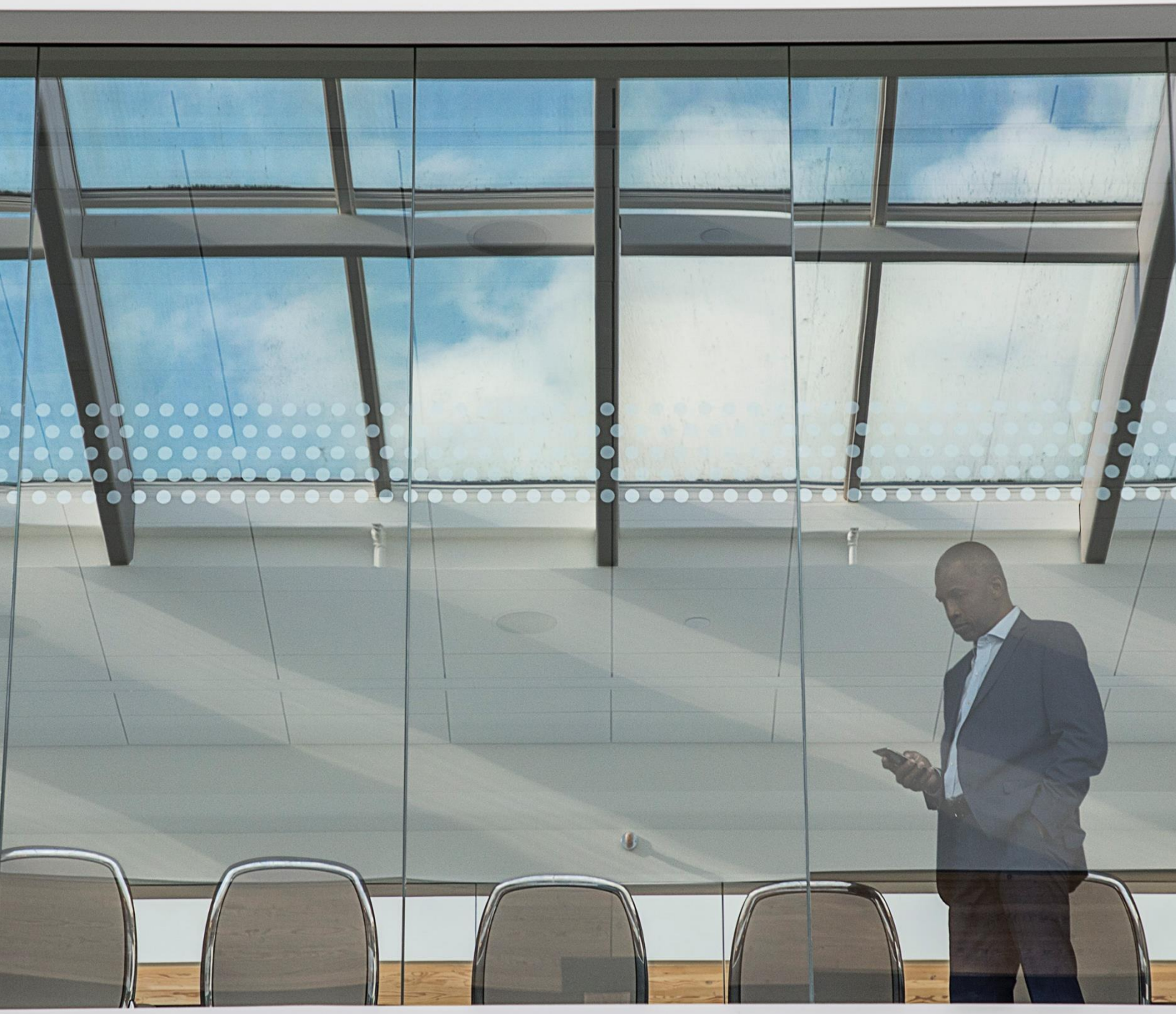
Microsoft



Cloud Migration Reference Guide for the end of support of SQL Server 2008 and 2008 R2

Migrating to Azure

Microsoft Corporation

First draft completed: September 2018

Version: 7.0

Authors: Scott Klein

Contributors: Frederico P. Rezende

Reviewers: Alain Dormehl, Eric Hudson, Ajay Jagannathan, Amit Banerjee, Jim Toland, Venkata Raj Pochiraju, Mine Tokus

For the latest documentation on the end of support of SQL Server 2008/R2, Azure SQL Database, Azure SQL Database Managed Instance, and Azure Migrations, please see:

- https://azure.microsoft.com/en-us/services/sql-database/
- https://azure.microsoft.com/en-us/migration/
- https://www.microsoft.com/en-us/sql-server/sql-server-2008

Disclaimer

Contents

Overview

Extended support for SQL Server 2008 and 2008 R2 will end on July 9th, 2019, with mainstream support having already ended on July 8th, 2014. After the period of extended support, no more updates or support for SQL Server 2008 and 2008 R2 will be provided, potentially leaving you vulnerable to security and compliance issues.

The purpose of this document is to provide a comprehensive look at the recommended migration and upgrade paths from SQL Server 2008 and 2008 R2 to SQL Server 2017, Azure SQL Database, and Azure SQL Database Managed Instance. Components described and covered as part of the migration and upgrade will cover the database and BI components. Consideration and guidance will be given for cases of migrating one or multiple databases, considering database downtime and migration performance.

