JPA Repository CRUD

JPA Repositori SQL .

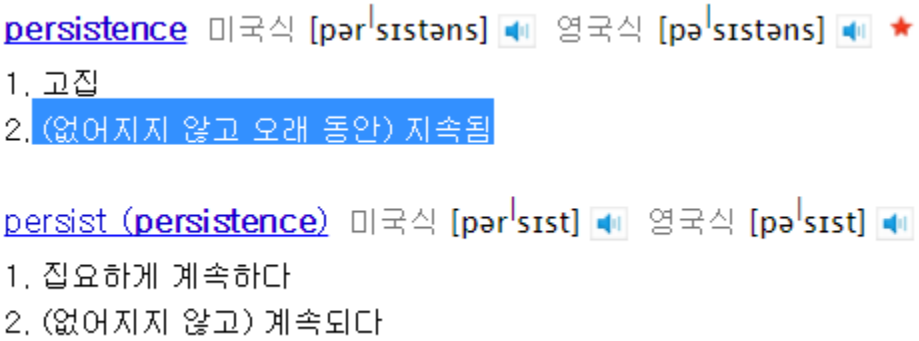
(Persistence) // .

CRUD () .

Code Link :http://git.webnori.com/projects/WEBF/repos/spring_jpa/browse/src/test/java/com/example/demo/jpa/JpaCRUD.java

Persitence

Persitence .




persistence , IT .

.

- persistent route : IP . .
- persitent FSM : AKKA , .
- redis persitence : DB redis IO .
- rdb persitence : . .

JPA Repository

JPA Repository 3 .

JpaRepository .

- PaginAndSortingRepository :
- CrudRepository :
- JpaRepository :

, SP,

.

: .

: , .

```
public interface AddressRepo extends CrudRepository<Address, Long>{

        List<Address> findBySex(String sex);

        List<Address> findBySex(String sex, Sort sort);

        List<Address> findBySexOrderByAgeDesc(String sex);

        List<Address> findByAgeGreaterThan(int age);

        List<Address> findByAgeGreaterThanEqual(int age);

        List<Address> findByAgeLessThan(int age);

        List<Address> findByAgeBetween(int low,int high);

        List<Address> findByAgeGreaterThanAndSex(int age,String sex);

        List<Address> findByAgeGreaterThanOrSex(int age,String sex);

        List<Address> findByAgeIn(int age[]);
}
```

JPQL

:SQL .

: SQL .

```
        @Query("select new com.example.demo.data.AddressStatistics(t.address,AVG(t.age)) from Address t where t.
age > :minage GROUP BY t.address HAVING AVG(t.age) > :filterage ")
        List<AddressStatistics> findRegionAvgage(
                        @Param("minage") int minage,
                        @Param("filterage") double filterage
                        );
```

Native

: SQL .

: , DB .

```
        @Query(value="SELECT address,age,name,phoneNbr,sex FROM address " +
                        "UNION " +
                        "SELECT address,age,name,phoneNbr,sex FROM address2 ",
                        nativeQuery = true)
        List<?> makeUniOn();
```

SQL

: .

: , SQL

```
        public void RankTest() {
             int[] score = {Integer.MIN_VALUE};
             int[] no = {0};
             int[] rank = {0};
             List<AddressAgeRank> ageRankList =
                                addressRepo.findByAgeBetween(10, 90).stream()
                                .sorted((a,b) -> b.getAge() - a.getAge() )
                                .map(p -> {
                               ++no[0];
                               if (score[0] != p.getAge()) rank[0] = no[0];
                               return new AddressAgeRank(p.getName(),score[0] = p.getAge(),  rank[0]
);
                                })
                                .collect(Collectors.toList());

             ageRankList.forEach(item ->{
                     System.out.println(item.toString());
             });
        }
```

QueryDSL

: , null .

