|  |
| --- |
| Erik Darling says: [Availability Group Direct Seeding: Extended Events and DMVs](http://brentozar.us2.list-manage1.com/track/click?u=9082566fb63d87be35c0662bc&id=e2873755e8&e=b137e91a2b)As of this writing, this is all undocumented I’m super interested in this feature, so that won’t deter me too much. There have been a number of [questions](http://brentozar.us2.list-manage.com/track/click?u=9082566fb63d87be35c0662bc&id=46aea032db&e=b137e91a2b) since Availability Groups became a thing about how to automate adding new databases. All of the solutions were kind of awkward scripts to backup, restore, join, blah blah blah. This feature aims to make that a thing of the past.  There’s also not a ton of information about how this works, the option hasn’t made it to the GUI, and there may still be some kinks to work out. Some interesting information I’ve come across has been limited to this [SAP on SQL blog post](http://brentozar.us2.list-manage1.com/track/click?u=9082566fb63d87be35c0662bc&id=b2e29e2c09&e=b137e91a2b), and a [Connect item](http://brentozar.us2.list-manage.com/track/click?u=9082566fb63d87be35c0662bc&id=034633dade&e=b137e91a2b) by the Smartest Guy At SanDisk, [Jimmy May](http://brentozar.us2.list-manage1.com/track/click?u=9082566fb63d87be35c0662bc&id=2ed9d5390f&e=b137e91a2b).  The SAP on SQL Server blog post says that this feature uses the same method as Azure databases to create replicas; opening a direct data link, and Jimmy’s Connect item points to it being a backup and restore behind the scenes. The Extended Events sessions point to it being a backup and restore, so let’s look at those first. Bring out your XML! We’re going to need two sessions, because there are two sets of collectors, and it doesn’t make sense to lump them into one XE session. If you look in the GUI, there’s a new category called dbseed, and of course, everything is in the super cool kid debug channel.  [New Extended Event Smell](http://brentozar.us2.list-manage.com/track/click?u=9082566fb63d87be35c0662bc&id=b128120663&e=b137e91a2b)  New Extended Event Smell  Quick setup scripts are below.  CREATE EVENT SESSION [DirectSeed] ON SERVER  ADD EVENT sqlserver.hadr\_ar\_controller\_debug(  ACTION(sqlserver.database\_id,sqlserver.sql\_text,sqlserver.tsql\_stack)),  ADD EVENT sqlserver.hadr\_automatic\_seeding\_failure(  ACTION(sqlserver.database\_id,sqlserver.sql\_text,sqlserver.tsql\_stack)),  ADD EVENT sqlserver.hadr\_automatic\_seeding\_start(  ACTION(sqlserver.database\_id,sqlserver.sql\_text,sqlserver.tsql\_stack)),  ADD EVENT sqlserver.hadr\_automatic\_seeding\_state\_transition(  ACTION(sqlserver.database\_id,sqlserver.sql\_text,sqlserver.tsql\_stack)),  ADD EVENT sqlserver.hadr\_automatic\_seeding\_success(  ACTION(sqlserver.database\_id,sqlserver.sql\_text,sqlserver.tsql\_stack)),  ADD EVENT sqlserver.hadr\_automatic\_seeding\_timeout(  ACTION(sqlserver.database\_id,sqlserver.sql\_text,sqlserver.tsql\_stack))  ADD TARGET package0.event\_file(SET filename=N'C:\XE\DirectSeed.xel',max\_rollover\_files=(10))  GO  CREATE EVENT SESSION [PhysicalSeed] ON SERVER  ADD EVENT sqlserver.hadr\_physical\_seeding\_backup\_state\_change(  ACTION(sqlserver.database\_id,sqlserver.sql\_text,sqlserver.tsql\_stack)),  ADD EVENT sqlserver.hadr\_physical\_seeding\_failure(  