Spring data: Query DSL

https://github.com/codefabrim/ps-guitar-db-master.git

Advantage

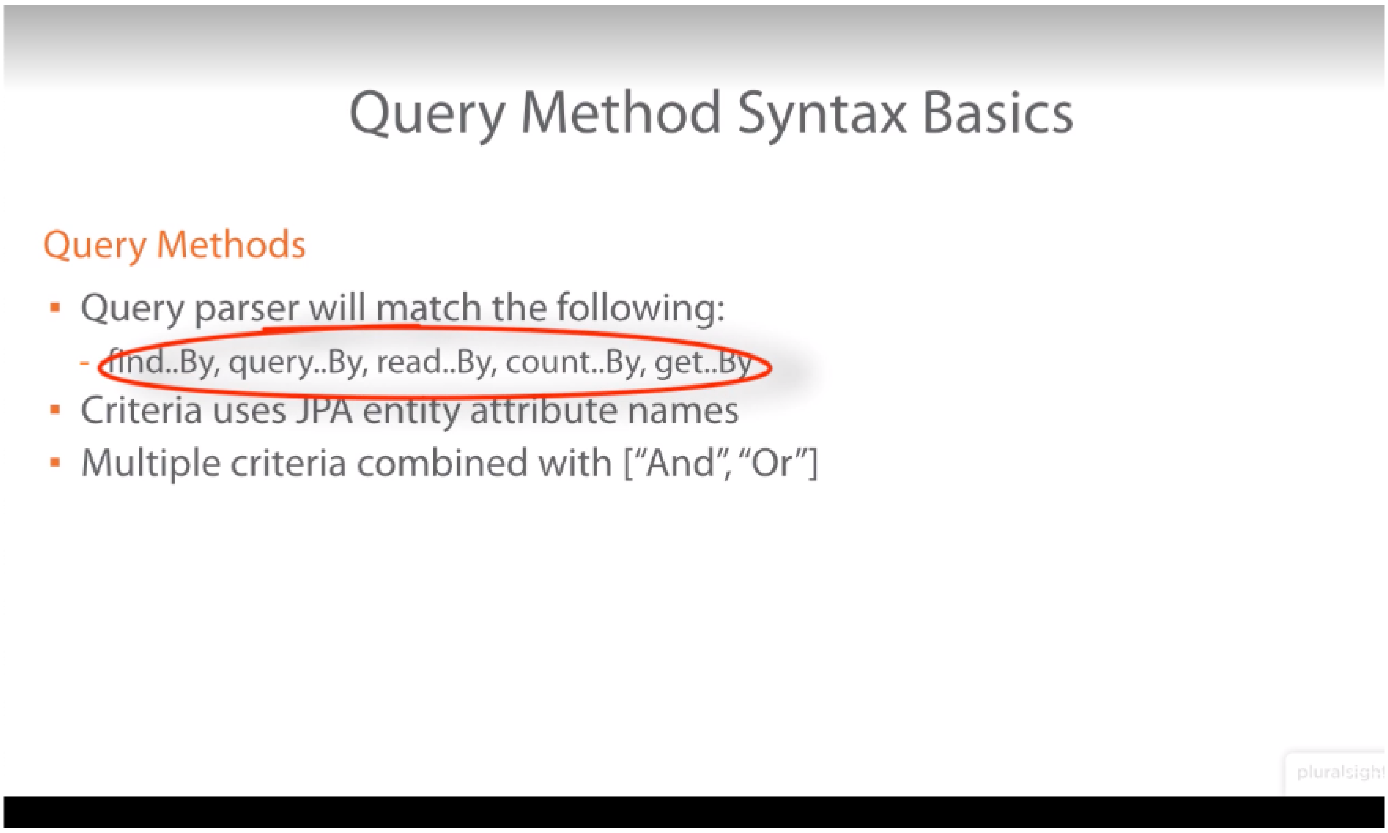
use the definition of JPA

check query at compile time

Query Methods

- find...By
- query...By
- read...By
- count...By
- get...By

Multiple Criteria can ber combine with AND , OR



Query Method Syntax Basics

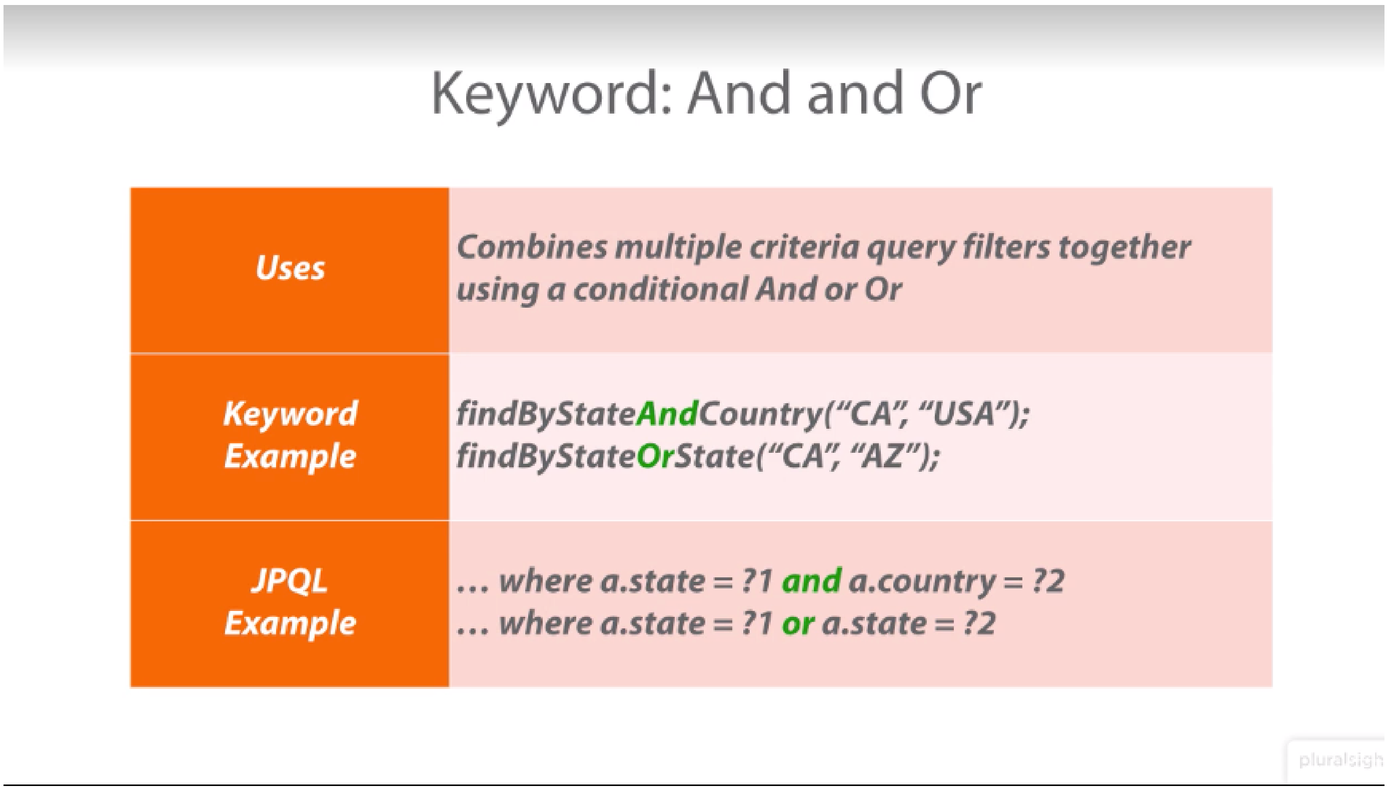
Query Methods

- Query parser will match the following:
  - find..By, query..By, read..By, count..By, get..By
- Criteria uses JPA entity attribute names
- Multiple criteria combined with ["And", "Or"]

add in LocationJPaRepository query findByStateLike and adapt test cole in LocationPersisitenceTest [LocationPersistence fixed]

