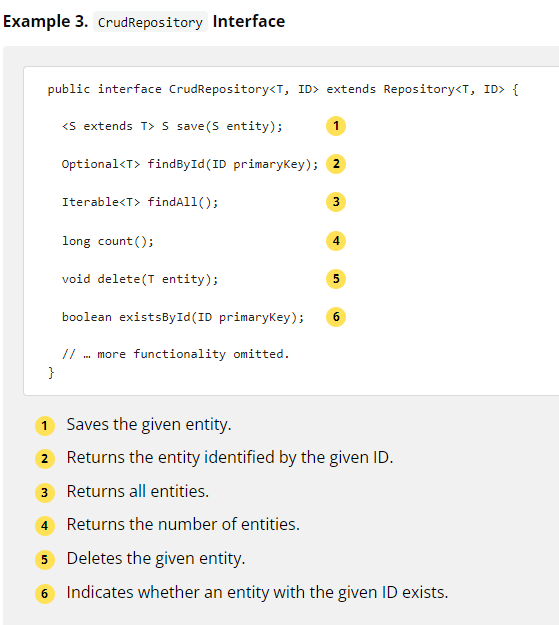
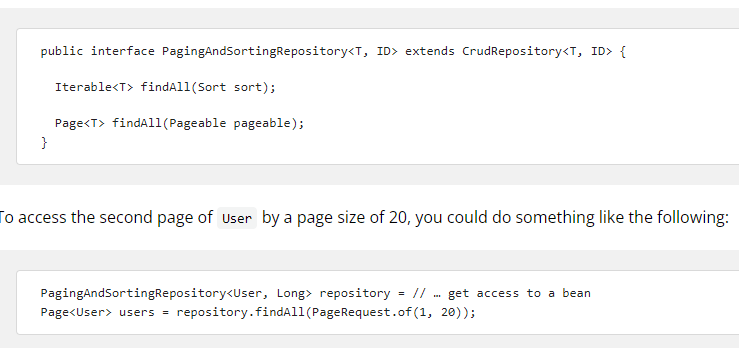
* The central interface in the Spring Data repository abstraction is **Repository**. It takes the *domain class* to manage as well as the *ID type* of the domain class as type arguments.
* The **CrudRepository** interface provides sophisticated CRUD functionality for the *entity* *class* that is being managed.



1. On top of the **CrudRepository**, there is a **PagingAndSortingRepository** abstraction that adds additional methods to ease paginated access to entities



**Query Methods:**

With Spring Data, declaring queries becomes a four-step process:

1. Declare an interface extending **Repository** or one of its sub interfaces and type it to the domain class and ID type that it should handle. Typically, your repository interface extends **Repository**, **CrudRepository**, or **PagingAndSortingRepository**

interface PersonRepository extends Repository<Person, Long> {…}

1. Declare query methods on the interface.

interface PersonRepository extends Repository<Person, Long> {

List<Person> findByLastname(String lastname);

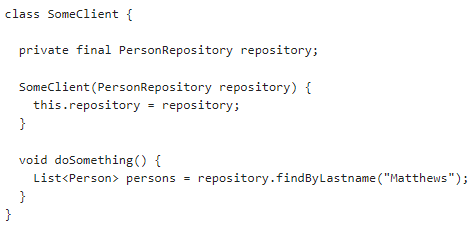
}

1. Set up Spring to create proxy instances for those interfaces, either with JavaConfig or with XML configuration.

@EnableJpaRepositories

class Config {…}

1. Inject the repository instance and use it



**Defining Query Methods:**

The repository proxy has two ways to derive a store-specific query from the method name:

* By deriving the query from the method name directly.
* By using a manually defined query.

Available options depend on the actual store. However, there must be a strategy that decides what actual query is created.

**Query Lookup Strategies:**

The following strategies are available for the repository infrastructure to resolve the query. For Java configuration, you can use the **queryLookupStrategy** attribute of the **Enable${store}Repositories** annotation.

* **CREATE**– If this option is used then Spring framework attempts to automatically construct a query from the query method name.
* **USE\_DECLARED\_QUERY**– For this option Spring framework tries to find a declared query. The query can be defined by an annotation like **@NamedQuery** (Used with Spring Data JPA) or **@Query**.
* **CREATE\_IF\_NOT\_FOUND–** This is the default option and it combines CREATE and USE\_DECLARED\_QUERY. It looks up a declared query first, and, if no declared query is found, it creates a custom method name-based query.

**Query Creation:**