http://www.egovframe.go.kr/wiki/doku.php?id=egovframework:rte2:psl:data:jpa:repository

# Description

The typical implementation of Data Access Layer involves massive volume of coding, With Spring Data, the demand of coding is controlled to the minimum to access the abstract repository, contributing to improved productivity, thanks to Query Method where you need no further information than the name of method to create queries. With the method made available under the specific protocol, Spring Data comes into play to auto-map the method to the corresponding database, coming up with the result.

In this Section, we'll take a brief look at CrudRepository where the repository interface is inherited to provide CRUD-related methods and PagingAndSortingRepository where Paging processing is available.

# User Guide

## CRUD Function Interface

Being the core interface of abstract repository for Spring Data, Repository is a marker interface that is made up of class type and ID type and has sub-interface of **CrudRepository** that features CRUD function. Refer to the following example for how to code CrudRepository:

public interface CrudRepository<T, ID extends Serializable> extends Repository<T, ID> {

<S extends T> S save(S entity);.....❶

T findOne(ID primaryKey);.....❷

Iterable<T> findAll();.....❸

Long count();.....❹

void delete(T entity);.....❺

boolean exists(ID primaryKey);.....❻

}

❶ The incoming entities are reposited.
❷ The identified entities are returned to the ID transferred.
❸ The entities are returned.
❹ The number of entities is returned.
❺ The incoming entities are deleted.
❻ The existence or non-existence of the entities concerning the transfered Ids is returned.

## Paging Function Interface

Both CrudRepository and **PagingAndSortingRespository**, the interface in charge of paging function where CrudRepository is inherited, are available in Spring Data.

public interface PagingAndSortingRepository<T, ID extends Serializable> extends
CrudRepository<T, ID> {
Iterable<T> findAll(Sort sort);
Page<T> findAll(Pageable pageable);
}

Refer to the following example for how **PagingAndSortingRepository** loads a total of 20 data from Page 2:

PagingAndSortingRepository<User, Long> repository =

Page<User> users = repository.findAll(new PageRequest(1, 20));

# References

http://static.springsource.org/spring-data/data-jpa/docs/current/reference/html/