

# Re: Help! Difficulty understanding DB -> Object mapping

---

*Source:* <http://coding.derkeiler.com/Archive/General/comp.object/2005-05/msg00088.html>

---

- *From:* "Daniel Parker" <[danielaparker@spam?nothanks.windupbird.com](mailto:danielaparker@spam?nothanks.windupbird.com)>
  - *Date:* Tue, 10 May 2005 01:29:05 -0400
- 

"H. S. Lahman" <[h.lahman@xxxxxxxxxxxxx](mailto:h.lahman@xxxxxxxxxxxxx)> wrote in message  
[news:FWNfe.6048\\$EC6.2329@xxxxxxxxxxxxx](mailto:news:FWNfe.6048$EC6.2329@xxxxxxxxxxxxx)

- >  
> Optimizing things like SQL queries depends upon the specific RDB schemas  
> and the way the particular applications needs to access data. (The design  
> issues for this could occupy an entire college-level course.) So there is  
> no single magic answer. However, one can go a long way by realizing a few  
> simple things:  
>  
> (1) Fewer DB accesses are better than many.  
>  
> (2) Virtual memory is cheap.  
>  
> (3) DB access time will usually be the major bottleneck in the application  
> unless it is a scientific application or does dazzling graphics.  
>  
> (4) DBAs hate long transactions (multiple reads/writes while locking the  
> data) because they tie up resources.

Right. Keep transactions short, where possible.

- >  
> (5) Accesses using joins should be minimized.  
>  
> Basically what this means is that one should grab as much data as one is  
> /likely/ to need all at once even if sometimes it isn't all used. The  
> bottleneck is getting it into memory as datasets; extracting particular  
> information from the datasets in memory is the fast part.

The problem here is that data loaded into memory has to be managed, it has to be invalidated if data on the server has changed, and this is generally not trivial. There is a strong case to be made for getting the data from the DBMS each time, for simplicity. The performance is usually fine, DBMS's are fast, and they cache too. Also there is the issue of volumes of data, granted, memory is cheap, but with data the only numbers that count are zero, one, and as many as you like.

- > Prefer single complex queries to multiple simple queries.

Re: Help! Difficulty understanding DB -> Object mapping

Yes, but doesn't that contradict your advice to minimize joins? Most DBMS's have been able to give good performance with joins for some time, although in the past you had to have a pretty good knowledge of how the query was executed in order to achieve that performance. When I worked with Sybase many years ago, we fondly referred to the "query optimizer" as a "query pessimizer", and the idea was to work around its shortcomings. But I think most DBMS vendors today have pretty good support for joins.

> If possible, create specialized indices and store "compiled" joins for  
> queries that are commonly used.

Not necessarily. "Compiled" queries, say in stored procedures, are typically tokenized on the first invocation, and a query plan is computed based on the passed parameters for that invocation. If those parameter values are atypical, the query plan may be off. Besides, processors are so fast these days that the time to compile a query is miniscule compared to the time to retrieve the data.

> When mapping to the problem solution's needs when performance is a big  
> problem, look for ways to use write caching or anticipatory reads. Cache  
> requests rather than opening long transactions whenever possible,  
> especially if the data is from user keyboard entry.  
>

The problem with delayed updates is that the cached data may become stale. Suppose you read a record, cache it, somebody else reads a record, changes a field and saves it, then you make your save, and overwrite the other users change. A common solution to that is optimistic locking. Basically, when you read your cached data, you also read a last updated timestamp on the record, and if you try to update when that timestamp has changed, your update fails.

Regards,  
Daniel Parker

---

• *Follow-Ups:*

- ◆ **Re: Help! Difficulty understanding DB -> Object mapping**  
◇ From: H. S. Lahman

• *References:*

- ◆ **Help! Difficulty understanding DB -> Object mapping**  
◇ From: usenet . news . account
- ◆ **Re: Help! Difficulty understanding DB -> Object mapping**  
◇ From: H. S. Lahman
- ◆ **Re: Help! Difficulty understanding DB -> Object mapping**  
◇ From: usenet . news . account
- ◆ **Re: Help! Difficulty understanding DB -> Object mapping**

Re: Help! Difficulty understanding DB -> Object mapping

◇ *From:* H. S. Lahman

- Prev by Date: ***Re: Lahman, how ya doing?***
- Next by Date: ***Re: polymorphism and dynamically typed languages***
- Previous by thread: ***Re: Help! Difficulty understanding DB -> Object mapping***
- Next by thread: ***Re: Help! Difficulty understanding DB -> Object mapping***
- Index(es):
  - ◆ ***Date***
  - ◆ ***Thread***