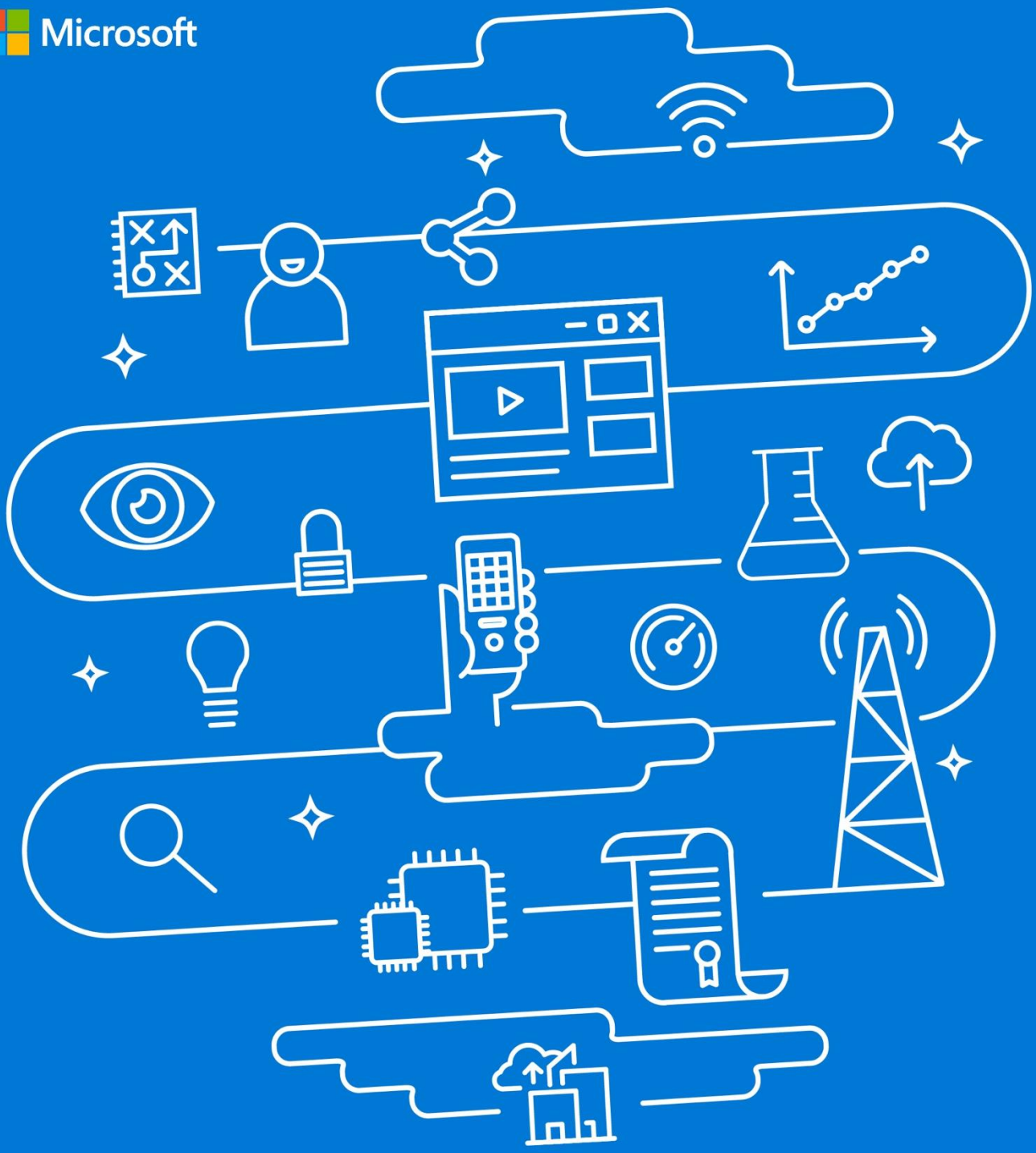
Microsoft



Choosing your database migration path to Azure

Microsoft Corporation

First draft completed: March 2018

Version: 7.0

Authors: Amber Williams, Becky Isserman, Steve Burkett

Contributors: Eric Hudson, Murat Ozturan, Borko Novakovic, Andrey Antjufejevs, Rag Guru, Venkata Raj Pochiraju, Ajay Jagannathan, Alain Dormehl

Reviewers: Alain Dormehl, Eric Hudson, Rag Guru, Ajay Jagannathan

For the latest documentation on Azure SQL Database, please see
https://azure.microsoft.com/en-us/services/sql-database/

TABLE OF CONTENTS

1 Introduction 4
1.1 Intended audience 4
1.2 Prerequisites 4
1.3 Out of scope 4
2 Overview 5
3 Initiate and Discover 6
3.1 Microsoft Tools and Services: Database Migration Guide 8
3.2 Microsoft Tools and Services: MAP Toolkit 10
3.3 Microsoft Tools and Services: Data Migration Assistant 14
4 Assessment 18
4.1 Assess Workloads for Migration 19
4.2 Assess Workload Criteria 20
4.3 Database assessment using Database Migration Assistant (DMA) 23
4.4 Assessment Steps using DMA 24
4.5 Look for high level red flags 29
5 Plan 30
5.1 Plan target platform 31
5.2 How to Choose the Right Target Platform 35
5.3 Choosing Target Platform by Usage Scenarios 35
5.4 Choosing Target Platform by Features 36
5.5 Choosing Target Platform by Cost 36
5.6 Migrating SSAS, SSIS and SSRS to an Azure Fully managed service offering 37
5.7 Migrating SSAS, SSIS and SSRS to Azure IaaS 38
5.8 Plan the Migration Tool 38
5.9 Target Platform Selection Examples 39
5.10 Example Summary – Target Platform Selection 42
5.11 Example Summary – Migration Tools Selection 44
6 Transform and Optimize 45
6.1 Transformation 46
6.2 Optimization 47
7 Migrate, Validate and Remediate 49
7.1 Migration Overview 50
7.2 Migration Tool Selection 52
8 Conclusion 59
9 Resources 60

1 INTRODUCTION

Azure SQL Database is a fully managed service that is comparable to a traditional on-premises SQL Server deployment, but greatly enhances SQL performance and robustness by making performance levels and storage capacity easily upgradable as well as providing standard high availability. Azure SQL Database delivers predictable performance at multiple service levels that provides dynamic scalability with no downtime, built-in intelligent optimization, global scalability and availability, and advanced security options — all with near-zero administration. These capabilities allow you to focus on rapid app development and accelerating your time to market, rather than allocating precious time and resources to managing virtual machines and infrastructure.

Azure SQL Database currently resides in 38 data centers around the world, with more data centers coming online regularly, enabling you to run your database in a data center near you.

With so many on-premises implementations at customer sites, how do you migrate from the traditional on-premises SQL Server implementation to modern Azure SQL Database technologies and benefit from what cloud database services can offer? This whitepaper will guide you through the thought process and steps required to migrate your database workloads from on-premises to Azure-based cloud services as well as SQL Server components such as SQL Server Reporting Services, SQL Server Analysis Services and SQL Server Integration Services.

1.1 Intended audience

This whitepaper is intended for data professionals, IT professionals, and IT decision makers who are looking to modernize their data estate by migrating on-premises SQL Server workloads to Microsoft Azure cloud services.

1.2 Prerequisites

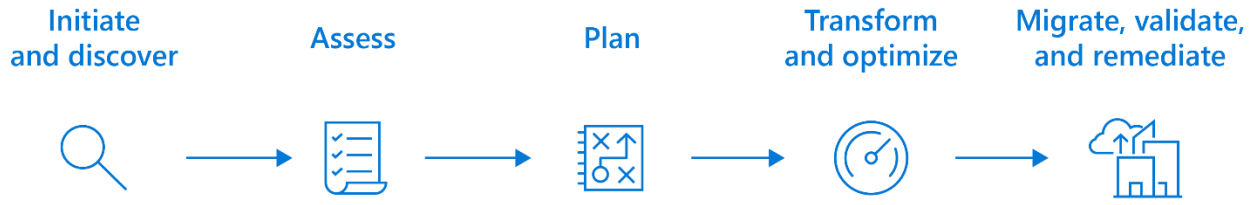
We assume that readers have some familiarity with Microsoft SQL Server and Azure cloud services.

1.3 Out of scope

While non-SQL Server workloads can certainly be migrated to Microsoft Azure cloud services, that is not the focus of this whitepaper.

2 OVERVIEW

The SQL Migration Roadmap consists of five stages, each encompassing several important tasks required to complete a successful migration to Azure cloud services.



The purpose of each stage can be summarized below, but we will look at each stage in more depth in the sections to follow:

Initiate and discover

Understand your database footprint and potential approaches to migration

Assess

Assess the discovered workload requirements and any dependencies

Plan

Plan and describe the workloads to be migrated, the tool to be used for migration and the target platform for the workload

Transform and optimize

Transform any workloads not currently compatible with modern data platforms. Optimize workloads to take advantage of new features

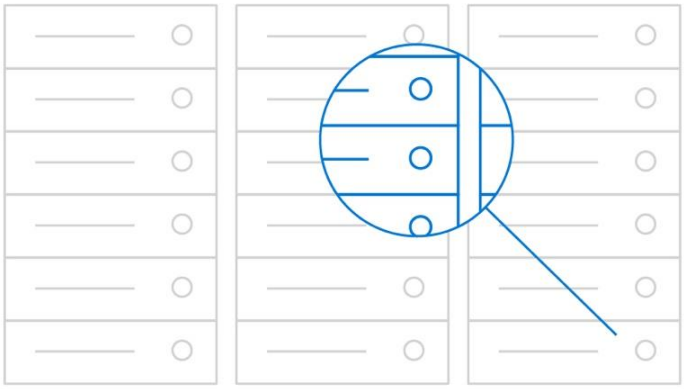
Migrate, validate and remediate

Perform migration, validate successful migration, and remediate applications where required

Tip: Don't forget, you don't need to perform all the work yourself.

Microsoft can provide assisted migrations through partners such as Movere and Cloudamize, who have intelligent cloud infrastructure analytics platforms available to automate most of the assessment, planning, migration, validation and on-going management of your cloud database deployments.

See https://www.cloudamize.com and https://www.movere.io for more information.



3 INITIATE AND DISCOVER

The first stage of the migration roadmap is Initiate and discover. In this first stage, the goal is to establish three things:

The inventory of your data estate

This constitutes what data is available, where it is located, what platforms it resides on and the size of the data.

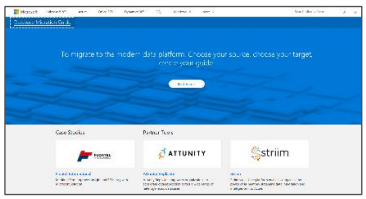
Application database dependencies

Applications will often utilize several databases or integrate with other applications that have their own databases. We need to know the database dependencies to other databases to logically group them together according to these relationships.

What databases move together

Once the logical groupings by relationship have been made, we can use them to form batches of databases for migrating up to Azure.

To achieve these goals, Microsoft has made available many resources and tools. These include:

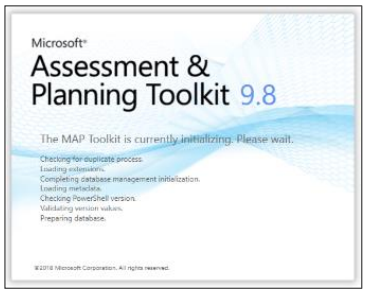


Database Migration Guide

The new Database Migration Guide is for enterprise customers, partners, and business decision makers who are interested in moving to Azure cloud services (i.e. migrating from Oracle or SQL Server to Azure Data Services).

The Database Migration Guide provides comprehensive, step-by-step guidance for performing migrations, as well as improves the discoverability of the guidance, tools, software, and programs that are available to assist customers in performing these migrations.

More info at https://datamigration.microsoft.com



Microsoft Assessment & Planning (MAP) Toolkit

The Microsoft Assessment & Planning (MAP) Toolkit makes it easy to assess your current IT infrastructure for a variety of technology migration projects. This solution accelerator provides a powerful inventory, assessment, and reporting tool to simplify the migration planning process. While not strictly aimed at database migrations, the MAP toolkit includes information on databases and server software that we can utilize in cloud migration planning.

More info at https://www.microsoft.com/en-us/download/details.aspx?id=7826

Data Migration Assistant (DMA)

Data Migration Assistant (DMA) enables you to upgrade to Azure data services by detecting compatibility issues that can impact database functionality on Azure SQL Database. It recommends performance and reliability improvements for your target environment. It allows you to not only move your schema and data, but also uncontained objects from your source server to your target service. DMA replaces the legacy SQL Server Upgrade Advisor tool and should be used for upgrades and migrations from most SQL Server versions.

More info at https://blogs.msdn.microsoft.com/datamigration/dma/

3.1 Microsoft tools and services: Database Migration Guide

The Database Migration Guide can be accessed via the Microsoft website at https://datamigration.microsoft.com/ and provides a centralized hub of data migration related information and resources.

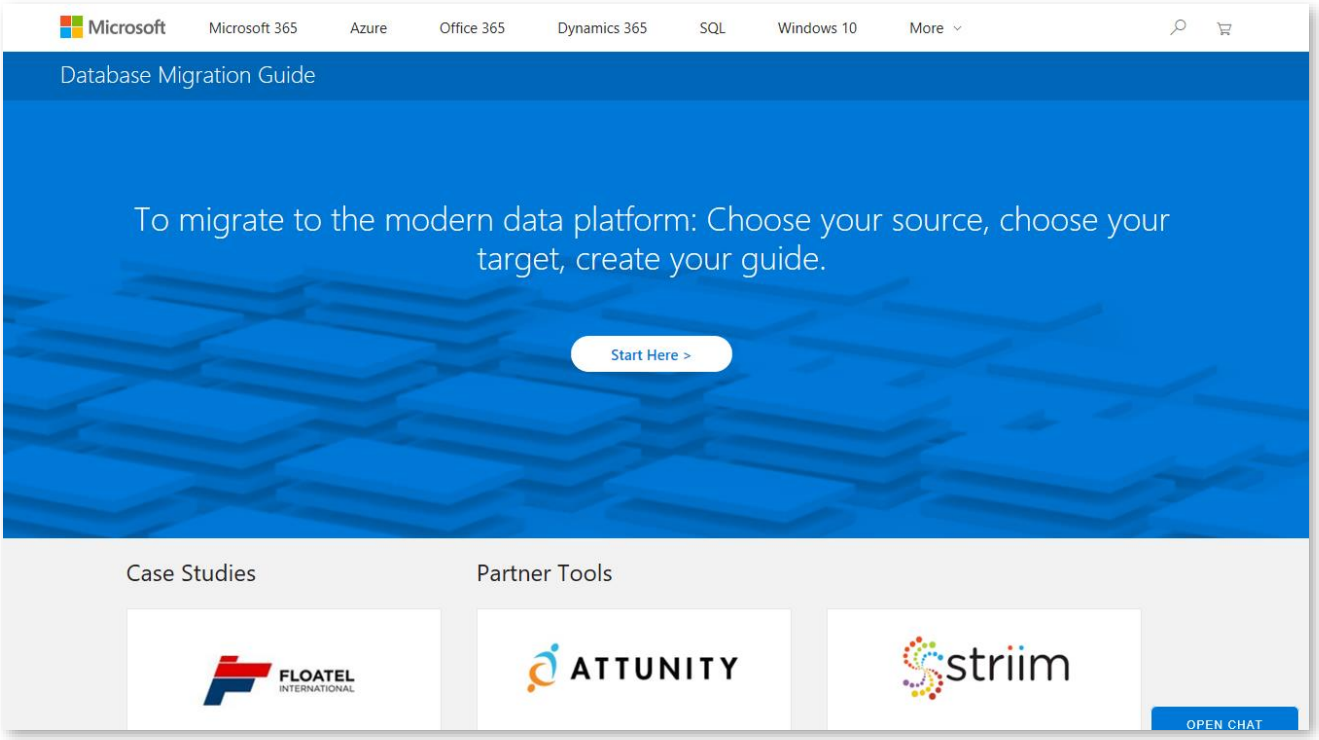


Figure 1 Database Migration Guide Introduction Page

You can create a comprehensive migration playbook for many database migration scenarios, customizing the playbook to suit your individual scenario by providing answers on where your data is currently stored and to where you wish to migrate your data.

The Database Migration Guide then compiles the information relevant to your scenario and presents it on screen as a document containing relevant migration-related information, discussion materials for making the correct design decisions, as well as links to numerous resources including best practice whitepapers, customer case studies, and training videos. This document can also be printed or emailed to a recipient for future reference.

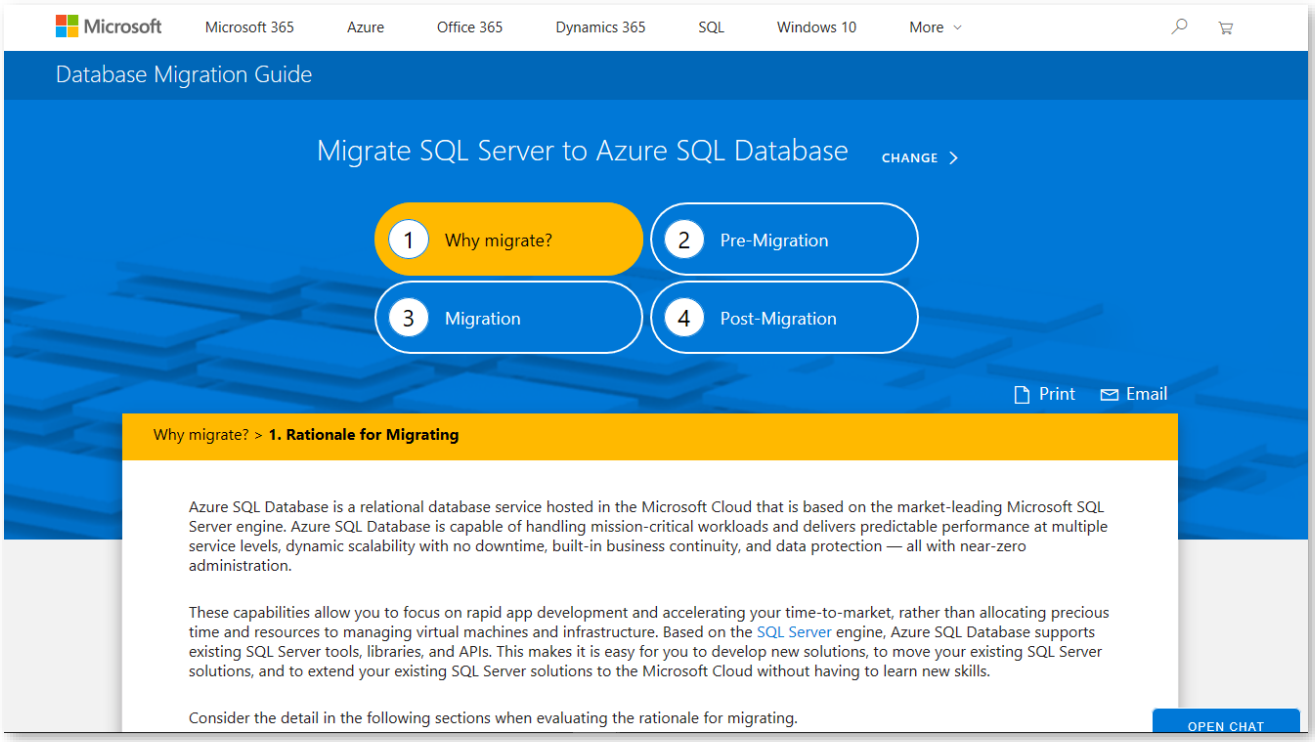


Figure 2 Database Migration Guide Completed Playbook

The Database Migration Guide caters not only to Microsoft SQL Server as the source platform, but also migrations from many other commercial and open-source platforms including Microsoft Access, Oracle, MySQL, PostgreSQL and MongoDB.

