## Module 09: Hibernate Optimization

### Exercise 09.1 – Data Fetching

#### *The Setup:*

In this exercise we will use System.nanotime() to check how long it takes for MySQL and Hibernate to retrieve the same dataset with different fetching strategies.

Start this exercise by opening the project from **C:\CS544\exercises\exercise09.1**.

#### *The Application:*

The imported application creates 1000 owner objects, each with 10 associated pet objects and saves them to the database. It then reads them and checks how long it took.

#### *The Exercise:*

Consider what the application does, and write down which strategy you think will perform best under these circumstances. Then test each of the strategies using the following steps:

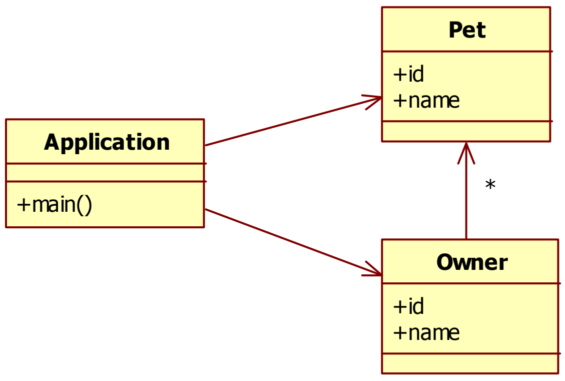
a) Run the application as is, without any optimization, and note the time. Run the application 3 times to get an average time (speed ups and slowdowns can be caused by background processes and / or caching).

b) Modify the mapping for **Owner.java** to use **batch fetching**, batch size 10, run the application three times again and note the new time. Also check the time when using batch size 5 and when using batch size 50.

c) Modify the mapping to use **sub-select** strategy instead of batch fetching, and run the application 3 times to note the time it took to retrieve the results.

d) Remove the sub-select strategy and use a **join fetch query** in Application.java to retrieve everything, again running the application 3 times and checking the time. Also check the difference between using a named query, or just a query directly in code.

e) Lastly modify the application to use the **always join** strategy. Run the application 3 times again and note the time.



Check to see if the strategy you thought would perform best was indeed the best for this situation. Remember, just because a strategy performed well under these circumstances does not necessarily mean it will perform well under other circumstances.