ACTION(sqlserver.database\_id,sqlserver.sql\_text,sqlserver.tsql\_stack)),  ADD EVENT sqlserver.hadr\_physical\_seeding\_forwarder\_state\_change(  ACTION(sqlserver.database\_id,sqlserver.sql\_text,sqlserver.tsql\_stack)),  ADD EVENT sqlserver.hadr\_physical\_seeding\_forwarder\_target\_state\_change(  ACTION(sqlserver.database\_id,sqlserver.sql\_text,sqlserver.tsql\_stack)),  ADD EVENT sqlserver.hadr\_physical\_seeding\_progress(  ACTION(sqlserver.database\_id,sqlserver.sql\_text,sqlserver.tsql\_stack)),  ADD EVENT sqlserver.hadr\_physical\_seeding\_restore\_state\_change(  ACTION(sqlserver.database\_id,sqlserver.sql\_text,sqlserver.tsql\_stack)),  ADD EVENT sqlserver.hadr\_physical\_seeding\_schedule\_long\_task\_failure(  ACTION(sqlserver.database\_id,sqlserver.sql\_text,sqlserver.tsql\_stack)),  ADD EVENT sqlserver.hadr\_physical\_seeding\_submit\_callback(  ACTION(sqlserver.database\_id,sqlserver.sql\_text,sqlserver.tsql\_stack))  ADD TARGET package0.event\_file(SET filename=N'C:\XE\PhysicalSeed',max\_rollover\_files=(10))  GO  ALTER EVENT SESSION [DirectSeed] ON SERVER STATE = START  ALTER EVENT SESSION [PhysicalSeed] ON SERVER STATE = START Since this is so new I haven’t quite narrowed down which are important and which yield pertinent information yet. Right now I’m grabbing everything. In a prelude to DBA days, I’m adding the StackOverflow database. With some session data flowing in, let’s figure out what we’re looking at. XML shredding fun is up next.  To get information out of the Automatic Seeding session…  IF OBJECT\_ID('tempdb..#DirectSeed') IS NOT NULL  DROP TABLE [#DirectSeed];  CREATE TABLE [#DirectSeed]  (  [ID] INT IDENTITY(1, 1)  NOT NULL ,  [EventXML] XML ,  CONSTRAINT [PK\_DirectSeed] PRIMARY KEY CLUSTERED ( [ID] )  );  INSERT [#DirectSeed]  ( [EventXML] )  SELECT CONVERT(XML, [event\_data]) AS [EventXML]  FROM [sys].[fn\_xe\_file\_target\_read\_file]('C:\XE\DirectSeed\*.xel', NULL, NULL, NULL)  CREATE PRIMARY XML INDEX [DirectSeedXML] ON [#DirectSeed]([EventXML]);  CREATE XML INDEX [DirectSeedXMLPath] ON [#DirectSeed]([EventXML])  USING XML INDEX [DirectSeedXML] FOR VALUE;  SELECT  [ds].[EventXML].[value]('(/event/@name)[1]', 'VARCHAR(MAX)') AS [event\_name],  [ds].[EventXML].[value]('(/event/@timestamp)[1]', 'DATETIME2(7)') AS [event\_time],  [ds].[EventXML].[value]('(/event/data[@name="debug\_message"]/value)[1]', 'VARCHAR(8000)') AS [debug\_message],  /\*hadr\_automatic\_seeding\_state\_transition\*/  [ds].[EventXML].[value]('(/event/data[@name="previous\_state"]/value)[1]', 'VARCHAR(8000)') AS [previous\_state],  [ds].[EventXML].[value]('(/event/data[@name="current\_state"]/value)[1]', 'VARCHAR(8000)') AS [current\_state],  /\*hadr\_automatic\_seeding\_start\*/  [ds].[EventXML].[value]('(/event/data[@name="operation\_attempt\_number"]/value)[1]', 'BIGINT') as [operation\_attempt\_number],  [ds].[EventXML].[value]('(/event/data[@name="ag\_id"]/value)[1]', 'VARCHAR(8000)') AS [ag\_id],  [ds].[EventXML].[value]('(/event/data[@name="ag\_db\_id"]/value)[1]', 'VARCHAR(8000)') AS [ag\_id],  [ds].[EventXML].[value]('(/event/data[@name="ag\_remote\_replica\_id"]/value)[1]', 'VARCHAR(8000)') AS [ag\_remote\_replica\_id],  /\*hadr\_automatic\_seeding\_success\*/  [ds].[EventXML].