3.2 Microsoft tools and services: MAP Toolkit

The Microsoft Assessment & Planning (MAP) Toolkit has been updated throughout the years for several different types of SQL and application migrations with new capabilities to support later releases of Windows and application software. The MAP Toolkit is freely available to download from the Microsoft site. The toolkit's light resource requirements allow it to be installed and executed on either a server or a workstation.

The MAP Toolkit's purpose is to discover and inventory computers and applications on the network for assisting with upgrades and migrations.

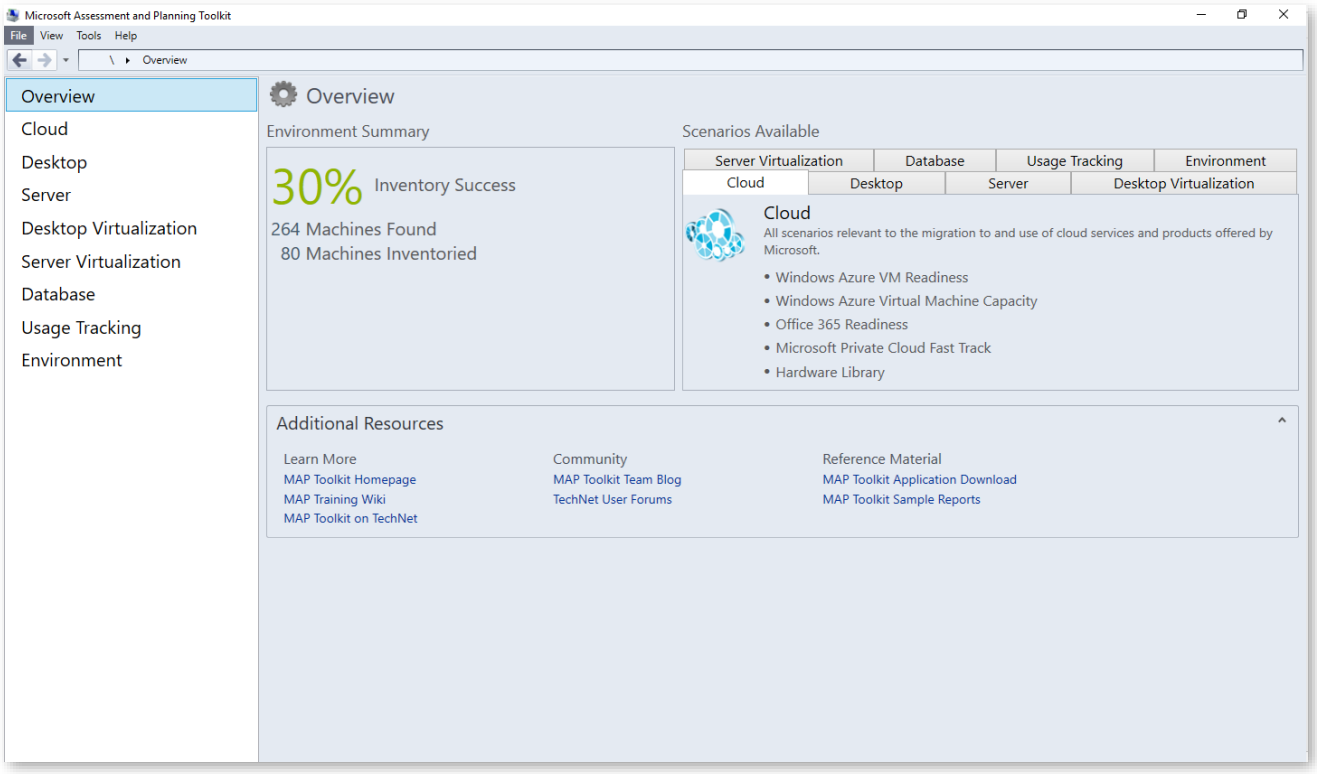


Figure 3 MAP Toolkit overview

The MAP Toolkit does this without requiring an agent to be installed on the discovered computers, instead discovering computers to be inventoried from Active Directory, System Center Configuration Manager (SCCM), scanning IP address ranges, or using a provided list of computer names. All collected information is stored in a locally installed SQL Server Express database, with no telemetry being sent to Microsoft.

Once a computer has been found, its hardware and software inventory is built using the sources inputted, including information gathered from Active Directory or queries using Windows Management Instrumentation (WMI), Remote Registry Service, and PowerShell. This causes minimal impact to the computers or network performing that tasks and is generally regarded as safe to do during business hours. For in-depth analysis of applications on the discovered computer, administrative rights are required on the target computer using a service account in Active Directory.

The information retrieved from Active Directory and discovered computers can be queried by the MAP Toolkit to produce reports and analysis. Microsoft provides out-of-the-box reports in MAP for many different scenarios including upgrade readiness for new versions of Microsoft operating systems and applications, but the reports generated around databases are most useful for database migration scenarios. For this, the Database tab can be used to show a categorization of each SQL Server version and the number of each version that exists in discovered environments.

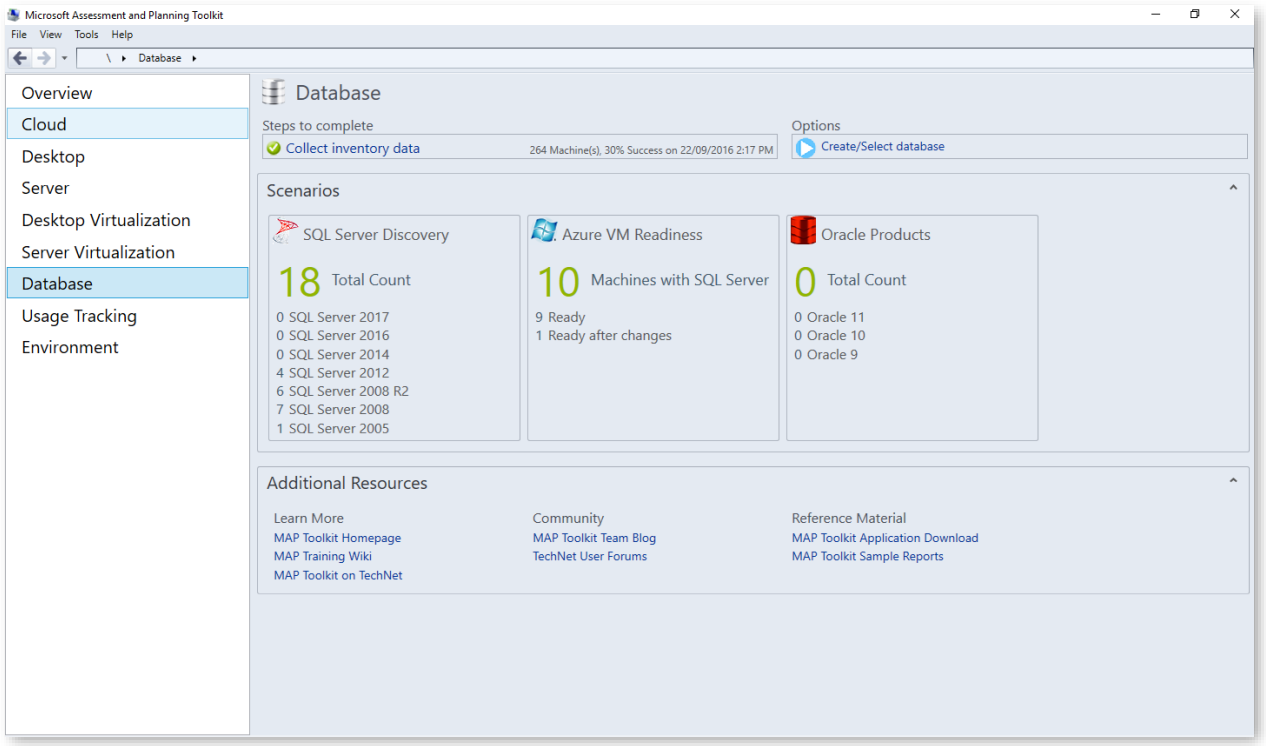
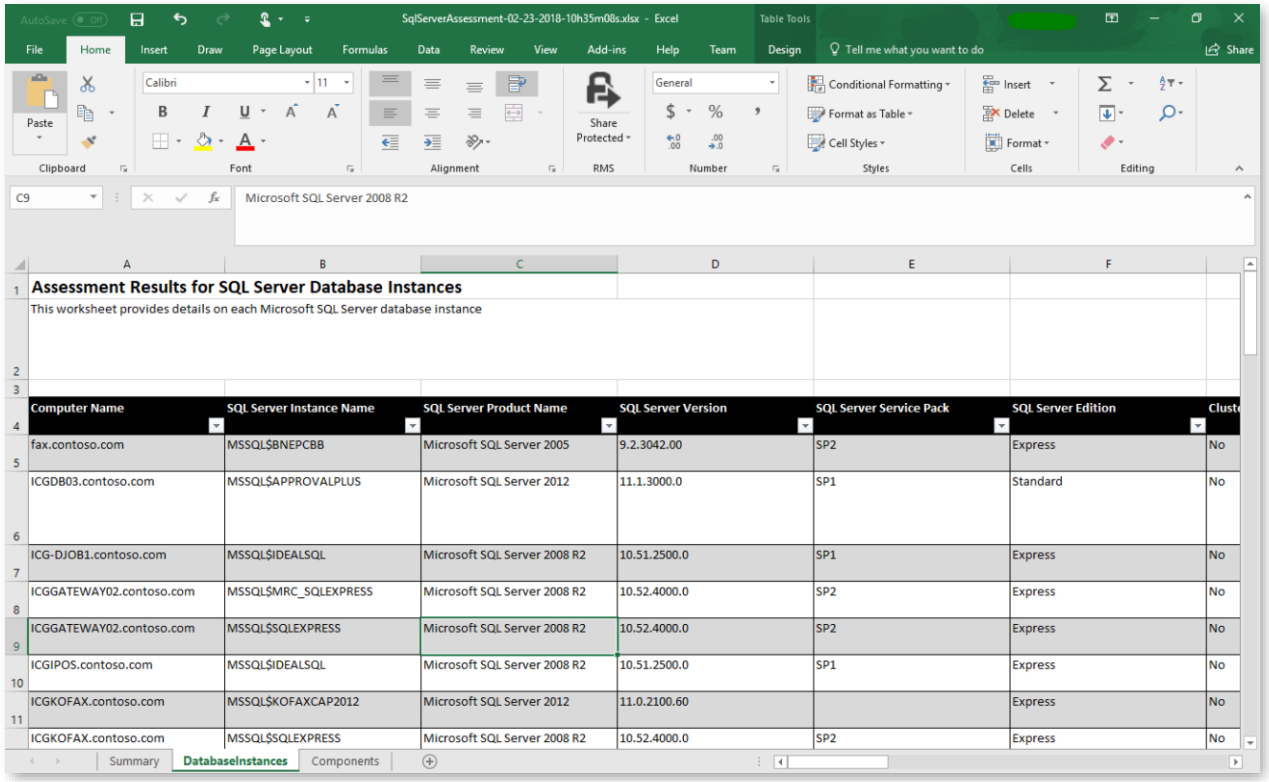


Figure 4 MAP Toolkit Database scenario

Clicking through SQL Server Discovery presents a further breakdown of not only the SQL Server versions found on inventoried computers, but also instances of SQL Server Reporting Services (SSRS), SQL Server Integration Services (SSIS) and SQL Server Analysis Services (SSAS) which might also form part of the overall migration.

Figure 5 MAP Toolkit SQL Server Discovery Scenario

Up in the top right are links to generate reports for SQL Server Assessment and SQL Server database details. Clicking these causes the MAP Toolkit to process the current inventoried data and produce a pre-canned report in the form of an Excel spreadsheet.



Assessment Results for SQL Server Database Instances

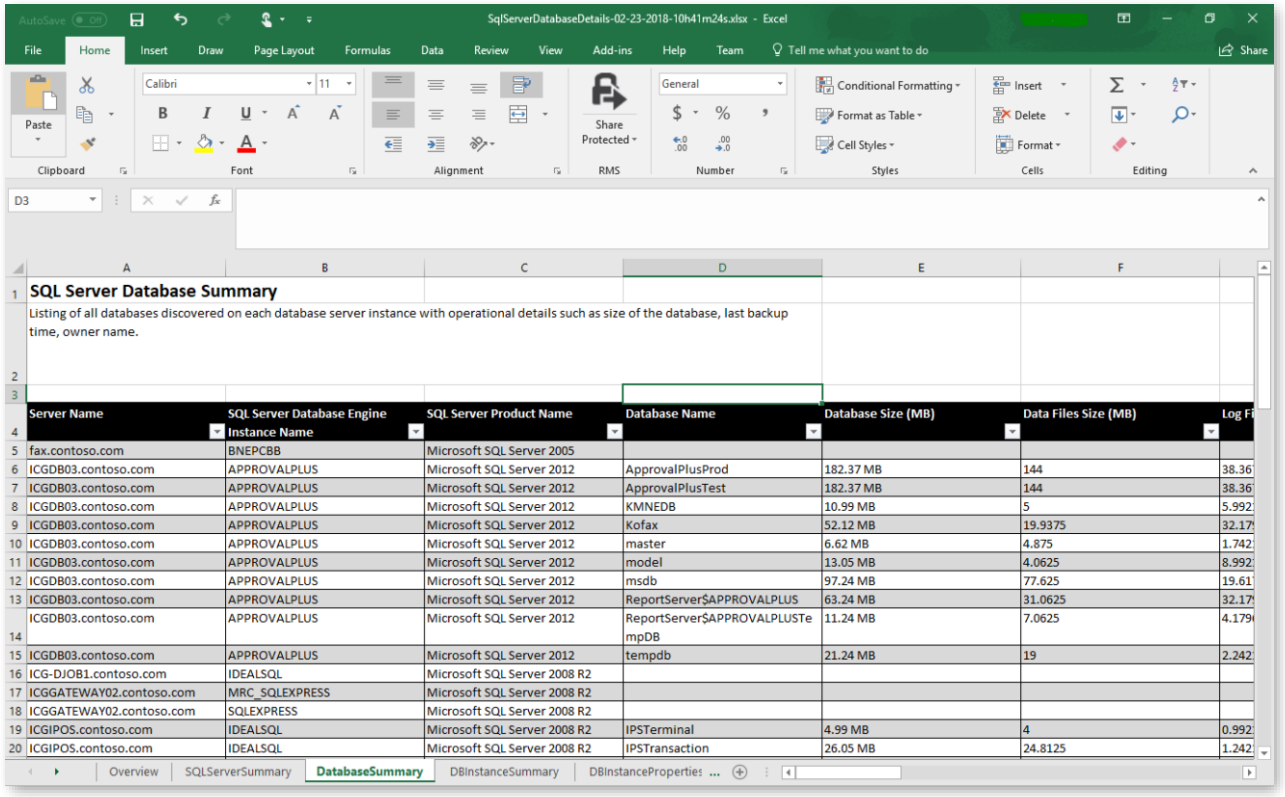
This worksheet provides details on each Microsoft SQL Server database instance

Computer Name	SQL Server Instance Name	SQL Server Product Name	SQL Server Version	SQL Server Service Pack	SQL Server Edition	Clust
fax.contoso.com	MSSQL$BNEPCBB	Microsoft SQL Server 2005	9.2.3042.00	SP2	Express	No
ICGDB03.contoso.com	MSSQL$APPROVALPLUS	Microsoft SQL Server 2012	11.1.3000.0	SP1	Standard	No
ICG-DJOB1.contoso.com	MSSQL$IDEALSQL	Microsoft SQL Server 2008 R2	10.51.2500.0	SP1	Express	No
ICGGATEWAY02.contoso.com	MSSQL$MRC_SQLEXPRESS	Microsoft SQL Server 2008 R2	10.52.4000.0	SP2	Express	No
ICGGATEWAY02.contoso.com	MSSQL$SQLEXPRESS	Microsoft SQL Server 2008 R2	10.52.4000.0	SP2	Express	No
ICGIPOS.contoso.com	MSSQL$IDEALSQL	Microsoft SQL Server 2008 R2	10.51.2500.0	SP1	Express	No
ICGKOFAX.contoso.com	MSSQL$KOFAXCAP2012	Microsoft SQL Server 2012	11.0.2100.60		Express	No
ICGKOFAX.contoso.com	MSSQL$SQLEXPRESS	Microsoft SQL Server 2008 R2	10.52.4000.0	SP2	Express	No

Figure 6 Example MAP Toolkit SQL Server Assessment Report

The **SQL Server Assessment Report** provides a good overview of the discovered SQL Server instances, the version and edition of each, the current service pack level, whether it is clustered, what language it is set to use, and many others. It also details the server that SQL Server is running on including how many processors are allocated, the assigned system memory, the number and size of logical disks, free disk space and whether the server is physical or virtual.

The more useful report for the database migration scenario is the SQL Server Database Details Report, which provides full details on all the SQL Server instances found, the names of databases housed on those SQL Server instances, their current sizing, as well as statistics on number of tables, views and stored procedures within those databases.



SQL Server Database Summary

Listing of all databases discovered on each database server instance with operational details such as size of the database, last backup time, owner name.

Server Name	SQL Server Database Engine Instance Name	SQL Server Product Name	Database Name	Database Size (MB)	Data Files Size (MB)	Log Fi
fax.contoso.com	BNEPCBB	Microsoft SQL Server 2005				
ICGDB03.contoso.com	APPROVALPLUS	Microsoft SQL Server 2012	ApprovalPlusProd	182.37 MB	144	38.36
ICGDB03.contoso.com	APPROVALPLUS	Microsoft SQL Server 2012	ApprovalPlusTest	182.37 MB	144	38.36
ICGDB03.contoso.com	APPROVALPLUS	Microsoft SQL Server 2012	KMNEDB	10.99 MB	5	5.992
ICGDB03.contoso.com	APPROVALPLUS	Microsoft SQL Server 2012	Kofax	52.12 MB	19.9375	32.17
ICGDB03.contoso.com	APPROVALPLUS	Microsoft SQL Server 2012	master	6.62 MB	4.875	1.742
ICGDB03.contoso.com	APPROVALPLUS	Microsoft SQL Server 2012	model	13.05 MB	4.0625	8.992
ICGDB03.contoso.com	APPROVALPLUS	Microsoft SQL Server 2012	msdb	97.24 MB	77.625	19.61
ICGDB03.contoso.com	APPROVALPLUS	Microsoft SQL Server 2012	ReportServer$APPROVALPLUS	63.24 MB	31.0625	32.17
ICGDB03.contoso.com	APPROVALPLUS	Microsoft SQL Server 2012	ReportServer$APPROVALPLUSTempDB	11.24 MB	7.0625	4.179
ICGDB03.contoso.com	APPROVALPLUS	Microsoft SQL Server 2012	tempdb	21.24 MB	19	2.242
ICG-DJOB1.contoso.com	IDEALSQL	Microsoft SQL Server 2008 R2				
ICGGATEWAY02.contoso.com	MRC_SQLEXPRESS	Microsoft SQL Server 2008 R2				
ICGGATEWAY02.contoso.com	SQLEXPRESS	Microsoft SQL Server 2008 R2				
ICGIPOS.contoso.com	IDEALSQL	Microsoft SQL Server 2008 R2	IPSTerminal	4.99 MB	4	0.992
ICGIPOS.contoso.com	IDEALSQL	Microsoft SQL Server 2008 R2	IPSTransaction	26.05 MB	24.8125	1.242

Figure 7 Example MAP Toolkit SQL Server Database Details Report

The MAP Toolkit also gathers performance metrics from computers which can be helpful to size virtual machines or Azure SQL Databases.

