**Ad Hoc Queries Disabled**

[Jason Brimhall](https://www.sqlservercentral.com/author/SQLRNNR), 2020-01-02 (first published: 2019-12-20)

FROM: <https://www.sqlservercentral.com/blogs/ad-hoc-queries-disabled>

[](http://jasonbrimhall.info/wp-content/uploads/2019/12/adhoc.png)

DBAs and [ad hoc](https://www.merriam-webster.com/dictionary/ad%20hoc) queries sometimes go together like oil and water. While we may prefer to avoid the ad hoc queries, sometimes it is the best method to achieve a task.

What exactly is an ad hoc query though? An ad hoc query is a query that should serve a single use purpose (not always the case), is routinely unplanned and quite possibly was untested (again, not always the case). Sometimes, however, the definitions of ad hoc don’t fully apply to queries in SQL Server. One such case is through the use of dynamic type queries and queries issued to linked servers via openrowset.

In my experience, queries used against a linked server and using the openrowset functionality is typically more of a prepared statement. However, it is treated as an ad hoc query. As such, you may encounter an error you may not have been expecting.

Msg 15281, Level 16, State 1, Line 161

SQL Server blocked access to STATEMENT ‘OpenRowset/OpenDatasource’ of component ‘Ad Hoc Distributed Queries’ because this component is turned off as part of the security configuration for this server. A system administrator can enable the use of ‘Ad Hoc Distributed Queries’ by using sp\_configure. For more information about enabling ‘Ad Hoc Distributed Queries’, search for ‘Ad Hoc Distributed Queries’ in SQL Server Books Online.

This article will help show how to resolve for this error so your queries may go on as intended.

Ad Hoc

Let’s first take a look at a pretty typical type of query I use routinely to check various things on my lab servers.

SELECT a.\*

FROM OPENROWSET('SQLNCLI', 'server=.DIXNEUFLATIN1;Trusted\_Connection=yes;',

'DECLARE @ServerMajorVersion DECIMAL(4, 2)

SELECT @ServerMajorVersion = CONVERT(DECIMAL(4, 2), PARSENAME(dt.fqn, 4) + ''.''

+ PARSENAME(dt.fqn, 3))

FROM ( SELECT CONVERT(VARCHAR(20), SERVERPROPERTY(''ProductVersion''))

) dt ( fqn );

select Case when @ServerMajorVersion = 14.00 then ''SQL Server 2017''

when @ServerMajorVersion = 13.00 then ''SQL Server 2016''

when @ServerMajorVersion = 12.00 then ''SQL Server 2014''

when @ServerMajorVersion = 11.00 then ''SQL Server 2012''

when @ServerMajorVersion = 15.00 then ''SQL Server 2019''

end as SQLVer

, @ServerMajorVersion as ServerMajorVersion

, xl.name, xl.description

from master.sys.dm\_xe\_objects xl

where xl.object\_type = ''target''') AS a

This is a query that I use (or something like it) to retrieve various Extended Events data from different lab servers. Notice, I use the openrowset method in order to query my linked server. If I do not have the server configured for [ad hoc distributed queries](https://docs.microsoft.com/en-us/sql/database-engine/configure-windows/ad-hoc-distributed-queries-server-configuration-option?view=sql-server-ver15) then I will receive the error previously noted. The solution for that error is to enable the setting. We can do that via the following query.

sp\_configure 'show advanced options', 1;

RECONFIGURE WITH OVERRIDE;

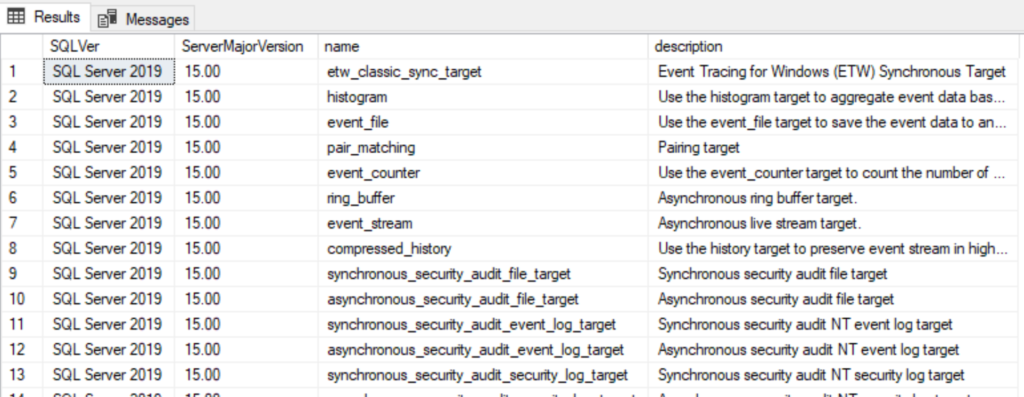
GO

sp\_configure 'Ad Hoc Distributed Queries', 1;

RECONFIGURE WITH OVERRIDE;

GO

Once enabled, then the openrowset queries will work across linked servers. This needs to be enabled on the server that is local in order to send the ad hoc query across to the remote server. With the setting now enabled, running the query that threw the error (from above) now returns the follow results.

[](http://jasonbrimhall.info/wp-content/uploads/2019/12/adhoc_linkedresults.png)



The Wrap

This article took a look at an error that may occur depending on your use of linked servers and the use of openrowset. Documentation states that this is something that should be done infrequently and alludes to the issue being resolved through the use of linked servers. In my case, this crops up when using openrowset to query my linked servers. I find that using openrowset to query the linked server is far more reliable, efficient, and better performing. YMMV.

This has been another post in the back to basics series. Other topics in the series include (but are not limited to): [Backups](http://bit.ly/2heI2eG), [backup history](https://jasonbrimhall.info/2016/12/20/database-backup-history-back-to-basics/) and [user logins](http://jasonbrimhall.info/2016/12/23/user-contains-invalid-characters-back-to-basics/).

https://jasonbrimhall.info/?feed-stats-post-id=6126

Related Posts:

* [Linked Servers and Stats](https://jasonbrimhall.info/2017/10/17/linked-servers-and-stats/) October 17, 2017
* [How Long is That Event Taking](https://jasonbrimhall.info/2018/11/20/how-long-is-that-event-taking/) November 20, 2018
* [Maintenance Plan Owner - Back to Basics](https://jasonbrimhall.info/2018/12/13/maintenance-plan-owner-back-to-basics/) December 13, 2018
* [Implicit Conversion Fail](https://jasonbrimhall.info/2019/05/31/implicit-conversion-fail/) May 31, 2019
* [Common Tempdb Trace Flags - Back to Basics](https://jasonbrimhall.info/2018/01/05/common-tempdb-trace-flags-back-to-basics/) January 5, 2018