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

import java.util.List;

public interface LocationJpaRepository extends JpaRepository <Location, Long
```

```

    List<Location> findByStateLike(String name);
}


//------

        @Test
        public void testFindWithLike() throws Exception {
        //      List<Location> locs = locationRepository.getLocationByStateN
                List<Location> locs = locationJpaRepository.findByStateLike(
                assertEquals(4, locs.size());
        }


```



Keyword: And and Or

Uses	*Combines multiple criteria query filters together using a conditional And or Or*

Keyword Example	*findByStateAndCountry("CA", "USA");*
	findByStateOrState("CA", "AZ");

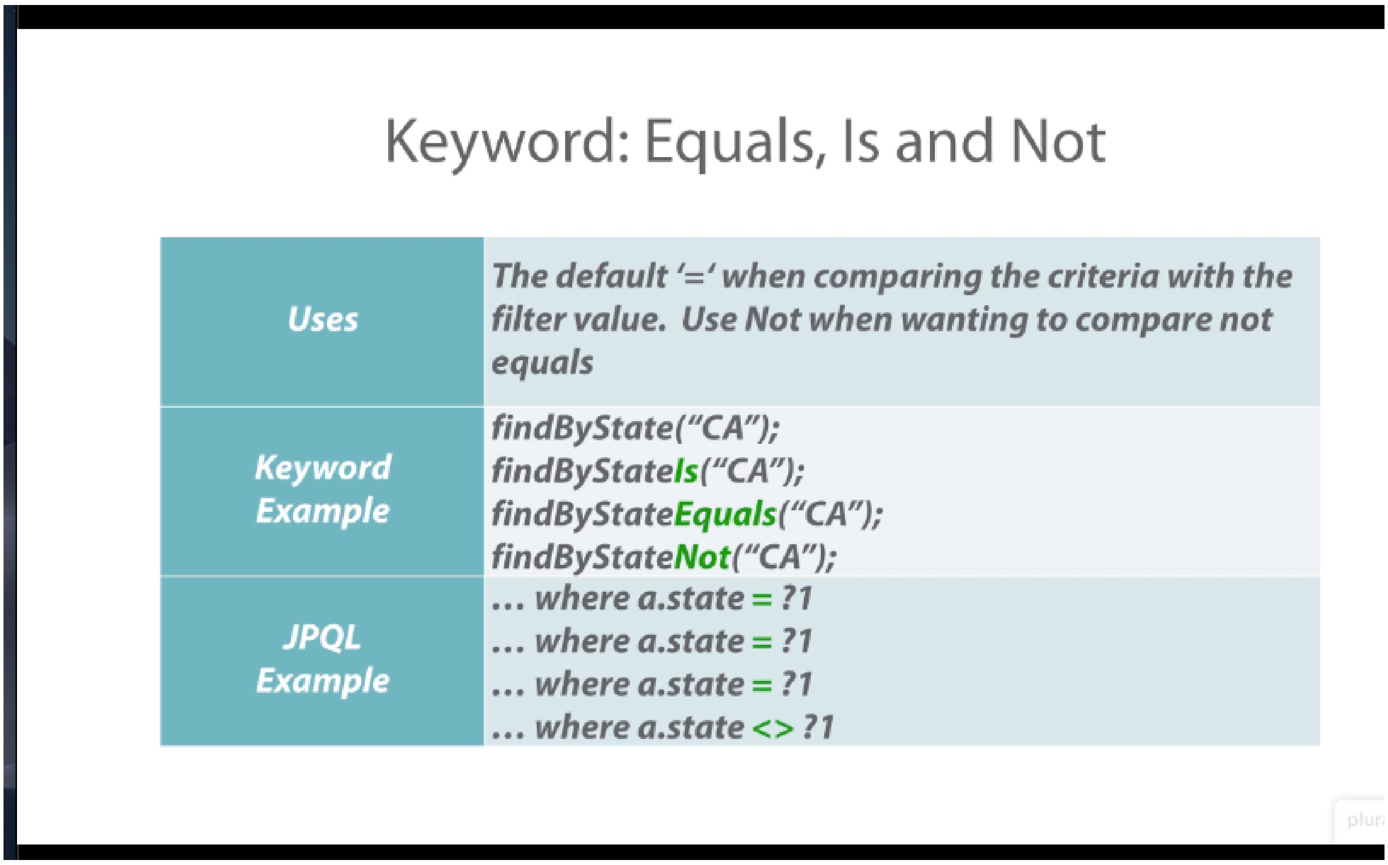
JPQL Example	*… where a.state = ?1 and a.country = ?2* *… where a.state = ?1 or a.state = ?2*

Query DSL: use of "Or" and "AND"

```

public interface LocationJpaRepository extends JpaRepository <Location, Long

    List<Location> findByStateLike(String name);


    List<Location> findByStateOrCountryLike(String value, String value2);

    List<Location> findByStateAndCountryLike(String state, String country);