Using all the data collected from the MAP Toolkit, you can generate a comprehensive list of databases and workloads that are in your environment and from it you can then create a short-list of databases and workloads that you wish to go ahead and assess for suitability to migrate to Azure data services.

Download the MAP Toolkit from the following URL: https://www.microsoft.com/en-us/download/details.aspx?id=7826

3.3 Microsoft Tools and Services: Data Migration Assistant

After using the MAP Toolkit to help generate a shortlist of databases and workloads, likely migration candidates can be fed into a tool called Microsoft Data Migration Assistant (DMA) for comprehensive assessment.

DMA is another freely available download from the Microsoft website to help with the migration of on-premises SQL Server instances to Azure SQL Database or to a modern SQL Server instance hosted on an Azure Virtual Machine. Usually run locally on your workstation, DMA replaces the legacy SQL Server Upgrade Advisor tool and has been fully extended to support cloud platforms as eligible targets.

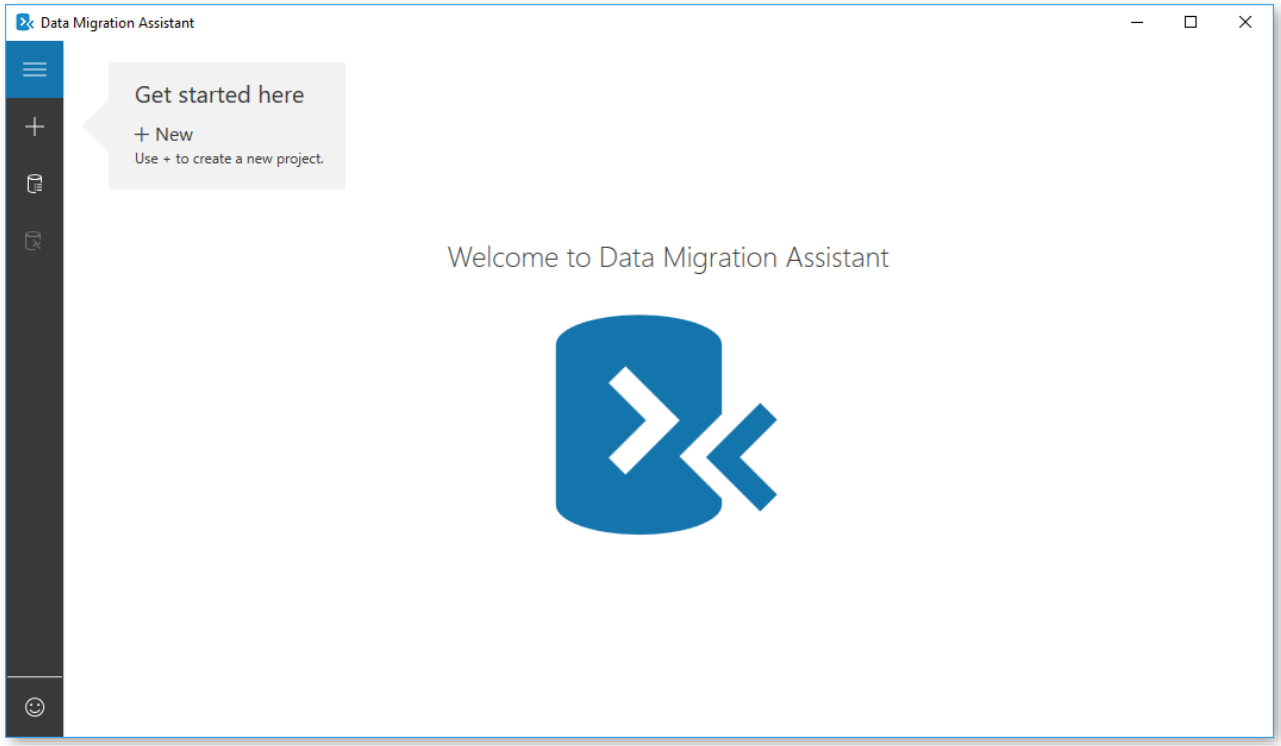


Figure 8 Data Migration Assistant

DMA will be explored in depth in the following section, but the following is a brief overview of what DMA offers. DMA allows you to define projects for data assessment or for data migration. For both types, you define the required source and target types as you create the project.

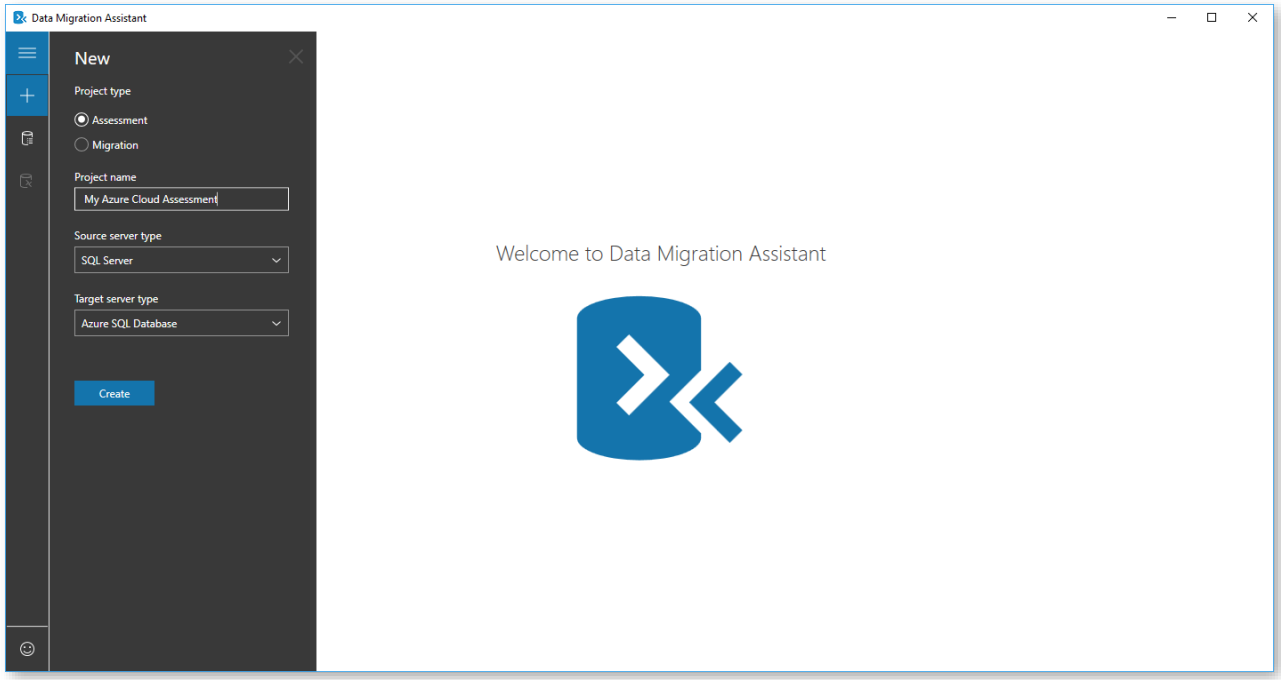


Figure 9 Creating New DMA Assessment Project

The assessment project uses DMA's assessment workflow to help you detect issues that can affect Azure SQL Database migration and then provides detailed guidance on how to resolve them.

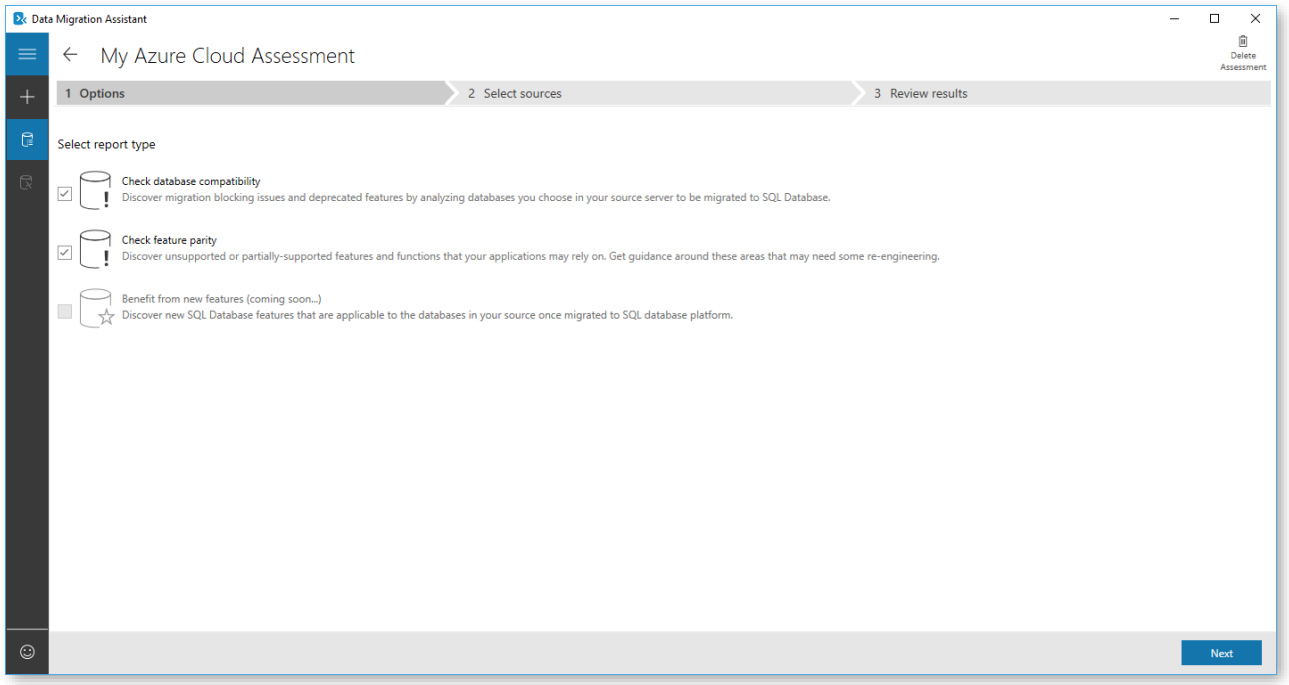
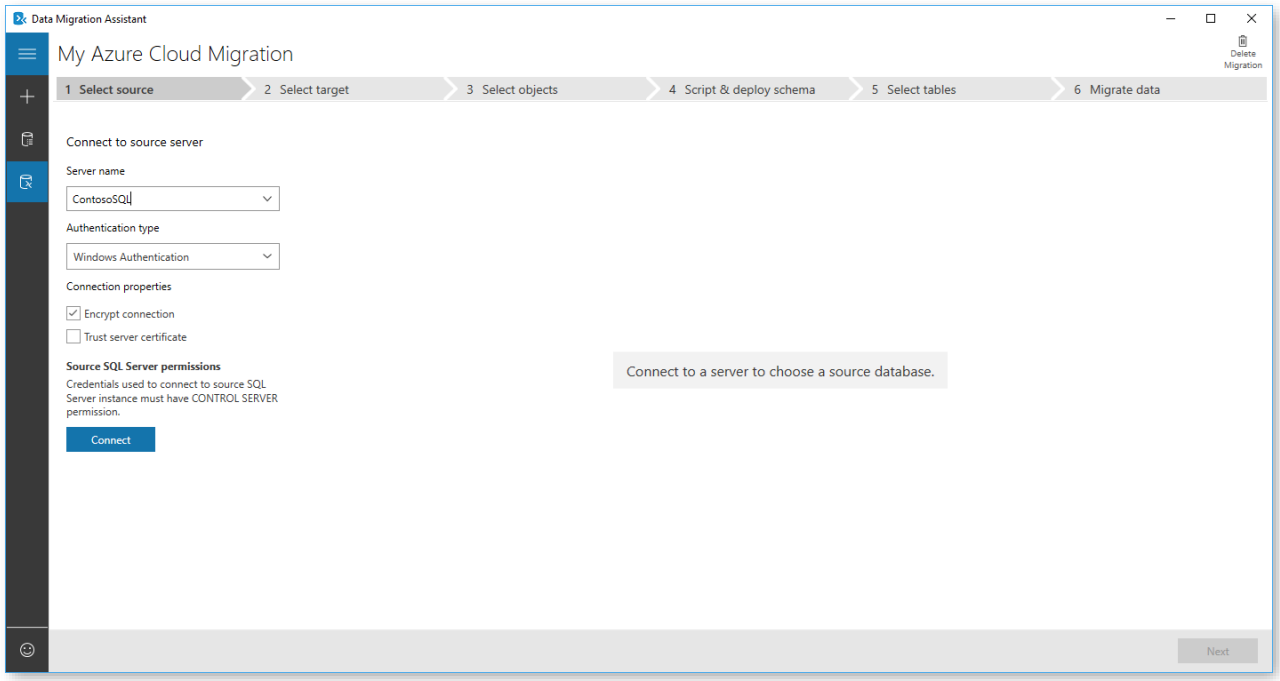


Figure 10 Assessing for database compatibility with DMA

These might include migration blocking issues such as compatibility problems that prevent successful migration from an on-premises SQL Server database to Azure SQL Database, or the highlighting of partially or unsupported features that are currently in use at the source SQL Server. It then provides recommendations to help remediate those issues as well as alternate approaches for migration.

The Migration project will use DMA's migration workflow to help you to migrate selected data from source to target, handling the copying process between the two entities.



Furthermore, DMA will soon be able to create benefits by indicating new features in the target SQL Server platform that the database can benefit from after an upgrade. These could help improve the database solution in the areas of performance, security or storage. This feature of DMA is expected shortly.

By making use of the MAP Toolkit we found all our database assets as well as their characteristics such as size and hosting server. DMA is a tool that can be fed databases found using the MAP Toolkit for further analysis and assessment. We will cover assessment using DMA in more detail in the following section.

4 ASSESSMENT

We now know what workloads we're dealing with, where they are, how big they are and what they're used for. This data obtained from the initiate and discover phase can now be used as input into the second phase, assessment. The data will need to be compiled and analyzed to achieve our goals for this phase, which is to identify:

The migration blockers

A migration will not be able to proceed until these issues are resolved.

Breaking changes

A migration will be able to proceed but the workload will need to be fixed post-migration to be functional.

Features to leverage

Available Azure features that when utilized can maximize the benefit of migrating to Azure services.

Effort involved to fix issues

An estimate of the time and processes required to rectify the above highlighted issues.

To realize these goals, a closer examination of workloads is done with emphasis placed on the following areas:

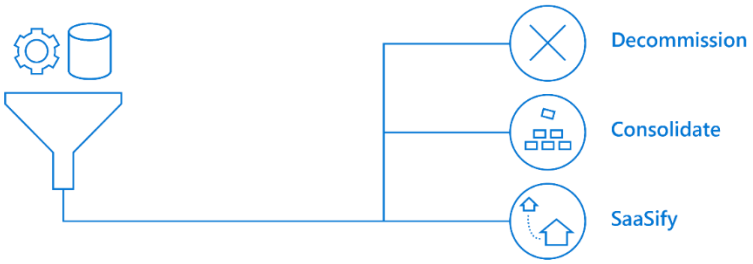
- Assess workloads for migration
- Assess workload criteria
- Assess database using Data Migration Assistant

4.1 Assess workloads for migration

To establish a complete migration plan, a thorough assessment of your workloads prior to migration will help determine which databases will need to be migrated to the cloud as well as the quantities involved.

If intending to migrate all on-premises workloads to Azure cloud services, an initial assessment pass with the intent of consolidating or decommissioning legacy workloads where possible can help to reduce the final number of database workloads needing to be migrated.

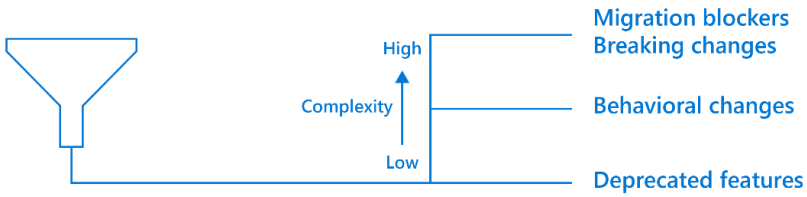
Investigations should be made into whether on-premises applications in use now have a SaaS-based or hosted deployment model available, and if so consider moving to that platform to lower administrative costs.



Proceed down the cloud migration path by looking to migrate any low effort, high impact databases. That is, look to segregate the discovered workloads according to their business impact.

Workloads for applications used by a select number of users in the enterprise should have a smaller scope for disruption than applications used widely across the business. Non-critical workloads such as development, testing and training platforms would make good candidates for the first wave of migrations.

Next, workloads can be further ranked by the severity of issues highlighted during the initiate and discover phase. Migration blockers or known breaking changes might require substantial remediation work, and position workloads well down the migration list.



Similarly, behavioral changes might mean that some workloads need additional investigation and planning before they can make the transition to the cloud to fully understand any impacts. Any workloads making use of deprecated features should still be migratable but warrant investigation later to remove their dependency on those deprecated features.

Using continuous migrations

Critical applications used within the business may not be able to afford the measurable downtime windows needed to migrate workloads to the cloud. They may also be dependent on other workloads which will need to be moved in the same grouping, the size of the data across all grouped applications dictating the length of time required for an outage window while the data copies across.

By making use of continuous migration methodologies, applications can continue to use the source database while the bulk of the data is synchronized to the cloud in the background. Any data changed during the migration process is replicated on the fly to the target platform, ensuring all data transactions are retained.

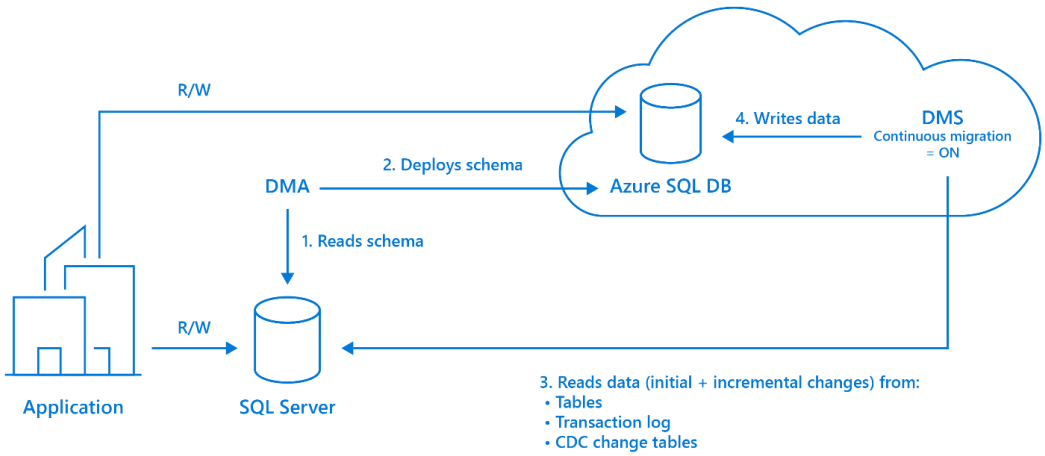


Figure 11 Continuous migrations workflow using DMA and DMS

This methodology affords a greatly reduced overall downtime as downtime is limited to the time taken to complete the final step of repointing the consuming application to the target database.

