

Re: RETURN_VALUES ??? done... What have I gained?

Source:

<http://coding.derkeiler.com/Archive/Delphi/borland.public.delphi.database.ado/2005-05/msg00091.html>

- *From:* "Del M" <Del.Murray@xxxxxxxxxxxxxxxx>
 - *Date:* Fri, 6 May 2005 14:04:45 -0400
-

Dr Codd is the IBM guy who theorized and developed everything we know about SQL and relational databases. His theories are the goal toward which all major DB vendors have been marching for years. I dont think it is a Pepsi vs Coke thing. I think its a disciplined well ordered approach at all times vs a "this works so it must be ok" approach which results in everything always haveing a diffent slant on it. But, remember, this is more art and abstract thinking than it is science.

Well normalized data is good. A tuned db is one that has the "best" sql statments, stored procedures, etc. and the best index useage based on the business problem. For instance., you will hardly find clustered indexes in MS Sql2000 databases but yet MS tells us over and over, use clustered indexes. MS Sql is just one big flat file. If you have to read the file in spot A to find the index, why make it read the data in spot B when all the data could be had just by reading A and being done with it. In the past that didn't work well at all because of the technical implementation but in SQL Server 2000 that was fixed. A friend and I make pretty good money changing queries and indexes on Databases (of course you have to know how to interpret execution plans and how to correctly interpret and run sql monitor) with out changing anycode and they think we are heros (which we aren't)... actually it's like walking on water, it's a piece of cake if you know where all the stones are.

I just have a strong personal preference that says if something is a return code, then it should be used for that, not to return data. If M\$ ever makes a change in their philosophy about return codes from stored procedures a whole bunch of people are gonna get caught with their pants down and I dont want to be one of them, I've seen it happen before ... consider the "registry".. That turned out to be such a bad idea that I understand that in ..Net stuff, M\$ has abandoned it. One more OS release and I'll bet the registry will not be in the OS and everone who jumped on that bandwagon, will have to change their software, whereas those who continued to use their own control file for their software (even if it was a .INI file) will still be in control and running.

Just dumb thoughts from an old programmer type to you young wipper-snappers who haven't been roasted in every IBM, GE, Honeywell, Burroughs and M\$ fire

Re: RETURN_VALUES ??? done... What have I gained?

... yet.

"Betsy" <betsy.a.tainer@xxxxxxxxxxxxxx> wrote in message

news:427b956b@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

> ok... we have a yeah and a neah!

>

> Who is Doctor Codd?

>

> I don't think that taking this approach is so complex that the person coming

> behind me would have any difficulty interpreting it. I'm all for the 'keep

> it simple stupid' rule and have an overwhelming desire, even need, to keep

>

> How would I be 'thrashing the db' ?

>

> Explain 'tuned' to me. If I have a 'normalized' – 'well formed'

> datastructure and my sql statements are 'clean/well structured', then I'm

> tuned... right? (with no angst, por favor!)

>

> I used the TADOCCommand component as Brian suggested... seems clean enough.

> Doesn't seem to really be less code... other then I will be asking numerous

> similar questions via stored proc and can set up a function to do the work

> with a proc name and return the value. I'm 'guessing' that if I need a

> value vs a dataset that somehow it would be more efficient to retrieve the

> value... but it's just a guess... I'm thinking that regardless of 1 row or

> 100 rows and 1 or numerous columns that there must be some amount of

> overhead required to accomplish that. (that was at the heart of the original

> question). Taking this approach DOES seem to have a performance gain,

> visually (immediate), over reading the field from the dataset... it wasn't

> really a measurable length of time to read the field... but now it seems

> even less so.

>

> After reading numerous posts I started replacing my TadoTable components

> with TadoDataset and did notice a considerable gain in performance and have

> since found that I prefer working with the dataset component for a lot of

> reasons. I was not aware of how TadoCommand played into that scenario...

> (good info, thank you).

>

> Is this just a pepsi vs coke thing? (matter of personal preference) or is

> there a legitimate reason to take one approach over another.

>

> Your input greatly appreciated!

>

> thanx

> b

Re: RETURN_VALUES ??? done... What have I gained?

Re: RETURN_VALUES ??? done... What have I gained?