}

```

```
//----

        @Test
        public void testJpaOr() throws Exception {
                List<Location> locs = locationJpaRepository.findByStateOrCou
        assertEquals("Maryland", locs.get(0).getState());
        }
```

Is , Equals



Keyword: Equals, Is and Not

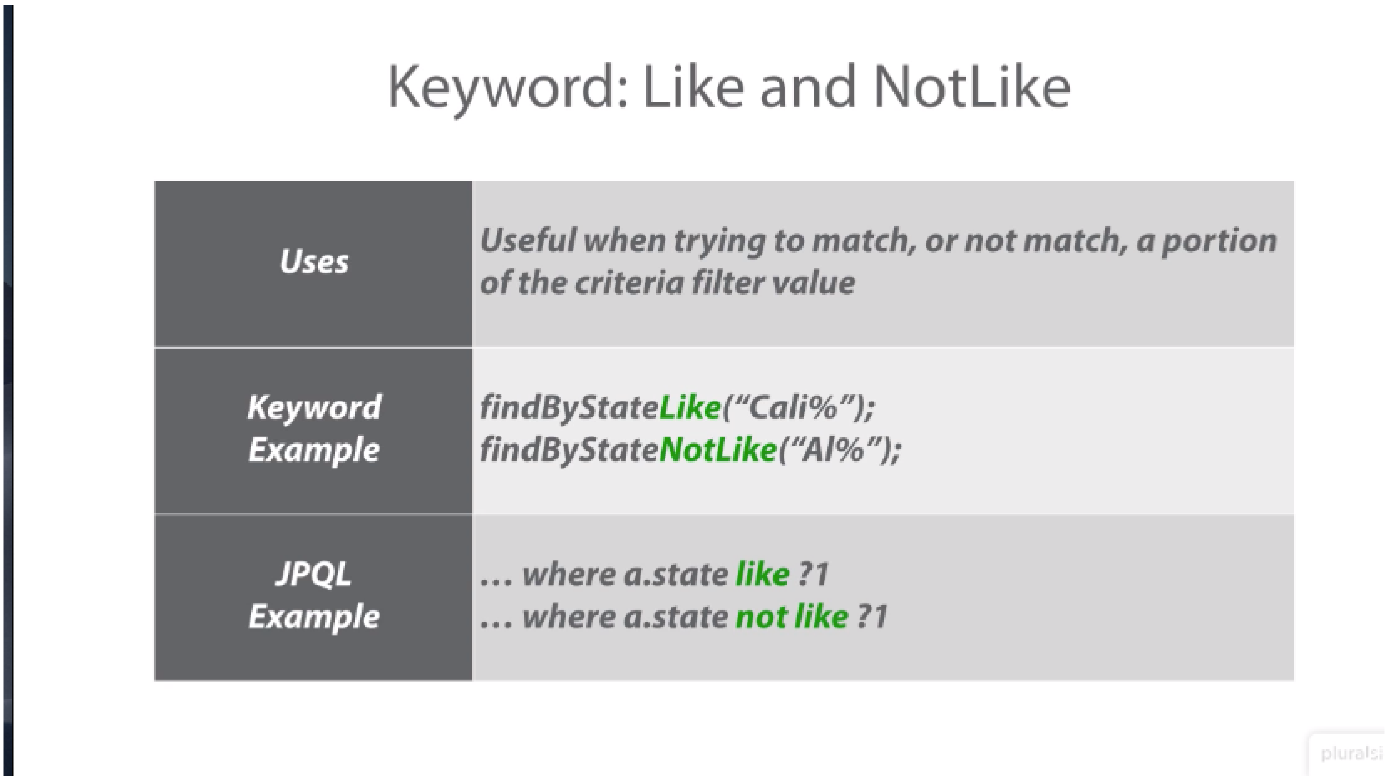
Uses	*The default '=' when comparing the criteria with the filter value. Use Not when wanting to compare not equals*

Keyword Example	*findByState("CA");*
	findByStateIs("CA");
	findByStateEquals("CA");
	findByStateNot("CA");

JPQL Example	*… where a.state = ?1* *… where a.state = ?1* *… where a.state = ?1* *… where a.state <> ?1*

```

    List<Location> findByStateIsOrCountryEquals(String value, String value2)


    List<Location> findByStateNot(String state);

 //----

        @Test
        public void testJpaAnd() throws Exception {
                List<Location> locs = locationJpaRepository.findByStateNot("
                assertNotSame("Maryland", locs.get(0).getState());

    }
        @Test
        public void testJpaOr() throws Exception {
                List<Location> locs = locationJpaRepository.findByStateIsOrC
        assertEquals("Maryland", locs.get(0).getState());
        }


```

Like and Not Like



Keyword: Like and NotLike

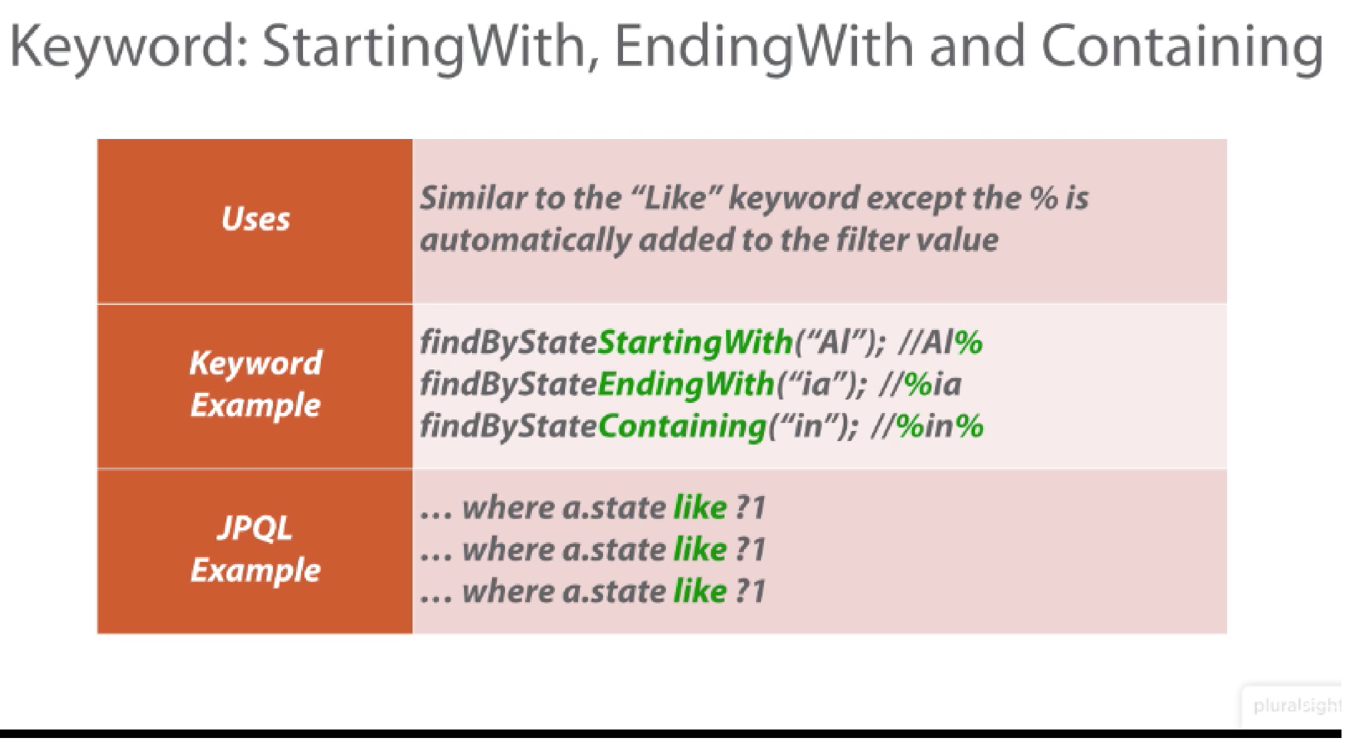
Uses	*Useful when trying to match, or not match, a portion of the criteria filter value*

Keyword Example	*findByStateLike("Cali%");*
	findByStateNotLike("Al%");

JPQL Example	*… where a.state like ?1* *… where a.state not like ?1*

```
public interface LocationJpaRepository extends JpaRepository <Location, Long

    List<Location> findByStateLike(String name);
    List<Location> findByStateNotLike(String name);
/////------