This document will also describe the terminology, tools, technologies and services, principles and best practices, and processes necessary to successfully plan and migrate from SQL Server 2008 and 2008 R2 on-premises to the solution in Microsoft Azure appropriate for your database environment.

This document contains four main sections which will cover the migration and upgrade scenarios:

1. **Assessment** – an overview of the tools used to assist in the database compatibility assessment and migration
2. **Migration** – pure lift-and-shift scenarios,
3. **Hybrid** – public cloud-private cloud and public cloud-public cloud scenarios, and
4. **Rebuild** – scenarios where most of the refactoring, re-architecting, and rebuilding are needed, especially in the Business Intelligence stack.

This document is a living document and will be updated as technologies and features evolve.

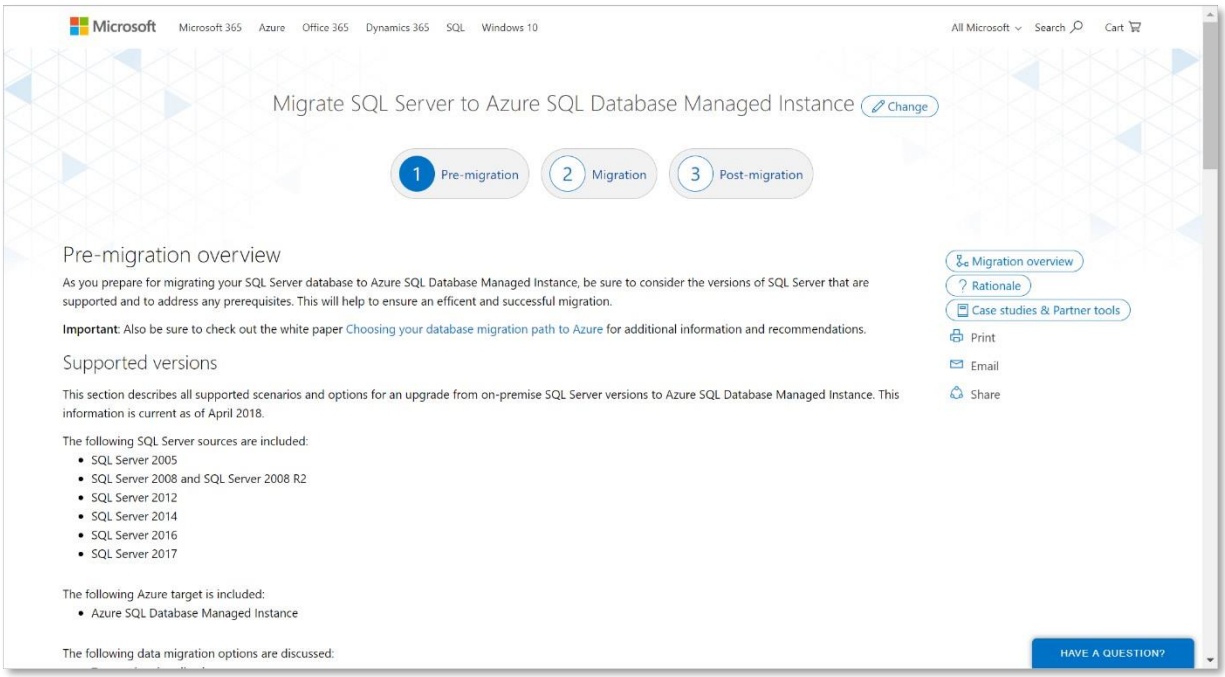
Assessment

The key to any migration from SQL Server 2008 and 2008 R2 begins with the assessment, regardless of the final destination of your database, whether it be SQL Server 2017 on-premises, an Azure Virtual Machine, Azure SQL Database, or SQL Database Managed Instance.

The assessment itself, and tools used in the assessment, provide an in-depth analysis of your database and its workload in preparation for migration to a modern data platform. The tools available perform compatibility and functionality checks, recommend performance and reliability improvements, perform performance A/B working testing, and perform the actual migration.

Any assessment should begin by following the recommendations in the Azure Database Migration Guide (https://datamigration.microsoft.com/), or DM Guide. The DM Guide prompts you to select the source platform currently hosting your database and the target platform, such as Azure SQL Database, SQL Server in an Azure VM, and more, to which you want to migrate.

The DM Guide then provides detail in three sections (**Pre-migration**, **Migration**, and **Post-migration**) that offer customers information about the migration process, the tools and services available to support each stage of the process, and step-by-step instructions for completing the process.



