EOF-Using EOF-Raw Rows

Overview

Think Twice Before Using Raw Rows

Jerry W. Walker

Is there a better way to obtain just a single column from the database that is just the values and not the key/value pair?

Yes, use a SQL solution. So long as you're thinking in terms of columns in a relational database and their values, you're thinking in the terms around which SQL, as a language, is wrapped. I'll paraphrase a message I posted on another thread regarding this same issue.

The question is framed from the perspective of "I have a table of rows with which I want to do something, now how do I get WO to help me do that thing to my table in the same way I've always done these things with SQL?"

Experienced WO developers would think instead of the object graph that we are manipulating in our object oriented Java application rather than the rows we have stored in our relational database. Rephrasing the question, "I have a set of objects 'O' with an attribute 'a'. How do I obtain the value of that attribute for each of those objects?"

The obvious answer to your original question is dump WebObjects. It's a poor technology for doing pure SQL manipulation. Just issue a SQL "select myColumn from myTable;" and live with the rest of SQL's shortcomings as an enterprise application development technology.

On the other hand, if you want to use WebObjects, you'll find it much easier to think in the terms in which it frames the questions, namely classes, objects, object graphs and object oriented concepts. In these terms, the database is more appropriately thought of as an "object store", or place to hold my objects rather than a relational database. In these terms, you'll find it much easier to simply fetch the objects from the database and extract their values as you would the values of any other objects.

Perhaps those objects have already been cached and we don't need to fetch raw rows. That is, if they've already been fetched earlier for some other operation, their snapshots are already cached in memory and we would have been doing a redundant fetch rather than economizing.

Failing those possibilities, I would still write the fetch without referencing raw rows and get it to work. Then, if the result is clearly taking too long or requiring too much memory, I would profile where the delays are occurring or the memory is being consumed and, if in database fetches, then would start to consider raw rows. The point being that referencing raw rows is a last resort rather than a way to move into the comfort zone of old habits with result sets.

So what are the down sides to just moving immediately to raw rows?

- a missed opportunitiy to get used to the WO way
- the short circuiting of several elegant features built into WO to help one deal with objects rather than rows
- potential extra database accesses to objects already cached in memory
- the higher probability of weak object oriented design built up from this raw row decision

If you take the natural WebObjects route and stop trying to get a single value out of a table, the advice I gave in my earlier response still holds in general. Except, instead of dealing with NSDictionaries, you will be dealing with Enterprise Objects, or EOs. The code sample I supplied would look more like this, in that case:

Presume that myFetchSpec exists, that the class name for the EO that you've persisted in that table of the database is MyObject, that the name of the element of that class whose value you are interested in is myAttribute, and that you've fetched an array named myObjects as follows:

```
NSArray myObjects = myEditingContext.objectsWithFetchSpecification(myFetchSpec);  // Fetches instances of
MyObject
NSDictionary anObject;  // Iteration variable
NSDictionary selectedObject;  // used to hold selected element from WOPopUpButton
```

Now, set up your WOPopUpButton with the following bindings:

```
list = myObjects;
item = anObject;
displayString = anObject.myAttribute
selection = selectedRow
```

You'll find this easier, in general, than working with the raw rows, because you can do the bindings visually and directly in WebObjects Builder rather than having to type them in as you would have to do with the NSDictionaries resulting from a raw rows fetch. The WOPopUpButton should display what you want.