        @Test
        public void testFindWithLike() throws Exception {
                List<Location> locs = locationJpaRepository.findByStateLike(
                assertEquals(4, locs.size());

                 locs = locationJpaRepository.findByStateNotLike("New%");
                assertEquals(46, locs.size());
        }
```

StartingWith, EndingWith, Containing



Keyword: StartingWith, EndingWith and Containing

Uses	*Similar to the "Like" keyword except the % is automatically added to the filter value*

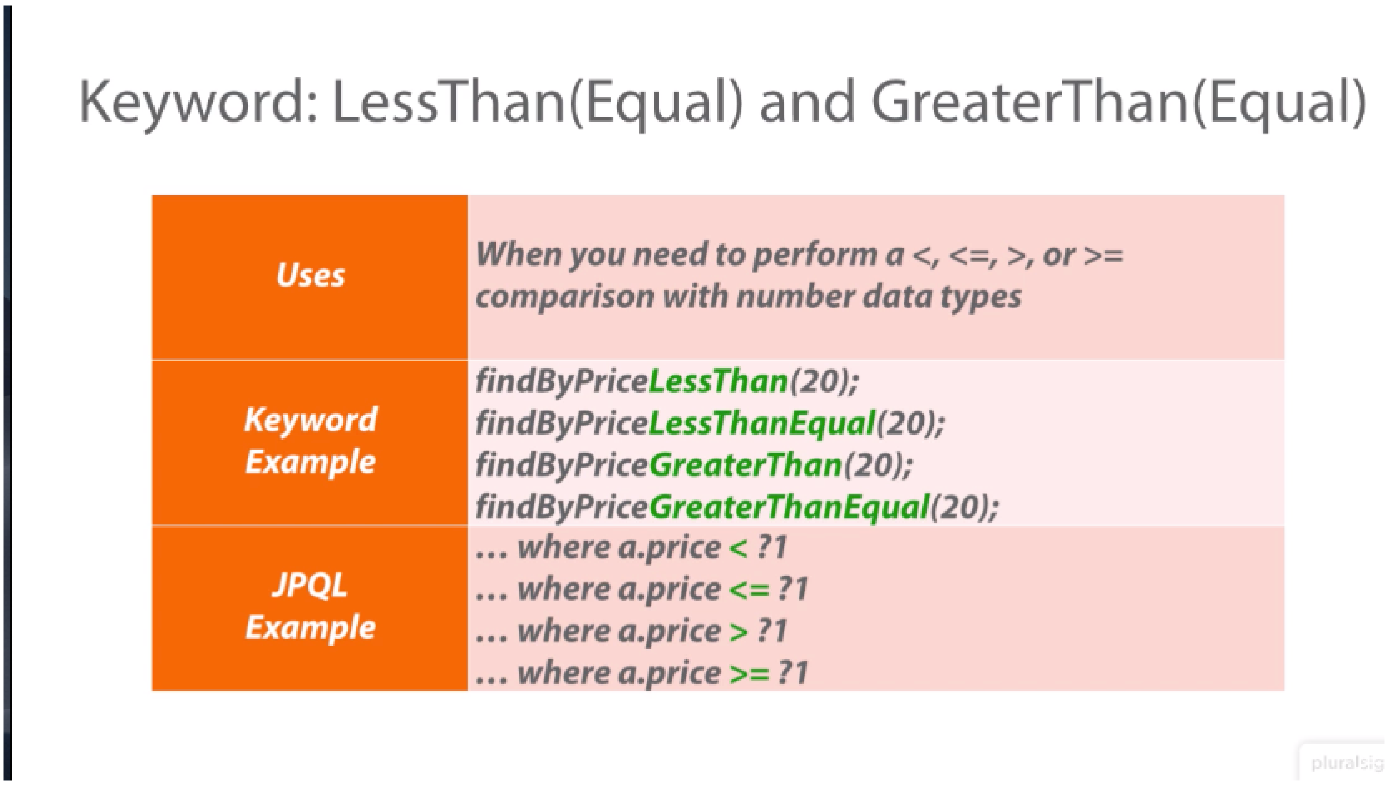
Keyword Example	*findByStateStartingWith("Al"); //Al%*
	findByStateEndingWith("ia"); //%ia
	findByStateContaining("in"); //%in%

JPQL Example	*… where a.state like ?1* *… where a.state like ?1* *… where a.state like ?1*

pluralsight

```

public interface LocationJpaRepository extends JpaRepository <Location, Long

    List<Location> findByStateLike(String name);
    List<Location> findByStateStartingWith(String name);
    List<Location> findByStateNotLike(String name);

//----
        @Test
        public void testFindWithLike() throws Exception {
                List<Location> locs = locationJpaRepository.findByStateStart
                assertEquals(4, locs.size());

                 locs = locationJpaRepository.findByStateNotLike("New%");
                assertEquals(46, locs.size());

```

Less than(Equal) , GreaterThan(Equal)



Keyword: LessThan(Equal) and GreaterThan(Equal)

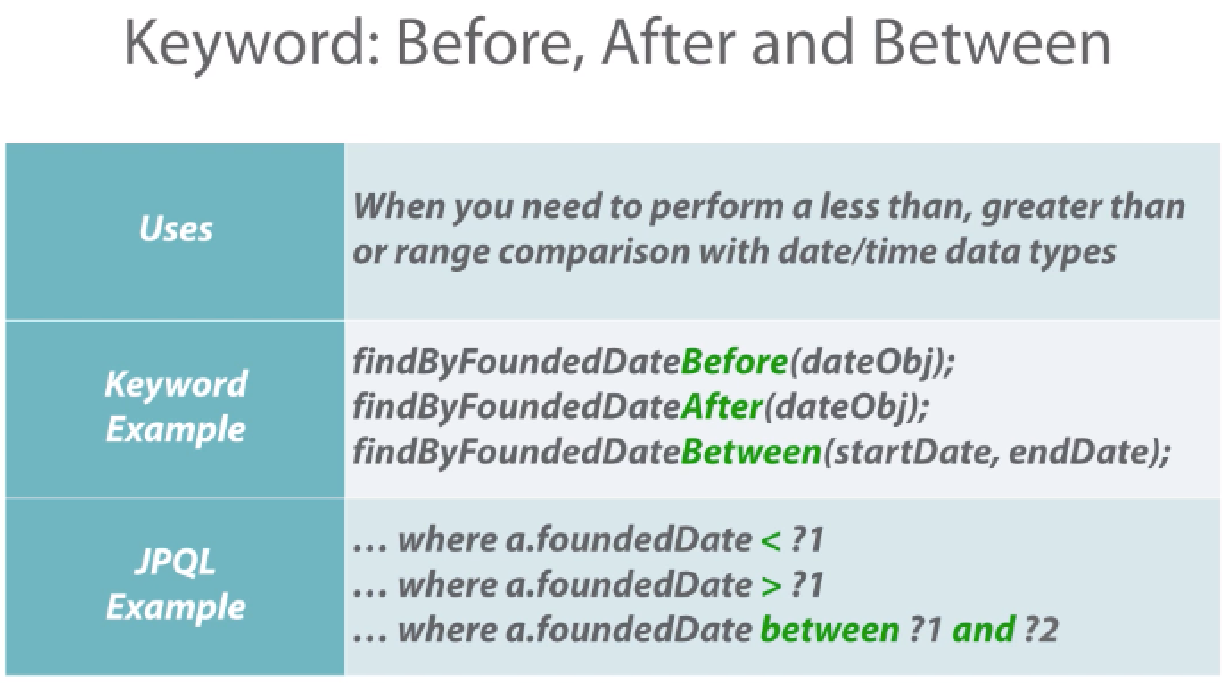
Uses	When you need to perform a <, <=, >, or >= comparison with number data types

Keyword Example	findByPriceLessThan(20);
	findByPriceLessThanEqual(20);
	findByPriceGreaterThan(20);
	findByPriceGreaterThanEqual(20);

JPQL Example	… where a.price < ?1 … where a.price <= ?1 … where a.price > ?1 … where a.price >= ?1

