

## Re: Creating a view that uses a sp to retrieve data

**Source:**

<http://www.tech-archive.net/Archive/SQL-Server/microsoft.public.sqlserver.programming/2004-03/3562.html>

---

**From:** Uri Dimant ([urid\\_at\\_iscar.co.il](mailto:urid_at_iscar.co.il))

**Date:** 03/15/04

Date: Mon, 15 Mar 2004 09:26:54 +0200

oj

>Uri: yes, you want to qualify the database if the view is >created in a different

>db other than the stored procedure.

Something wrong is going here

Use Northwind

```
exec sp_serveroption 'server','data access','true'
```

```
go
```

```
create view _v
```

```
as
```

```
select *
```

```
from openquery(Server,'exec CustOrdersDetail 10248')x
```

```
go
```

Note: I have tried to create a view in the same database where i created stored procedure.

And I still should qualify the database name.

This works fine

Use Northwind

```
create view _v
```

```
as
```

```
select *
```

```
from openquery(Server,'exec northwind.dbo.CustOrdersDetail 10248')x
```

```
go
```

"oj" <[nospam\\_ojngo@home.com](mailto:nospam_ojngo@home.com)> wrote in message [news:ekPBavICEHA.3804@TK2MSFTNGP09.phx.gbl...](mailto:news:ekPBavICEHA.3804@TK2MSFTNGP09.phx.gbl...)

> Okay, let see if I can answer all posts at once...

>

> Ryan: sp\_who2 uses #tmp table and distributed query tries to determine the metadata using 'set fmtonly on' statement. Thus, you want to turn it off.

>

> Uri: yes, you want to qualify the database if the view is created in a different

> db other than the stored procedure.

microsoft.public.sqlserver.programming: Re: Creating a view that uses a sp to retrieve data

```
>
> Steve: the sp can return as many resultsets it wants. However, only the
first
> will be accepted and redirected by the view.
>
> e.g.
>
> use tempdb
> go
> create proc usp
> as
> set nocount on
> create table #tmp1(OrderID int, CustomerID sysname)
> create table #tmp2(EmployeeID int, Freight money)
>
> insert #tmp1
> select top 1 OrderID, CustomerID
> from Northwind..Orders
>
> insert #tmp2
> select top 1 EmployeeID, Freight
> from Northwind..Orders
>
> select * from #tmp1
> select * from #tmp2
> go
>
> use Northwind
> go
> create view _v
> as
> select *
> from openquery(myserver, 'set fmtonly off; exec tempdb..usp')x
> go
> select * from _v
> go
>
> drop view _v
> go
> exec tempdb..sp_executesql N'drop proc usp'
> go
>
>
> --
> -oj
> http://www.rac4sql.net
>
>
> "Steve Kass" <skass@drew.edu> wrote in message
> news:ujDh64kCEHA.2796@TK2MSFTNGP09.phx.gbl...
> > Ryan,
```

Re: Creating a view that uses a sp to retrieve data

```
> >
> > Try this:
> >
> > create view V as
> > select *
> > from OpenQuery(myserver, 'set fmtonly off; exec
msdb..sp_help_jobhistory')
> > go
> >
> > select * from V
> >
> > For sp's that return a single result set, I think this should work.
> >
> > SK
> >
> >
> > Ryan Simpson [MSFT] wrote:
> >
> > > Hi Oj,
> > >
> > > thanks for the heads up on this one – I hadn't thought of this, but
having
> > > looked into it a little more, it does have varied success depending on
the
> > > proc called – for example, this works brilliantly with sp_who, but not
> > > sp_who2 or sp_help_jobhistory.
> > >
> > > Many thanks
> > >
> > > Ryan
> > >
> > >
> > > "oj" <nospam_ojngo@home.com> wrote in message
> > > news:OJXHvYjCEHA.576@TK2MSFTNGP11.phx.gbl...
> > >
> > >
> > > > Ryan,
> > > >
> > > > A view can call a stored procedure via loop back linked server.
> > > >
> > > > e.g.
> > > > exec sp_serveroption 'myserver','data access','true'
> > > > go
> > > > create view _v
> > > > as
> > > > select *
> > > > from openquery(myserver,'exec sp_who')x
> > > > go
> > > >
> > > > select * from _v
> > > >
```

microsoft.public.sqlserver.programming: Re: Creating a view that uses a sp to retrieve data

