
SELECT c.user_guid AS UserID, u.state, u.zip, d.dog_guid AS DogID, d.breed, d.breed
_type, d.breed_group
FROM dogs d, complete_tests c, users u
WHERE d.dog_guid=c.dog_guid
   AND c.user_guid=u.user_guid
   AND c.test_name="Yawn Warm-up";
```

This query focuses the relationships primarily on the complete_tests table. However, it turns out that our Dognition dataset has only NULL values in the user_guid column of the complete_tests table. If you were to execute the query above, you would not get an error message, but your output would have 0 rows. However, the power of relational databases will come in handy here. You can use the dogs table to link the complete_tests and users table (pay attention to the difference between the WHERE statement in this query vs. the WHERE statement in the query above):

```
SELECT d.user_guid AS UserID, u.state, u.zip, d.dog_guid AS DogID, d.breed, d.breed
_type, d.breed_group
FROM dogs d, complete_tests c, users u
WHERE d.dog_guid=c.dog_guid
   AND d.user_guid=u.user_guid
   AND c.test_name="Yawn Warm-up";
```

Of note, joins are very resource intensive, so try not to join unnecessarily. In general, the more joins you have to execute, the slower your query performance will be.

Question 6: How would you extract the user_guid, membership_type, and dog_guid of all the golden retrievers who completed at least 1 Dognition test (you should get 711 rows)?

In [45]:
```
%%sql
SELECT DISTINCT d.user_guid AS UserID, u.membership_type, d.dog_guid AS DogID,
 d.breed, d.total_tests_completed
FROM dogs d, complete_tests c, users u
WHERE d.dog_guid = c.dog_guid
AND d.user_guid = u.user_guid
AND d.breed = "Golden Retriever"
AND d.total_tests_completed >= 1
LIMIT 20
```

```
20 rows affected.
```

Out[45]:

UserID	membership_type	DogID	breed	total_tests_completed
ce135ab8-7144-11e5-ba71-058fbc01cf0b	1	fd27b6b4-7144-11e5-ba71-058fbc01cf0b	Golden Retriever	2
ce13507c-7144-11e5-ba71-058fbc01cf0b	1	fd27b79a-7144-11e5-ba71-058fbc01cf0b	Golden Retriever	11
ce1389d4-7144-11e5-ba71-058fbc01cf0b	1	fd27efb2-7144-11e5-ba71-058fbc01cf0b	Golden Retriever	20
ce21f122-7144-11e5-ba71-058fbc01cf0b	2	fd3d03fc-7144-11e5-ba71-058fbc01cf0b	Golden Retriever	4
ce220bb2-7144-11e5-ba71-058fbc01cf0b	2	fd3d10cc-7144-11e5-ba71-058fbc01cf0b	Golden Retriever	20
ce2237f4-7144-11e5-ba71-058fbc01cf0b	2	fd3d3b24-7144-11e5-ba71-058fbc01cf0b	Golden Retriever	4
ce2243e8-7144-11e5-ba71-058fbc01cf0b	2	fd3d4b8c-7144-11e5-ba71-058fbc01cf0b	Golden Retriever	20
ce240b24-7144-11e5-ba71-058fbc01cf0b	2	fd3fe96e-7144-11e5-ba71-058fbc01cf0b	Golden Retriever	6
ce135cf2-7144-11e5-ba71-058fbc01cf0b	2	fd7af63a-7144-11e5-ba71-058fbc01cf0b	Golden Retriever	20
ce24476a-7144-11e5-ba71-058fbc01cf0b	2	fd404ff8-7144-11e5-ba71-058fbc01cf0b	Golden Retriever	37
ce245cdc-7144-11e5-ba71-058fbc01cf0b	2	fd4059e4-7144-11e5-ba71-058fbc01cf0b	Golden Retriever	20
ce245cdc-7144-11e5-ba71-058fbc01cf0b	2	fd45eb98-7144-11e5-ba71-058fbc01cf0b	Golden Retriever	1
ce246be6-7144-11e5-ba71-058fbc01cf0b	2	fd40738e-7144-11e5-ba71-058fbc01cf0b	Golden Retriever	26

ce246be6-7144-11e5-ba71-058fbc01cf0b	2	fd45d6ee-7144-11e5-ba71-058fbc01cf0b	Golden Retriever	14
ce249774-7144-11e5-ba71-058fbc01cf0b	1	fd408ed2-7144-11e5-ba71-058fbc01cf0b	Golden Retriever	20
ce249c6a-7144-11e5-ba71-058fbc01cf0b	2	fd40945e-7144-11e5-ba71-058fbc01cf0b	Golden Retriever	20
ce2258a6-7144-11e5-ba71-058fbc01cf0b	2	fd6487c4-7144-11e5-ba71-058fbc01cf0b	Golden Retriever	16
ce24c1c2-7144-11e5-ba71-058fbc01cf0b	2	fd40ee72-7144-11e5-ba71-058fbc01cf0b	Golden Retriever	11
ce24a106-7144-11e5-ba71-058fbc01cf0b	2	fd40f822-7144-11e5-ba71-058fbc01cf0b	Golden Retriever	20
ce24d572-7144-11e5-ba71-058fbc01cf0b	2	fd4107cc-7144-11e5-ba71-058fbc01cf0b	Golden Retriever	28

Practice inner joining your own tables!

Question 7: How many unique Golden Retrievers who live in North Carolina are there in the Dognition database (you should get 30)?

In [37]:
```
%%sql
SELECT u.state AS state, d.breed AS breed, COUNT(DISTINCT d.dog_guid)
FROM users u, dogs d
WHERE d.user_guid=u.user_guid AND breed="Golden Retriever"
GROUP BY state
HAVING state="NC";
```

```
1 rows affected.
```

Out[37]:

state	breed	COUNT(DISTINCT d.dog_guid)
NC	Golden Retriever	30

Question 8: How many unique customers within each membership type provided reviews (there should be 2900 in the membership type with the greatest number of customers, and 15 in the membership type with the fewest number of customers if you do NOT include entries with NULL values in their ratings field)?

In [41]:
```
%%sql
SELECT COUNT(DISTINCT u.user_guid), u.membership_type
FROM users u, reviews r
WHERE u.user_guid = r.user_guid
AND r.rating IS NOT NULL
GROUP BY u.membership_type
```

5 rows affected.

Out[41]:

COUNT(DISTINCT u.user_guid)	membership_type
2900	1
1120	2
238	3
816	4
15	5

Question 9: For which 3 dog breeds do we have the greatest amount of site_activity data, (as defined by non-NULL values in script_detail_id)(your answers should be "Mixed", "Labrador Retriever", and "Labrador Retriever-Golden Retriever Mix"?

In [44]:
```
%%sql
SELECT COUNT(DISTINCT script_detail_id) activity, d.breed
FROM dogs d, site_activities s
WHERE d.dog_guid = s.dog_guid
AND s.script_detail_id IS NOT NULL
GROUP BY d.breed
ORDER BY activity DESC
LIMIT 3
```

3 rows affected.

Out[44]:

activity	breed
295	Mixed
295	American Pit Bull Terrier
295	Labrador Retriever

Practice any other inner joins you would like to try here!

In [ ]:
```

```