Assessment and Migration Tools

As part of the pre-migration recommendations and steps, the Database Migration Guide lists and suggests several tools to help in the assessment and migration process by identifying and discovering details about the selected data sources to gain a deep understanding of the data sources to plan for a successful migration. Depending on the destination, the migration guide will suggest one or more of the following tools:

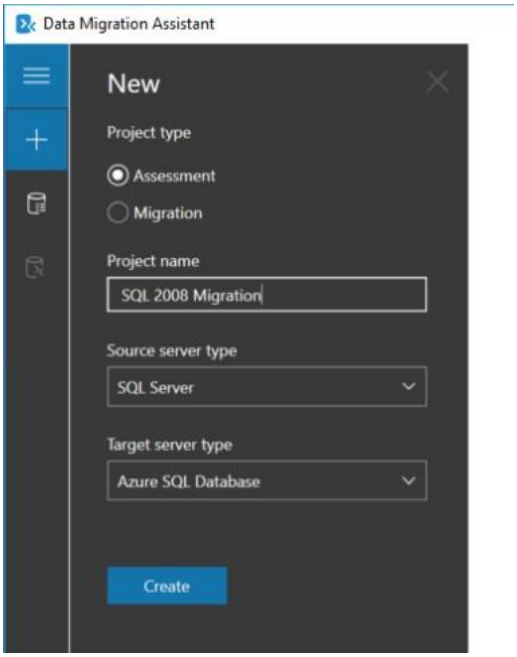
- Microsoft Data Migration Assistant
  - Download
  - Documentation

- Microsoft Database Experimentation Assistant
  - Download
  - Documentation
- Microsoft Assessment and Planning Toolkit
  - Download
  - Documentation
- Azure Database Migration Service
  - Documentation

The following sections discuss each of these tools and services.

Data Migration Assistant

The Microsoft Data Migration Assistant (DMA) is a database assessment and migration tool, detecting database compatibility and functionality issues between SQL Server 2008/R2 and the destination version of SQL Server or SQL Server on Azure VM or Azure SQL Database or Azure SQL Database Managed Instance, then ultimately executing the migration from source to destination.



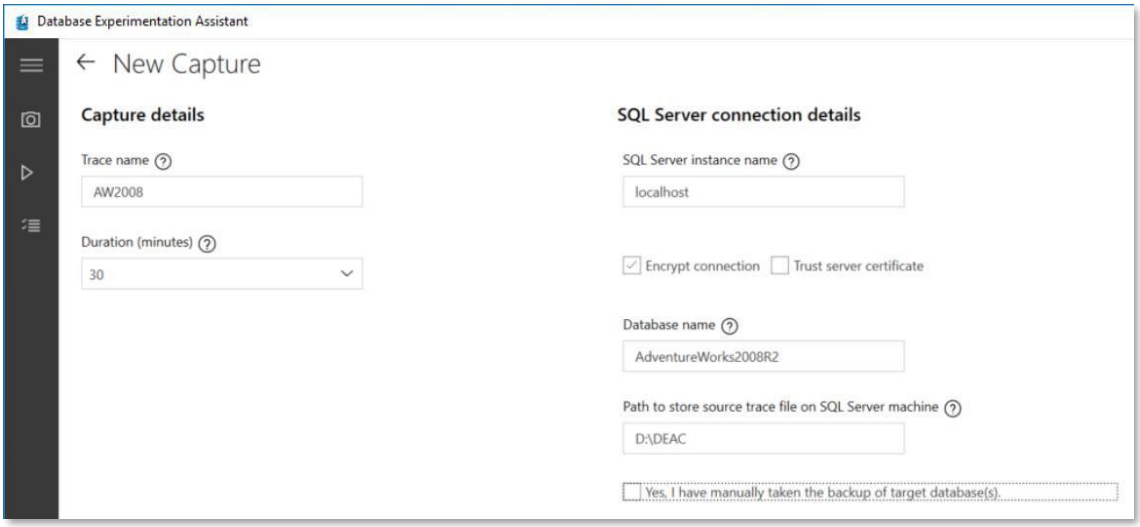
The assessment portion of the DMA detects migration blocking issues as well as unsupported or partially supported features that will impede a migration to the selected SQL Server or SQL Server on Azure VM, Azure SQL DB or Azure SQL Database Managed Instance destination version, such as breaking feature or behavior changes and features that have been deprecated. DMA will also recommend new features in the target version which the migrated database could benefit from post-migration, such as security and performance features and enhancements.

After the assessment is done, DMA can be used to migrate the selected on-premises database, including the schema, data, roles, and logins to modern on-premises SQL Server or SQL Server on Azure VM. DMA can also be used to deploy the schema to target Azure SQL Database. While DMA will let you to migrate

data to Azure SQL Database, we only recommend using DMA for smaller databases up to the size of 100 GB, especially when you are doing POC and test migrations. Use Azure Database Migration Service (DMS) for scale and resilient production database migrations

Database Experimentation Assistant

The Microsoft Database Experimentation Assistant (DEA) is a new tool which provides workload evaluation testing using an A/B testing process. During the assessment and analysis phase of migrating from SQL Server 2008/R2, the DEA can be used to capture workloads on source database systems and then replay the trace file on the target system. The resulting analysis and metrics are then used to determine the appropriate Azure SQL Database service level or virtual machine size to ensure a successful and high-confidence migration and upgrade.



The DEA gathers workload metrics from analysis such as degraded queries, query compatibility errors, query plans, and other workload comparison data to provide the appropriate recommendations.

Assessment and Planning Toolkit

The Microsoft Assessment and Planning Toolkit (MAP) is a powerful assessment, inventory, and reporting tool that can be used to assist in the migration process. MAP simplifies the process of accessing your current IT infrastructure for generating detailed assessment reports with hardware and software evidence and information.

Azure Database Migration Service

The Azure Database Migration Service (DMS) is a fully managed PaaS service specifically designed for smooth migrations of multiple database sources to Azure data platforms with as little downtime as possible.