Tip: Microsoft has partnered with Attunity to provide their Attunity Replicate for Microsoft Migrations product to Microsoft customers for no additional cost.

Attunity Replicate continuously migrates databases from many commercial and open-source platforms, including Oracle, PostgreSQL, MySQL, and SAP Sybase ASE to the Microsoft data platform with virtually no downtime. The source systems stay operational during the migration process and any data changes in the source databases are continuously replicated to your target database, so you are always working with real-time data.

For additional information see: https://aka.ms/attunity-replicate

4.2 Assess workload criteria

Performance requirements

It is important to understand if each workload is a high or low user of resources, and gauge how many Azure resources will be required post-migration. If looking to transition to SQL Server on Azure IaaS VMs, this might simply amount to matching the number of compute cores currently allocated to those on the target platform. If moving to Azure SQL Databases this might require computing the number of Database Transaction Units (DTU) or virtual cores (vCores) needed for each database. Azure SQL Database provides two different models for measuring and purchasing compute: DTU-based and vCore-based.

What are Database Transaction Units (DTUs)?

Azure SQL Database performance is measured using the Database Transaction Unit or DTU, which is an aggregated metric of CPU, memory and I/O. DTU's are useful for understanding the relative amount of resources allocated between different Azure SQL Databases as varying available performance levels are characterized by their allocated DTUs. For example, the basic performance level has a maximum DTU count of 5, whereas Standard performance levels start at a maximum DTU count of 10 for an 'S0' instance and rise to a maximum of 3000 for an 'S12' instance. P15 is the highest available performance level, where up to 4000 DTUs can be achieved. The DTU-based model provides simplicity for those who want a pre-configured solution.

Tip: Justin Henriksen has created a useful free tool called the Azure SQL Database DTU Calculator which can help you determine the number of DTUs for your existing SQL Server database(s) as well as provide a recommendation of the minimum performance level and service tier that you need before you migrate to Azure SQL Database. The Azure SQL Database DTU Calculator also supports calculating requirements for elastic pools.

For additional information see: http://dtucalculator.azurewebsites.net/

What are vCores?

A virtual core (vCore) represents the logical CPU offered with a choice between generations of hardware. Gen 4 logical CPUs are based on Intel E5-2673 v3 (Haswell) 2.4 GHz processors and Gen 5 logical CPUs are based on Intel E5-2673 v4 (Broadwell) 2.3 GHz processors. The vCore-based model provides additional choice and flexibility for those who want to optimize their workloads in the cloud by allowing compute, storage and I/O to be configured independently. For example, Azure SQL Database Managed Instance allows you to choose from 8, 16 or 24 vCore instances and up to 8TB of storage.

Compliance requirements

Determine if there are any specific security or regulatory requirements. Microsoft's Trusted Cloud initiative is built around the four foundational principles of security, privacy, compliance and transparency which is reflected in the platforms and services offered through Azure. Azure data centers comply with strict regulations and compliance standards, to help customers meet international data protection laws and industry requirements. Data residency laws might also mean that data for a given application must be kept in country or geographic region, restricting which Azure data centers can be utilized.

You can learn more about Azure compliance practices at the Azure Trust Center: https://azure.microsoft.com/support/trust-center/compliance/.

Migration downtime

Understand the business requirements around the workload to be migrated. Is any downtime acceptable? This will impact the migration approach, toolsets used, and timeframes involved.

Availability

Following on from migration downtime, what are the ongoing availability requirements for the workload? Azure SQL Databases are locally highly-available as standard, with three copies of your database used to keep the data online and accessible during patching and transient hard failures. SQL Server on Azure VMs would require HA technologies such as Always On Failover Clustering, Always On availability groups, database mirroring or log shipping.

Disaster recovery

Establish if there are disaster recovery requirements for the application workloads supported by the database and understand RTO and RPO requirements. Implementing disaster recovery on Azure SQL Database needs just a few clicks to establish a database replica out-of-region for minimal cost, the geo-replication feature protecting your database and application against wider regional failures. SQL Server on Azure IaaS VM doesn't have readily available DR support and might require implementing SQL Server Enterprise Edition using Always On Availability Groups to meet aggressive RTO requirements for mission critical workloads. Lower priority workloads using Azure Site Recovery would normally suffice where protection at the virtual server level is acceptable.

Custom workloads

There may be databases that have 3rd party tool integrations which are not currently supported on Azure SQL Database. The 3rd party vendor might need to be approached for a compatible version or alternate products considered.

Azure has a target platform for pretty much any database workload. Understanding the source criteria is key to determining where and how the workload should land.

4.3 Database assessment using Database Migration Assistant (DMA)

As previously mentioned, Data Migration Assistant (DMA) is a freely downloadable tool from Microsoft that is installed and executed locally. DMA enables you to upgrade to a modern data platform by detecting compatibility issues that can impact database functionality before attempting to migrate to a new version of SQL Server or on to Azure SQL Database. It also provides recommendations on how to remediate those issues.

- Assess on-premises SQL Server instances migrating to Azure SQL Database
- Discover issues that can affect an upgrade to an on-premises SQL Server
- Discover new features in the target SQL Server platform that the database can benefit from after an upgrade
- Migrate an on-premises SQL Server instance to a modern SQL Server instance

An overview of DMA is available at: https://docs.microsoft.com/en-us/sql/dma/dma-overview

When using DMA, it should first assess and identify issues in the source database that would prevent a successful migration. Armed with this information, you must fix the root cause or implement an alternate methodology for each highlighted issue. The assessment and fix processes are then repeated until the source database passes all DMA tests, at which point the schema of the source database can be deployed to the target database in the cloud with a high degree of confidence.

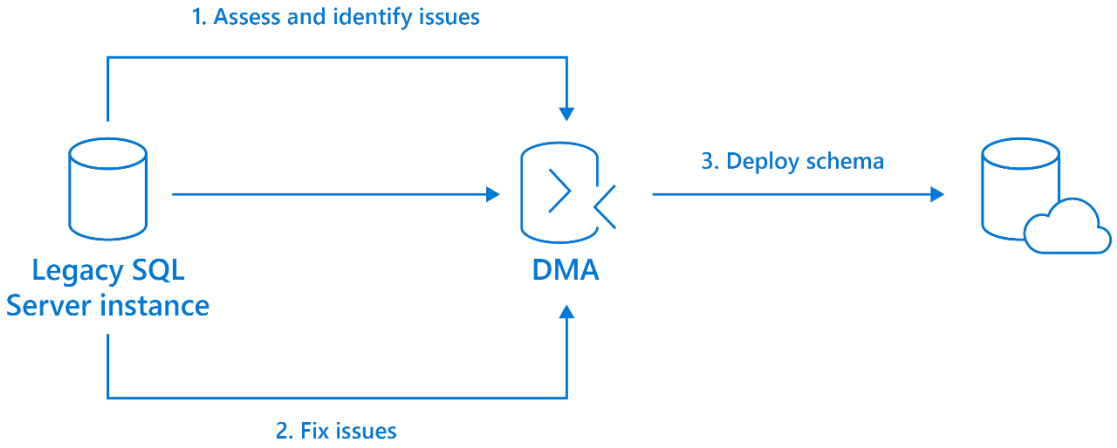


Figure 12 Assessment and fix workflow using DMA

4.4 Assessment steps using DMA

To use DMA to create an assessment, perform the following steps.

1. **Download** DMA, and then install it.
2. Create a **New Assessment** project.
   a. Select the New (+) icon, select the **Assessment** project type, specify a project name, select **SQL Server** as the source and **Azure SQL Database** as the target, and then select **Create**.

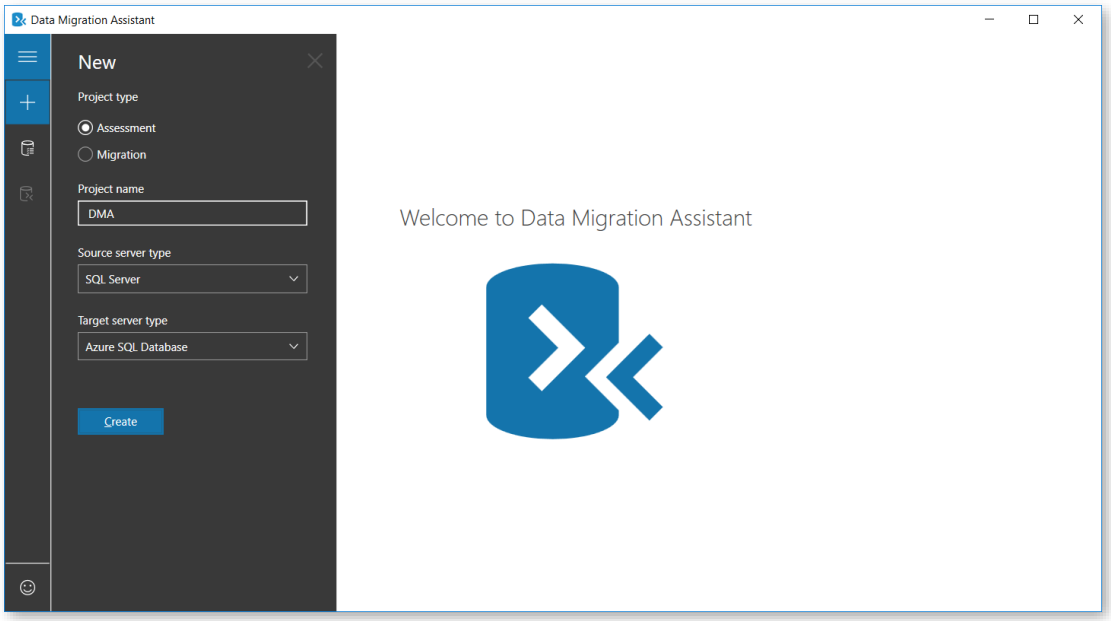


Figure 13 Creating a New DMA Project

   b. Select one or both assessment report types (**Check database compatibility** and **Check feature parity**), and then select **Next**.

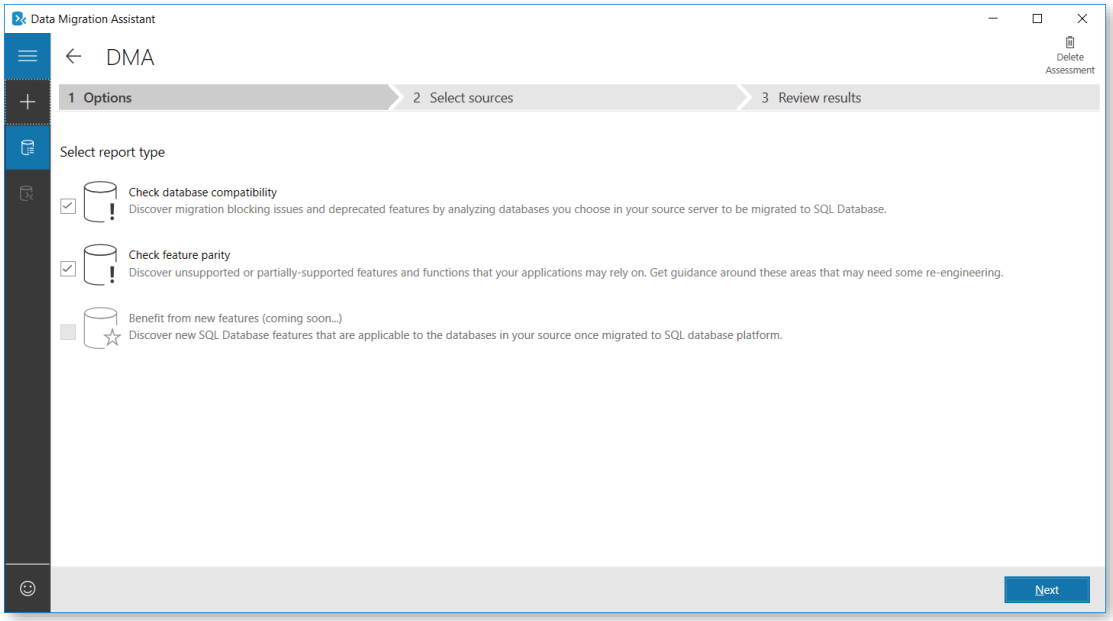


Figure 14 DMA assessment options

c. In the connect to a server blade, specify the name of the SQL Server instance to connect to, specify the authentication type and connection properties, and then select Connect.

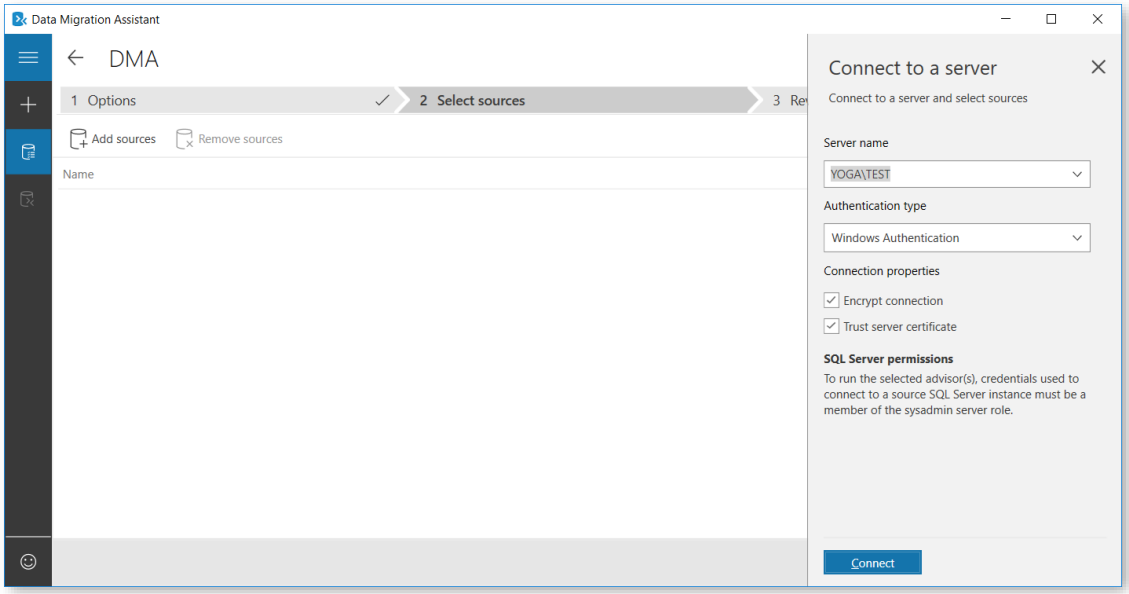


Figure 15 Connect to Source for DMA Assessment

d. In the Add sources fly-out, select the database(s) that you want to assess, and then select Add.

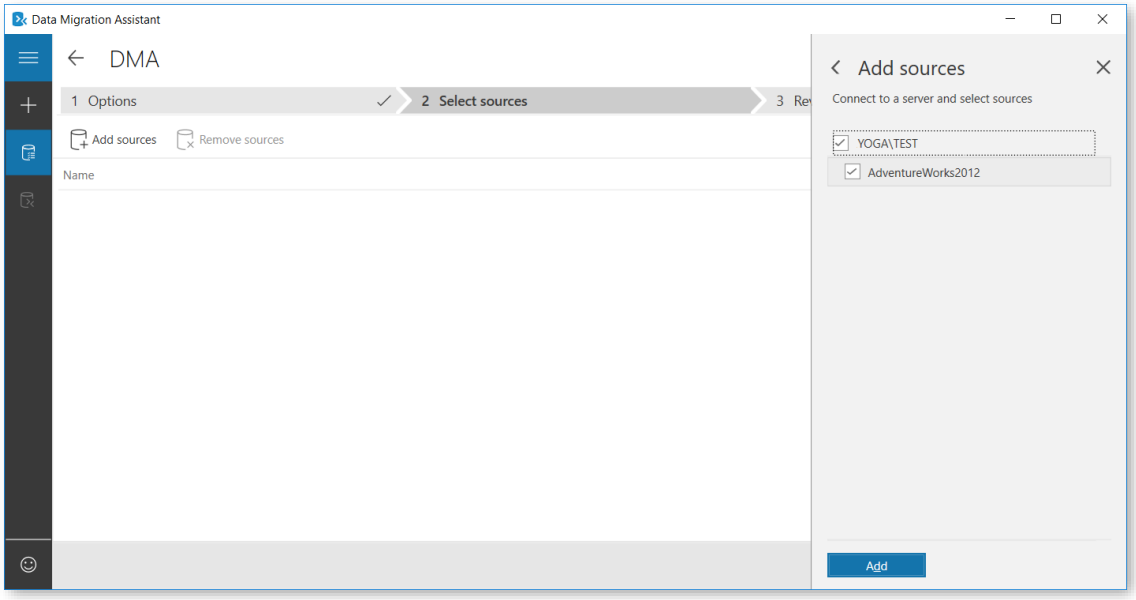


Figure 16 Source Selection in DMA

e. Select Start Assessment.

Now wait for the assessment results; the duration of the assessment depends on the number of databases added and the schema size of each database. Results will be displayed per database as soon as they are available.

f. Select the database that has completed assessment, and then select Compatibility issues to review incompatible objects categorized under Migration blockers, Behavior changes, and Deprecated features.

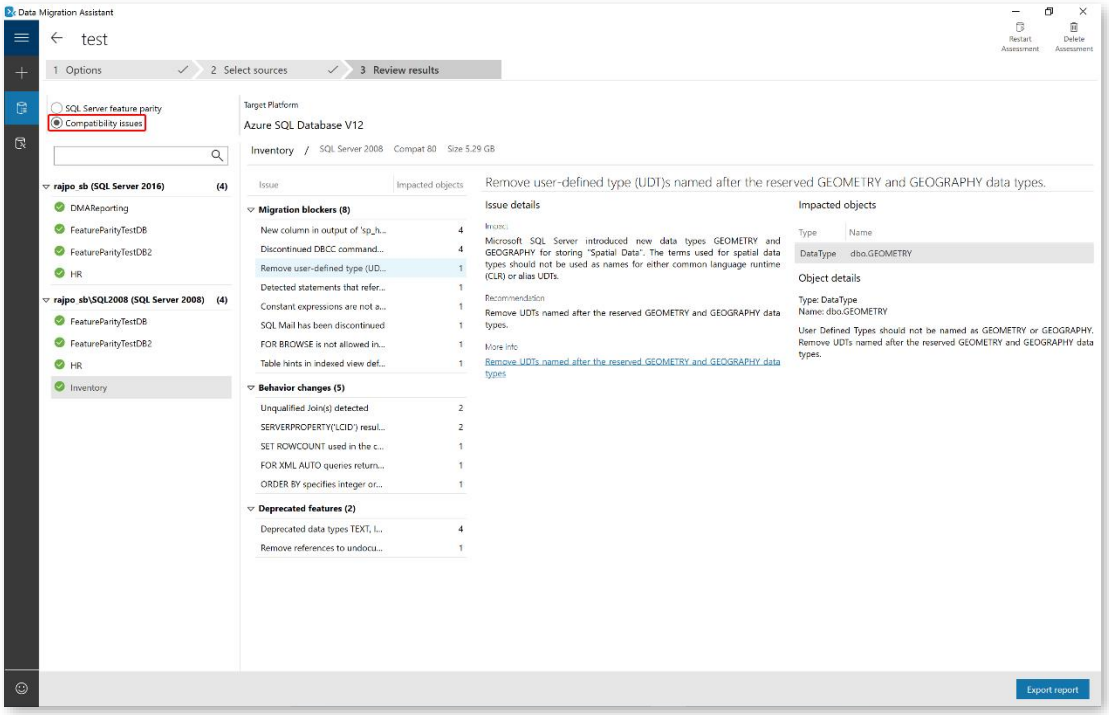


Figure 17 DMA compatibility findings and recommendations

Similarly, you can review recommendations across Performance, Storage, and Security areas. Feature recommendations cover a variety of features such as In-Memory OLTP and Columnstore, Stretch Database, Always Encrypted, Dynamic Data Masking, and Transparent Data Encryption.

g. Select SQL Server feature parity to review the unsupported features or partially supported features in Azure SQL Database.