```
@Repository
public interface ModelJpaRepository  extends JpaRepository<Model, Long> {

    List<Model> findByPriceGreaterThanEqualAndPriceLessThanEqual(BigDecimal



-----
//ModelRepository.java
        public List<Model> getModelsInPriceRange(BigDecimal lowest, BigDecim
        /**     @SuppressWarnings("unchecked")
                List<Model> mods = entityManager
                                .createQuery("select m from Model m where m.
                                .setParameter("lowest", lowest)
                                .setParameter("highest", highest).getResultL
                */
                List<Model> mods = modelJpaRepository.findByPriceGreaterThan
                return mods;
        }
----
        @Test
        public void testGetModelsInPriceRange() throws Exception {
//              List<Model> mods = modelRepository.getModelsInPriceRange(Big
//              List<Model> mods = modelJpaRepository.getModelsInPriceRange(
                List<Model> mods = modelJpaRepository.getModelsInPriceRange(
                assertEquals(4, mods.size());
        }

```

##Beforer, After, Between



Keyword: Before, After and Between

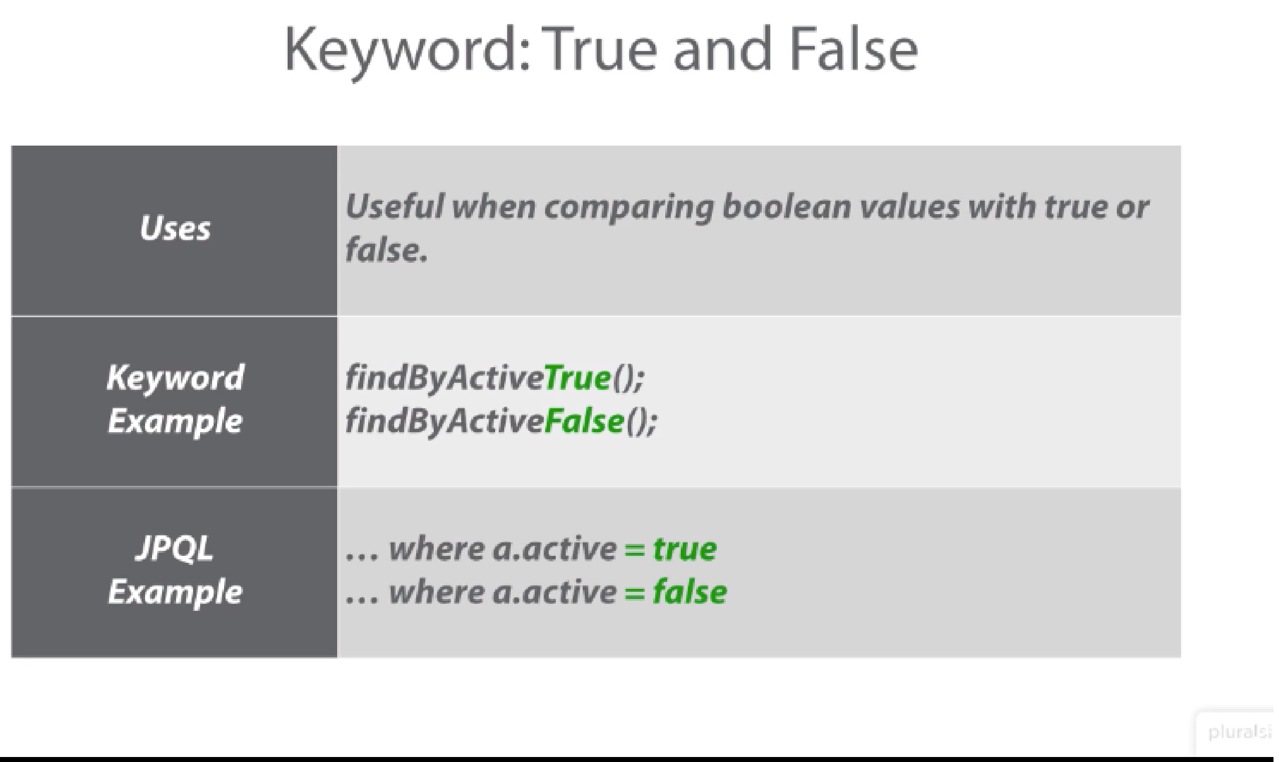
Uses	*When you need to perform a less than, greater than or range comparison with date/time data types*

Keyword Example	*findByFoundedDate**Before**(dateObj);*
	*findByFoundedDate**After**(dateObj);*
	*findByFoundedDate**Between**(startDate, endDate);*

JPQL Example	***… where a.foundedDate < ?1*** ***… where a.foundedDate > ?1*** ***… where a.foundedDate between ?1 and ?2***

```
@Repository
public interface ManufacturerJpaRepository  extends JpaRepository <Manufactu

    List<Manufacturer> findByFoundedDateBefore(Date d);