The DMS is accessed via the Microsoft Azure portal and should be used in conjunction with the aforementioned, and discussed, Data Migration Assistant to assess the source database prior to performing the migration to check for any database compatibility and functionality issues. The Data Migration Service does require the implementation of Azure ExpressRoute or VPN in order to connect to on-premises sources.

A key benefit of the Data Migration Service is that it is a fire-and-forget process, meaning that once the migration process is started, the service will perform the appropriate and necessary migration steps.

Regardless of the migration scenario chosen as outlined in the following sections, it is assumed that the initial step will be to run an assessment as outlined in the Database Migration Guide prior to performing any type of migration or upgrade to SQL Server or Azure SQL Database.

Limitations

The Azure Database Migration Guide is designed to guide you in your process for migrating your database to a modern data platform of your choice, such as Azure SQL Database, SQL Data Warehouse, or Azure SQL Database Managed Instance, to name a few. Where it lacks in information is how to migrate ancillary components and services, if possible, to the same platform. As such, each section will discuss and provide additional guidance and information for migrating items such as BI components and failover clusters.

Migration

In a pure lift-and-shift migration scenario, the database and BI components are simply rehosted in an environment that requires little-to-no refactoring, re-architecting, or rebuilding of functionality, components, or objects.

Deploying the source database to SQL Server in an Azure Virtual Machine enables organizations to maintain flexibility and control of the database and operating system while taking advantages of the benefits of Microsoft Azure. This lift-and-shift scenario reduces overhead and time to market while offering consistency across on-premises and the Azure platform.

Azure SQL Database Managed Instance

The primary lift-and-shift migration scenario is Azure SQL Database Managed Instance. Quickly gaining momentum, Managed Instance is a pure PaaS (Platform-as-a-Service) offering, providing near 100% compatibility with on-premises SQL Server Enterprise Edition. Managed Instance allows organizations to easily lift and shift their on-premises databases to Azure with little-to-no database changes while still taking advantage of the PaaS capabilities, such as automatic patching, automated backups, high availability, integration with other PaaS services, and version updates, among others.

Organizations still have control over the SQL Server instance while allowing complete isolation of instances with native VNET support, but do not need to worry about the infrastructure components such as the operating system.

As suggested above, proper assessment should be executed to determine the database compatibility and appropriate service tier for your database for the given workload. Both service tiers provide a 99.99% availability SLA as well as low latency and optimal performance.

More information on Azure SQL Database Managed Instance here.

Database

Managed Instance supports two main database migration options:

- Native – native backup and restore of a database backup (.bak files) using Backup to URL and Restore from URL T-SQL syntax. This option requires some database downtime.
- Database Migration Service – at-scale migration with near-zero downtime.

These two options provide a pure lift-and-shift of an applications back end without the need to re-architect the application.

The following videos illustrate the two different migration options. The first video provides a step-by-step tutorial showing the steps and process of migrating an on-premises SQL Server database to Azure SQL Database Managed Instance. The second video shows how to use the Database Migration Service (DMS) to perform at-scale database migrations, i.e., migrating multiple databases with near-zero downtime.

https://www.youtube.com/watch?v=KQ7laP_su4g&t=2s

https://www.youtube.com/watch?v=onP5y2TNhN0&t=23s

BI Components

When migrating the database to Azure SQL Database Managed Instance, the BI components will either need to be migrated to an Azure virtual machine or kept on premises, thus resulting in a hybrid scenario. This scenario is discussed in the Hybrid section later in this document.

Azure Virtual Machine

After the assessment phase is complete and issues addressed, the database and any BI components are ready for migration to the Azure Virtual Machine.

Database

The recommended method of migrating the database to SQL Server in an Azure Virtual Machine is by using the suggested methods as outlined in the Database Migration Guide for this particular scenario. Depending on the scenario, several options will be suggested, including, but not limited to, the following:

- Using the DMA to migrate the schema and data to the Azure VM.
- Manually backup the database, copy the .bak file to the virtual machine, then preform a restore.
- Backup the database to Blob using Backup to URL, then restore the database into the Azure VM from the URL.
- Detach and copy the data (.mdf) and log (.ldf) files to Azure Blob storage, copy the data and log files from Azure Blob storage into the Azure VM, then attach them in the Azure VM. Likewise, it is possible to attach the data and log files from the Blob URL.

Most of the options listed will keep the compatibility level set at the SQL Server 2008 or SQL Server 2008 R2. Once the database is migrated, the database compatibility level will need to be manually changed to SQL Server 2017 (140).

BI Components

When fully migrating from on-premises to an Azure Virtual Machine, the BI components can be migrated in a similar manner. Whether the choice is made to run the BI components on the same virtual machine as the database, or a secondary virtual machine is used for BI, is up to the organization.

Regardless of where the BI components are installed, migrating the BI components are similar to that of migrating the database. For example, SSIS packages and SSRS reports can be copied from on-premises to the virtual machine. One of the common ways to migrate is by using the Azure Data Factory SSIS Runtime, which is discussed later in this document.

In some cases, the BI components may initially be left out of the scope of migration to focus on the database aspect of the migration and migrated later. This is a common scenario and will be discussed in the Hybrid section later in this document.

Compatibility Level Considerations

An additional item to consider when migrating to an Azure VM is the database compatibility. With the end of support of SQL Server 2008 and 2008 R2 coming in July of 2019, organizations should consider migrating to SQL Server 2017 in an Azure VM sooner rather than later for several reasons.