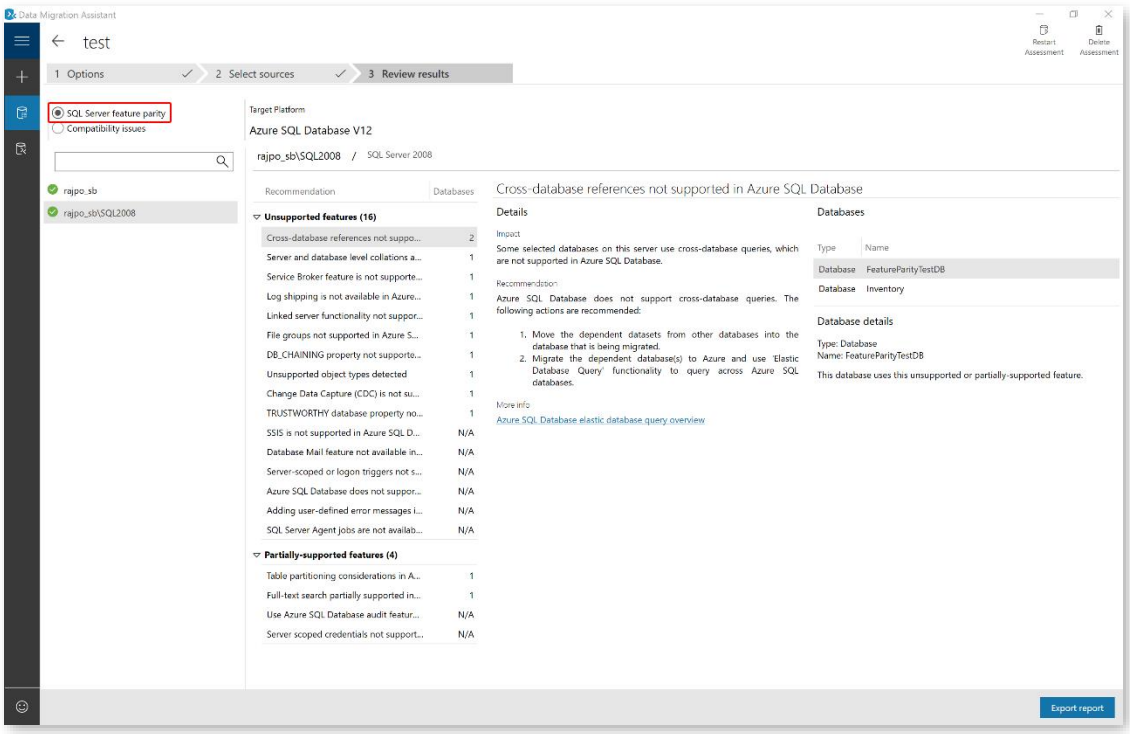


Figure 18 DMA Feature Parity Results and Findings

DMA also provides a comprehensive set of recommendations, alternative approaches available in Azure, and mitigating steps.

3. Review assessment results
   a. After all database assessments are complete, select Export report to export the results to either JSON or CSV file to analyze the data at your convenience.

4.5 Look for high level red flags

The findings surfaced during the initiate and discover phase and the assessment tools used during the assessment phase should now be considered. Possible options to work around identified issues need to be identified or flagged as potential migration blockers if a workable solution cannot be found.

Are you using features such as Database Mail, SQL Agent? Such features are available in Azure SQL Database Managed Instance, which is highly compatible with on-premises SQL Server.

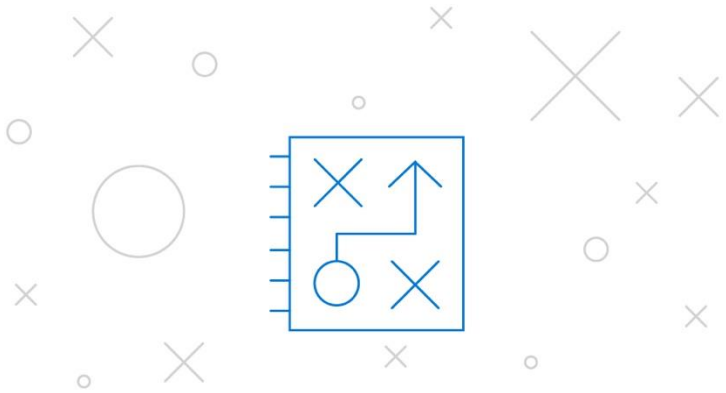
If the database is not using advanced SQL Server features such as MSDTC, MDS or QTS, then Azure SQL Database or Azure SQL Database elastic pools would be a good choice as Microsoft Operations takes care of the majority of infrastructure management drastically reducing administrative overhead costs.

Are you looking to also migrate SSRS, SSAS or SSIS? Unfortunately, not all SQL Server components currently have an Azure data services equivalent.

SSRS currently has no direct cloud-based equivalent, but reports could be rewritten based around Microsoft Power BI. Alternatively, SSRS can be deployed using SQL Server on an Azure VM.

SSAS can be migrated to Azure Analysis Services which is largely compatible with recent versions of SQL Server Analysis Services Enterprise Edition. Alternatively, SSAS can be deployed using SQL Server on an Azure VM.

SSIS packages can be invoked using stored procedures in Azure Data Factory. Alternatively, SSIS can be deployed using SQL Server on an Azure VM.



5 PLAN

The third stage of the migration roadmap is plan. In this most important stage, the goal is to establish two key things for each workload:

Target platform

This is the final location for each workload.

One-time migration versus continuous sync

A one-time migration means that a workload can be taken offline, whereas a continuous sync means that the workload source database needs to be available during the migration.

The following section will guide you through making these decisions and help formulate a plan of action for each workload, using the following topics:

- Plan the target platform
- How to choose the right target platform
- Migrating SSAS, SSIS and SSRS to Azure
- Plan the migration tool
- Target platform selection examples

5.1 Plan target platform

After completing assessment on the source environment and understanding your workload requirements, then you can pick your target location:

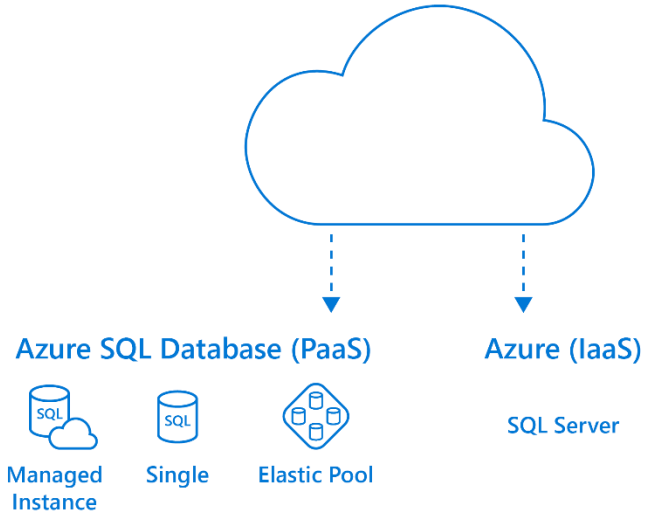


Figure 19 Available Azure Database Platforms

Azure SQL Database

Azure SQL Database is a fully managed service offering in Azure with which you can create a database with most of the functionality of running your own SQL Server in a virtual machine, but without having to worry about operating the virtual machine part of it. The associated maintenance and administration overhead goes away as Microsoft Operations takes care of all the underlying operating system and application for you.

Azure SQL Database offers three service tiers within its DTU-based model to support lightweight to heavyweight database workloads:

- Basic
- Standard
- Premium

The service tier affects the specification and characteristics of your database, involving size, performance level, availability, and concurrency. The tier of service selected dictates the limits on achievable performance, measured in Database Transaction Units (DTUs), as well as the database sizing.

- Basic: Best suited for small databases particularly those in development phases. Limited to 2GB in size and are allocated only limited compute resources.
- Standard: Best for general purpose databases with moderate performance requirements, so will likely constitute the bulk of your Azure SQL Databases. Supports databases with sizing up to 250GB.
- Premium: Designed for mission critical databases which have high performance and high availability requirements. The premium tier has low latency and can support high input/output workloads as well as databases with sizing up to 4TB.

You can build your first app on a small, single database at a low cost per month and then change its service tier manually or programmatically at any time to meet the needs of your solution. You can adjust performance without

downtime to your app or to your customers. Dynamic scalability enables your database to transparently respond to rapidly changing resource requirements and enables you to only pay for the resources that you need when you need them.

Azure SQL Database offers two service tiers within its vCore-based model:

- General Purpose
- Business Critical

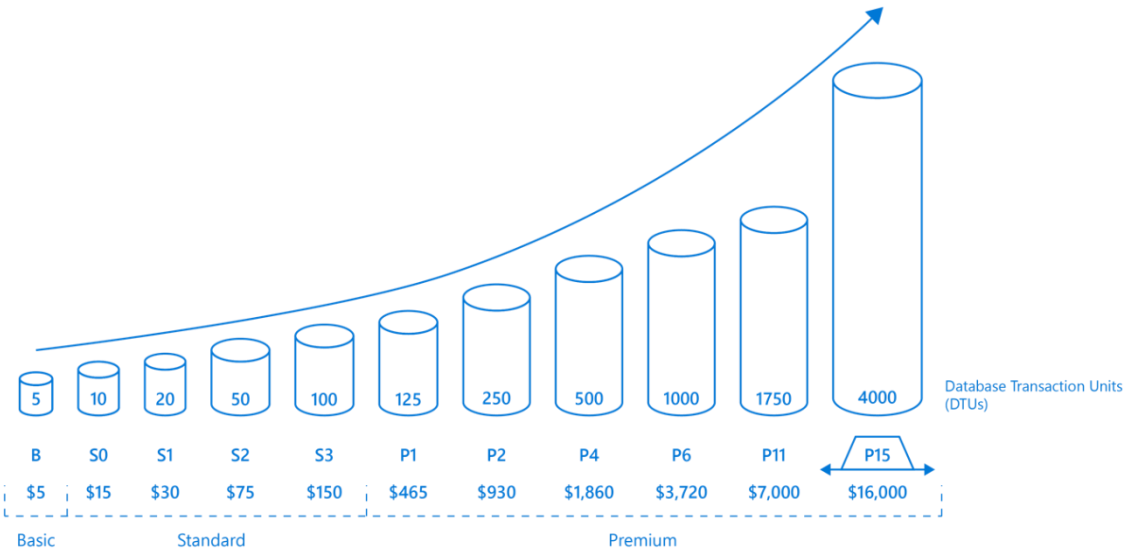
For existing SQL Database applications using the DTU model, the General Purpose service tier is comparable to Standard edition. The Business Critical service tier is comparable to Premium edition. The vCore-based service tiers provide flexibility through independent control over compute and storage configurations so that you can optimize it to exactly what the application requires and pay accordingly.

- General Purpose: Most business workloads. Offers budget oriented balanced and scalable compute and storage options.
- Business Critical: Best for business applications with high IO requirements. Offers highest resilience to failures using several isolated replicas.

Why Use Azure SQL Database?

Azure SQL Database delivers predictable performance at multiple service levels that provides:

- SQL Server engine compatibility and native virtual network (VNET) support
- Dynamic scalability with no downtime
- Built-in intelligent optimization, global scalability and availability, and advanced security options
- Eliminates hardware costs and reduces administrative costs
- Built-in fault tolerance infrastructure capabilities, Azure SQL Database provides features, such as automated backups, Point-In-Time Restore, geo-restore, and active geo-replication to increase business continuity for applications hosting data in Azure SQL Database
- Databases of up to 4 TB or larger databases that can be horizontally or vertically partitioned using a scale-out pattern

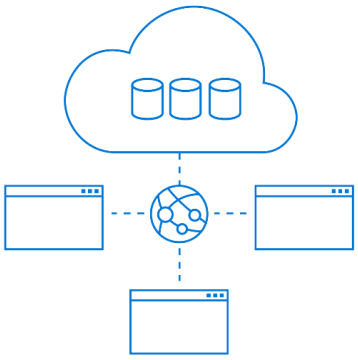


Microsoft Azure SQL Database can change between service tiers, so that you can easily allocate more resources and capacity (for an additional cost) as the database's needs grows over time. Changing the tier via the Azure Portal or PowerShell script can trigger a background operation to create a replica of the database followed by a small outage, only a few seconds, as the switchover occurs. This also means that underestimation during the initial sizing process can easily be rectified at a later stage.

Azure SQL Database Elastic Pool

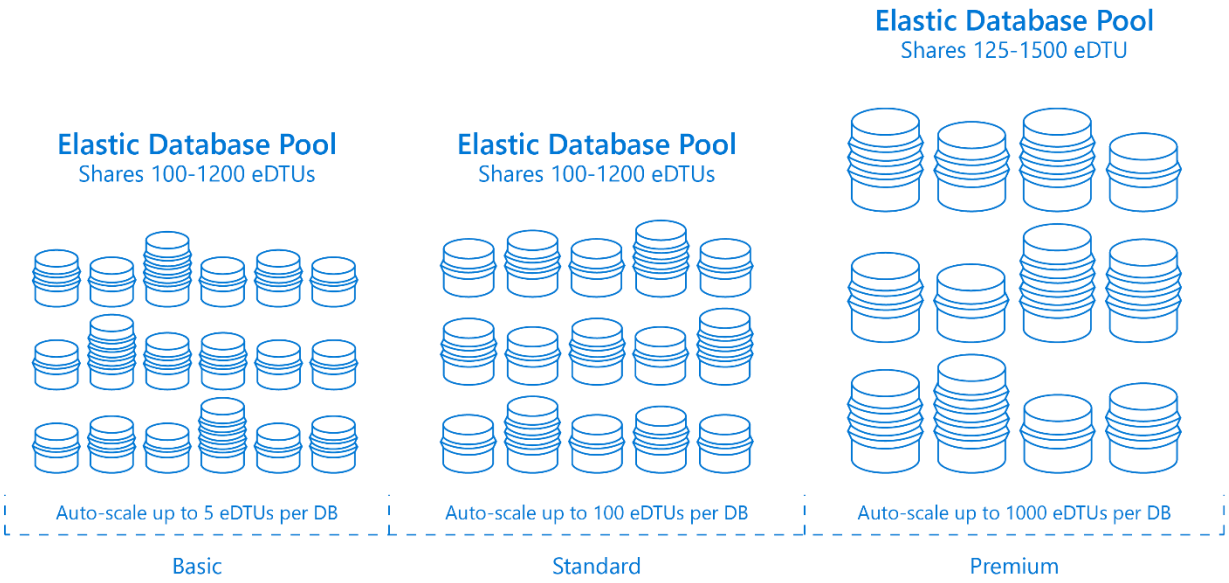
For many business processes and applications, being able to create single databases and dial performance up or down on demand is enough, especially if usage patterns are relatively predictable. But if you have unpredictable usage patterns, it can make it hard to manage costs and your business model. Elastic pools are designed to solve this problem. You allocate performance resources to a pool rather than an individual database and pay for the collective performance resources of the pool rather than for single database performance.

Why use Azure SQL Database Elastic Pools?



Elastic pools are best suited for applications with many databases that have generally low utilization with occasional spikes. This is particularly useful for Software-as-a-Service offerings with multiple tenants that each have their own database. Substantial cost savings can be achieved by using Azure SQL Database Elastic Pools in this situation, with greater savings seen when more databases are added in to the pool.

If all the DTUs in the elastic pool are consumed, performance in the pool will be throttled with each database receiving an equal amount of compute resourcing, like the Resource Governor feature in SQL Server.



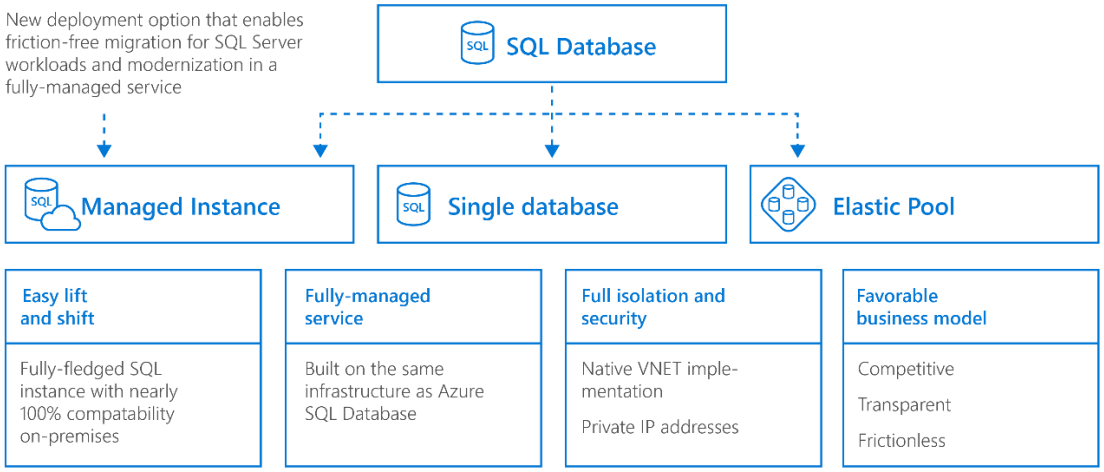
Technical Overview:

https://docs.microsoft.com/en-us/azure/sql-database/sql-database-technical-overview

Azure SQL Database Managed Instance

Azure SQL Database Managed Instance offers broad SQL Server compatibility and network isolation making it easy to lift-and-shift SQL Server databases to Azure. You can now simply backup an on-premise database and restore it into an Azure SQL Database Managed Instance. Built on the same fully-managed service offering infrastructure as Azure SQL Database and maintaining all the Azure SQL Database features like active geo-replication, high availability, automatic backups, database advisor, threat detection, intelligent insights, and vulnerability assessment. It also adds support for database sizes up to 8TB and SQL Server features like SQL Agent, cross-database querying and replication.

For organizations looking to migrate large numbers of SQL Server databases from on-premises or VM/hosted, self-built or ISV provided, with as low effort as possible, Managed Instance provides a simple, secure and economical migration destination.



Why Use Azure SQL Database Managed Instance:

- Isolated environment (single-tenant service with VNET, dedicated compute and storage resources)
- Customer configurable backup retention and recovery
- Database Advisor and Log Analytics for advanced workload analysis
- Automatic database tuning and maintenance for predictable performance
- Monitor, troubleshoot and manage at scale
- Azure Portal functionality for manual service provisioning and scaling
- Azure AD authentication, single sign-on support
- Adheres to same compliance standards as Azure SQL Database
- Encryption of the data in transit and rest with customer provided encryption keys
- No patching and version upgrade overhead

SQL Server on Azure VMs

In this section SQL Server installed and hosted in the cloud on Windows Server Virtual Machines (VMs) running on Azure, also known as an Infrastructure-as-a-Service (IaaS), are discussed. SQL Server on Azure virtual machines are optimized for migrating existing SQL Server applications. All the versions and editions of SQL Server are available. VMs offer 100% compatibility with SQL Server, allowing you to host as many databases as needed and executing cross-database transactions. VMs allow full control on SQL Server and Windows.

