


















Spring Data JPA

- ✓  SBDemo1 [boot] [devtools]
 - ✓  src/main/java
 - >  com.rit
 - >  com.rit.p1intro
 - >  com.rit.p2mapping
 - >  com.rit.p3aop
 - >  com.rit.p4crud
 - >  com.rit.p5crud
 - >  com.rit.p6request
 - >  com.rit.p7crud
 - ✓  com.rit.p8jpa
 - >  Address.java
 - >  QueryController1.java
 - >  QueryController2.java
 - >  Staff.java
 - >  StaffController.java
 - >  StaffRepository.java

```
15 @Entity
16 public class Staff {
17
18     @Id
19     @GeneratedValue(strategy=GenerationType.IDENTITY)
20     private Long id;
21     private String firstName;
22     private String department;
23
24     @JsonFormat(pattern="dd/MM/yyyy", shape = JsonFormat.Shape.STRING)
25     private LocalDate dob;
26     private Integer exp;
27     private Boolean active;
28
29     @OneToOne(cascade=CascadeType.ALL)
30     private Address address;
31
32     public static void printList(List<Staff> staffs) {
33         System.out.println("\n-----");
34         for(Staff staff:staffs)
35             System.out.println("Staff ["+staff.getId() + ", "
36                 + staff.getFirstName() + ", " + staff.getDepartment()+"]");
37     }
38     //Constructors, toString, Getter & Setter
```

Address.java ×

```
8 @Entity
9 public class Address {
10
11     @Id
12     @GeneratedValue(strategy=GenerationType.IDENTITY)
13     private Long id;
14     private String street;
15     private String city;
16     private Integer zipcode;
17
18     //Constructors, toString, Getter & Setter
```

