Spring Data JPA Query, Spring Boot, NamedQuery 실습

**[예제]**

STS -> Spring Stater Project ,name : **jpamethodquery**, Type : MAVEN

다음화면에서 Core -> Lombok, SQL -> JPA, MySQL 선택

롬복(Lombok)설치는 다음 URL 참조

<http://ojc.asia/bbs/board.php?bo_table=LecSpring&wr_id=561>

마리아 DB 설치는 다음 URL 참조

<http://ojc.asia/bbs/board.php?bo_table=LecSpring&wr_id=524>

**src/main/resources/application.properties**

spring.datasource.platform=mysql

spring.datasource.sql-script-encoding=UTF-8

spring.datasource.url=jdbc:mysql://localhost/jpamethodquery?createDatabaseIfNotExist=true

spring.datasource.username=root

spring.datasource.password=1111

spring.datasource.driver-class-name=com.mysql.jdbc.Driver

spring.jpa.show-sql=true

spring.jpa.hibernate.ddl-auto=create

logging.level.jpa=DEBUG

**jpa.model.Emp.java**

package jpa.model;

import java.io.Serializable;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.Id;

import javax.persistence.NamedQuery;

import lombok.Getter;

import lombok.Setter;

import lombok.ToString;

@Entity

@Getter

@Setter

@ToString

@NamedQuery(name="Emp.findBySalAndJob",

query="select e from Emp e where e.sal > ?1 and job = ?2")

public class Emp {

@Id

@GeneratedValue

private Long empno;

private String ename;

private String job;

private Long sal;

}

**jpa.repository.EmpRepository.java**

package jpa.repository;

import java.util.List;

import org.springframework.data.jpa.repository.JpaRepository;

import jpa.model.Emp;

public interface EmpRepository extends JpaRepository<Emp, Long> {

Emp findByEmpno(Long empno);

**//Emp엔티티에서 정의된 NameQuery지정**

List<Emp> findBySalAndJob(Long sal, String job);

**//직무(job)를 조건으로 이름(ename) 내림차순검색**

List<Emp> findByJobOrderByEnameDesc(String job);

**//이름(ename) like 조건으로 이름(ename) 오름차순검색**

List<Emp> findByJobLikeOrderByEnameAsc(String job);

**//급여(sal)가 sal1 ~ sal2 사이인 사원 추출**

List<Emp> findBySalBetween(Long sal1, Long sal2);

**//직무가 job 이거나 급여가 sal 보다 크거나 같은 사원 추출**

List<Emp> findByJobOrSalGreaterThanEqual(String job, Long sal);

}

**jpa.JpamethodqueryApplication.java**

package jpa;

import java.util.List;

import javax.persistence.Tuple;

import javax.transaction.Transactional;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.boot.CommandLineRunner;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import jpa.model.Emp;

import jpa.repository.EmpRepository;

@SpringBootApplication

public class JpamethodqueryApplication implements CommandLineRunner {

private static final Logger logger = LoggerFactory.getLogger(JpamethodqueryApplication.class);

@Autowired

private EmpRepository empRepository;

public static void main(String[] args) {

SpringApplication.run(JpamethodqueryApplication.class, args);

}

@Override

@Transactional

public void run(String...strings) throws Exception {

Emp e1 = new Emp();

e1.setEname("1길동"); e1.setJob("연구원"); e1.setSal(1111L);

empRepository.save(e1);

Emp e2 = new Emp();

e2.setEname("2길동"); e2.setJob("연구원"); e2.setSal(2222L);

empRepository.save(e2);

Emp e3 = new Emp();

e3.setEname("3길동"); e3.setJob("교사"); e3.setSal(3333L);

empRepository.save(e3);

Emp e4 = new Emp();

e4.setEname("4길동"); e4.setJob("연구원"); e4.setSal(4444L);

empRepository.save(e4);

Emp e5 = new Emp();

e5.setEname("5길동"); e5.setJob("교사"); e5.setSal(5555L);

empRepository.save(e5);

Emp e = empRepository.findByEmpno(1L);

System.out.println(e);