First, organizations can migrate their database to an Azure VM running SQL Server 2017, but keep the database compatibility level set to 100 (for SQL Server 2008). Microsoft guarantees no performance degradation when keeping the database compatibility level at 100, and full compatibility level 100 operational capabilities.

Second, organizations can take advantage of the performance improvements made in the SQL Server 2017 engine while planning to make, and making, the necessary compatibility and functionality changes necessary to run in compatibility level 140.

Hybrid

The term hybrid cloud typically refers to an environment which combines a public cloud with a private cloud (such as an on-premises environment). In this type of hybrid cloud, businesses have the ability to seamlessly scale their on-premises infrastructure to a public cloud to handle any overflow. Organizations gain the ability to scale computing resources on demand to meet their workload needs while eliminating initial capital expenditure.

However, an often-overlooked term for hybrid cloud refers to the use of multiple cloud platform offerings, such as Platform-as-a-Service (PaaS) and Infrastructure-as-a-Service (IaaS). In this type of hybrid cloud solution, organizations can still take advantage of eliminating capital expenditure while scaling resources on demand but have the additional benefit of choosing which platform offering to use for the database and any secondary components (BI, etc.).

SQL Server 2008 and 2008 R2 can leverage both scenarios.

Azure Virtual Machine with BI on-premises

As organizations plan to migrate to Azure, a common scenario is one where the database is migrated to SQL Server in an Azure Virtual Machine while some of the secondary components (such as the BI stack) still run on-premises.

As discussed in the previous section, the recommendations and options for migrating an on-premises database to an Azure Virtual Machine also apply in this scenario. Additionally, this scenario also requires little-to-no refactoring, re-architecting, or rebuilding of functionality, components, or objects.

As eluded to earlier in this document, there are certain benefits for this scenario. In some cases, the BI components may initially be left out of the scope of migration to allow the organization to focus on the database aspect of the migration and ease the database migration process. The BI components can then be migrated once the databases are migrated, or at a later period once the database post-migration assessment has been done and the migration has been tested and deemed successful.

Another benefit briefly mentioned earlier, was that this scenario allows organizations to take advantage of the performance improvements made in the SQL Server 2017 engine while making the necessary compatibility and functionality changes necessary to run in compatibility level 140. Additionally, this scenario also allows organizations to keep sensitive data on-premises and allow the BI stack to interact with data in both locations, on-premises and in Azure.

A disadvantage to this scenario is that data must travel across the internet to and from the organization and the Azure region, resulting in performance latency and additional financial costs. While data going into the Azure region (data center) has no additional cost, any data which leaves an Azure region incurs an additional cost based on the amount of data being transferred.

In addition to using SQL Server Integration Services (SSIS) to move data back and forth between the database in the Azure VM and on-premises, a common technology used for this task is PowerShell. Some organizations have found PowerShell to be an option for data collection and movement, offering some flexibility for parallel bulk data operations where there are many databases and large quantities of data to move. It should be noted that command-line shells and scripting languages such as PowerShell call an underlying API which is not native to PowerShell. Thus, SSIS can be, and is, still used for the more traditional translation and manipulation and should be considered first.

Azure SQL Database or Azure SQL Database Managed Instance with BI in a Virtual Machine

The following two sections will discuss the process of migrating an on-premises SQL Server database to Azure SQL Database Managed Instance and Azure SQL Database. Both of the following sections assume that the appropriate options and selections have been made in the Azure Database Migration Guide and the pre-migration assessments have been performed.

Azure SQL Database Managed Instance

A second hybrid scenario, one which is quickly gaining popularity, is migrating the database to Azure SQL Database Managed Instance. Managed Instance is a recent capability of Azure SQL Database, and as discussed in the previous section, provides near 100% compatibility with on-premises SQL Server Enterprise Edition.

The difference between SQL Server running in an Azure Virtual Machine and Azure SQL Database Managed Instance is that Azure SQL Database Managed Instance is a pure PaaS (Platform-as-a-Service) offering, whereas SQL Server running in an Azure Virtual Machine is IaaS (Infrastructure-as-a-Service).

Please refer to the Azure SQL Database Managed Instance option in the Migration section earlier in this document for further information on database migration and information on Managed Instance.

Once the database is migrated, it is vitally important to track database performance and behavior post-migration as some changes are enabled once the database compatibility level has been changed. When a database is migrated to Managed Instance, the database keeps its original compatibility level in most cases. The lowest compatibility level in Managed Instance is 100, thus migrating an on-premises database with compatibility level 100 to Managed Instance will keep its compatibility level. However, migrating an on-premises database with compatibility level 90 will result in the compatibility level being set to 100 once migrated to Managed Instance. Therefore, to reduce migration risks, change the compatibility level after performance testing and monitoring to ensure optimal workload performance in Managed Instance.

Azure SQL Database

Similar to Azure SQL Database Managed Instance, but a scenario that has been around for quite a while, is migrating the on-premises database to Azure SQL Database. Also a PaaS offering, Azure SQL Database is based on the latest stable version of SQL Server, but differs from Azure SQL Database Managed Instance in that organizations do not have control over the SQL Server instance. Azure SQL Database is built on a database-scoped programming model and doesn't quite have 100% compatibility with on-premises SQL Server Enterprise Edition.

