## Spring Data JPA

顶层接口 `Repository` 用于宏宇类型并与之一起工作，并且被其它接口继承，用于扩展功能，都必须继承这个接口

```
public interface CrudRepository<T, ID extends Serializable>
  extends Repository<T, ID> {
  <S extends T> S save(S entity);
  Optional<T> findById(ID primaryKey);
  Iterable<T> findAll();
  long count();
  void delete(T entity);
  boolean existsById(ID primaryKey);
 // … more functionality omitted.
}
```

继续扩展功能

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

如果要使用上面的接口

```
PagingAndSortingRepository<User, Long> repository = // … get access to a bean
Page<User> users = repository.findAll(PageRequest.of(1, 20));
```

在这个接口类中，根据命名规则可以写自己的方法，Spring data jpa 会根据这些命名形成查询数据库语句。

```
interface UserRepository extends CrudRepository<User, Long> {
  long deleteByLastname(String lastname);
  List<User> removeByLastname(String lastname);
}
```

---

### 定义自己的查询

1. 可以根据命名去推断出查询语句

   `findByLastnameIgnoreCase(…)`)

   `findByLastnameAndFirstnameAllIgnoreCase(…)`

2. 写自己的查询语句

一些例子,在自己的查询语句中也可以增加`Pageable`等参数

```
Page<User> findByLastname(String lastname, Pageable pageable);
Slice<User> findByLastname(String lastname, Pageable pageable);
List<User> findByLastname(String lastname, Sort sort);
List<User> findByLastname(String lastname, Pageable pageable);
```

limiting 查询

```
User findFirstByOrderByLastnameAsc();
User findTopByOrderByAgeDesc();
Page<User> queryFirst10ByLastname(String lastname, Pageable pageable);
Slice<User> findTop3ByLastname(String lastname, Pageable pageable);
List<User> findFirst10ByLastname(String lastname, Sort sort);
List<User> findTop10ByLastname(String lastname, Pageable pageable);
```

返回结果可以是Java 8 的stream 类型

```
@Query("select u from User u")
Stream<User> findAllByCustomQueryAndStream();
Stream<User> readAllByFirstnameNotNull();
@Query("select u from User u")
Stream<User> streamAllPaged(Pageable pageable);
```

在外部也可以使用`Repository`

```
RepositoryFactorySupport factory = … // Instantiate factory here
UserRepository repository = factory.getRepository(UserRepository.class);
```

添加自定义方法(古老方法)

```
自定义Repository 接口
public interface CustomRepository {
    List<Blog> searchTitle(String key);
}

继承
@NoRepositoryBean  // 说明将 BlogRepository , 在启动springBoot 的时候不创建实例
public interface BlogRepository extends JpaRepository<Blog, Integer>,
CustomRepository {
}

实现
public class BlogRepositoryImpl implements CustomRepository {
    @PersistenceContext
```

```
    private EntityManager em;
    @Override
    public List<Blog> searchTitle(String key) {
        CriteriaBuilder builder = em.getCriteriaBuilder();
        CriteriaQuery<Blog> query = builder.createQuery(Blog.class);
        Root<Blog> root = query.from(Blog.class);
        query.where(builder.like(root.get("title"), "%" + key + "%"));
        return em.createQuery(query.select(root)).getResultList();
    }
}
```

---

## Querydsl && Specifications

*有专门笔记*

---