[value]('(/event/data[@name="required\_seeding"]/value)[1]', 'VARCHAR(8000)') AS [required\_seeding],  /\*hadr\_automatic\_seeding\_timeout\*/  [ds].[EventXML].[value]('(/event/data[@name="timeout\_ms"]/value)[1]', 'BIGINT') as [timeout\_ms],  /\*hadr\_automatic\_seeding\_failure\*/  [ds].[EventXML].[value]('(/event/data[@name="failure\_state"]/value)[1]', 'BIGINT') as [failure\_state],  [ds].[EventXML].[value]('(/event/data[@name="failure\_state\_desc"]/value)[1]', 'VARCHAR(8000)') AS [failure\_state\_desc]  FROM [#DirectSeed] AS [ds]  ORDER BY [ds].[EventXML].[value]('(/event/@timestamp)[1]', 'DATETIME2(7)') DESC  Every time I have to work with XML I want to go to culinary school and become a tattooed cliche on Chopped. Upside? Brent might hire me to be his personal chef. Downside? I’d only be cooking for Ernie. Here’s a sample of what we get back I’ve moved the ‘less interesting’ columns off to the right.  [Frenemy.](http://brentozar.us2.list-manage1.com/track/click?u=9082566fb63d87be35c0662bc&id=1cd236e4ed&e=b137e91a2b)  Frenemy.  These are my first clues that Jimmy is right about it being a backup and restore. One of the columns says “limit concurrent backups” and, we’re also sending file lists around. Particularly interesting is in the debug column from the hadr\_ar\_controller\_debug item. Here’s pasted text from it.  [HADR] [Secondary] operation on replicas [58BCC44A-12A6-449B-BF33-FAAF9D1A46DD]->[F5302334-B620-4FE2-83A2-399F55AA40EF], database [StackOverflow], remote endpoint [TCP://SQLVM01.darling.com:5022], source operation [55782AB4-5307-47A2-A0D9-3BB29F130F3C]: Transitioning from [LIMIT\_CONCURRENT\_BACKUPS] to [SEEDING].  [HADR] [Secondary] operation on replicas [58BCC44A-12A6-449B-BF33-FAAF9D1A46DD]->[F5302334-B620-4FE2-83A2-399F55AA40EF], database [StackOverflow], remote endpoint [TCP://SQLVM01.darling.com:5022], source operation [55782AB4-5307-47A2-A0D9-3BB29F130F3C]: Starting streaming restore, DB size [-461504512] bytes, [2] logical files.  [HADR] [Secondary] operation on replicas [58BCC44A-12A6-449B-BF33-FAAF9D1A46DD]->[F5302334-B620-4FE2-83A2-399F55AA40EF], database [StackOverflow], remote endpoint [TCP://SQLVM01.darling.com:5022], source operation [55782AB4-5307-47A2-A0D9-3BB29F130F3C]:  Database file #[0]: LogicalName: [StackOverflow] FileId: [1] FileTypeId: [0]  Database file #[1]: LogicalName: [StackOverflow\_log] FileId: [2] FileTypeId: [1]  [HADR] [Secondary] operation on replicas [58BCC44A-12A6-449B-BF33-FAAF9D1A46DD]->[F5302334-B620-4FE2-83A2-399F55AA40EF], database [StackOverflow], remote endpoint [TCP://SQLVM01.darling.com:5022], source operation [55782AB4-5307-47A2-A0D9-3BB29F130F3C]: RESTORE T-SQL String for VDI Client: [RESTORE DATABASE FROM VIRTUAL\_DEVICE='{AA4C5800-7192-4B77-863B-426246C0CC27}' WITH NORECOVERY, CHECKSUM, REPLACE, BUFFERCOUNT=16, MAXTRANSFERSIZE=2097152, MOVE 'StackOverflow' TO 'E:\SO\StackOverflow.mdf', MOVE 'StackOverflow\_log' TO 'E:\SO\StackOverflow\_log.ldf'] Hey look, a restore While I didn’t see an explicit backup command to match, we did pick up data like this:  [HADR] [Primary] operation on replicas [58BCC44A-12A6-449B-BF33-FAAF9D1A46DD]->[571F3967-FB40-4187-BF1E-36A88458C13A], database [StackOverflow], remote endpoint [TCP://SQLVM03.darling.com:5022], source operation [AFB86269-8284-4DB1-95F9-0128EB710825]: Starting streaming backup, DB size [-461504512] bytes, [2] logical files.  