However, one of the benefits that Azure SQL Database has is the concept of elastic pools; a cost-effective solution for managing and scaling multiple databases that have unpredictable usage patterns and demands. Refer to the following URL for further information on Azure SQL Database elastic pools.

https://docs.microsoft.com/en-us/azure/sql-database/sql-database-elastic-pool

One or Many Databases

Critical items to consider when migrating to either Azure SQL Database or Azure SQL Database Managed Instance is the number of databases being migrated and the size of the databases.

As shown and demonstrated in the video above, after SQL Database Managed Instance is set up and configured, databases can be migrated quite easily. The DMS is specifically built to provide and facilitate migrations of multiple database sources to Azure data platforms as quickly as possible and with the most minimal amount of downtime.

BI Components

Both Azure SQL Database Managed Instance and Azure SQL Database integrate very well with other Azure services, such that a common hybrid scenario is one where BI components can be hosted in an Azure VM and utilized from both Managed Instance and SQL Database.

For Managed Instance, the Managed Instance must be the only service in the subnet within the VNET, but an Azure VM can be created in a second subnet within the same VNET and interact with the Managed Instance easily.

Rebuild

With the availability of Azure SQL Database Managed Instance, Microsoft has made it possible to migrate your on-premises database to the Azure platform with little to no changes to the database, as Managed Instance provides near 100% compatibility with on-premises SQL Server Enterprise Edition.

In terms of the database itself, the migration and upgrade process is made easier due to the many tools available which provide guidance in discovering and addressing compatibility issues. As discussed in the previous section, a SQL Server database can be migrated to Azure SQL Database Managed Instance in a Rebuild scenario.

Azure SQL Database Managed Instance with Re-built BI components

The same cannot be said for the BI components, and ultimately, hosting the BI stack in an Azure VM long term is not a viable solution. BI functionality is slowly making its way into PaaS offerings as is evident with the release of Azure Analysis Services. Power BI has taken over for Reporting Services, and Azure Data Factory is quickly becoming the service to supplant Integration Services.

As a result, working with BI generally requires the majority of refactoring, re-architecting, and rebuilding.

BI Components

Depending on the business intelligence capability or feature that is being considered for a cloud migration will somewhat dictate how much rebuilding and rearchitecting will be needed. The following sections will discuss and highlight several options for each business intelligence feature.

Analysis Services

With the availability of Azure Analysis Services (AAS), migrating to AAS from on-premises SQL Server Analysis Services (SSAS) is quite intuitive, with Microsoft ensuring compatibility between on-premises SQL Server Analysis Services and Azure Analysis Services.

What this means is that it is quite simple to move your Symantec model from on-premises SSAS to AAS using the current tools used to work with SSAS, such as SQL Server Management Studio and SQL Server Data Tools.

With the AAS server provisioned and the server name obtained, SQL Server Data Tools can be used to deploy the Analysis Services Symantec model to AAS by providing the new AAS server name for the deployment server and then deploying the model to AAS. The model can still be managed and maintained as it was before.

The migration of the models can also be automated using several options including the ARM API or the APIs that come with Analysis Services to automate the deployment and management of the models.

A few points to highlight when considering migrating SSAS to AAS:

- Models must be in the Tabular format
- Models need to be in 1200 or higher compatibility levels
- Multidimensional models would need to be converted to Tabular format

An important and key item to highlight is that AAS has the ability to connect to data sources in Azure as well as on-premises. That means you can leave your data sources on-premises and connect to them via a gateway to take advantage of the benefits of Azure. Likewise, AAS can leverage a hybrid scenario where some of the data sources are on-premises and other data sources in Azure to have a full end-to-end data solution in the cloud.

By leveraging AAS, organizations can scale on demand as their data and resource needs grow by scaling capacity with a few simple clicks.

Integration Services

Organizations have some level of flexibility when it comes to migrating SQL Server Integration Services (SSIS) packages to Azure. Currently, there are three options:

1) Deploy SSIS Packages to Azure and execute them via the Azure Data Factory SSIS Integration Runtime.
2) Deploy SSIS Packages to Azure and execute them via an Azure Data Factory pipeline.
3) Re-architecting and rewriting the SSIS packages using other services and technologies, including Azure Databricks and Azure Data Factory.

The first two options require the creation of an Azure SQL Database in which to host the SSIS catalog, but do not require the rewriting of any SSIS packages. Azure Data Factory can be utilized to create the SSISDB, or it can be created separately and referenced within Azure Data Factory. With the SSISDB catalog created, familiar tools such as SQL Server Data Tools (SSDT) or SQL Server Management Studio (SSMS) can be used to deploy the SSIS packages to the Azure SSIS Catalog (SSISDB database).

The first option involves no integration with other data factory concepts other than using the integration runtime in Azure Data Factory to natively execute SSIS packages. The integration runtime is a compute infrastructure used by Azure Data Factory to provide data integration capabilities across different environments.

In the second option, the difference is that SSIS packages are invoked through an Azure Data Factory pipeline activity and not directly with the integration runtime, thus requiring more integration with Azure Data Factory components and concepts. The pipeline does still use the Integration Runtime under the covers, but using Azure Data Factory pipelines and activities allows organizations to construct end-to-end workflows, group activities, and have more control and flexibility over the execution of data movement and data processing scenarios.