: , DSL QEntityObject . ( )

```
//
public interface AddressRepoDSL extends JpaRepository<Address, Long>,
QueryDslPredicateExecutor<Address>{

}
//
        @Autowired
        private AddressRepoDSL addressRepoDSL;
        public void jpa_queryDslTest() {

                QAddress        userAddress = QAddress.address1;
                BooleanBuilder builder = new BooleanBuilder();

                builder.and(userAddress.name.eq("1") )
                           .and(userAddress.address.like("%" + "" +"%"));

                Iterable<Address> addressList =  addressRepoDSL.findAll(builder);
                addressList.forEach( item -> {
                        String itemString = String.format("%d%s %s %s %s %s",item.getId(),item.getName(),
                                        item.getPhoneNbr(), item.getSex(),item.getAddress(),item.getAge());
                        System.out.println(itemString);
                });

        }
```

:

- https://spring.io/blog/2011/02/10/getting-started-with-spring-data-jpa/
- https://docs.spring.io/spring-data/jpa/docs/1.5.0.RELEASE/reference/html/jpa.repositories.html

(Insert,Update,Delete)

: SP .

: , , JPA 100%

, TPS EntityManager Max ()

```
CREATE OR REPLACE FUNCTION addaddress(
        address character varying,
        age integer,
        name character varying,
        sex character varying,
        phonenbr character varying )

RETURNS integer AS $BODY$
        BEGIN
                INSERT INTO address ( address, age, name,sex,phonenbr) VALUES( address, age, name,sex,phonenbr);
                return 0;
        END;
$BODY$
LANGUAGE plpgsql VOLATILE
```

```
public interface AddressRepo extends CrudRepository<Address, Long>{
        @Procedure
        public Integer addaddress(String address,Integer age, String name, String sex, String phonenbr);
}


//SP  :   , Loop   .
addressRepo.addaddress("", 45, "", "", "010-1233-3321");


// :    5 6.  ,OUT  .
Hibernate:
    {call addaddress(?,?,?,?,?,?)}
```

more info: http://roufid.com/3-ways-to-call-a-stored-procedure-with-hibernate-jpa-2-1/

Persitence Context

PersitenceContext JPA , .

, JPA Persitence .

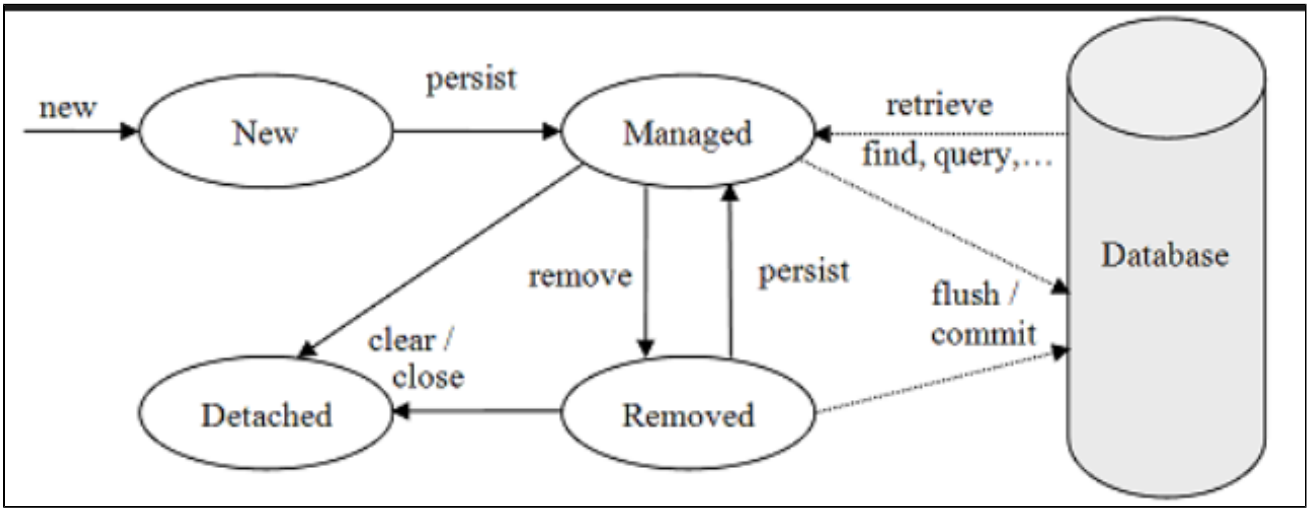
DBMS , IO .