Why use SQL Server on Azure VMs?

VMs are great for existing applications that require fast migration to the cloud with minimal changes. VMs are well suited for rapid development and test scenarios when you do not want to buy on-premises non-production SQL Server hardware.

Other reasons to use VMs for SQL Server deployment in the cloud:

- Configure and manage high availability, disaster recovery, and patching for SQL Server easier than on-premises machines
- Customized environment with full administrative rights
- SQL Server instances with up to 64 TB of storage and as many databases as needed
- Fully supports SQL Server transactional replication, AlwaysOn Availability Groups, Integration Services, Log Shipping to replicate data, and traditional SQL Server backups

5.2 How to choose the right target platform

When looking to choose an appropriate target platform for each workload, there are three considerations to be made:

- Usage Scenarios
- Features
- Total Cost of Ownership

5.3 Choosing target platform by usage scenarios

Azure SQL Database single databases and elastic pools

Azure SQL Database is ideally suited for customers developing new SaaS multi-tenant applications or intentionally transforming their existing on-premises applications into a SaaS multi-tenant application. There are enough differences between Azure SQL Database single databases and elastic pools and on-premises SQL Server that it is not usually trivial to lift-and-shift on-premises database workloads to Azure SQL Database. Similarly, third-party applications do not yet support the Azure SQL Database platform.

This version of SQL Server is designed for a high level of uptime, with high availability coming as standard, which can be extended to provide geo-replicated topologies.

Azure SQL Database Managed Instance

Managed Instances are good for customers looking to migrate several applications from on-premises or VM/hosted, self-built or ISV provided, with as low migration effort as possible.

Additionally, these features make Managed Instances more desirable:

- High level of compatibility with on-premises SQL Server
- Support for isolation of workloads from the public internet using VNET support with private IP addresses and VPN to on-premises networks

SQL Server on Azure VMs

Virtual machines can help customers that need to customize the operating system or the database server, as well as customers having specific requirements in terms of running third party apps side-by-side with SQL Server (on the same VM).

5.4 Choosing target platform by features

Azure SQL Database single databases and elastic pools

Azure SQL Database would be appropriate for use if the application surface area is database scoped.

If the application uses some SQL features, then Azure SQL Database may not be appropriate as not all are yet available.

Azure SQL Database Managed Instance

Would be appropriate for use if the application surface area is instance scoped and requires features not available in Azure SQL Database such as:

- SQL Agent
- MSDTC
- DQS
- MDS
- Database Mail
- Filestream
- Filetable
- Polybase

Additional features include:

- Support for Linked Servers
- Supports new Azure cloud services such as Threat Detection

SQL Server on Azure IaaS

Use if application surface area is instance scoped and requires features not available in Azure SQL Database Managed Instance

Additionally, supports local instances of:

- SSRS
- SSAS
- SSIS

5.5 Choosing Target Platform by Cost

Azure SQL Database

The Platform-as-a-Service nature of Azure SQL Database greatly reduces administration and management costs over the more traditional SQL Server on Azure IaaS topology, as most of the required work is completed silently in the background for you by Microsoft Operations. This is evident at scale where considerable savings in time and effort can be made.

Azure SQL Database Elastic Pools

Azure SQL Database Elastic Pools can offer considerable savings if used by multiple databases that have varying and unpredictable usage demands. The sharing of compute resources amongst all databases in the pool means that customers are not required to over-provision resources for all databases to meet their infrequent spikes in usage.

Further savings are made on lowered server maintenance and administrative costs as most of the required work is completed silently in the background by Microsoft Operations.

Azure SQL Database Managed Instance

Managed Instances is offered to those customers who want a fully managed service offering, where they can easily lift and shift their on-premises environment with minimal configuration changes. The environment offers a minimum of 8 cores and up to 35 TB of storage and sits in an isolated virtual network. This offering is great for customers that want to quickly get to the cloud and want to avoid the overhead of virtual machines.

SQL Server on Azure VMs

VMs impose higher compute, storage, and management costs over the Azure SQL Database offerings but grants greater control across the SQL Server and infrastructure.

Tip: Azure Hybrid Benefit for additional savings

If you have active Software Assurance on existing Windows or SQL Server licenses, then you can use the Azure Hybrid Benefit to receive savings on Window Server or vCore-based SQL Database options.

See https://azure.microsoft.com/en-us/pricing/hybrid-benefit/ for more information.

5.6 Migrating SSAS, SSIS and SSRS to an Azure fully managed service offering

Are you looking to also migrate SSRS, SSAS or SSIS? Unfortunately, not all SQL Server components have an Azure data services equivalent currently.

SQL Services Analysis Services (SSAS)

SSAS can be migrated to Azure Analysis Services which is largely compatible with recent versions of SQL Server Analysis Services Enterprise Edition.

For more info see Azure Analysis Services videos on Channel 9.

SQL Server Integration Services (SSIS)

Use the Azure SSIS Integration Runtime, which is the compute infrastructure used by Azure Data Factory.

SSIS packages can be invoked using stored procedures in Azure Data Factory to provide true first-class support of SSIS package execution.

For more info see Azure SSIS lift-and-shift overview and Azure Data Factory tutorial.

SQL Server Reporting Services (SSRS)

SSRS currently has no direct cloud-based equivalent, but reports could be re-written using Microsoft Power BI or migrate to SSRS running on an Azure VM.

5.7 Migrating SSAS, SSIS and SSRS to Azure VMs

First, install services on an Azure VM and connect to Azure SQL Database or Managed Instance.

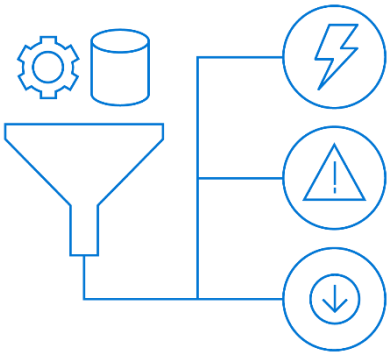
Some reference links for SSRS and SSAS migrations:

- SSAS multidimensional
- SSAS tabular
- SSRS data sources
- SSRS connection type

Alternatively, it's still possible to utilize an existing on-premises SSRS server to connect to your Azure SQL Database or Managed Instance for reporting purposes.

5.8 Plan the migration tool

Often the acceptable downtime or maintenance window stipulated by the application owner will dictate which migration method needs to be used, with a corresponding migration tool to match.



Critical (Zero downtime) – SQL Server Management Studio (SSMS) transactional replication

High (Small maintenance window) – Azure Database Migration Service (DMS)

Low (Large maintenance window) – SQL Server Management Studio BACPAC export/import

For critical workloads, which must remain online and available, the use of transactional replication technologies can copy most of the data to Azure in the background and then keep the target data in-step with the source data until a switch-over can occur. SQL Server Management Studio can be used to establish this copy process.

For important applications that can afford some downtime, the Azure Database Migration Service should be used to perform the initial assessment and migrate the data in a consistent and correct manner.

Finally, SQL Server Management Studio can be used to export the data and schema of a database in the form of a BACPAC file. For larger databases, the time taken to export and import the BACPAC can be considerable, so this method is best suited for low priority workloads with large maintenance windows available.

5.9 Target Platform Selection Examples

In this section we'll look at four examples of customer workloads and requirements and decide on an appropriate target platform as well as the method of migration we'd use to get it there.

Example 1 – Azure SQL Database Single Database

The customer has an application that uses an on-premises SQL Server 2008 R2 installation. This application is 24x7 business critical with significant impact from any downtime during the year. This strict operational requirement has meant no scheduled maintenance windows are available and unscheduled maintenance is unacceptable. To facilitate this, the underlying infrastructure is designed for high availability with full redundancy across all components. The actual database has minimal growth per year, but an extremely high transaction rate that requires sizeable amounts of compute resources coupled with low latency/high throughput storage and networking. The redundancy requirements across all components means there are a lot of SQL Servers, virtual machines, storage, and networking to keep the DBAs and Sysadmins busy for significant amounts of time, which they'd rather spend on improving the performance and security of the application.

Solution:

In this scenario, utilizing the **Azure SQL Database** fully managed service offering platform would be beneficial as it removes the issue of managing compute and storage requirements. With Azure SQL Database including local high availability as standard for a 99.99% uptime SLA, and the possibility of using geo-located replicas for regional high availability and disaster recovery, the high uptime requirements should be easily met. Azure SQL Database's premium performance tier is capable of 2ms IO latency with IO throughput measured at approximately 48 IOPS per DTU, a performance level on-par with enterprise flash-based SAN storage appliances.

The lack of an allowed maintenance window means that it would not be possible to migrate the on-premises databases to Azure SQL Database using a backup-and-restore technique due to the sheer size of the data involved. It would take too long to copy the backup files over the WAN connection. Instead, transactional replication would be used to synchronize the data in the background while keeping the source database online and available.

Moving to Azure SQL Database would save costs on hardware and management overhead by removing the need to monitor, patch, and fix the numerous servers in the on-premises solution.

The application would also benefit from Azure SQL Database's built-in intelligent optimization and monitoring. Azure SQL Database Advisor can make recommendations of missing indexes that should be added or unused indexes that could be removed, to proactively improve application performance. Azure SQL Database Intelligent Insights analyzes SQL Database performance by comparing the current database workload with a historical baseline to highlight performance degradations and their possible causes. Threat Detection can be utilized through Azure Security Center, to detect and respond to potential threats as they occur.

Example 2 – Azure SQL Database Elastic Pools

The customer has an application that uses many databases currently residing in an on-premises version of Microsoft SQL Server 2008. The total database footprint is large, and rapidly growing by several terabytes per year. The existing SAN storage that the databases are located on is almost at capacity, expensive to expand, and nearing the end of its life. The application is critical to the company, with a moderate transaction rate, and any downtime would have significant business impact. Small maintenance windows are available in which to make changes to maximize the availability of the application. The high growth rate has seen more and more time being spent by DBAs and sysadmins just to keep everything running.

Solution:

In this scenario, again utilizing the **Azure SQL Database** fully managed service offering platform would be beneficial as it removes the issue of managing constantly increasing storage requirements, as well as increasing compute resource requirements. Provisioning of storage and compute in the Azure data centers is managed by Microsoft Operations, with dialing up resources for your Azure SQL Databases turned into a trivial task via the Azure Portal. This leads to considerable costs savings in both hardware costs and saving man-hours and freeing up DBAs and sysadmins to add more value to the business in other areas.

By choosing the **Elastic Pools** option of Azure SQL Database, the allocated resources can be shared amongst all the databases with the idea that not all databases will require the maximum allocated resources at any one time. This can produce considerable cost savings as much lower total resources are required to service all the databases in the pool. It also means as the workload grows, additional resources can be added to the elastic pool which benefits all the databases using the pool, which greatly enhances the scalability of the consuming applications. Sharding can also be implemented to spread out chunks of the database to help stay under the 4TB limitation.

A small maintenance window means that Database Migration Services (DMS) in Azure could be used, because it has a minimal downtime window. Multiple databases can be moved concurrently during maintenance hours as several scheduled jobs run together in a centralized hub.

Example 3 – Azure SQL Managed Instance

The customer has a custom-built application based on an on-premises SQL Server that contains sensitive intellectual property-related data. The application code has had some quirky development practices used in the past, which have caused compatibility issues over the years during upgrades from SQL Server 2000 to 2005 to 2008 to, finally, 2012. Any changes made to this application are costly as the development work has always been handled by a third-party development team. The application also does many cross-database queries for reporting and analysis reasons.

A scheduled outage of the application would have a medium level impact on the business but would be acceptable with some forward planning. The customer is not convinced their current backup and recovery solution is reliable, with frequent failures occurring due to lack of free disk space or hung backup agent processes. The customer would like to remove the headaches of these operational tasks such as performing backups, patching, and version upgrades.

The customer also has heard that the multi-tenant nature of cloud-based services means they will need yet another set of user credentials for each user to remember, which could cause extra overhead to users.

Solution:

The preferred platform of Azure SQL Database Single Databases doesn't yet support all the features and levels of compatibility that a traditional on-premises SQL Server might, with one feature that it lacks being the ability to perform cross-database queries, something the customer has pointed out they need. Rather than have to re-engineer the application to fit a solution that which will be both costly and time consuming, Azure SQL Database Managed Instance can be utilized to bring a high level of compatibility with on-premises SQL Server, while still enjoying many of the benefits of the cloud.

To migrate, a native SQL backup can be taken of the on-premises SQL Server databases, uploaded to Azure Blob storage, and restored straight into Azure SQL Database Managed Instance.

Once on Azure SQL Database Managed Instance, the built-in automated backup and point-in-time restore capabilities can remove the headaches of ensuring reliable data protection, but the customer configurable backup retention and recovery means they still have control when needed.

Furthering their data protection, Azure SQL Database Managed Instance's native support of encryption means that valuable intellectual property data can be secured by encrypted data in transit and at rest using customer provided encryption keys.

Cost savings can be made on server maintenance and administration areas, with no patching and version upgrade overhead, so that administrators can fulfil higher priority tasks. Additional savings could be made by bringing their own SQL licenses with active Software Assurance using the Azure Hybrid Benefit for SQL Server.

Lastly, by synchronizing their on-premises active directory to Azure Active Directory using the free Microsoft AADConnect directory synchronization tool, it's possible to provide a single sign-on experience, so that Azure SQL Database Managed Instance databases are accessible using Windows user credentials without any additional login prompts being displayed. Managed Instances also adhere to compliance standards available to Azure SQL Server, so that customers do not need to maintain a lot of administrative overhead to keep up with new standards.

Example 4 – SQL Server on Azure Infrastructure as a Service (IaaS)

Our customer in this example has a custom application that makes use of the Filestream feature in SQL Server to store large sound files to disk and need to be read back at high speeds. Several third-party tools integrate with this SQL Server instance to provide advanced processing of the related meta-data and need to be installed locally. The vendors of these tools have yet to produce a version that works with Azure SQL Database.

Solution

Unfortunately, the requirement for the Filestream feature rules out Azure SQL Database as it is not yet supported by this platform. Also, the need to install the third-party tools locally on the SQL Server rules out Azure SQL Database Managed Instance, whereas the full SQL Server instance is exposed to the end-user under Managed Instance, the underlying operating system is not.

Therefore, in this example, the solution needs to use SQL Server on Azure VMs (IaaS), which offers a customized virtual environment to run SQL Server and includes full administrative rights to allow installation of third party tools. The full specifications of SQL Server can be utilized, including support for up to 64TB of storage, as many databases per instance as needed, SQL Server transactional replication, AlwaysOn Availability Groups, Integration Services, Log Shipping to replicate data, and native SQL Server backups.

The downfall of using SQL Server on Azure VMs over Azure SQL Database is that many server maintenance and administration costs still exist, as does the need to manually configure and manage high availability, disaster recovery, and patching, creating extra administrative overhead.

To migrate, Azure Site Recovery could be used to lift-and-shift the existing SQL Server to the Azure data center. This produces an exact replica of the server, complete with already installed and configured third-party tools which saves time and reduces the risk of mistakes when installing from scratch. The server data is synchronized to Azure in the background while the on-premises server remains online and available for service requests, with a minimal outage required to failover to the completely synchronized server image at an agreeable time.

5.10 Example Summary – Target Platform Selection

The following tables summarize the scenarios from the previous examples where each of the target platforms are suited.

Common across Azure SQL platforms:

Target platform	Indicators to look for	Benefits
Azure SQL Database Single Database Azure SQL Database Elastic Pools Azure SQL Database Managed Instances	Current capacity or management issues Requires high availability	Compute and storage layers are provided and managed for you Near limitless capacity available on demand Premium tier available to meet higher performance and throughput requirements Highly available as standard Options available for regional high availability and disaster recovery protection Azure manages backups, upgrades and patching for you Azure provides automated analysis and recommendations for performance and security events. Support for data encryption Support for single-sign on with Azure AAD

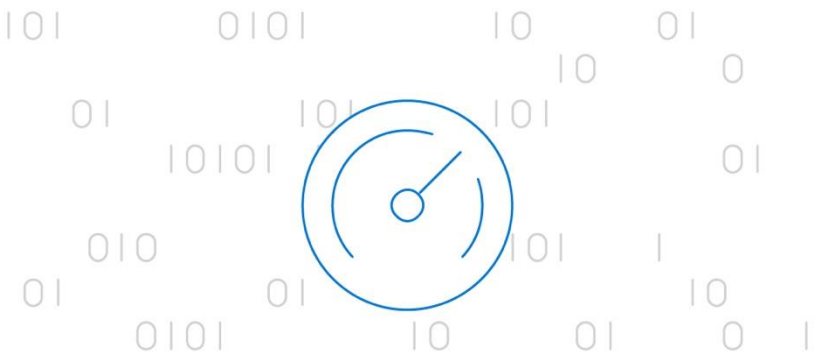
Specific to each platform:

Target platform	Indicators to look for	Unique Benefits
Azure SQL Database Single Database	Small number of databases or many databases but all with high steady usage	Lowest cost for single databases
Azure SQL Database Elastic Pools	Many databases or multi-tenant deployments	Cost effective as pooled resources are shared amongst multiple databases
Azure SQL Database Managed Instances	Do not own the application code or expensive to modify Requires high level of compatibility Uses features of SQL Server not yet supported by Azure SQL Databases	Fully managed service whilst retaining high-level of compatibility with SQL Server Supports SQL features such as cross-database queries which are unavailable in Azure SQL Database
SQL Server on Azure Infrastructure as a Service	Use of features of SQL Server not yet supported by Azure SQL Databases Third-party tools installed locally on SQL Server	Fully compatible with on-premises SQL Server Third-party applications and plug-ins highly likely to work as-is Full access to underlying operating system to install third-party tools and services

5.11 Example summary – migration tools selection

The following table summarizes the scenarios from the previous examples where each of the migration tools are suited:

Migration Tool	Indicators to Look for	Unique Benefits
Transaction replication	Critical database with small or non-existent maintenance window Large databases (>1TB)	Smallest possible outage requirements for switch over as source database remains online and servicing requests during synchronization of data
Azure Database Migration Service (DMS)	Many databases to migrate with moderate maintenance window allowance Large databases (>1TB)	Supports moving multiple databases concurrently
BACPAC export/import	Small number of ad hoc databases to migrate Small to medium sized databases (<1TB) Low availability requirements with relaxed maintenance windows	Quick and easy with no real set up requirements
Azure Site Recovery	Existing SQL Server to be moved as-is to Azure	Lift-and-shifts an exact replica of server to Azure IaaS VM Source server remains online and servicing requests during synchronization of server data



6 TRANSFORM AND OPTIMIZE

With a solid idea now of what is being migrated, where its migrating to, and how that will be achieved, the next step is to transform and optimize and make any required changes to the source environment to prepare for the upcoming migration phase. After completing the transform and optimize phase we will have:

Schema compatible with target

The database schema will be in a supported state for the target platform and ready to migrate

Preparations complete for data migration

All errors will be rectified, and data is ready to be moved.