The third option is more daunting but provides greater flexibility within ETL processes. Azure Databricks is an Apache Spark-based analytics platform optimized for the Microsoft Azure cloud platform. Among its deep and rich analytics capabilities, Azure Databricks provides the ability to perform ETL operations, extracting data from wide range of data sources, execute rich transformations on the data, then load the transformed data into an equally wide range of destinations.

There are pros and cons to each of the three options. While re-writing SSIS packages is not on the very top of everyone's "to-do" list, Azure Databricks is more of a developer-friendly environment, where more imperative code rather than declarative code is used. Databricks also has the ability to blend in not only ETL processes, but streaming, analytics, and ML processing in the same code and job.

On the flip side, there is a cost to Azure Databricks, where there is not with Azure Data Factory. Thus, running and executing existing SSIS packages using Azure Data Factory requires only the cost of the SSIS catalog.

Reporting Services

With the investments being made into Power BI, many organizations are migrating from SQL Server Reporting Services (SSRS) to Power BI. Organizations can migrate from SQL Server 2008 Reporting Services to Power BI Report Server which provides self-service BI and enterprise reporting capabilities of SSRS in one solution.

Power BI Report Server can be run on-premises or in an Azure VM. Migrating to Power BI Report Server adds the flexibility to move to the cloud through PowerBi.com when the organization is ready.

Currently, there is no migration path from SQL Server Reporting Services (SSRS) to PowerBi.com due to paginated reports (.rdl) are not currently supported in PowerBi.com. There are plans on making this change in the near future, at which point this document will be updated.

Thus, the current solution for migrating SSRS to the cloud is either running SSRS in an Azure VM, as discussed previously, or running Power BI Report Server in an Azure VM and migrating an SSRS report server installation to Power BI Report Server.

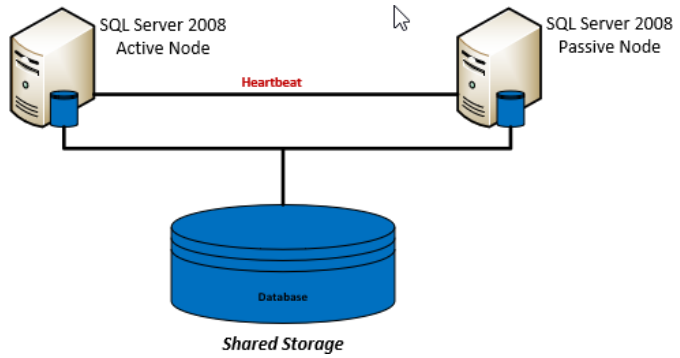
The process for migrating from SSRS to Power BI Report Server requires the following:

- Backing up the SSRS report server database, application and configuration files, and encryption key
- Cloning the report server database
- Configure Power BI Report Server to connect to the cloned database

More information about migrating from SSRS to Power BI Report Server can be found at the following URL: https://docs.microsoft.com/en-us/power-bi/report-server/migrate-report-server

Failover Clusters

High availability in SQL Server 2008 and SQL Server 2008 R2 was accomplished via failover clusters; the concept of installing one or more SQL Server instances into a Windows Failover Cluster using shared storage, as shown in the following diagram.



In a failover cluster, organizations are responsible for the maintenance and upkeep of the environment, ensuring that the failover cluster has been properly configured.

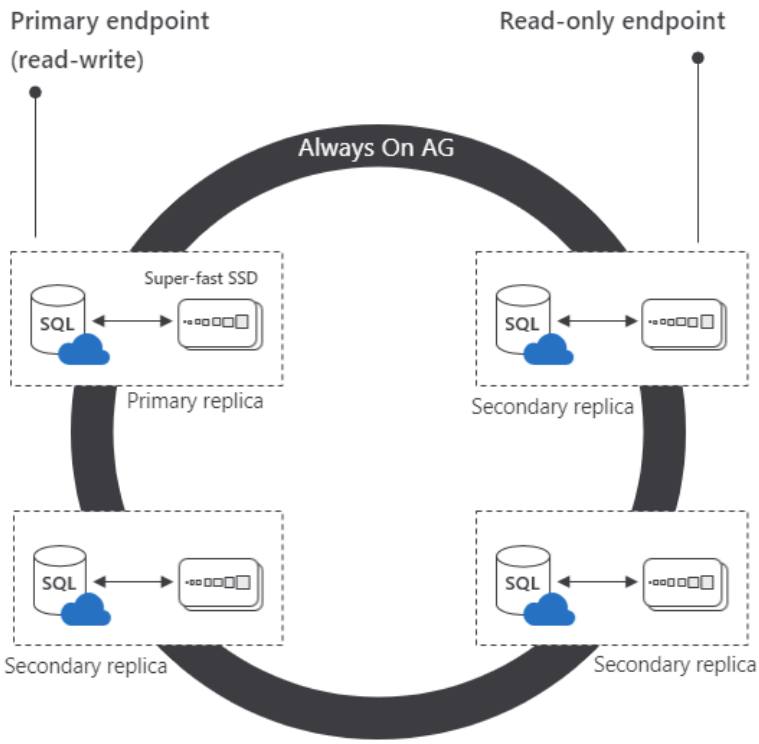
However, as mentioned previously in this document, Azure SQL Database Managed Instance provides near 100% compatibility with on-premises SQL Server Enterprise Edition. However, a common question asked from organizations who run their databases within failover clusters for high availability want to know how to get the same high availability guarantee in a Managed Instance.

