How much should one DataSet represent?

Asked 16 years, 3 months ago Modified 9 years ago Viewed 272 times

1

How much should one DataSet represent? Using the example of an ordering system: While showing your order I also show a list of items similar to one of yours as well as a list of our most popular items. While your items are tangled in a web of relationships involving you and your past orders, preferred suppliers, and the various other kinds of information related to you as a client, the other items do not have these same relationships. The set of queries I use to navigate the set of stuff representing you is different than the queries I use for one of these other lists of items.

My inclination is to create different DataSets for different kinds of relationships but then I create ten separate item DataTables and that seems wrong. When I instantiate the larger DataSet even though I am only interested in a small subset that seems wrong, and when I try to pack all of these into one DataSet I have a big messy looking thing with several items tables next to each other and I am pretty sure that IS wrong.

Maybe I am over-valuing the relationship feature of DataSets or maybe I just need to get over myself, either way I could use some guidance.

.net dataset

Share

Improve this question

Follow

edited Dec 20, 2015 at 22:59

Termininja

6,998 ● 12 ● 50 ● 50

asked Aug 30, 2008 at 19:08

Harry

13 ● 5

2 Answers

Sorted by: Highest score (default)

▲

4

▼

The DataSet is vastly overrated and overused. Use strongly-typed collections (thank you, generics and automatic properties!). As icing on the cake, you can now even do cool query things against your custom objects with LINQ.

Good Esposito article on datasets versus custom objects:

http://msdn.microsoft.com/en-us/magazine/cc163751.aspx

Automatic properties:

http://weblogs.asp.net/dwahlin/archive/2007/12/04/c-3-0-features-automatic-properties.aspx

LINQ with your objects:

http://blogs.msdn.com/wriju/archive/2006/09/16/linq-custom-object-query.aspx

Share Improve this answer Follow

answered Aug 30, 2008 at 19:23

rp.
17.6k ● 14 ● 65 ● 79

▲
1
▼
✓

This is why I don't use datasets. If you use strongly-typed datasets you benefit from the strong typing but you pay for it in terms of the time it takes to create one even if you're just using part of it and its extensibility in terms of the code base. If you want to modify an existing one and you modify a row definition then this will create "shotgun" breaks in the code base as each definition for adding a new row will have to be modified as it wont compile anymore.

To avoid the above scenario the most sensible approach is to generally give up on sensible re-use. Define a dataset per purpose and per use. However the main issue with this is API use, you end up with dataset that is simliar to another dataset but because it is a different dataset type you have to transform it to use the common API which is both painful and inelegant.

This, plus the fact that strongly typed datasets make your code look horrid (the length of the type declarations) are pretty much the reasons i've given up on datasets and switched to business objects instead.

Share Improve this answer

Follow

answered Aug 30, 2008 at 19:25

Quibblesome

25.4k ● 10 ● 62 ● 104