DBMS / JPA .

DBMS , .

, DBMS IO

JPA Persitence .

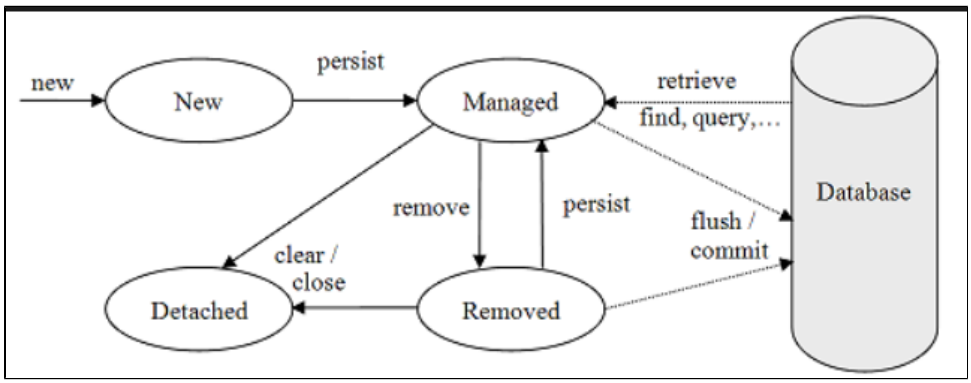
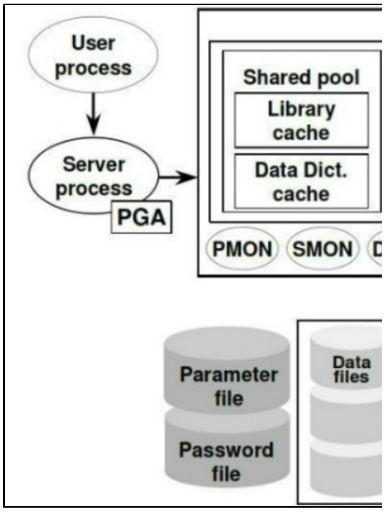
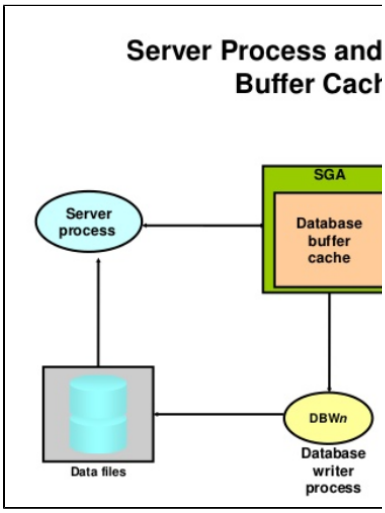


4

- (new/transient) :
- (managed) :
- (detached) :
- (removed) :

- '@Id' ( )
- (Managed) , JPA Flush
- : 1,, ,,

JPA

JPA	DBMS
	 
• https://www.java-forums.org/ocpjbcd/52967-tutorial-review-jpa-entitymanager-operations-component-developer-exam.html • http://wckhg89.tistory.com/10	• http://o2sunn.tistory.com/21 • http://sarc.io/index.php/oracledatabas

> *JPA DBMS Buffer Cache .*
>
> *JPA .*
>
> *JPA DBMS*
>
> *.*

CRUD Repository

JPA //

. .

CrudRepository .

() ,

.

```
/
public interface UserRepository extends CrudRepository<User, Long> {
}
```

, save,delete,find 3

.

```
public class UserUpdateService{
....
  @Autowired
  private UserRepository userRepository;

  public User jpatest1C(String groupname,String name) {
                GroupInfo newGroup = new GroupInfo();
                newGroup.setName(groupname);
                groupRepository.save(newGroup);
                //
        User addUser = new User();
        addUser.setName(name);
        addUser.setEmail("test@x.com");
        addUser.setGroupInfo(newGroup);
        return userRepository.save(addUser);
        }

        public void jpatest1U(User user) {
                user.setName(user.getName() + "_mody");
                userRepository.save(user);
        }

        public void jpatest1D(User user) {
                userRepository.delete(user);
        }

        public void japtestList() {
        //
        Iterable<User> userList = userRepository.findAll();
        userList.forEach(item->System.out.println( String.format("V1 Name:%s  GroupName:%s", item.getName(),
item.getGroupInfo().getName() )  ));
        }
```

OOP() , .

