


```
DynamicFinderEntity findByNameNotNull();

DynamicFinderEntity findByCampo1IsNullNameNotNull();

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>
```

Exemplo:

Java Code

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

Uso do Dynamic Finder

Para usarmos um dynamic finder, devemos recorrer 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'
```