

Testando alteração e deleção

Até agora testamos operações de inserção e seleção. Chegou a hora de testar remoção e atualização. Veja o método `deleta()` no `UsuarioDao`:

```
public void deletar(Usuario usuario) {  
    session.delete(usuario);  
}
```

Ele deleta o usuário que é passado. Para testar uma deleção, precisamos primeiro inserir um elemento, depois deletá-lo, e depois fazer uma consulta para garantir que ele não está lá. Vamos aos testes:

```
@Test  
public void deveDeletarUmUsuario() {  
    Usuario usuario =  
        new Usuario("Mauricio Aniche", "mauricio@aniche.com.br");  
  
    usuarioDao.salvar(usuario);  
    usuarioDao.deletar(usuario);  
  
    Usuario usuarioNoBanco =  
        usuarioDao.porNomeEEEmail("Mauricio Aniche", "mauricio@aniche.com.br");  
  
    assertNull(usuarioNoBanco);  
}
```

Excelente, o teste está da maneira que pretendíamos. Mas se rodarmos, ele falha! A pergunta é: por que?

Muitas vezes nossa camada de acesso à dados não envia as consultas SQL para o banco até que você finalize a transação. É o que está acontecendo aqui. Precisamos então forçar o envio do INSERT e do DELETE para o banco, para que depois o SELECT não traga o objeto! Essa operação é chamada de "flush":

```
@Test  
public void deveDeletarUmUsuario() {  
    Usuario usuario =  
        new Usuario("Mauricio Aniche", "mauricio@aniche.com.br");  
  
    usuarioDao.salvar(usuario);  
    usuarioDao.deletar(usuario);  
  
    // envia tudo para o banco de dados  
    session.flush();  
  
    Usuario usuarioNoBanco =  
        usuarioDao.porNomeEEEmail("Mauricio Aniche", "mauricio@aniche.com.br");  
  
    assertNull(usuarioNoBanco);  
}
```

Agora sim nosso teste passa!

É muito comum fazer uso de "flush" sempre que fazemos testes com banco de dados. Dessa forma, garantimos que a consulta realmente chegou ao banco de dados, e que as futuras consultas levarão mesmo em consideração as consultas anteriores!

Veja que problemas como esses aparecerão sempre em testes de integração. Precisamos entender bem do sistema que está do outro lado para conseguirmos escrever um bom teste!