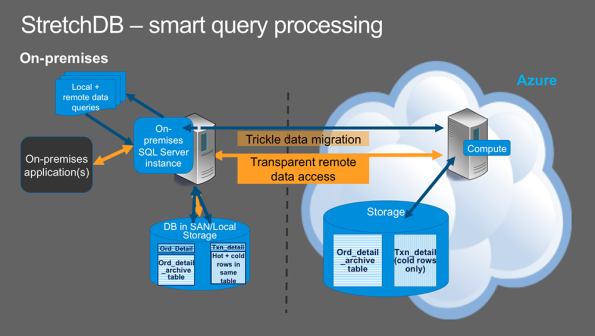
**Stretch your on-premise Database/Table to Azure SQL Database with StretchDB – SQL Server 2016**

A **Stretch Database** lets you **archive** your **historical data** transparently and securely in the **Azure SQL Database** (i.e. Microsoft PaaS Cloud).

The idea is to have an On-Prem database configured to allow extension of one or more of its tables to a SQL Azure Database.

This extension to Azure can archive transparently “cold” data without change access to information requirements when the table is queried, all this while enjoying the very attractive storage cost proposed by Microsoft Azure.

[](https://sqlwithmanoj.files.wordpress.com/2015/09/sql-server-2016-stretchdb-jpg1.png)

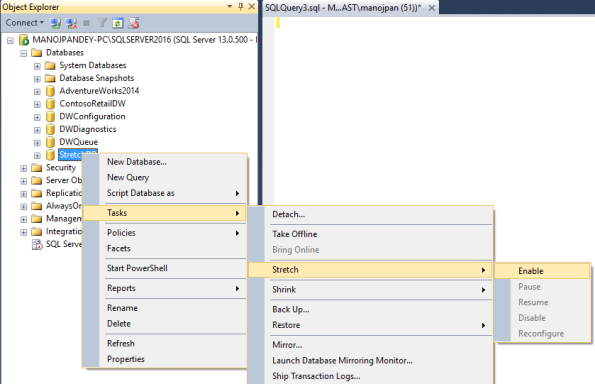
**–>** We will see here how we can configure a Database for Stretch and enable it in a table in simple steps.

**Step #1. Configure SQL Server instance to enable “Remote Data Archive”**

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | USE [master]  GO    EXEC sp\_configure 'remote data archive', '1';  RECONFIGURE;  GO |

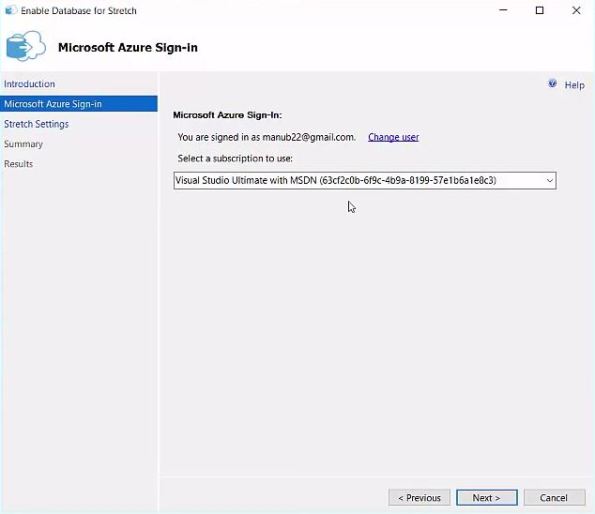
**Step #2. Enable Database for Stretch to Azure**