------
ManufacturerRepository.java
        /**
         * Custom finder
         */
        public List<Manufacturer> getManufacturersFoundedBeforeDate(Date dat
//              @SuppressWarnings("unchecked")
//              List<Manufacturer> mans = entityManager
//                              .createQuery("select m from Manufacturer m w
//                              .setParameter("date", date).getResultList();
                List<Manufacturer> mans = manufacturerJpaRepository.findByFo
                return mans;
        }
```

True and False



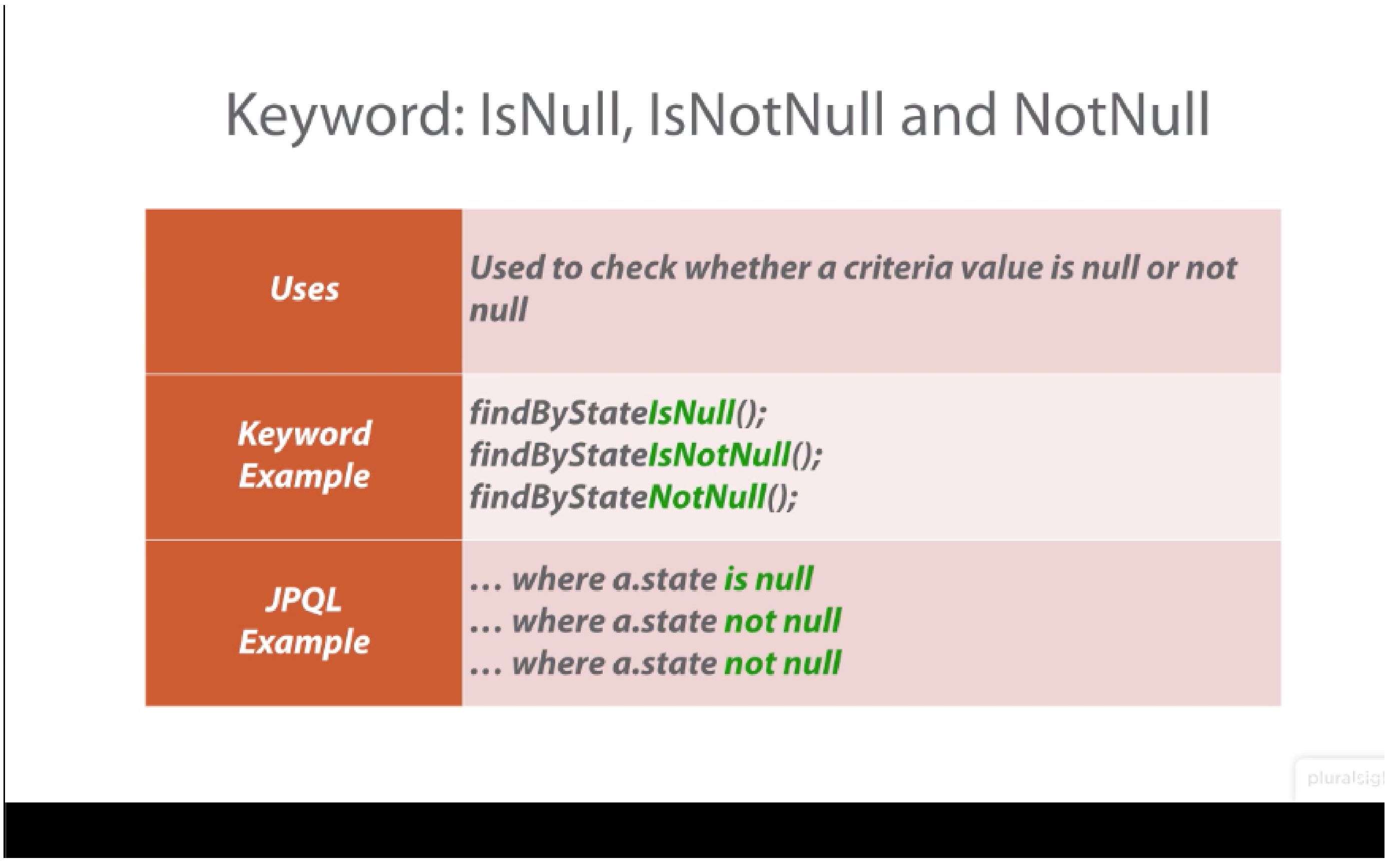
Keyword: True and False

Uses	*Useful when comparing boolean values with true or false.*

Keyword Example	*findByActive**True**();*
	*findByActive**False**();*

JPQL Example	*… where a.active = **true*** *… where a.active = **false***

```
public interface ManufacturerJpaRepository  extends JpaRepository <Manufactu

    List<Manufacturer> findByFoundedDateBefore(Date d);

    List<Manufacturer> findByActiveTrue();
    List<Manufacturer> findByActiveFalse();