, /// .

Transactional , .

CRUDRepository CRUD

.

> *CrudRepository ,*
>
> *JpaRepository Repository , .*
>
> *,*
>
> *.*

```
@Transactional
public void jpa_test1() {
                User userA = jpatest1C("A","minsu");
                User userB = jpatest1C("B","minsu");
                jpatest1U(userA);
                jpatest1D(userB);
                japtestList();
        }
```

SQL .(SQL)

insert com.example.demo.data2.User – minsu,A

insert com.example.demo.data2.User – minsu,B

update com.example.demo.data2.User – minsu

delete com.example.demo.data2.User – minsu data

`japtestList() ==> ( minsu`

V1 Name:minsu_mody GroupName:A

Custom Repository

CrudRepository

. ( )

.

JPA .

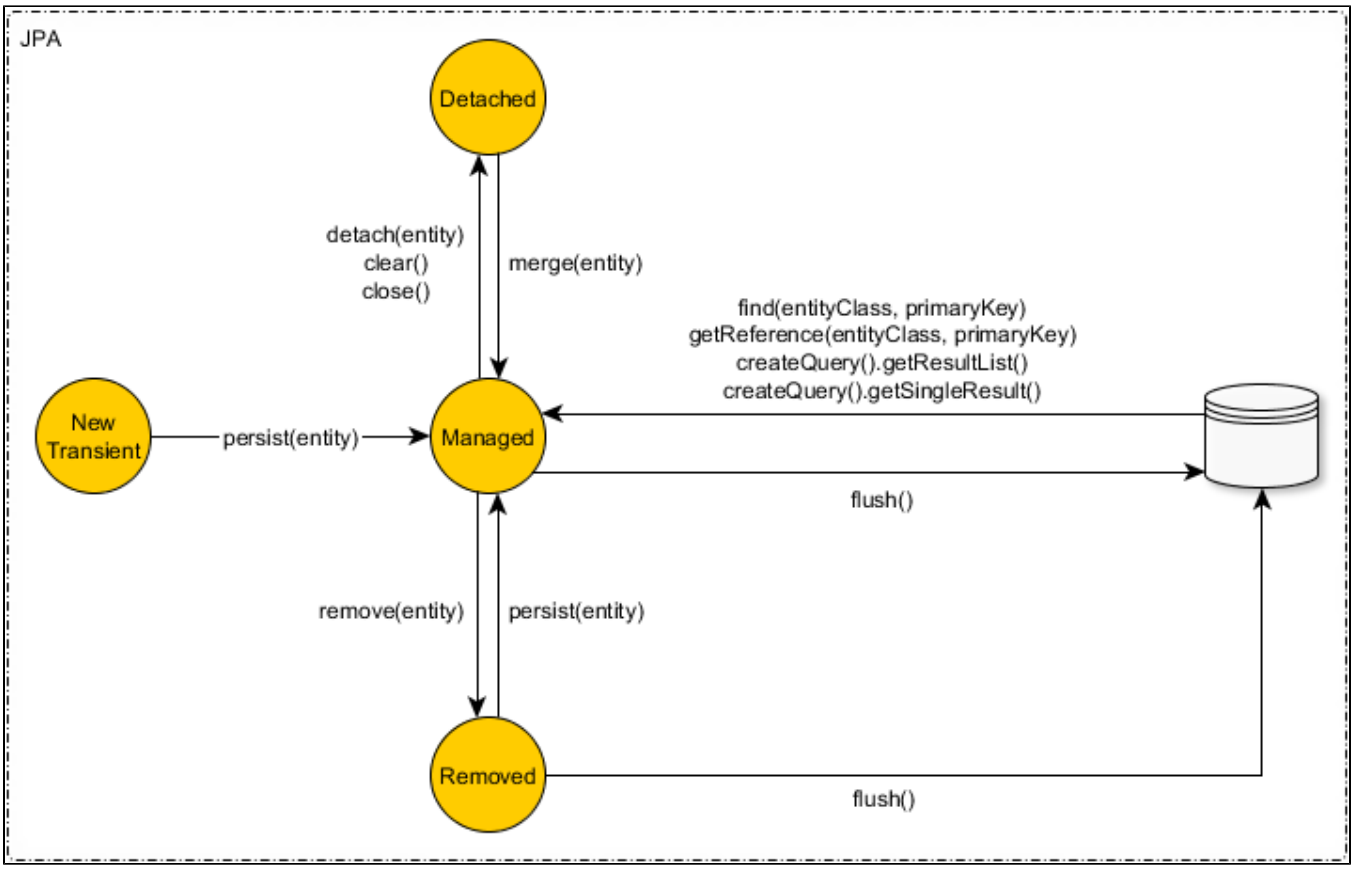
```
public interface MyAddressRepository  {
        void someTest(Address2 address2);
}


public class AddressRepositoryImpl implements MyAddressRepository {
        @PersistenceContext
        private EntityManager em;

        @Override
        @Transactional
        public void someTest(Address2 address2) {
        //    em  .
                em.refresh(address2);
                em.detach(address2);
                em.persist(address2);
                em.lock(address2, null);
                em.flush();
        }
}
public interface AddressRepository extends CrudRepository<Address2, Long>,MyAddressRepository {

}
```