>>>>  
>>>>--  
>>>>-oj  
>>>><http://www.rac4sql.net>  
>>>>  
>>>>  
>>>>"Ryan Simpson" <ryansimpson@online.microsoft.com> wrote in message  
>>>>news:umUPIQjCEHA.2600@TK2MSFTNGP09.phx.gbl...  
>>>>  
>>>>  
>>>>>Hi Glen,  
>>>>>  
>>>>>if I understand you currently, you want to return the execution of a  
>>>>>  
>>>>>  
>>>>stored  
>>>>  
>>>>  
>>>>>procedure via a view. As a view cannot call a stored procedure, the  
only  
>>>>>  
>>>>>  
>>>>way  
>>>>  
>>>>  
>>>>>to achieve this functionality is via a multistatement user defined  
>>>>>  
>>>>>  
>>>>function.  
>>>>  
>>>>  
>>>>>I've put together an example which returns jobhistory for you, but I  
>>>>>  
>>>>>  
>>>>would  
>>>>  
>>>>  
>>>>>recommend reworking this example (mainly around the data types of the  
>>>>>declared table variable), as I haven't tested this other to compile  
and  
>>>>>  
>>>>>  
>>>>run  
>>>>  
>>>>  
>>>>>it.  
>>>>>  
>>>>>Best regards  
>>>>>  
>>>>>Ryan  
>>>>>

