**Custom Queries:**

As known, repository in Spring Boot JPA generates implementation of few methods, by default, and these methods interact with underlying Database/tables, to perform required CRUD operations.

However, in some situations these default methods provided are not sufficient for certain requirements, and there is necessity for additional custom methods in repository.

How to provide these additional custom methods in repository?

Below are the different approaches for the same

**#1. Provide declaration of required custom methods**, and follow specific patterns for methods names(which includes specific operation and field names), along with the appropriate parameters/return types.

Exception occurs, if method name or fields are not recognized.

**#2. Use @Query annotation along with method declarations.** In this case name of the method does not matter, unlike above approach #1, but the query specified with @Query annotation matters.

Exception occurs, if the query specified with @Query annotation is not understandable by Sprign Boot JPA.

Along with @Query we have two possible alternatives, JPQL query and Native query.

Incase of JPQL Query, the query is based on Entity class name and fields in Entity class, where as in Native Query, the query is based on underlying db table and db table columns.

Due to portability reasons, JPQL query is preferred over Native query approach.

Now, the question is, who provides the definition/implementation of methods in either of above, #1 or #2. Spring Boot JPA is so powerful that it can generate method definitions in runtime, even in above cases.

And we developers just need to invoke above custom methods, from Service Layer, as usual.

More details, in upcoming Lectures, meanwhile if any queries please contact the Instructor.