**2.a. Enable a Database for Stretch from SSMS:**

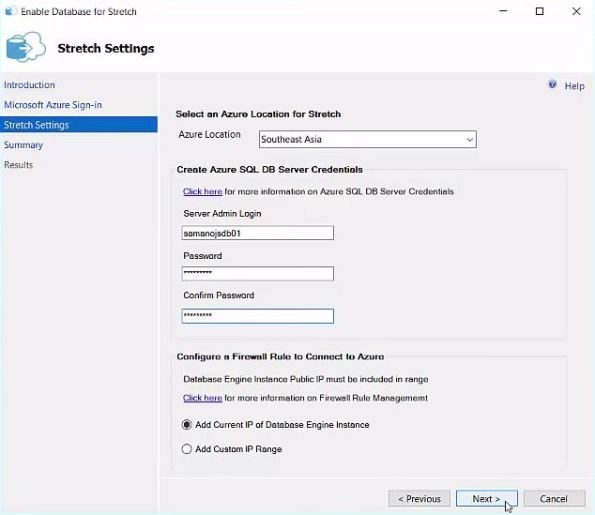
[](https://sqlwithmanoj.files.wordpress.com/2015/09/sql-server-2016-stretchdb-01.png)

Report this ad

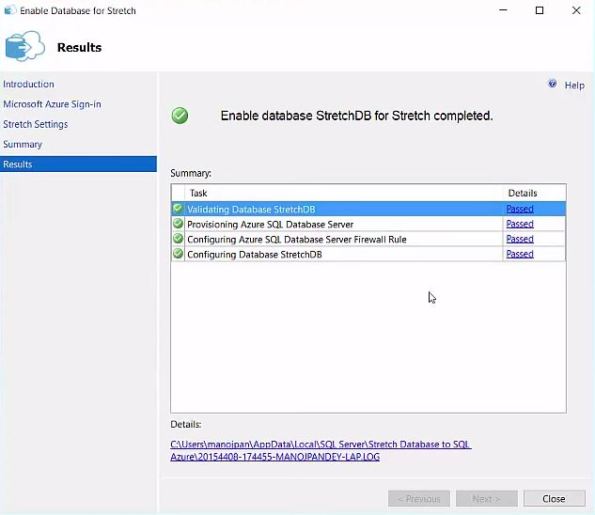
**2.b. Sign-in to Microsoft Azure with your subscription ID:**

[](https://sqlwithmanoj.files.wordpress.com/2015/09/sql-server-2016-stretchdb-02.jpg)

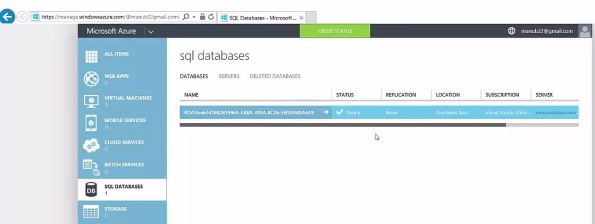
**2.c. Provide your preference for Azure Datacenter and login credentials:**

[](https://sqlwithmanoj.files.wordpress.com/2015/09/sql-server-2016-stretchdb-03.jpg)

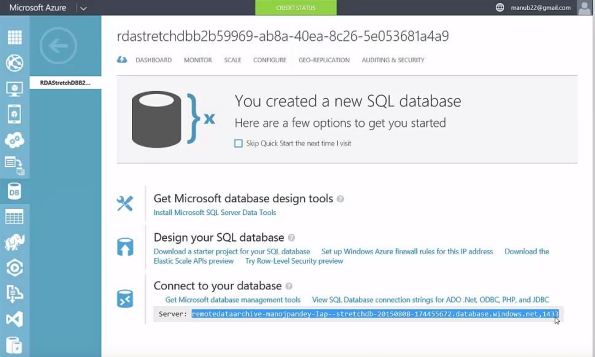
**2.d. Final page showing successful completion of enabling StretchDB:**

[](https://sqlwithmanoj.files.wordpress.com/2015/09/sql-server-2016-stretchdb-04.jpg)

**2.e. Now go to the Azure portal and check the new SQL DB created by the Enable process:**

[](https://sqlwithmanoj.files.wordpress.com/2015/09/sql-server-2016-stretchdb-05.jpg)