A streaming backup! How cute. There’s more evidence in the Physical Seeding session, so let’s look there. Prerequisite XML horrors to follow.  IF OBJECT\_ID('tempdb..#PhysicalSeed') IS NOT NULL  DROP TABLE [#PhysicalSeed];  CREATE TABLE [#PhysicalSeed]  (  [ID] INT IDENTITY(1, 1)  NOT NULL ,  [EventXML] XML ,  CONSTRAINT [PK\_PhysicalSeed] PRIMARY KEY CLUSTERED ( [ID] )  );  INSERT [#PhysicalSeed]  ( [EventXML] )  SELECT CONVERT(XML, [event\_data]) AS [EventXML]  FROM [sys].[fn\_xe\_file\_target\_read\_file]('C:\XE\PhysicalSeed\*.xel', NULL, NULL, NULL)  CREATE PRIMARY XML INDEX [PhysicalSeedXML] ON [#PhysicalSeed]([EventXML]);  CREATE XML INDEX [PhysicalSeedXMLPath] ON [#PhysicalSeed]([EventXML])  USING XML INDEX [PhysicalSeedXML] FOR VALUE;  SELECT  [ds].[EventXML].[value]('(/event/@name)[1]', 'VARCHAR(MAX)') AS [event\_name],  [ds].[EventXML].[value]('(/event/@timestamp)[1]', 'DATETIME2(7)') AS [event\_time],  [ds].[EventXML].[value]('(/event/data[@name="old\_state"]/text)[1]', 'VARCHAR(8000)') as [old\_state],  [ds].[EventXML].[value]('(/event/data[@name="new\_state"]/text)[1]', 'VARCHAR(8000)') as [new\_state],  [ds].[EventXML].[value]('(/event/data[@name="seeding\_start\_time"]/value)[1]', 'DATETIME2(7)') as [seeding\_start\_time],  [ds].[EventXML].[value]('(/event/data[@name="seeding\_end\_time"]/value)[1]', 'DATETIME2(7)') as [seeding\_end\_time],  [ds].[EventXML].[value]('(/event/data[@name="estimated\_completion\_time"]/value)[1]', 'DATETIME2(7)') as [estimated\_completion\_time],  [ds].[EventXML].[value]('(/event/data[@name="transferred\_size\_bytes"]/value)[1]', 'BIGINT') / (1024. \* 1024.) as [transferred\_size\_mb],  [ds].[EventXML].[value]('(/event/data[@name="transfer\_rate\_bytes\_per\_second"]/value)[1]', 'BIGINT') / (1024. \* 1024.) as [transfer\_rate\_mb\_per\_second],  [ds].[EventXML].[value]('(/event/data[@name="database\_size\_bytes"]/value)[1]', 'BIGINT') / (1024. \* 1024.) as [database\_size\_mb],  [ds].[EventXML].[value]('(/event/data[@name="total\_disk\_io\_wait\_time\_ms"]/value)[1]', 'BIGINT') as [total\_disk\_io\_wait\_time\_ms],  [ds].[EventXML].[value]('(/event/data[@name="total\_network\_wait\_time\_ms"]/value)[1]', 'BIGINT') as [total\_network\_wait\_time\_ms],  [ds].[EventXML].[value]('(/event/data[@name="is\_compression\_enabled"]/value)[1]', 'VARCHAR(8000)') as [is\_compression\_enabled],  [ds].[EventXML].[value]('(/event/data[@name="failure\_code"]/value)[1]', 'BIGINT') as [failure\_code]  FROM [#PhysicalSeed] AS [ds]  ORDER BY [ds].[EventXML].[value]('(/event/@timestamp)[1]', 'DATETIME2(7)') DESC  And a sampling of data…  [What an odd estimated completion date.](http://brentozar.us2.list-manage.com/track/click?u=9082566fb63d87be35c0662bc&id=d7897d515d&e=b137e91a2b)  What an odd estimated completion date.  The old state and new state columns also point to backup and restore operations. I assume the completion date points to 1600 BECAUSE THIS IS ABSOLUTE WITCHCRAFT.    [Ooh! Metrics!](http://brentozar.us2.list-manage.com/track/click?u=9082566fb63d87be35c0662bc&id=93f2f45b04&e=b137e91a2b)  Ooh! Metrics!  