What database tests should I write so that refactoring database is easy [closed]

Asked 15 years, 11 months ago Modified 15 years, 11 months ago

Viewed 196 times

3

Closed. This question needs to be more focused. It is not currently accepting answers.

Want to improve this question? Update the question so it focuses on one problem only by editing this post.

Closed 7 years ago.

Improve this question

Are there any guidelines on writing database tests so that you can refactor database "without fear" while doing evolutionary database design?

What aspects of database should be put to test while developing it? Any example would be great..

database refactoring agile

Share

edited Jan 14, 2009 at 17:34

Improve this question

Follow

asked Jan 14, 2009 at 17:28

StackUnderflow

25.2k ● 15 ● 56 ● 77

4 Answers

Sorted by: Highest score (default)

2

I write tests that call my dal code and after I check if the inserts/updates/deletes actually occurred, so called state tests. These are pr definition not Units test but rather integration tests, but they have actually helped me many times doing database changes. Over the time I have more and better tests and even bigger database changes go rather seamlessly.

Share Improve this answer

Follow

answered Jan 14, 2009 at 17:42

terjetyl

9,565 ● 5 ● 56 ● 73

1

The concept of having tests, according to TDD, is that you test the way things are meant to work *now*. When you need to change something, you change the tests that address that, (ensure the current codebase fails), and then make the code changes until the tests pass. The tests that didn't change give you confidence that other aspects of your software is still doing what it used to and that your refactoring hasn't broken anything.

So you don't write tests with refactoring in mind; rather, you test your requirements, and then when your requirements change you update the tests accordingly, and refactor the *code* so it passes the new *tests*.

Share Improve this answer

Follow

answered Jan 14, 2009 at 17:31

Andrzej Doyle

104k ● 33 ● 191 ● 231

1

Always fear every change you make to a database. This will compel you to double check every thing you change and narrow down the possible errors.

Keep in mind that databases are usually the backbone of one or more application, so every change you make must be planned carefully and properly tested.

Share Improve this answer

Follow

answered Jan 14, 2009 at 17:33

Sergio

8,251 ● 10 ● 48 ● 77

-1

When writing stored procedures, I liberally add Debug / Print statements. With @Debug as a input parameter of type bit :

```
IF @Debug = 1 PRINT @MyDynamicVariable
```

Share Improve this answer

Follow

answered Jan 14, 2009 at 18:22

Ash Machine

9,891 ● 11 ● 47 ● 52

An optional (and last) input variable, defaulted to 0.
– Mark Brackett Jan 14, 2009 at 18:27