`MyAddressRepository`	, . : ++ My + Address + Repository
`AddressRepositoryImpl`	. : ++ MyAddressRepositoryImpx (X)
`AddressRepository`	. JPA . : +



,

, .

2 .

Remove deatach() .

, () .

```
public class AddressRepositoryImpl implements MyAddressRepository {
        @PersistenceContext
        private EntityManager em;

        @Override
        @Transactional
        public void addAddress() {

                //(    )
                Address2 address2 = new Address2();
                address2.setName("");
                address2.setSex("");
                address2.setAddress("/");

                // , ()
                em.persist(address2);
                em.detach(address2);

                //    ...
                Address2 firstAddress = em.find(Address2.class, 1L );
                em.remove(firstAddress);

                //
                em.merge(address2);
                //    .
                address2.setName("2");

                //  Db()
                em.flush();

                //
                //em.detach(address2);
                //
                //em.merge(address2);
                //-
                //assertTrue(false);
        }
```

Hibernate:
/* insert com.example.demo.data.Address2
*/ insert
into
Address2
(address, age, name, phoneNbr, sex)
values
(?, ?, ?, ?, ?)

Hibernate:
select
address2x0_.ADDRESS_ID as ADDRESS_1_1_0_,
address2x0_.address as address2_1_0_,
address2x0_.age as age3_1_0_,
address2x0_.name as name4_1_0_,
address2x0_.phoneNbr as phoneNbr5_1_0_,
address2x0_.sex as sex6_1_0_
from
Address2 address2x0_
where
address2x0_.ADDRESS_ID=?

```
Hibernate:
/* load com.example.demo.data.Address2 */ select
address2x0_.ADDRESS_ID as ADDRESS_1_1_0_,
address2x0_.address as address2_1_0_,
address2x0_.age as age3_1_0_,
address2x0_.name as name4_1_0_,
address2x0_.phoneNbr as phoneNbr5_1_0_,
address2x0_.sex as sex6_1_0_
from
Address2 address2x0_
where
address2x0_.ADDRESS_ID=?


Hibernate:
/* delete com.example.demo.data.Address2 */ delete
from
Address2
where
ADDRESS_ID=?
```

:

- '@Transactional' , , .
- SQL/SP . SQL .
- , DB flush() .

:

- .
- , OOP .
- SP SP . SQL .

:

- DB .
- , .
- .
- ? .