**2.f. Check the SQL Instance settings and DB Server name to connect it from SSMS:**

[](https://sqlwithmanoj.files.wordpress.com/2015/09/sql-server-2016-stretchdb-06.jpg)

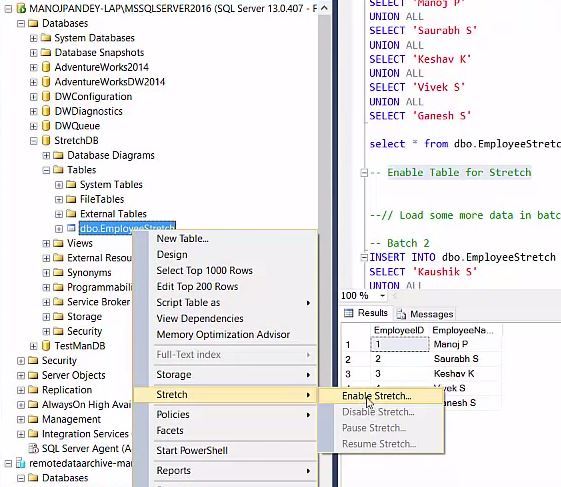
Now we have a new Azure SQL Database created that is paired up with our On-prem SQL Server instance Database. We can not enable individual tables that we want to stretch to Azure.

**Step #3. Now Create a table and insert some sample data in batches**

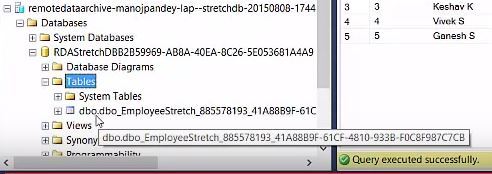
|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | USE [StretchDB]  GO    CREATE TABLE dbo.EmployeeStretch (      EmployeeID  INT IDENTITY(1,1),      EmployeeName VARCHAR(1000)  )    -- Batch 1  INSERT INTO dbo.EmployeeStretch (EmployeeName)  SELECT 'Manoj P'  UNION ALL  SELECT 'Saurabh S'  UNION ALL  SELECT 'Keshav K'  UNION ALL  SELECT 'Vivek S'  UNION ALL  SELECT 'Ganesh S'    select \* from dbo.EmployeeStretch |

**Step #4. Enable Table for Stretch**

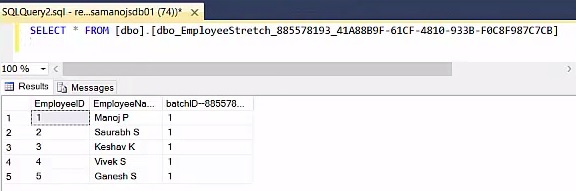
With SSMS only you can enable a table for Stretch with 1-2 simple clicks, check the image below:

[](https://sqlwithmanoj.files.wordpress.com/2015/09/sql-server-2016-stretchdb-07.jpg)

So, as soon as you enable this table for stretch the process creates an mutually equivalent table in Azure SQL DB with similar name which can be seen immediately via SSMS, check below:

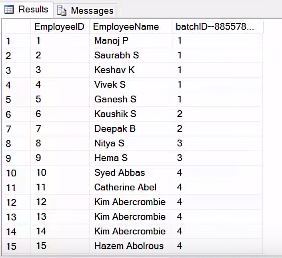
[](https://sqlwithmanoj.files.wordpress.com/2015/09/sql-server-2016-stretchdb-08.jpg)

On Querying this table the Stretch process adds an extra column with name **“batchid–[object\_id]”** suffixed with the table ID belonging to On-prem SQL DB, check below:

[](https://sqlwithmanoj.files.wordpress.com/2015/09/sql-server-2016-stretchdb-09.jpg)

