

Re: DBMS and lisp, etc.

Source: <http://coding.derkeiler.com/Archive/Lisp/comp.lang.lisp/2004-05/1874.html>

From: Will Hartung (willh_at_msoft.com)

Date: 05/21/04

Date: Fri, 21 May 2004 14:08:09 -0700

"Chris C apel" <chris@ibanktech.net> wrote in message
news:69d0f981.0405201630.5d97a155@posting.google.com...

> *That's a great point. I'm imagining a sort of data access layer where*
> *I have around one function for every different time I access my data,*
> *tailored to that specific instance. Otherwise, it's too easy to write*
> *a bunch of data access functions that do small pieces and then try to*
> *combine multiple calls to these methods. That's, as you say, a *bad*
> *thing* when dealing with RDBMS's. But with (almost) one function for*
> *each data access point, I force myself to think about how I can get my*
> *data over the line in the most efficient way possible. Is this a good*
> *design principle for DAL's in general?*

First, ANY DAL is better than none. It's one thing to have SQL bundled in your code, but it's another to have it bundled in your logic. Specifically, having at least 1 level of abstraction between how you get your data, and where you use your data is, I think, important, as it gives you that much more of an opportunity to refactor things later compared to having the access directly in your logic.

i.e.

```
(let ((orders (sql-results (format nil "SELECT * FROM ORDERTABLE WHERE  
CUSTCODE=~A`" custcode))))  
  ....)
```

is worse than

```
(let ((orders (get-orders-for-cust custcode)))  
  ...)
```

Even if `get-orders-for-cust` is a simple wrapper.

In Days Of Yore, with ISAM style databases, this was the typical paradigm for processing.

```
use OrderCustomerIndex
```

```
Find Order 'CUST001'  
Fetch Order  
while(order.custcode = 'CUST001')  
  use OrderDetailOrderIndex  
  Find OrderDetail order.orderno  
  Fetch OrderDetail  
  while(orderdetail.orderno = order.orderno)  
    use ItemItemCodeIndex  
    Find Item orderdetail.itemcode  
    Fetch Item  
    ...  
    Fetch Next OrderDetail  
  end while  
  Fetch Next Order  
end while
```

So, if you had 10 orders with 5 lines items each, this code would "hit" the DB library 100 times (10 for each o