-----


        /**
         * Native Query finder
         */
        public List<Manufacturer> getManufacturersThatSellModelsOfType(Strin
//              @SuppressWarnings("unchecked")
//              List<Manufacturer> mans = entityManager
//                              .createNamedQuery("Manufacturer.getAllThatSe
//                              .setParameter(1, modelType).getResultList();
//              return mans;
                List<Manufacturer> mans = manufacturerJpaRepository.findByAc
                return mans;
        }
```



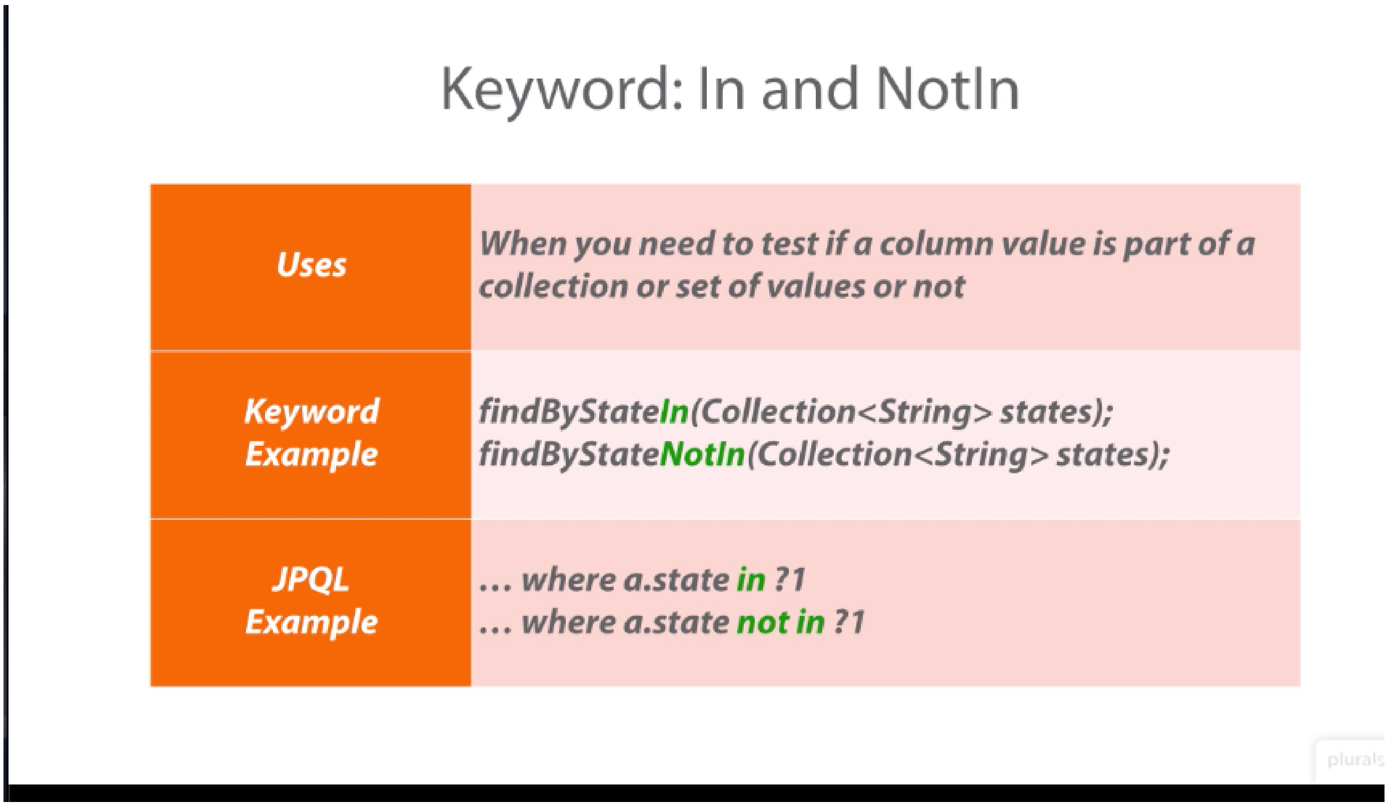
Keyword: IsNull, IsNotNull and NotNull

Uses	Used to check whether a criteria value is null or not null

Keyword Example	findByStateIsNull();
	findByStateIsNotNull();
	findByStateNotNull();

JPQL Example	… where a.state is null … where a.state not null … where a.state not null

In , NotIn



Keyword: In and NotIn

Uses	When you need to test if a column value is part of a collection or set of values or not

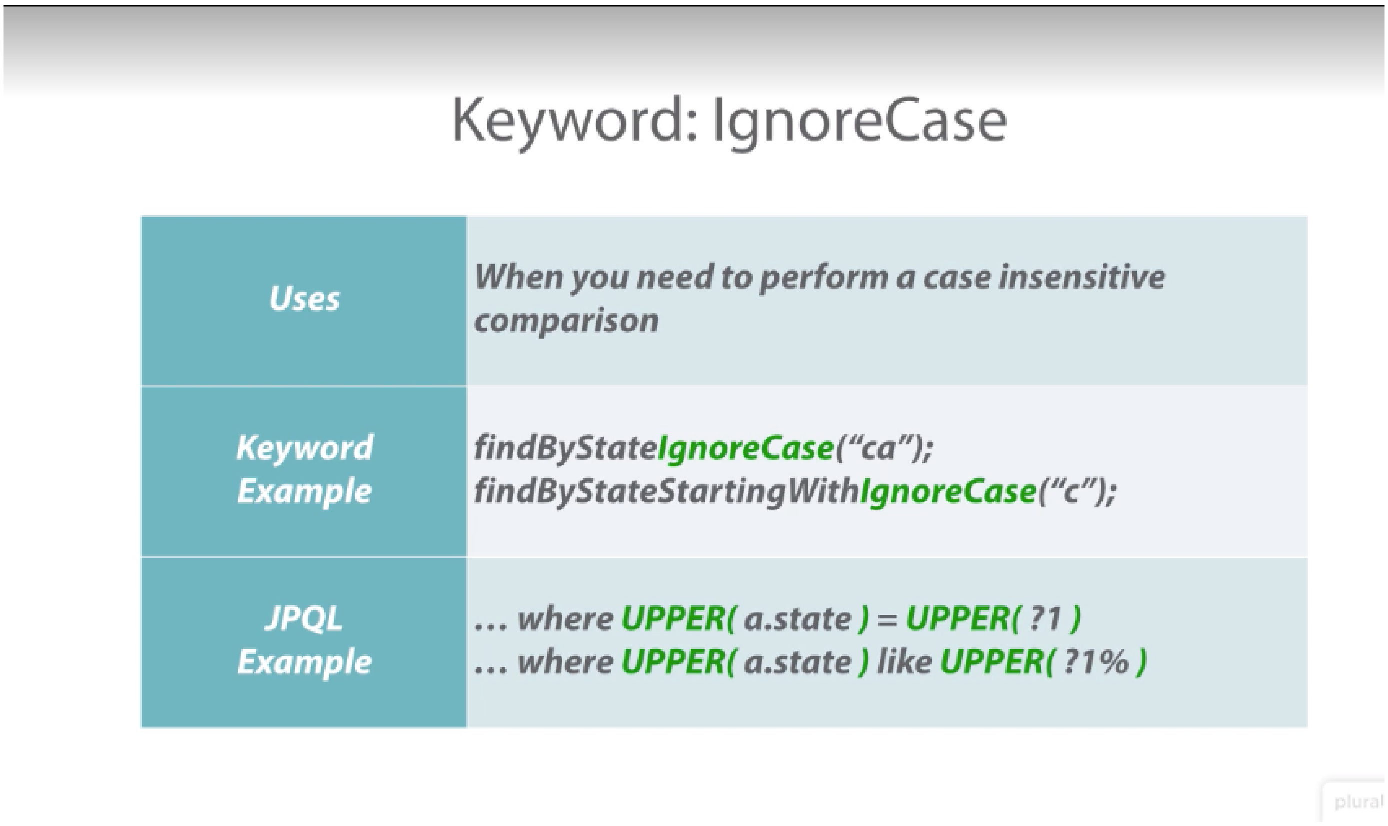
Keyword Example	findByStateIn(Collection<String> states);
	findByStateNotIn(Collection<String> states);

JPQL Example	… where a.state in ?1 … where a.state not in ?1

```
public interface ModelJpaRepository  extends JpaRepository<Model, Long> {

    List<Model> findByPriceGreaterThanEqualAndPriceLessThanEqual(BigDecimal

    List<Model> findByModelTypeNameIn(List<String> types);

```

```

}
///----
@Test
        public void testGetModelsByTypes() throws Exception {

                List<String > types = new ArrayList<String>();
                types.add("Electric");
                types.add("Acoustic");
                types.add("Bass");
                List<Model> mods = modelJpaRepository.findByModelTypeNameIn(

                mods.forEach((model) ->  {
                        assertTrue(
                                        (model.getModelType().getName().equa
                                                        (model.getModelType(
                        );
                });
                assertEquals(4, mods.size());
        }
```

IgnoreCase



Keyword: IgnoreCase

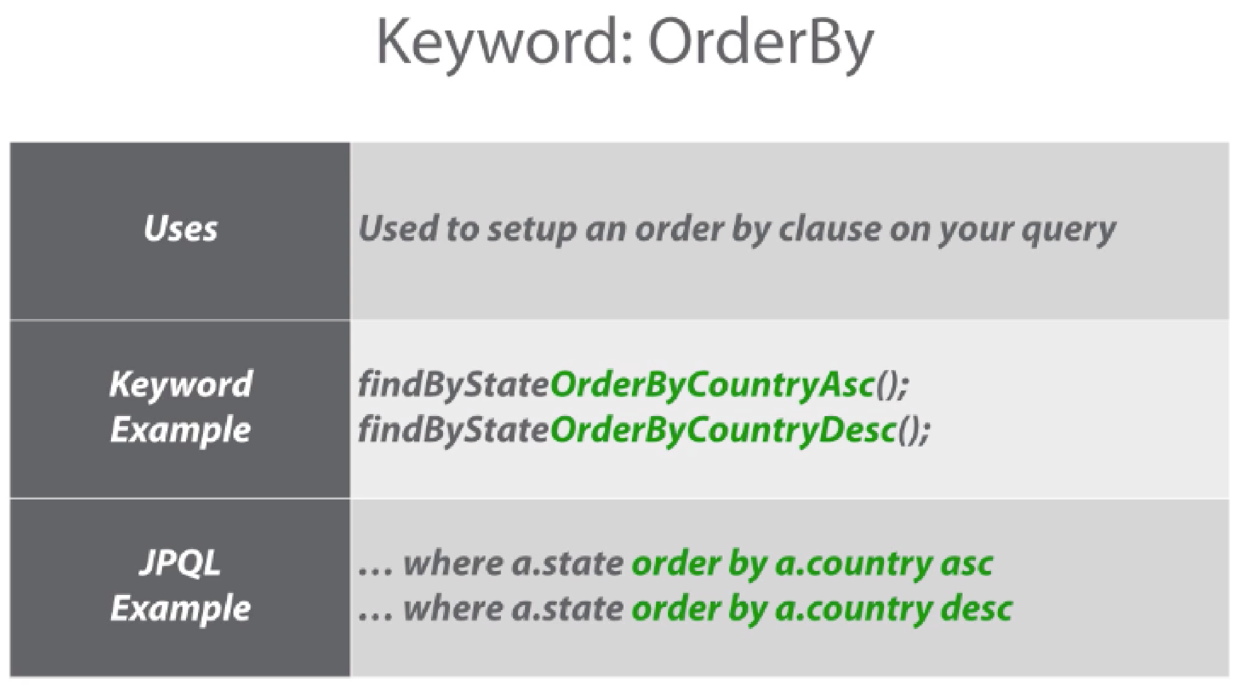
Uses	*When you need to perform a case insensitive comparison*

Keyword Example	*findByStateIgnoreCase("ca");*
	findByStateStartingWithIgnoreCase("c");

JPQL Example	*… where UPPER(a.state) = UPPER(?1)* *… where UPPER(a.state) like UPPER(?1%)*