Throughout the remainder of this section we will investigate how to transform the source data or mechanisms used to fix any issues and look into possible optimizations that can be made to take full advantage of the Azure SQL platform.

6.1 Transformation

In an ideal world, migrating data from on-premises to the cloud would just work, but it's highly possible that some things will need to be changed to ensure a functional and successful migration. Areas to focus necessary changes on would include:

Update and check database schemas

The findings from the earlier assessment tasks should have highlighted any changes needed to be made to the database schema and these modifications should be implemented in this stage.

Implement any version upgrade requirements for the environment

The SQL Server source must generally be on at least SQL Server 2005 or greater to use the migration tools provided by Microsoft, such as Azure Database Migration Service (Azure DMS) and Data Migration Assistant (DMA). If the source SQL Server doesn't meet this minimum requirement, then an upgrade will need to be factored in before these migration tools can be used to ease the remainder of the migration.

Remediation of any errors or warnings provided by the migration assessment tools

Some features must be supplemented or removed due to not being available in Azure SQL Database, including the use of cross-database references, service broker, log shipping, or linked servers. Deprecated SQL syntax might need to be updated or rewritten to meet the required version of SQL on Azure SQL Database.

Migrate existing integrated database services into Azure

As we previously found, Azure doesn't yet have a like-for-like comparable cloud service for SSIS or SSRS. These workloads will require implementing new Azure services that can partially support the required workloads, keeping the workloads on-premises, or implementing them in virtual machines.

Handling SSIS workloads in the cloud

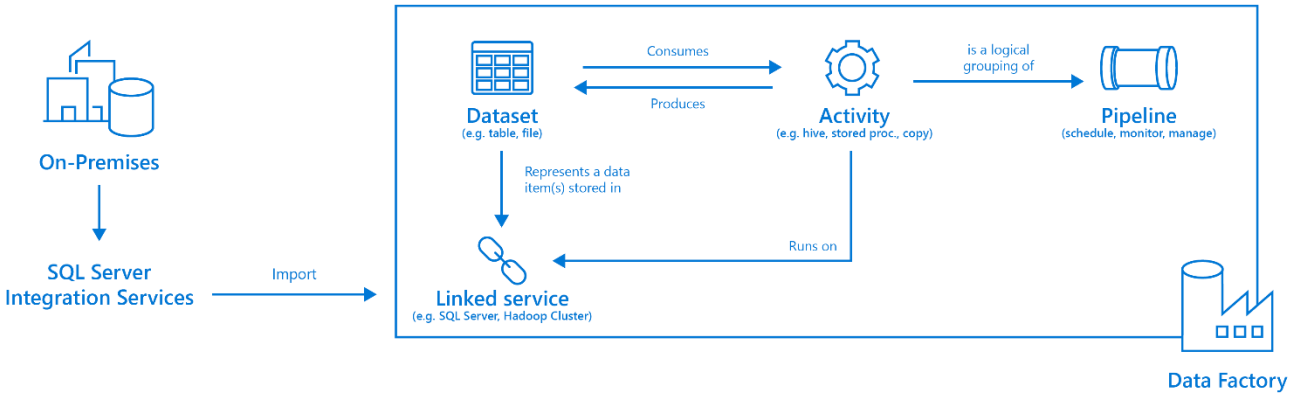
Reasons you might want to move your on-premises SSIS workloads to Azure can reduce operational costs and reduce the burden of managing infrastructure that can run SSIS on-premises or on Azure Virtual Machines. High availability can be increased by specifying multiple nodes per cluster, as well as using the high availability feature in Azure SQL Database. Finally, you can increase scalability with the ability to specify multiple cores per node (scale up) and multiple nodes per cluster (scale out).

Slightly different from on-premises SSIS, where the SSIS runtime is hosted by the SQL Server, in Azure it is Azure Data Factory that hosts the runtime engine for SSIS packages. The runtime engine is called the Azure SSIS Integration Runtime (SSIS IR). The SSIS Catalog Database that SSIS uses (called SSISDB) is provisioned on Azure SQL Database, which should be implemented in the same Azure region as the SSIS IR.

Creation of the Azure Data Factory is easily accomplished via the Azure Portal or PowerShell. The necessary Azure-SSIS integration runtime can then be created and started, making it ready to service SSIS packages. To deploy SSIS packages to Azure you can use either SQL Server Data Tools (SSDT) or SQL Server Management Studio (SSMS), which connects to the Azure SQL Server that hosts the SSIS Catalog (SSISDB).

SSIS can also be used to migrate data from on-premises to Azure. Here, Azure Data Factory again hosts the Azure-SSIS integration runtime. A sample set of steps for implementation of a data migration mechanism using SSIS packages might look something like:

- Create an Azure Data Factory
- Create an Azure-SSIS Integration Runtime (SSIS IR)
- Create SQL Server and Azure Storage linked services
- Create SQL Server and Azure Blob datasets
- Create Azure SQL Table
- Create a pipeline with a copy activity to move data
- Start a pipeline run
- Monitor the pipeline run
- Test for completion



6.2 Optimization

Optimization might include the following steps:

Assess what new features may be available on the target platform

New features that were previously too complex or cost inhibitive to warrant implementation on-premises may now be available via a few clicks in the Azure Portal. These features should be considered as to whether they would bring good benefits to each workload and then should be implemented as appropriate.

Re-structure workloads into more cost effective or performance effective sets

This might include allocating databases that make up a workload into various service levels and performance tiers on Azure SQL Database. These were previously lumped together on the same on-premises SQL Server due to hardware and licensing costs, but with the fully managed service offering of Azure SQL Database it's now cost effective to grant individual databases additional resources if beneficial.

Ensure workloads are right-sized

Look to realign workloads into the more appropriate service levels and performance tiers. Previously they shared a combined pool of compute and storage resources on the physical server where they resided that was underutilized to allow for future growth. Now with the fully managed service offering of Azure SQL Database it's possible to get a more accurate size of the databases by using tools such as the Azure SQL Database DTU calculator or comparing on-premises core requirements to vCores and dial up the allocated resources only if required.

The following list contains recommendations for best performance during the import process:

Choose the highest service level and performance tier that your budget allows for the migration time to maximize the transfer performance.

While migrating, the database will be performing an enormous quantity of write operations and by under-sizing the selected performance tier you may unintentionally throttle the throughput, causing a much-extended migration timeline. Instead, choose a higher performance tier than needed temporarily during the migration, and then scale back down after the migration to minimize costs.

Minimize the distance between your BACPAC file and the destination data center

By minimizing the physical distance between your BACPAC file and the destination data center, the network latency will be reduced. This in turn will increase overall migration throughput as more read and write operations to the target database can be completed in the same period.

Disable auto-statistics during migration

On Azure SQL Database, statistics objects have "Auto update" turned on by default. The auto update of the statistics is done when a sufficient amount of change to a table has occurred. During the import process, when nearly all the rows in all tables are changing, this trigger is repeatedly met, causing continuous attempts to update the statistics. This update uses valuable IO resources to complete, which detracts from the overall pool of IO resources available for the import process and extends the migration timeline.

Partition tables and indexes

Portioning tables and indexes can help with the transfer and access of data during a migration. The data can be partitioned into one or more subsets that are similar and will allow for the transfer of data quicker. Partitioning large datasets can also reduce lock contention, because lock escalation can be activated at the partition level without hurting the entire dataset. Once the data is moved to the cloud, then queries may performance faster and reduce overhead costs for applications. Overall partitioning tables and indexes helps the migration cost and mitigates future risk after the migration by helping increase performance of the data.

Drop indexed views and recreate them once finished

When an indexed view is used, every time data is modified on an underlying table Azure SQL maintains the index entries on those tables, but also the index entries on the view. This can affect write performance and again reduce IO resources available for the import process, extending the migration timeline. In addition, they also have the potential to cause other issues such as lock contentions.

Remove rarely queried historical data to another database and migrate this historical data to a separate Azure SQL Database. You can then query this historical data using **elastic queries**.

By purging historical data from a database, the size of the database and thus the amount of data needing to be migrated can be drastically reduced. This helps meet tight maintenance window targets as the core data can be moved to Azure SQL in a much shorter time, enabling the application to be brought back online sooner. The rarely queried historical data can be migrated in a less aggressive timeframe given it is a much lower priority.

7 MIGRATE, VALIDATE AND REMEDIATE

We move on to the final stage, the data migration itself. The previous planning, assessment, and transformations stages will have ensured everything is ready to be migrated and functioning correctly once moved to Azure. Therefore, all that's left to do is to prepare the migration tools required, perform the migration, and run post-migration functional and performance validations.

7.1 Migration overview

Consider the maintenance windows that are available to the application and database targeted for migration

If they are critical workloads, they may only be able to go offline for a few minutes at a very specific point in time. Alternatively, a workload may be used for historical reporting purposes and can easily be taken offline most days of the week without impacting end-users. These differences will help decide which migration technique needs to be used.

Start with low priority databases first

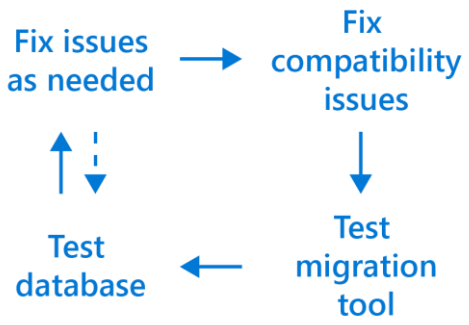
This can help ensure the migration process works and help gauge how long the migration is likely to take when you get to your more critical workloads.

Fix compatibility issues outlined in Data Migration Assistant

These issues should be fixed during the Transform and Optimize phase, but validate that DMA no longer presents any remaining issues.

Run a test migration with chosen tool

Before migrating the database, run a test migration of the database to confirm the amount of time the migration will take, and any issues encountered during the migration process.



Test database for issues

When the test migration completes, perform validation steps to confirm that the data is migrated in full and check for any issues encountered on the Azure SQL platform.

Repeat issue fixes until the database is fixed

For each issue discovered during testing, find a fix, and then retest. Keep repeating this test-fix cycle until all issues have been found and repaired.

Re-write third-party applications for the cloud as needed

Third party applications may benefit from the Azure Application Architecture Guide as it discusses older on-premises versus cloud architecture models that could help optimize performance and decrease overhead with leaner coding approaches. Each application should be analyzed on a case by case basis to see if a lift and shift or a re-write in necessary.

Test third-party applications

Confirm that any third-party applications will still function as expected in the cloud as each application is moved, including any dependencies.

Take old databases and application offline

Remember to take the source database and application offline before starting the migration process to avoid confusion and preserve the original data in case there is a need to refer to them or perform a roll-back.

Create new disaster recovery and maintenance plans

Take the time to update your disaster recovery plans, as data has now moved locations and is accessed in a different manner. Consider improving disaster recovery plans by utilizing the geo-replication features of Azure to

protect data that may previously have been too complex or costly to protect when on-premises. Maintenance plans will also need to be reviewed as Azure now performs many of those maintenance tasks automatically for you in the background, removing the need to perform them manually.

Use toolsets to give you greater insight into your environment and greatly assist with the migration process

Numerous Microsoft and third-party tools exist that can help keep tabs on your SQL environments both on-premises and in the cloud, including:

- Azure Database Migration Service (DMS) helps track progress of large scale migrations of data to Azure.
- Microsoft Operations Management Suite can help monitor and visualize SQL workloads both on-premises and in the cloud, including SQL Server version count, current CPU performance, job successes and failures, and any logged events.
- Azure SQL Database Intelligent Insights is a tool that uses built-in intelligence to continuously monitor database usage and detect disruptive events that cause poor performance, offering recommendations for improvements that could help with functionality.

Assess migration tools based on disruption to help lower the risk of database downtime

In the coming sections we'll look at which migration tools require downtime to complete, and which ones can work in the background while the workload remains online and available.

Understand your workload requirements as a starting point

Requirements might include storage size, storage throughput, and high availability.

Create a plan to mitigate risk associated with downtime and compatibility issues

Many of the discussed points in this whitepaper will help to reduce the risk of errors during the migration. Conduct test migrations before doing the final migration by getting to know any errors before getting to more critical workloads and have a rollback plan prepared in case of an emergency.

Understand feature parity between versions of SQL Server and use assessment tools to mitigate choosing the wrong target option

Tools such as Data Migration Assistant (DMA) will help identify if the source workload is using features unavailable on some platforms in Azure.

Select non-critical workloads for migration initially

This can help ensure the migration process works and help gauge how long the migration is likely to take when you get to your critical workloads.

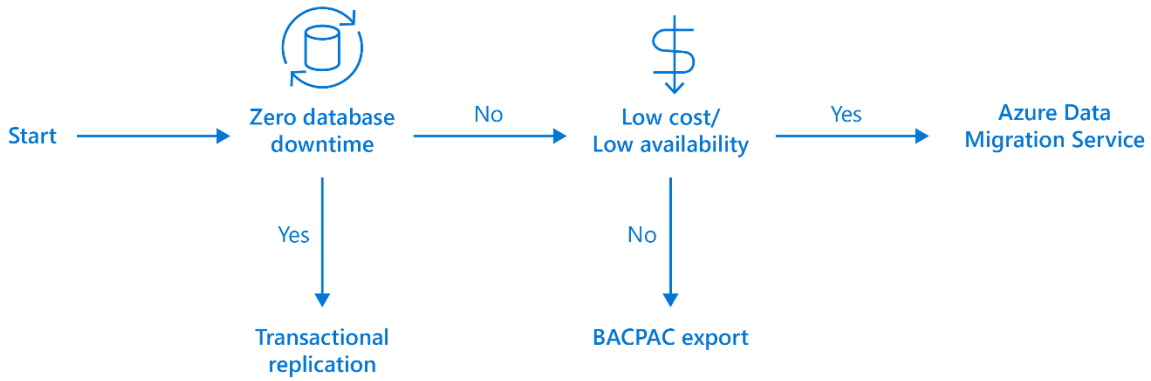
Continually iterate on your migration process

During the first migrations small changes will be found, documentation or processes will need to be created, or unnecessary migration steps will need to be removed. These findings should be fed back in to the migration process that you are following to optimize the remaining higher priority migrations.

7.2 Migration tool selection

The migration tools employed to move data to Azure will be selected based on the criticality of the workload and how long the application can be offline during the switchover.

Here is a simple workflow that can help with tool selection:



For critical workloads, which can afford zero database downtime, SQL Server Transactional Replication should be used to synchronize all data between on-premises and Azure while keeping the source database online and servicing requests. For general workloads, where small amounts of downtime are acceptable, Azure Data Migration Service can be used to manage the migration process for all these databases. For all other workloads that can be taken offline at a scheduled time, exporting a BACPAC file containing the data and schema of the source data base and importing this in to Azure would be a good fit.

Different tools can be used for different needs and no single tool must be used for all database migrations. We will investigate each of these further in the sections to come.

Migration using SQL Server Transactional Replication

Transactional Replication gradually migrates a SQL Server database to the cloud, while leaving production servers online and creating transactions. As new transactions are created at the source, these too are migrated to the target database, keeping the source and target in lock-step. This technique allows for a high level of availability as the only downtime involved will be switching over the application to point to the newly migrated Azure SQL Database. It's also especially suited for hybrid scenarios where a gradual or partial migration is desired rather than a bulk migration of all data.

Transactional replication can be configured using SQL Server Management Studio (SSMS) or T-SQL statements, with the Azure SQL Database set up as a push subscriber of the source SQL Server publisher. The required distribution database and replication agents cannot be placed on the SQL Database that is being migrated.

Snapshot and one-way transactional replication is supported, but peer-to-peer transactional replication and merge replication is not supported.

To use this method, the source database must meet the requirements for transactional replication and be compatible with Azure SQL Database. All versions of SQL Server from SQL Server 2012 and later are supported for use in a transactional replication configuration but might require a certain service pack and cumulative update installed before they can be used.

The SQL Server Sources supported with Transactional Replication are:

- SQL Server 2016 RTM
- SQL Server 2014 SP1 CU3
- SQL Server 2014 RTM CU10
- SQL Server 2012 SP2 CU8

To use this solution, then configure your Azure SQL Database as a subscriber to the SQL Server instance that you wish to migrate. The transactional replication distributor synchronizes data from the database to be synchronized (the publisher) while new transactions continue to occur.

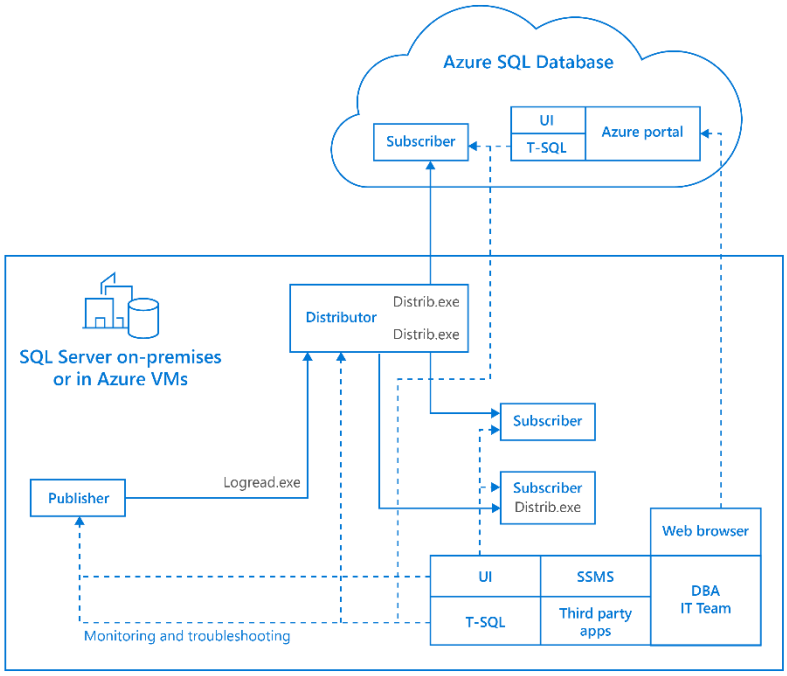


Figure 20 Transactional Replication with SQL Server

With transactional replication, all changes to your data or schema show up in an Azure SQL Database. Once the synchronization is complete and the data is ready for a switch over, then change the connection string of your applications to point to the Azure SQL Database and publish the application to production.

As transactional replication finishes any changes left on your source database and all your applications point to the new Azure SQL Database, then you can uninstall transactional replication.

Migration using Azure Data Migration Service (DMS)

Azure Data Migration Service is a fully managed migration service designed to enable seamless migrations from multiple database sources to Azure data platforms with minimal downtime. To achieve this, Azure DMS couples together multiple Microsoft migration engines such as the Data Migration Assistant (DMA), the Database Experimentation Assistant (DEA), and SQL Server Migration Assistant (SSMA) to cover a wide range of scenarios.

Azure DMS is accessed via the Azure Portal (https://portal.azure.com) where an Azure DMS instance can be created based on different regions with a variety of vCore options available. By assigning more vCores to the service you can provide for faster migrations to meet your intended timeline, but at the expense of added cost.