>
> "Del M" <Del.Murray@xxxxxxxxxxxxxxxx> wrote in message
> [news:427ad8a9\\$1@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx](mailto:news:427ad8a9$1@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx)
> > What's the real question ?? is one more efficient than the other ?? If
the
> > database is not properly tuned then it doesn't matter which you use, all
> > efficiency might be lost by thrashing the db.
> > tAdoDataSet is really easy to use. You can set the commandtext to
standard
> > sql statement or point to a stored procedure. Eitherway, you "open" the
> > dataset even though you are pointing to a stored procedure. The stored
> > procedure can be as simple as "select sum(x) where .." or it can be a
hugh
> > join with unions and cross db joins etc, which could result in one row
or
> > one column or a gazillion rows. When you use a tADOSTored procedure, it
is
> > usually because the stored proc itself is doing something and you dont
> > need a result set, you just want to know if it was successful or not. On
> the
> > surface, if you want data, do a tadodataset , if you want the stored
> > procedure to do it's thing and tell you it is ok, then do a
> tadostoredproc.
> > But dont use a stored proc return code to return data data is not a
> > return code, it is data, and return codes are not data, they are return
> > codes. That way the person comming behind you will see a tAdodataset and
> > know that you are retrieving data (even if it is 1 row, 1 column) and
they
> > will be better able to understand what you are trying to do. Doctor Codd
> > would frown on you using a stored procedure return code to get the sum
of
> an
> > amount column. He would rather you use a stored procedure to execute a
> > precompiled sql statment which returns in a recordset, the answer that
you
> > need.
> >
> > "Betsy" <betsy.a.tainer@xxxxxxxxxxxxxxxx> wrote in message
> > news:427aa075@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
> > > General question...
> > >
> > > -using an ADOSTOREDPROC component... is there any reason why I should
> use
> > > something else, say ADOQuery ???? With adodataset I was getting an
error
> > > about a result set and didn't see anywhere where I had the opportunity
> to
> > > Execproc vs Open... (did I miss something there?)
> > >
> > > -the only advantage I see is that I'm not trying to return a dataset
> when

Re: RETURN_VALUES ??? done... What have I gained?

Re: RETURN_VALUES ??? done... What have I gained?

```
>>> all I need is a stinkin value, amount, string, what-have-you... there
is
>>> nothing especially clean and simple about it... so... (the question)
>>> (FINALLY!)... Did I really gain anything??? (other then some tiny
sense
> of
>>> accomplishment)
>>>
>>> That is a serious question by the way... is it somehow more effecient
to
>>> retrieve one value/result when that is all you need?
>>>
>>> ***for the inquiring mind***
>>> stored proc:
>>> CREATE PROCEDURE dbo.getPayments @meetingCode char(6), @guestID
> char(10),
>>> @total money OUTPUT AS
>>> /* used to retrieve the payments made by a member*/
>>> select @total = sum(amount)
>>> from payments
>>> where (meetingCode= @meetingCode) and (guestid = @guestid)
>>> RETURN
>>> GO
>>>
>>> call in a function:
>>> function Tdm.GetPayments: currency;
>>> begin
>>> result := 0.00; //default
>>> if adodsMeeting.Active then //a meeting is opened
>>> begin
>>>
>>> with ADOSP do
>>> begin
>>> close;
>>> Prepared := false;
>>> Parameters.Clear;
>>>
>>> ProcedureName := 'getPayments';
>>>
>>> Parameters.CreateParameter('meetingCode',ftString,pdInput,6,
>>> trim(adodsGuests.Fieldbyname('meetingCode').AsString));
>>> Parameters.CreateParameter('guestID',ftString,pdInput,10,
>>> trim(adodsGuests.Fieldbyname('guestid').AsString));
>>> Parameters.CreateParameter('total',ftCurrency,pdOutput,10,0);
>>>
>>> prepared := true;
>>> ExecProc;
>>>
>>> if varisnull(Parameters.ParamByName('total').Value) then
>>> result := 0.00
>>> else
```

Re: RETURN_VALUES ??? done... What have I gained?

Re: RETURN_VALUES ??? done... What have I gained?

```
>>> result := Parameters.ParamByName('total').Value;
>>> end;
>>> end;
>>> end;
>>>
>>>
>>> ps... thanx Ron noone@xxxxxxxxxxxxxxxx your notes helped
tremendously...
> I
>>> find microsoft help to be a little criptic.
>>>
>>>
>>
>>
>
>
```

• **References:**

- ◆ **[RETURN_VALUES ??? done... What have I gained?](#)**
 - ◇ From: Betsy
 - ◆ **[Re: RETURN_VALUES ??? done... What have I gained?](#)**
 - ◇ From: Del M
 - ◆ **[Re: RETURN_VALUES ??? done... What have I gained?](#)**
 - ◇ From: Betsy
-
- Prev by Date: **[DB null value returned by TDataSet's FieldValues](#)**
 - Next by Date: **[Re: DB null value returned by TDataSet's FieldValues](#)**
 - Previous by thread: **[Re: RETURN_VALUES ??? done... What have I gained?](#)**
 - Next by thread: **[Re: RETURN_VALUES ??? done... What have I gained?](#)**
 - Index(es):
 - ◆ **[Date](#)**
 - ◆ **[Thread](#)**