Re: Creating a view that uses a sp to retrieve data

```
> > >>>The information in this email is provided 'as-is' with no warranties
> > >>>
> > >>>
> > >either
> > >
> > >>>inferred or implied.
> > >>>
> > >>>-- example of use:
> > >>>
> > >>>select * from
> > >>>
> > >>>
> > >>>
> >
>dbo.jobhistory(null,null,null,null,null,null,null,null,null,null,null,
n
> > >
> > >
> > >>>ull,null) -- unfortunately all paramaters are required
> > >>>
> > >>>-- use msdb
> > >>>-- go
> > >>>-- sp_helptext sp_help_jobhistory
> > >>>-- go
> > >>>
> > >>>create function dbo.jobhistory
> > >>>(
> > >>> @job_id UNIQUEIDENTIFIER = NULL,
> > >>> @job_name sysname = NULL,
> > >>> @step_id INT = NULL,
> > >>> @sql_message_id INT = NULL,
> > >>> @sql_severity INT = NULL,
> > >>> @start_run_date INT = NULL, -- YYYYMMDD
> > >>> @end_run_date INT = NULL, -- YYYYMMDD
> > >>> @start_run_time INT = NULL, -- HHMMSS
> > >>> @end_run_time INT = NULL, -- HHMMSS
> > >>> @minimum_run_duration INT = NULL, -- HHMMSS
> > >>> @run_status INT = NULL, --
SQLAGENT_EXEC_X
> > >>>
> > >>>
> > >code
> > >
> > >
> > >>> @minimum_retries INT = NULL,
> > >>> @oldest_first INT = 0, -- Or 1
> > >>> @server NVARCHAR(30) = NULL
> > >>>-- @mode VARCHAR(7) = 'SUMMARY' -- Or 'FULL'
or
> > >>>'SEM' -- removing as this will change the output select list
```

```
>>>>)
>>>>returns @jobhistory table (
>>>>[instance_id] int, -- This is included just for ordering purposes
>>>>[job_id] uniqueidentifier,
>>>>[name] sysname,
>>>>step_id int,
>>>>step_name int,
>>>>sql_message_id int,
>>>>sql_severity int,
>>>>message int,
>>>>run_status int,
>>>>run_date int,
>>>>run_time int,
>>>>run_duration int,
>>>>operator_emailed int,
>>>>operator_netsent int,
>>>>operator_paged int,
>>>>retries_attempted int,
>>>>server nvarchar(30)
>>>>)
>>>>AS
>>>>BEGIN
>>>> DECLARE @retval INT
>>>> DECLARE @order_by INT -- Must be INT since it can be -1
>>>>
>>>>-- SET NOCOUNT ON
>>>>
>>>> -- Remove any leading/trailing spaces from parameters
>>>> SELECT @server = LTRIM(RTRIM(@server))
>>>> --SELECT @mode = LTRIM(RTRIM(@mode))
>>>>
>>>> -- Turn [nullable] empty string parameters into NULLs
>>>> IF (@server = N'') SELECT @server = NULL
>>>>
>>>>-- we cannot use exec from within a user defined function
>>>> -- Check job id/name (if supplied)
>>>>-- IF ((@job_id IS NOT NULL) OR (@job_name IS NOT NULL))
>>>>-- BEGIN
>>>>-- EXECUTE @retval = sp_verify_job_identifiers '@job_name',
>>>>-- '@job_id',
>>>>-- @job_name OUTPUT,
>>>>-- @job_id OUTPUT
>>>>-- IF (@retval <> 0)
>>>>-- RETURN(1) -- Failure
>>>>-- END
>>>>
>>>> -- Check @start_run_date
>>>>-- IF (@start_run_date IS NOT NULL)
>>>>-- BEGIN
>>>>-- EXECUTE @retval = sp_verify_job_date @start_run_date,
>>>>'@start_run_date'
```

```
> > > > -- IF (@retval <> 0)
> > > > -- RETURN(1) -- Failure
> > > > -- END
> > > >
> > > > -- -- Check @end_run_date
> > > > -- IF (@end_run_date IS NOT NULL)
> > > > -- BEGIN
> > > > -- EXECUTE @retval = sp_verify_job_date @end_run_date,
> > > >
> > > >
> > > > '@end_run_date'
> > > >
> > > >
> > > > -- IF (@retval <> 0)
> > > > -- RETURN(1) -- Failure
> > > > -- END