Currently, Managed Instance provides two pricing tiers; General Purpose and Business Critical. The Business-Critical service tier is designed and architected for applications which require enterprise level uptime, which is provided via several isolated Always On replicas. Thus, high availability is provided automatically behind the scenes.

With the General-Purpose pricing tier, Managed Instance provides a secondary non-readable replica, while the Business-Critical pricing tier provides two replicas; one readable and one that is non-readable. Behind the scenes, the Azure Service Fabric is coordinating the high availability in both cases.

Therefore, HA configuration is configured at the database level and is automatically applied when executing DATABASE RESTORE and DATABASE CREATE statements. Additionally, when using the Database

Migration Service, or natively backing up and restoring a database, built-in HA will be configured automatically. That is, high availability is achieved by replication of compute and storage deployed and managed automatically through Always On Availability Groups.



Business Critical service tier: collocated compute and storage

A key point with high availability with Managed Instance is the fact that Managed Instance provides HA for the databases, including master, model, and msdb.

A few additional points for consideration:

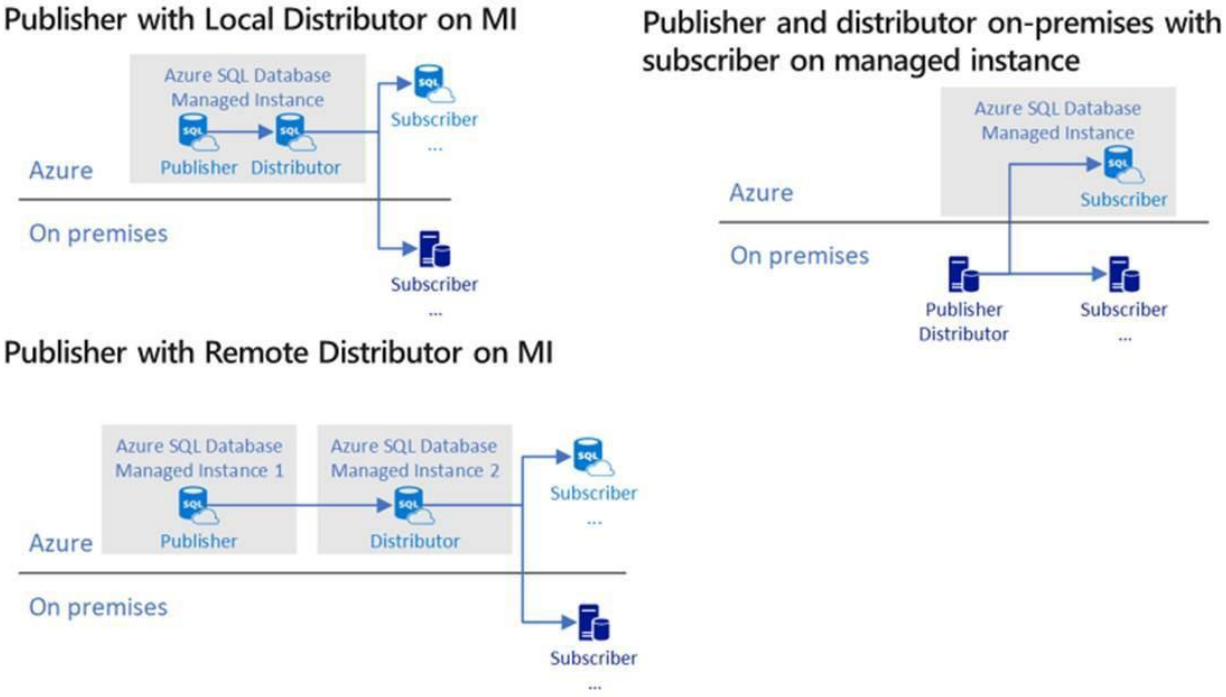
- Transactional replication from on-premises to Managed Instance, and visa-versa, is supported.
- An on-premises AG cannot be extended to a Managed Instance.
- Database Mirroring and Log Shipping cannot target a Managed Instance.

Since High Availability is built into Managed Instance and therefore cannot be controlled by the user, there are a few T-SQL statements which are not supported:

- CREATE ENDPOINT..FOR DATABASE_MIRRORING
- CREATE/ALTER/DROP AVAILABILITY GROUP
- SET HADR clause of the ALTER DATABASE statement

Replication

SQL Database Managed Instance supports several replication scenarios, as shown in the following figure. These scenarios allow the flexibility to replicate data across environments with minimal impact.



Snapshot replication and Transactional replication are both supported in these scenarios, with Snapshot replication supporting the same functionality as on-premises. However, Transaction replication has a few restrictions, including:

- Updateable subscriptions are not permitted
- Publisher and distributor must be in the same location
- Connections to the Distributor must use SQL Authentication
- Azure file share must be used to store data if the publisher and distributor are in a Managed Instance.

Merge replication and peer-to-peer replication is currently not supported, as well as heterogeneous replication.

Summary

As SQL Server 2008 and 2008 R2 nears its end of support, organizations now have more options than ever to leverage Microsoft's modern data platform. Enterprises have the choice and flexibility to span their environment and workload across on-premises and the cloud in a true lift-and-shift or hybrid scenarios.