**Step #5. Load some more data in batches**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | -- Batch 2  INSERT INTO dbo.EmployeeStretch (EmployeeName)  SELECT 'Kaushik S'  UNION ALL  SELECT 'Deepak B'    select \* from dbo.EmployeeStretch    -- Batch 3  INSERT INTO dbo.EmployeeStretch (EmployeeName)  SELECT 'Nitya S'  UNION ALL  SELECT 'Hema S'    select \* from dbo.EmployeeStretch    -- Batch 4  INSERT INTO dbo.EmployeeStretch (EmployeeName)  SELECT TOP 100 CONCAT([FirstName], ' ', [LastName]) FROM [AdventureWorks2014].[Person].[Person]    select \* from dbo.EmployeeStretch |

As we’ve done INSERT in separate 4 Batches, thus you can see Batch IDs assigned to records from 1-4 under the batchid column of the Stretch table on Azure SQL DB, below:  
[](https://sqlwithmanoj.files.wordpress.com/2015/09/sql-server-2016-stretchdb-10.jpg)

**–> Check the stats, i.e. space used by the table in on-prem and Azure DB:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | sp\_spaceused @objname = 'dbo.EmployeeStretch'  --Output:  --name                 rows  reserved  data   index\_size  unused  --dbo.EmployeeStretch  109   48 KB     16 KB  32 KB   0 KB    sp\_spaceused @objname = 'dbo.EmployeeStretch', @mode = N'LOCAL\_ONLY'  --Output:  --name                 rows  reserved  data   index\_size  unused  --dbo.EmployeeStretch  0     16 KB     8 KB   8 KB        0 KB    sp\_spaceused @objname = 'dbo.EmployeeStretch', @mode = N'REMOTE\_ONLY'  --Output:  --name                 rows  reserved  data   index\_size  unused  --dbo.EmployeeStretch  109   32 KB     8 KB   24 KB       0 KB |

The above stats clearly shows that the space used by Stretch enabled table in On-premise and On Azure SQL DB. As the data is moved behind the scenes from On-Prem to Azure SQL DB the space used by the table in On-prem is negligible i.e. some minimal default space.

**–> Let’s load some more data as a 5th batch**

|  |  |
| --- | --- |
| 1  2  3  4  5 | -- Batch 5  INSERT INTO dbo.EmployeeStretch (EmployeeName)  SELECT TOP 1000 CONCAT([FirstName], ' ', [LastName]) FROM [AdventureWorks2014].[Person].[Person]    select \* from dbo.EmployeeStretch |

**–> Now again check the space used:**

|  |  |
| --- | --- |
| 1  2  3  4 | sp\_spaceused @objname = 'dbo.EmployeeStretch', @mode = N'REMOTE\_ONLY'  --Output:  --name                 rows  reserved  data   index\_size  unused  --dbo.EmployeeStretch  1509  288 KB    80 KB  80 KB       128 KB |

This again shows that whatever data you load on an On-Prem Stretch-enabled table it moves all data to Azure SQL DB without taking any space in On-prem DB, but on Azure SQL DB.

**–> Some DMVs that you can use to check the stats of the data migration and Database’s & Table’s Stretch status:**

|  |  |
| --- | --- |
| 1  2  3  4  5 | select \* from sys.dm\_db\_rda\_migration\_status    select \* from sys.remote\_data\_archive\_databases    select object\_name (885578193), \* from sys.remote\_data\_archive\_tables |

**–> Let’s try to DELETE the Stretch Enabled table:**

|  |  |
| --- | --- |
| 1 | DELETE FROM dbo.EmployeeStretch WHERE EmployeeID >= 10 |

It throws following error:

**Msg 14826, Level 14, State 1, Procedure trigger\_RemoteDataArchive\_885578193, Line 94  
Update and delete of rows eligible for migration in table ‘EmployeeStretch’ is not allowed because of the use of REMOTE\_DATA\_ARCHIVE.  
Msg 3609, Level 16, State 1, Line 79  
The transaction ended in the trigger. The batch has been aborted.**

**Please Note:** As of CTE 2.3 release you can’t run UPDATE or DELETE operations on a Stretch-enabled table.