Ignore the smaller sizes at the bottom. I’ve clearly been doing this with a few different databases. The disk IO and network metrics are pretty awesome. Now I have to backtrack a little bit…  The SAP on SQL Server blog post talks about Trace Flag 9567 being used to enable compression. It says that it only has to be enabled on the Primary Replica to work, but even with it turned on on all three of my Replicas, the compression column says false. Perhaps, like parallel redo logs, it hasn’t been implemented yet. I tried both enabling it with DBCC TRACEON, and using it as a startup parameter. Which brings us to the next set of collectors… DMVs These are also undocumented, and that kind of sucks. There are two that ‘match’ the XE sessions we have.  [sys].[dm\_hadr\_physical\_seeding\_stats]  [sys].[dm\_hadr\_automatic\_seeding]  These can be joined around to other views to get back some alright information. I used these two queries. If you have anything better, feel free to let me know.  SELECT  ag.name as ag\_name,  adc.database\_name,  r.replica\_server\_name,  start\_time,  completion\_time,  current\_state,  failure\_state\_desc,  number\_of\_attempts,  failure\_condition\_level  FROM sys.availability\_groups ag  JOIN sys.availability\_replicas r ON ag.group\_id = r.group\_id  JOIN sys.availability\_databases\_cluster adc on ag.group\_id=adc.group\_id  JOIN sys.dm\_hadr\_automatic\_seeding AS dhas  ON dhas.ag\_id = ag.group\_id  LEFT JOIN sys.dm\_hadr\_physical\_seeding\_stats AS dhpss  ON adc.database\_name = dhpss.local\_database\_name  WHERE database\_name = 'StackOverflow'  ORDER BY completion\_time DESC  SELECT  database\_name,  transfer\_rate\_bytes\_per\_second,  transferred\_size\_bytes,  database\_size\_bytes,  start\_time\_utc,  end\_time\_utc,  estimate\_time\_complete\_utc,  total\_disk\_io\_wait\_time\_ms,  total\_network\_wait\_time\_ms,  is\_compression\_enabled  FROM sys.availability\_groups ag  JOIN sys.availability\_replicas r ON ag.group\_id = r.group\_id  JOIN sys.availability\_databases\_cluster adc on ag.group\_id=adc.group\_id  JOIN sys.dm\_hadr\_automatic\_seeding AS dhas  ON dhas.ag\_id = ag.group\_id  LEFT JOIN sys.dm\_hadr\_physical\_seeding\_stats AS dhpss  ON adc.database\_name = dhpss.local\_database\_name  WHERE database\_name = 'StackOverflow'  ORDER BY completion\_time DESC  But we get sort of different information back in a couple places. This is part of what makes me wonder how fully formed this feature baby is. The completion estimate is in this century, heck, even this YEAR. The compression column is now a 0. Just a heads up, when I DIDN’T have Trace Flag 9567 on, that column was NULL. Turning it on changed it to 0. Heh. So uh, glad that’s… there.  [I smell like tequila.](http://brentozar.us2.list-manage.com/track/click?u=9082566fb63d87be35c0662bc&id=c2ac241ec1&e=b137e91a2b)  I smell like tequila. Oh look, it’s the end I know I said it before, but I love this new feature. There’s apparently still some stuff to work out, but it’s very promising so far. I’ll post updates as I get more information, but this is about the limit of what I can get without some official documentation.  Thanks for reading!  *Wanna shape sp\_Blitz and the rest of our scripts?* [*Check out our new Github repository.*](http://brentozar.us2.list-manage1.com/track/click?u=9082566fb63d87be35c0662bc&id=36d08e0867&e=b137e91a2b) [Read the comments.](http://brentozar.us2.list-manage.com/track/click?u=9082566fb63d87be35c0662bc&id=a19f0c40e8&e=b137e91a2b) |