@Repository

public interface StaffRepository **extends** JpaRepository<Sta

```
List<Staff> findByFirstName(String name);  
List<Staff> findByFirstNameIs(String name);  
List<Staff> findByFirstNameEquals(String name);
```

```
List<Staff> findByFirstNameNot(String name);  
List<Staff> findByFirstNameIsNot(String name);
```

```
List<Staff> findByFirstNameIsNull();  
List<Staff> findByFirstNameIsNotNull();
```

```
List<Staff> findByFirstNameStartingWith(String name);  
List<Staff> findByFirstNameEndingWith(String name);  
List<Staff> findByFirstNameContaining(String name);
```

```
List<Staff> findByDepartmentOrderByFirstName(String dept);
List<Staff> findByDepartmentOrderByFirstNameAsc(String dept);
List<Staff> findByDepartmentOrderByFirstNameDesc(String dept);
List<Staff> findDistinctByDepartment(String dept);

List<Staff> findByFirstNameLike(String pattern);

List<Staff> findByExpGreaterThan(Integer exp);
List<Staff> findByExpGreaterThanEqual(Integer exp);
List<Staff> findByExpLessThan(Integer exp);
List<Staff> findByExpLessThanEqual(Integer exp);

List<Staff> findByExpBetween(Integer from, Integer to);
List<Staff> findByDepartmentAndExpGreaterThan(String d, Integer e);
```

```
List<Staff> findByActive(Boolean status);
List<Staff> findByActiveTrue();
List<Staff> findByActiveFalse();
List<Staff> findByFirstNameAndActive(String f, Boolean b);
List<Staff> findByDobBefore(LocalDate dob);
List<Staff> findByDobAfter(LocalDate dob);

List<Staff> findByAddressZipcode(Integer zip);

List<Staff> findFirstByOrderByFirstName();
List<Staff> findFirst5ByOrderByFirstName();
List<Staff> findTop5ByOrderByFirstNameDesc();
}
```

```
10
11 @RestController
12 @RequestMapping("/jpa")
13 public class StaffController {
14
15     @Autowired
16     StaffRepository repo;
17
18     @PostMapping
19     public Staff post(@RequestBody Staff staff) {
20         return repo.save(staff);
21     }
22
23     @GetMapping
24     public void get1(){
25         Staff.printList(repo.findAll());
26     }
27
28     @GetMapping("/{id}")
29     public void get2(@PathVariable long id){
30         System.out.println(repo.findById(id).get());
31     }
32 }
33
```

```
@RestController
@RequestMapping("/c1")
public class QueryController1 {

    @Autowired
    StaffRepository repo;

    @GetMapping("/fname")
    public void get1(){

        Staff.printList(repo.findByFirstNameIsNull());
        Staff.printList(repo.findByFirstNameIsNotNull());
        Staff.printList(repo.findFirstOrderByFirstName());
        Staff.printList(repo.findFirst5OrderByFirstName());
        Staff.printList(repo.findTop5OrderByFirstNameDesc());
    }
}
```



```
@GetMapping("/fname/{name}")
public void get2(@PathVariable String name){
    Staff.printList(repo.findByFirstName(name));
    Staff.printList(repo.findByFirstNameIs(name));
    Staff.printList(repo.findByFirstNameEquals(name));

    Staff.printList(repo.findByFirstNameNot(name));
    Staff.printList(repo.findByFirstNameIsNot(name));

    Staff.printList(repo.findByFirstNameStartingWith(name));
    Staff.printList(repo.findByFirstNameEndingWith(name));
    Staff.printList(repo.findByFirstNameContaining(name));
}
```

```
@GetMapping("/fname/{p1}/{p2}")  
public void get3(@PathVariable String p1, @PathVariable String p2){  
    Staff.printList(repo.findByFirstNameLike(p1+"%" + p2));  
}
```

```
@GetMapping("/dept/{dept}")  
public void get11(@PathVariable String dept){  
    Staff.printList(repo.findByDepartmentOrderByFirstName(dept));  
    Staff.printList(repo.findByDepartmentOrderByFirstNameAsc(dept));  
    Staff.printList(repo.findByDepartmentOrderByFirstNameDesc(dept));  
    Staff.printList(repo.findDistinctByDepartment(dept));  
}
```

```
@GetMapping("/exp/{exp}")
public void get1(@PathVariable Integer exp){
    Staff.printList(repo.findByExpGreaterThan(exp));
    Staff.printList(repo.findByExpGreaterThanOrEqualTo(exp));
    Staff.printList(repo.findByExpLessThan(exp));
    Staff.printList(repo.findByExpLessThanOrEqualTo(exp));
}
```

```
@GetMapping("/exp/{from}/{to}")
public void get2(@PathVariable Integer from, @PathVariable Integer to){
    Staff.printList(repo.findByExpBetween(from,to));
}
```

```
@GetMapping("/deptandexp/{dept}/{exp}")
public void get3(@PathVariable String dept, @PathVariable Integer exp){
    Staff.printList(repo.findByDepartmentAndExpGreaterThan(dept,exp));
    Staff.printList(repo.findByDepartmentOrExpGreaterThan(dept,exp));
}
```

```
@GetMapping("/active/{status}")
public void get4(@PathVariable Boolean status){
    Staff.printList(repo.findByActive(status));
    Staff.printList(repo.findByActiveTrue());
    Staff.printList(repo.findByActiveFalse());
    Staff.printList(repo.findByFirstNameAndActive("Raj",status));
}
```

```
@GetMapping("/dob/{dob}")
public void get5(@PathVariable("dob")
    @DateTimeFormat(pattern = "yyyy-mm-dd") LocalDate dob){
    Staff.printList(repo.findByDobBefore(dob));
    Staff.printList(repo.findByDobAfter(dob));
}
```

```
@GetMapping("/zip/{zip}")
public void get6(@PathVariable Integer zip){
    Staff.printList(repo.findByAddressZipcode(zip));
}
```

POST

http://localhost:8080/jpa

Send

Body

ooo

raw

JSON

Beautify

```
1  {
2    .... "firstName": "Abdul",
3    .... "department": "Accounts",
4    .... "dob": "23/04/1991",
5    .... "exp": 3,
6    .... "active": true,
7    .... "address": {
8    .... | .... "street": "13th st",
9    .... | .... "city": "Chennai",
10   .... | .... "zipcode": 600100
11   .... }
12 }
```

Body



200 OK

214 ms

329 B

...

Pretty

JSON



```
1  {
2    "id": 1,
3    "firstName": "Abdul",
4    "department": "Accounts",
5    "dob": "23/04/1991",
6    "exp": 3,
7    "active": true,
8    "address": {
9      "id": 1,
10     "street": "13th st",
11     "city": "Chennai",
12     "zipcode": 600100
13   }
14 }
```

Thank you