**//Sal가 1111보다 크고 job이 “연구원” SELECT**

List<Emp> emps1 = empRepository.findBySalAndJob(1111L, "연구원");

emps1.forEach(System.out::println);

**//job이 “연구원” SELECT, 이름 내림차순 정렬**

List<Emp> emps2 = empRepository.findByJobOrderByEnameDesc("연구원");

emps2.forEach(System.out::println);

**//job "교"로 시작하는 사원 SELECT, 이름 오름차순 정렬**

List<Emp> emps3 = empRepository.findByJobLikeOrderByEnameAsc("교%");

emps3.forEach(System.out::println);

**//job "교"로 시작하는 사원 SELECT, 이름 오름차순 정렬**

List<Emp> emps4 = empRepository.findBySalBetween(2000L, 5000L);

emps4.forEach(System.out::println);

**//job이 "연구원" 이거나 sal 값이 3333보다 크거나 같은 사원 추출**

List<Emp> emps5 = empRepository.findByJobOrSalGreaterThanEqual("연구원", 3333L);

emps5.forEach(System.out::println);

}

}

**실행 결과**

Hibernate: insert into emp (ename, job, sal) values (?, ?, ?)

Hibernate: insert into emp (ename, job, sal) values (?, ?, ?)

Hibernate: insert into emp (ename, job, sal) values (?, ?, ?)

Hibernate: insert into emp (ename, job, sal) values (?, ?, ?)

Hibernate: insert into emp (ename, job, sal) values (?, ?, ?)

select emp0\_.empno as empno1\_0\_, emp0\_.ename as ename2\_0\_, emp0\_.job as job3\_0\_, emp0\_.sal as sal4\_0\_ from emp emp0\_ where emp0\_.empno=?

Emp(empno=1, ename=1길동, job=연구원, sal=1111)

select emp0\_.empno as empno1\_0\_, emp0\_.ename as ename2\_0\_, emp0\_.job as job3\_0\_, emp0\_.sal as sal4\_0\_ from emp emp0\_ where emp0\_.sal>? and emp0\_.job=?

Emp(empno=2, ename=2길동, job=연구원, sal=2222)

Emp(empno=4, ename=4길동, job=연구원, sal=4444)

select emp0\_.empno as empno1\_0\_, emp0\_.ename as ename2\_0\_, emp0\_.job as job3\_0\_, emp0\_.sal as sal4\_0\_ from emp emp0\_ where emp0\_.job=? order by emp0\_.ename desc

Emp(empno=4, ename=4길동, job=연구원, sal=4444)

Emp(empno=2, ename=2길동, job=연구원, sal=2222)

Emp(empno=1, ename=1길동, job=연구원, sal=1111)

select emp0\_.empno as empno1\_0\_, emp0\_.ename as ename2\_0\_, emp0\_.job as job3\_0\_, emp0\_.sal as sal4\_0\_ from emp emp0\_ where emp0\_.job like ? order by emp0\_.ename asc

Emp(empno=3, ename=3길동, job=교사, sal=3333)

Emp(empno=5, ename=5길동, job=교사, sal=5555)

select emp0\_.empno as empno1\_0\_, emp0\_.ename as ename2\_0\_, emp0\_.job as job3\_0\_, emp0\_.sal as sal4\_0\_ from emp emp0\_ where emp0\_.sal between ? and ?

Emp(empno=2, ename=2길동, job=연구원, sal=2222)

Emp(empno=3, ename=3길동, job=교사, sal=3333)

Emp(empno=4, ename=4길동, job=연구원, sal=4444)

select emp0\_.empno as empno1\_0\_, emp0\_.ename as ename2\_0\_, emp0\_.job as job3\_0\_, emp0\_.sal as sal4\_0\_ from emp emp0\_ where emp0\_.job=? or emp0\_.sal>=?

Emp(empno=1, ename=1길동, job=연구원, sal=1111)

Emp(empno=2, ename=2길동, job=연구원, sal=2222)

Emp(empno=3, ename=3길동, job=교사, sal=3333)

Emp(empno=4, ename=4길동, job=연구원, sal=4444)

Emp(empno=5, ename=5길동, job=교사, sal=5555)