```
public interface LocationJpaRepository extends JpaRepository <Location, Long
//    List<Location> findByStateStartingWith(String name);
    List<Location> findByStateIgnoreCaseStartingWith(String name);
    List<Location> findByStateNotLike(String name);

    ------

    @Test
        public void testFindWithLike() throws Exception {
    //          List<Location> locs = locationJpaRepository.findByStateStart
                List<Location> locs = locationJpaRepository.findByStateIgnor
                assertEquals(4, locs.size());

                 locs = locationJpaRepository.findByStateNotLike("New%");
                assertEquals(46, locs.size());

```

OrderBy



Keyword: OrderBy

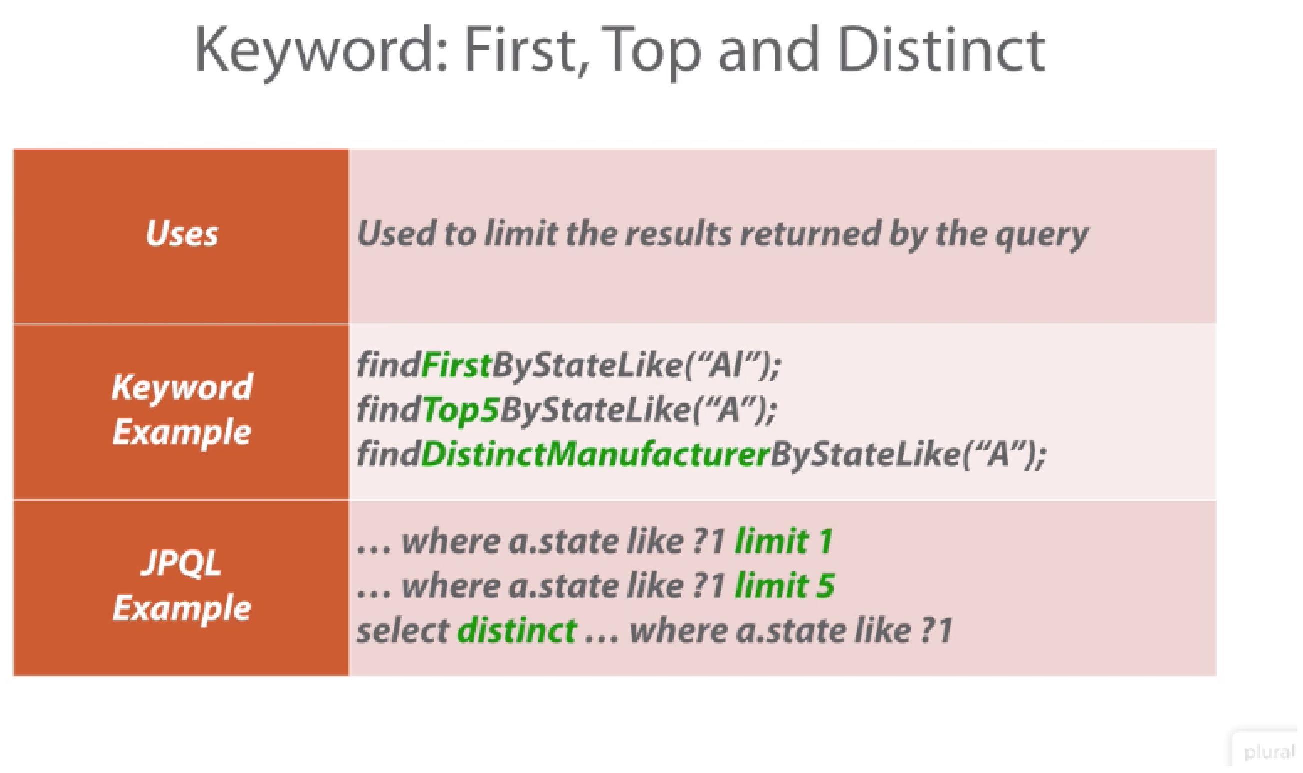
Uses	*Used to setup an order by clause on your query*

Keyword Example	*findByStateOrderByCountryAsc();*
	findByStateOrderByCountryDesc();

JPQL Example	*… where a.state order by a.country asc* *… where a.state order by a.country desc*

```
public interface LocationJpaRepository extends JpaRepository <Location, Long

//    List<Location> findByStateStartingWith(String name);
    List<Location> findByStateIgnoreCaseStartingWith(String name);

    List<Location> findByStateNotLikeOrderByStateAsc(String name);

//-----
public void testFindWithLike() throws Exception {
                List<Location> locs = locationJpaRepository.findByStateIgnor
                assertEquals(4, locs.size());

           locs = locationJpaRepository.findByStateNotLikeOrderByStateAsc("N
                assertEquals(46, locs.size());

                locs = locationJpaRepository.findByStateNotLikeOrderByStateA
                assertEquals("Alabama", locs.get(0).getState());

                locs.forEach(
                                (item) -> {
                                        System.out.println(item.getState());
                                });
        }


```

First, Top, Distinct



Keyword: First, Top and Distinct

Uses	*Used to limit the results returned by the query*

Keyword Example	*findFirstByStateLike("Al");*
	findTop5ByStateLike("A");
	findDistinctManufacturerByStateLike("A");

JPQL Example	*… where a.state like ?1 limit 1* *… where a.state like ?1 limit 5* *select distinct … where a.state like ?1*

```
public interface LocationJpaRepository extends JpaRepository <Location, Long

//    List<Location> findByStateStartingWith(String name);
    List<Location> findByStateIgnoreCaseStartingWith(String name);


    List<Location> findFirstByStateIgnoreCaseStartingWith(String name);
    ///----test
        locs   = locationJpaRepository.findFirstByStateIgnoreCaseStartingWith
                assertEquals("Alabama", locs.get(0).getState());


```