ARTIGO: 11370
Usando Dynamic Finders
Introdução

Nesse artigo vemos ver um recurso muito interessante chamado *Dynamic Finder*. Esse recurso permite que através do nome do método seja construída uma consulta SQL.

Uso

Os Dynamic Finders devem seguir um padrão de construção:

Java Code

```
<Tipo do retorno>  [<identificação do retorno> | <condição> | <nome do atributo> | <ordenação>]
```

As palavras chave disponíveis para a contrução são:

Java Code

```
* Tipo de consulta. Toda busca dinâmica deve começar com uma dessas key words
findBy - retorna um resultado
findAllBy - Retorna uma lista de resultados
findAllOrderBy - select * from order by field
countBy - Retorna uma contagem (int)


* Condições pré atributo
And = where field01 = value01 and field02 = value02
Between = between value01 and value02
IgnoreCase = where upper(field) = upper(value)
Not = where field != value

* Complementos pós atributo.
StartsWith = where  field like 'value%'
AnyWhere = where field like '%value%'
EndsWith = where field like '%value'
IsNull = where field is not null
NotIsNull = where field is null

* Ordenação pós atributo
OrderBy = order by field
OrderDesc = order by field desc
```

Exemplo de uso:

Java Code

```
public static interface Finder {

        List<DynamicFinderEntity> findAllByName(String nome);

        List<DynamicFinderEntity> findAllByNameStartsWith(String nome);

        List<DynamicFinderEntity> findAllByNameEndsWith(String nome);

        List<DynamicFinderEntity> findAllByNameAndCampo1AndCampo2AndCampo3(String args1, String args2, String args3, String args4);

        List<DynamicFinderEntity> findAllByNameStartsWithAndCampo1StartsWithAndCampo2StartsWithAndCampo3StartsWith(String args1, String args2, String args3,
String args4);

        List<DynamicFinderEntity> findAllByNameEndsWithAndCampo1EndsWithAndCampo2EndsWithAndCampo3EndsWith(String args1, String args2, String args3, String
args4);

        List<DynamicFinderEntity> findAllByNameAnyWhere(String ri);

        List<DynamicFinderEntity> findAllByNameAndCampo1AnyWhere(String args1, String args2);

        DynamicFinderEntity findByName(String args);

        DynamicFinderEntity findByNameStartsWith(String args);

        DynamicFinderEntity findByNameEndsWith(String args);

        DynamicFinderEntity findByNameStartsWithAndCampo1EndsWith(String args, String args2);

        DynamicFinderEntity findByNameStartsWithNot(String args);

        List<DynamicFinderEntity> findAllOrderByName();

        List<DynamicFinderEntity> findAllOrderByNameOrderDesc();

        List<DynamicFinderEntity> findAllByNameStartsWithOrderByName(String args1);

        List<DynamicFinderEntity> findAllByNameStartsWithOrderByNameOrderDesc(String args1);

        DynamicFinderEntity findByAgeBetweenOrderByName(int arg1, int arg2);

        DynamicFinderEntity findByNameIsNull();
```

```
        DynamicFinderEntity findByNameNotIsNull();

        DynamicFinderEntity findByCampo1IsNullNameNotIsNull();

        DynamicFinderEntity findByNameAndCampo1IsNull(String arg);

        List<DynamicFinderEntity> findAllByNameAndCampo1IsNull(String arg);

        int countByName(String arg);

        int countByNameAndId(String arg, Long id);
    }
```

Por padrão já existe um modelo de repositório que pode ser usado para consultas dinâmicas. Esse modelo já define boa parte das operações necessárias para a manipulação do banco de dados, como save, delete e update.

Java Code

```
br.com.mobilemind.veloster.orm. RepositoryModel<TipoDaEntidade>
```

Excemplo:

Java Code

```
interface FinderModel extends RepositoryModel<DynamicFinderEntity> {
      List<DynamicFinderEntity> findAllByNameAnyWhere(String ri);
}
```

Uso do Dynamic Finder

Para usarmos um dynamic finder, devemos recorer ao repositório de serviços, chamando o método *getDynamicFinder* passando como parametro a interface que define as operações dinâmicas e o tipo da entidade persistente

Java Code

```
Finder finder = VelosterRepository.getDynamicFinder(Finder.class, FinderEntity.class);
List<FinderEntity> items = finder.findAllByNameEndsWith("foo") //select * from FinderEntity where name like '%foo'
```