> > > >
> > > > -- Check @start_run_time
> > > > -- EXECUTE @retval = sp_verify_job_time @start_run_time,
> > > >
> > > >
> > > > '@start_run_time'
> > > >
> > > >
> > > > -- IF (@retval <> 0)
> > > > -- RETURN(1) -- Failure
> > > >
> > > > -- -- Check @end_run_time
> > > > -- EXECUTE @retval = sp_verify_job_time @end_run_time,
'@end_run_time'
> > > > -- IF (@retval <> 0)
> > > > -- RETURN(1) -- Failure
> > > >
> > > > -- -- Check @run_status
> > > > -- IF ((@run_status < 0) OR (@run_status > 5))
> > > > -- BEGIN
> > > > -- RAISERROR(13266, -1, -1, '@run_status', '0..5')
> > > > -- RETURN(1) -- Failure
> > > > -- END
> > > >
> > > > -- -- Check mode
> > > > -- SELECT @mode = UPPER(@mode)
> > > > -- IF (@mode NOT IN ('SUMMARY', 'FULL', 'SEM'))
> > > > -- BEGIN
> > > > -- RAISERROR(14266, -1, -1, '@mode', 'SUMMARY, FULL, SEM')
> > > > -- RETURN(1) -- Failure
> > > > -- END
> > > >
> > > > SELECT @order_by = -1
> > > > IF (@oldest_first = 1)
> > > > SELECT @order_by = 1
```

```
> > >>>
> > >>> -- Return history information filtered by the supplied parameters.
> > >>> -- NOTE: SQLDMO relies on the 'FULL' format; ** DO NOT CHANGE IT **
> > >>>-- IF (@mode = 'FULL')
> > >>>-- BEGIN
> > >>> INSERT INTO @jobhistory
> > >>> SELECT sjh.instance_id, -- This is included just for ordering
> > >>>
> > >>>
> > >>>
> > >>>purposes
> > >
> > >
> > >>> sj.job_id,
> > >>> job_name = sj.name,
> > >>> sjh.step_id,
> > >>> sjh.step_name,
> > >>> sjh.sql_message_id,
> > >>> sjh.sql_severity,
> > >>> sjh.message,
> > >>> sjh.run_status,
> > >>> sjh.run_date,
> > >>> sjh.run_time,
> > >>> sjh.run_duration,
> > >>> operator_emailed = so1.name,
> > >>> operator_netsent = so2.name,
> > >>> operator_paged = so3.name,
> > >>> sjh.retries_attempted,
> > >>> sjh.server
> > >>> FROM msdb.dbo.sysjobhistory sjh
> > >>> LEFT OUTER JOIN msdb.dbo.sysoperators so1 ON
> > >>>(sjh.operator_id_emailed = so1.id)
> > >>> LEFT OUTER JOIN msdb.dbo.sysoperators so2 ON
> > >>>(sjh.operator_id_netsent = so2.id)
> > >>> LEFT OUTER JOIN msdb.dbo.sysoperators so3 ON
> > >>>(sjh.operator_id_paged = so3.id),
> > >>> msdb.dbo.sysjobs_view sj
> > >>> WHERE (sj.job_id = sjh.job_id)
> > >>> AND ((@job_id IS NULL) OR (@job_id = sjh.job_id))
> > >>> AND ((@step_id IS NULL) OR (@step_id =
sjh.step_id))
> > >>> AND ((@sql_message_id IS NULL) OR (@sql_message_id =
> > >>>sjh.sql_message_id))
> > >>> AND ((@sql_severity IS NULL) OR (@sql_severity =
> > >>>sjh.sql_severity))
> > >>> AND ((@start_run_date IS NULL) OR (sjh.run_date >=
> > >>>@start_run_date))
> > >>> AND ((@end_run_date IS NULL) OR (sjh.run_date <=
> > >>>@end_run_date))
> > >>> AND ((@start_run_time IS NULL) OR (sjh.run_time >=
> > >>>@start_run_time))
> > >>> AND ((@end_run_time IS NULL) OR (sjh.run_time <=
```

```
> > >>> @end_run_time))
> > >>> AND ((@minimum_run_duration IS NULL) OR (sjh.run_duration >=
> > >>> @minimum_run_duration))
> > >>> AND ((@run_status IS NULL) OR (@run_status =
> > >>> sjh.run_status))
> > >>> AND ((@minimum_retries IS NULL) OR (sjh.retries_attempted
> =
> > >>> @minimum_retries))
> > >>> AND ((@server IS NULL) OR (sjh.server = @server))
> > >>> ORDER BY (sjh.instance_id * @order_by)
> > >>> -- END
> > >>> -- ELSE
> > >>> -- IF (@mode = 'SUMMARY')
> > >>> -- BEGIN
> > >>> -- -- Summary format: same WHERE clause just a different SELECT
list
> > >>> -- SELECT sj.job_id,
> > >>> -- job_name = sj.name,
> > >>> -- sjh.run_status,
> > >>> -- sjh.run_date,
> > >>> -- sjh.run_time,
> > >>> -- sjh.run_duration,
> > >>> -- operator_emailed = substring(so1.name, 1, 20),
> > >>> -- operator_netsent = substring(so2.name, 1, 20),
> > >>> -- operator_paged = substring(so3.name, 1, 20),
> > >>> -- sjh.retries_attempted,
> > >>> -- sjh.server
> > >>> -- FROM msdb.dbo.sysjobhistory sjh
> > >>> -- LEFT OUTER JOIN msdb.dbo.sysoperators so1 ON
> > >>> (sjh.operator_id_emailed = so1.id)
> > >>> -- LEFT OUTER JOIN msdb.dbo.sysoperators so2 ON
> > >>> (sjh.operator_id_netsent = so2.id)
> > >>> -- LEFT OUTER JOIN msdb.dbo.sysoperators so3 ON
> > >>> (sjh.operator_id_paged = so3.id),
> > >>> -- msdb.dbo.sysjobs_view sj
> > >>> -- WHERE (sj.job_id = sjh.job_id)
> > >>> -- AND ((@job_id IS NULL) OR (@job_id =
sjh.job_id))
> > >>> -- AND ((@step_id IS NULL) OR (@step_id =
> > >>>
> > >>>
> > >>> sjh.step_id))
> > >
> > >
> > >>> -- AND ((@sql_message_id IS NULL) OR (@sql_message_id =
> > >>> sjh.sql_message_id))
> > >>> -- AND ((@sql_severity IS NULL) OR (@sql_severity =
> > >>> sjh.sql_severity))
> > >>> -- AND ((@start_run_date IS NULL) OR (sjh.run_date >=
> > >>> @start_run_date))
> > >>> -- AND ((@end_run_date IS NULL) OR (sjh.run_date <=
```

```
> > >>> @end_run_date))
> > >>>-- AND ((@start_run_time IS NULL) OR (sjh.run_time >=
> > >>> @start_run_time))
> > >>>-- AND ((@end_run_time IS NULL) OR (sjh.run_time <=
> > >>> @end_run_time))
> > >>>-- AND ((@minimum_run_duration IS NULL) OR (sjh.run_duration >=
> > >>> @minimum_run_duration))
> > >>>-- AND ((@run_status IS NULL) OR (@run_status =
> > >>> sjh.run_status))
> > >>>-- AND ((@minimum_retries IS NULL) OR
(sjh.retries_attempted
> > >>>
> > >>>
> > >>=
> > >>
> > >>
> > >>> @minimum_retries))
> > >>>-- AND ((@server IS NULL) OR (sjh.server =
@server))
> > >>>-- ORDER BY (sjh.instance_id * @order_by)
> > >>>-- END
> > >>>-- ELSE
> > >>>-- IF (@mode = 'SEM')
> > >>>-- BEGIN
> > >>>-- -- SQL Enterprise Manager format
> > >>>-- SELECT sjh.step_id,
> > >>>-- sjh.step_name,
> > >>>-- sjh.message,
> > >>>-- sjh.run_status,
> > >>>-- sjh.run_date,
> > >>>-- sjh.run_time,
> > >>>-- sjh.run_duration,
> > >>>-- operator_emailed = so1.name,
> > >>>-- operator_netsent = so2.name,
> > >>>-- operator_paged = so3.name
> > >>>-- FROM msdb.dbo.sysjobhistory sjh
> > >>>-- LEFT OUTER JOIN msdb.dbo.sysoperators so1 ON
> > >>>(sjh.operator_id_emailed = so1.id)
> > >>>-- LEFT OUTER JOIN msdb.dbo.sysoperators so2 ON
> > >>>(sjh.operator_id_netsent = so2.id)
> > >>>-- LEFT OUTER JOIN msdb.dbo.sysoperators so3 ON
> > >>>(sjh.operator_id_paged = so3.id),
> > >>>-- msdb.dbo.sysjobs_view sj
> > >>>-- WHERE (sj.job_id = sjh.job_id)
> > >>>-- AND (@job_id = sjh.job_id)
> > >>>-- ORDER BY (sjh.instance_id * @order_by)
> > >>>-- END
> > >>>--
> > >>>RETURN
> > >>>END
> > >>>
```