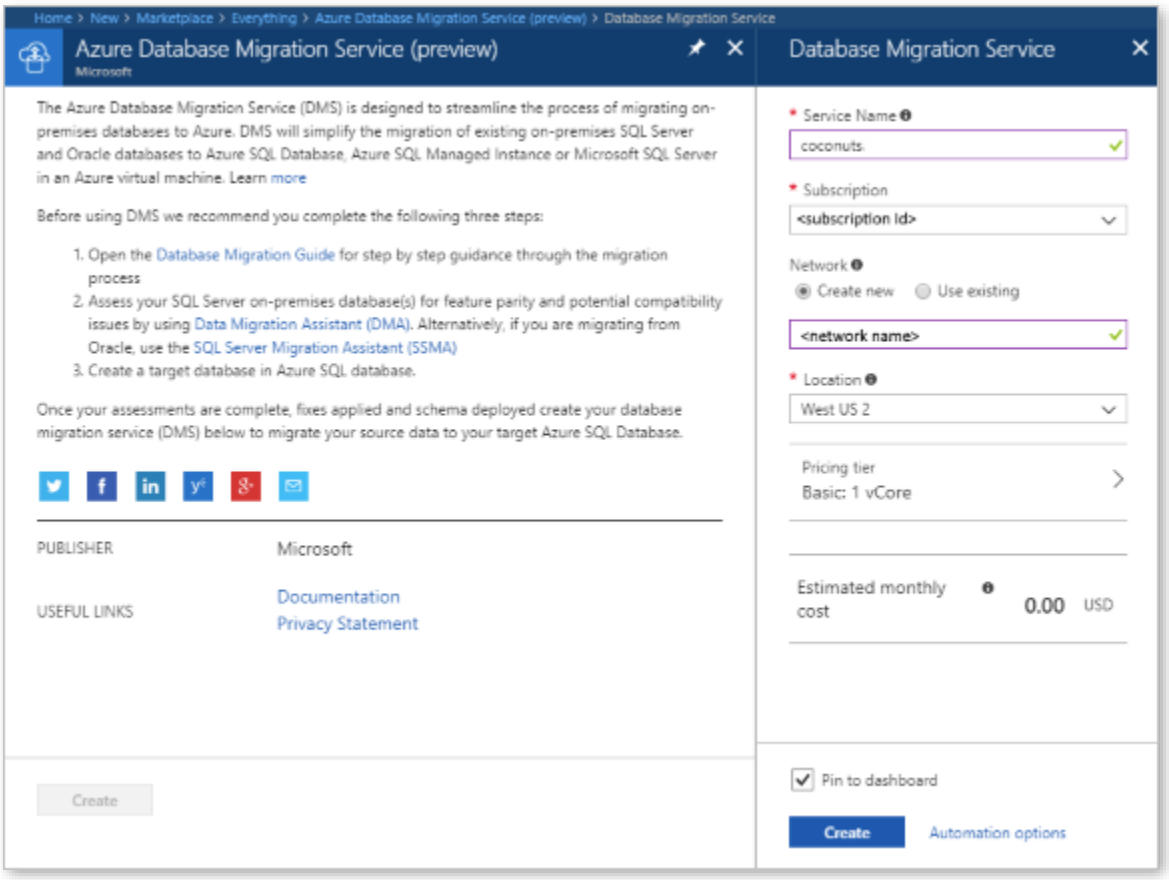


Figure 21 Creating an Azure Database Migration Service

Azure DMS supports migrating to all service options of Azure SQL Database (Single, Elastic, and Managed Instance) as well as SQL Server on an Azure IaaS Virtual Machine.

From there it's possible to create projects that allow you to perform source assessment, schema, data conversion, and validation activities which help prepare the source for migration. Migration tasks can also be created easily, such as proof of concept migrations and automation scripts.

For more information refer to: https://azure.microsoft.com/en-us/services/database-migration/

The SQL Server database needs to be assessed for any blockers before you can migrate data from an on-premises SQL Server instance to Azure SQL Database. Database Migration Assistant is used to perform this task, as previously shown in the Assessment section.

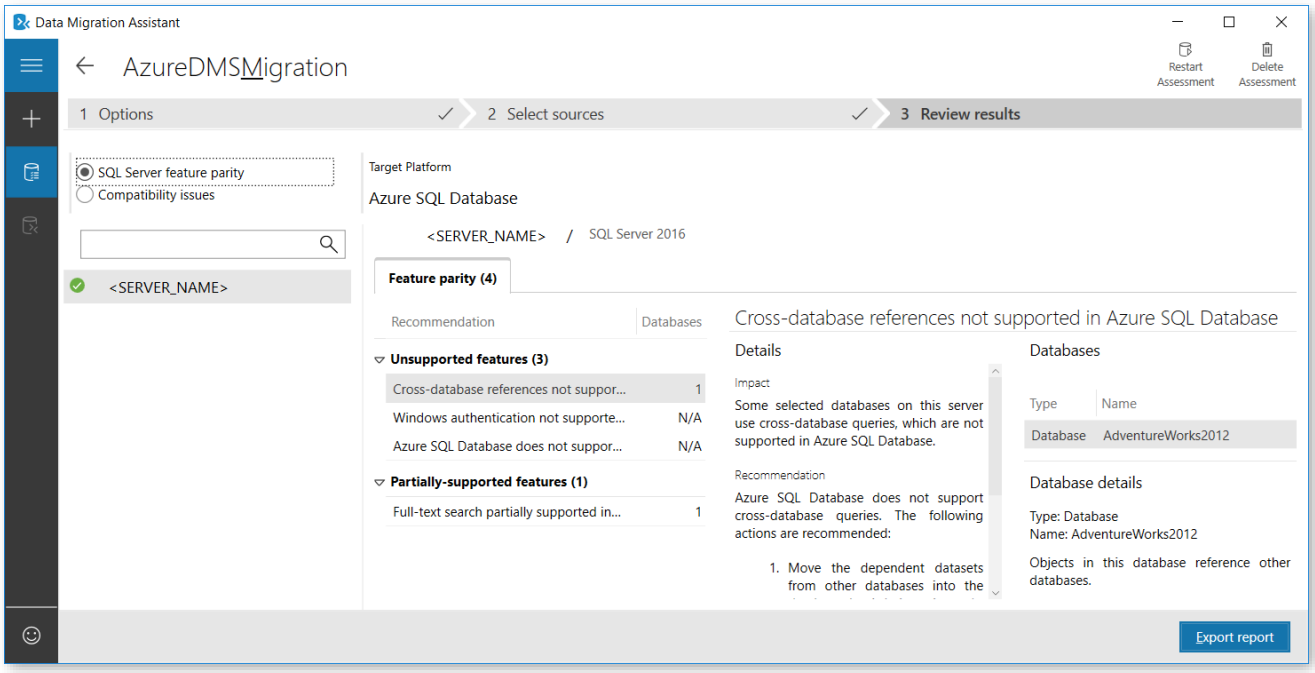


Figure 22 Reviewing Assessment Results with DMA

After completing the assessment and finding the selected database is a good candidate for migration to Azure SQL Database, then the Data Migration Assistant is used to migrate the database schema to Azure SQL Database.

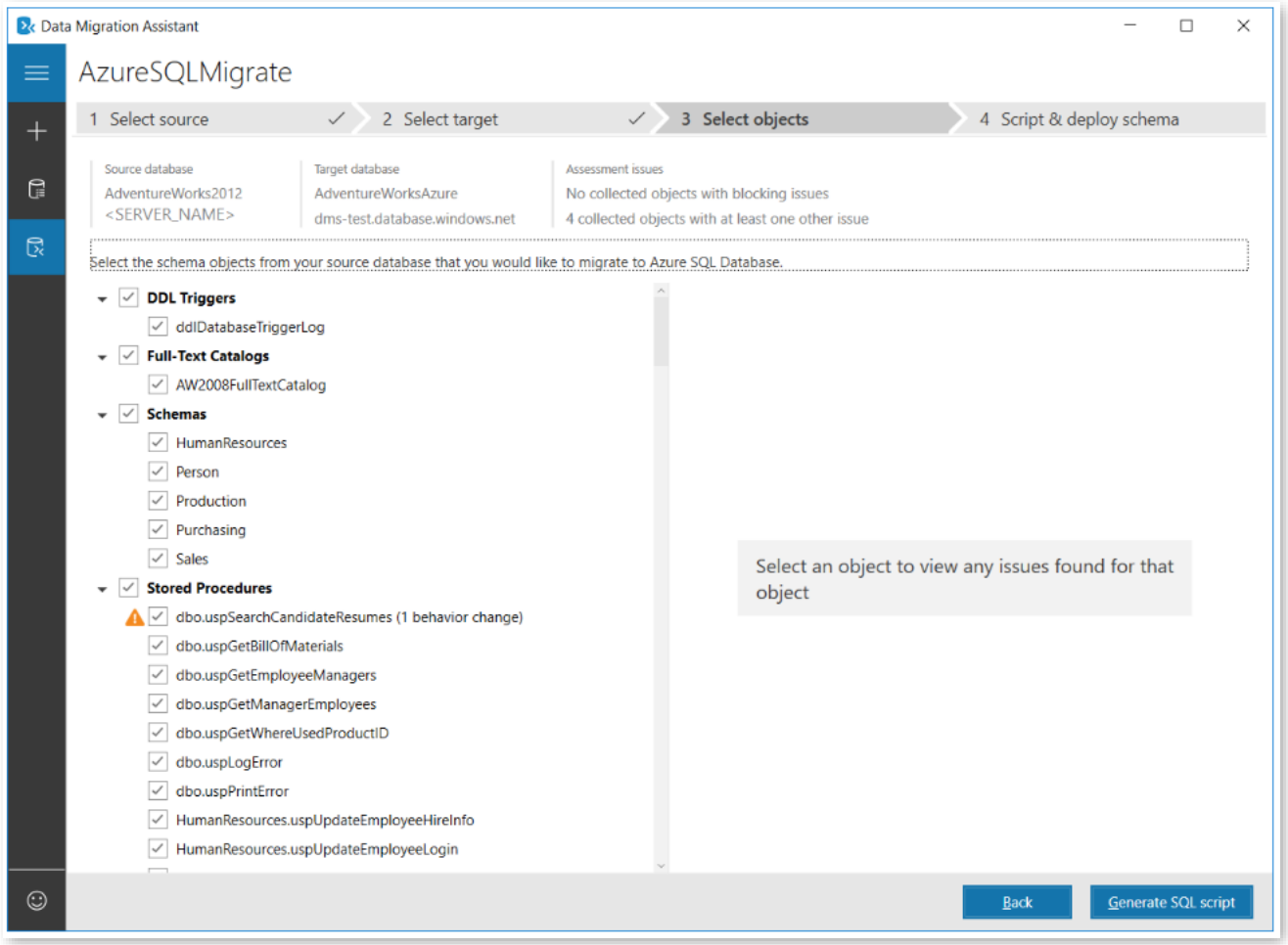


Figure 23 Select Objects to Migrate with DMA

DMA does this by generating a SQL Script which is then replayed on the Azure SQL Database to establish the database schema and prepare the target database for data insertion.

The Azure DMS service is then employed to migrate the data to Azure data services.

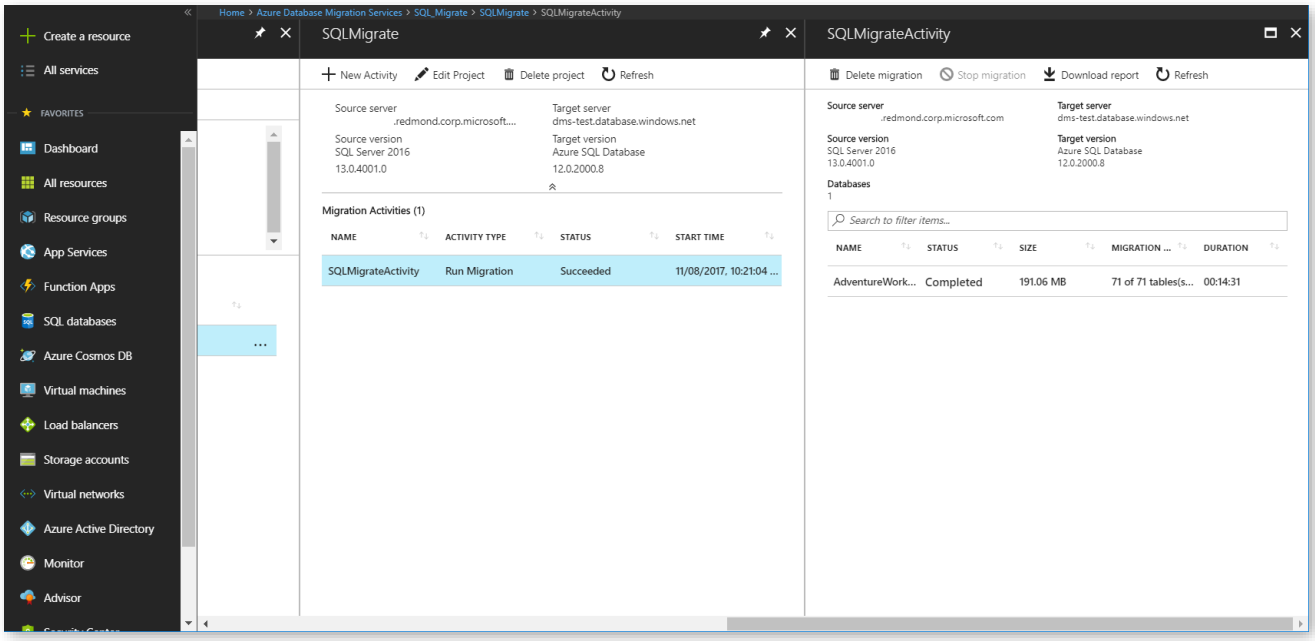


Figure 24 Monitoring data migration progress with DMS

Migration using data-tier application export/import (BACPAC)

SQL Server Management Studio can export a BACPAC file that encapsulates the database schema as well as the data stored in a database application. This can be useful as the BACPAC file can be easily imported to an Azure SQL Database to provide a simple means of migrating data between on-premises and Azure.

To ensure the exported BACPAC contains all data in a complete and consistent state, workloads using the source database need to be taken offline during the export process, so that transactions are not being made while exporting. This means that scheduled outages will be required to export a BACPAC file, which may need a substantial amount of time. It's possible to export up to 200GB to Blob Storage, so migration using BACPAC files is only good for smaller databases. This limitation might be a moot point as the time taken to export larger databases to a BACPAC file, copy the BACPAC file to Azure Blob Storage, and then import the BACPAC file to an Azure SQL Database can be substantial and other migration techniques would be better suited to minimize downtime.

For more information on migrating from SQL Server to Azure SQL Database using BACPAC files refer to the following link: https://blogs.msdn.microsoft.com/sqlcat/2016/10/20/migrating-from-sql-server-to-azure-sql-database-using-bacpac-files/

To import a BACPAC file into Azure SQL Database

1. Log on to the Azure Platform Management Portal at https://portal.azure.com
2. Click **New > Data Services > SQL Database > Import**. This will open the **Import Database** dialog.
3. Navigate to the .bacpac file to import: Click **Storage account > Container > BACPAC** and then click **Open**.
4. Specify a name for the new SQL database. The database name must be unique on the server, so the name cannot be the same as another SQL Server, and the name must comply with SQL Server rules for identifiers. For more information, see Identifiers.
5. Specify **Subscription, Edition, Max Size**, and host **Server** details. To continue, click the **Arrow** at the bottom of the dialog.
6. Specify login details for the host server.
7. To start the import operation, click the **Check mark** at the bottom of the dialog. The portal will display status information in the ribbon at the bottom of the page.
8. To view your new database, click **SQL Databases** in the navigation pane and refresh the page.

Import a BACPAC file to a new Azure SQL Database:
https://docs.microsoft.com/en-us/azure/sql-database/sql-database-import

8 CONCLUSION

Cloud computing is a major evolutionary step for the IT industry and while the benefits are evident, the path to get from the traditional on-premises data estate to one in the cloud has until recently proven to be challenging and time consuming. Continued investment by Microsoft into the tools and services useful for data migrations has vastly simplified the process and it is now it possible to get from on-premises to the cloud in far shorter timeframes with much less effort than before.

This whitepaper has guided you through the stages of discovering what you must migrate and where, before moving on to assessing databases prime for migration and how much work is required to prepare them. We then compared the possible target workload options in Azure and ran through some sample customer migration use-cases. Next, we discussed how to adapt workloads for Azure and how to optimize them once they are moved to the cloud. Finally, we considered the available tools and mechanisms for migrating the data to the cloud, and when to use each one.

However due to the fast-changing nature of the cloud, check the Microsoft Docs and Microsoft team blogs, mentioned in the resources section below, for changes in migrations tools and best practices, as new methodologies or supported configurations are being announced by Microsoft each month.

9 RESOURCES

Azure SQL Database Feature comparisons

Features	Azure SQL DB Single/Elastic	Azure SQL DB Managed Instance	SQL Server on Azure VM
Business model			
Units for pricing	DTU/eDTU	vCore/Storage/IOPS	vCPU/Storage
Premium/SSDs	Yes	Yes	Yes
Size limitation	4 TB	8 TB per instance	500 TB
Licensing	Included	Included or Azure Hybrid Benefit	Included or BYOL
Business continuity/disaster recovery			
High availability	Built-in	Built-in	User managed
Availability SLA	99.99%	99.99%	99.9%
Automatic backups	Yes	Yes	Options available to configure
Point in time restore	Yes	Yes	No
Backup retention	35 Days	7 days+	Depends
Geo-replication	Yes	Yes⁺	Yes, user managed
SQL Server surface area			
Native backup/restore	No	Yes	Yes
Cross-DB Transactions	Partially	Yes	Yes
SQL Agent	No	Yes	Yes
Database mail	No	Yes	Yes
Encryption	TDE/Always Encrypted	TDE/Always Encrypted⁺	Yes
Filestream/filetable	No	No*	Yes
Built-in intelligence			
Automatic tuning	Yes	Yes⁺	Partial (plan corrections)
Threat detection	Yes	Yes⁺	No
Vulnerability Assess.	Yes	No*	No
Intelligent Insights	Yes	Yes⁺	No

⁺ Before GA of the service
* To be considered post-GA

Cost Considerations

Component	Tier	Description	Cost guidance resource
Azure SQL Database (Single)	Basic	Non-production workloads	Cost guidance
	Standard	Moderate workloads with moderate availability	Cost guidance
	Premium	High transaction workloads with high availability requirements	Cost guidance
Azure SQL Database (Elastic Pools)	Basic	Non-production workloads	Cost guidance
	Standard	Moderate workloads with moderate availability	Cost guidance
	Premium	High transaction workloads with high availability requirements	Cost guidance
Azure SQL Database Managed Instance	General Purpose	Moderate workloads with moderate availability	Cost guidance
	Business Critical	High transaction workloads with high availability requirements	Cost guidance
SQL Server on Azure IaaS	Varies	Dependent upon Virtual Machine sizing, redundancy and resiliency considerations.	Cost guidance

Azure SQL Database related blogs and documentation

Microsoft Database blog
https://azure.microsoft.com/en-us/blog/topics/database/

Microsoft Azure SQL Database blog
https://azure.microsoft.com/en-us/blog/tag/azure-sql-database/

Microsoft SQL Server Database Engine blog
https://blogs.msdn.microsoft.com/sqlserverstorageengine/

Microsoft Data Migration blog
https://blogs.msdn.microsoft.com/datamigration/

Microsoft Azure SQL documentation
https://docs.microsoft.com/en-us/azure/sql-database/

Microsoft Azure Data Migration Assistant documentation
https://docs.microsoft.com/en-us/sql/dma/dma-overview

Microsoft Azure Database Migration Service documentation
https://docs.microsoft.com/en-us/azure/dms/dms-overview

Microsoft SQL Server documentation
https://docs.microsoft.com/en-us/sql/sql-server/sql-server-technical-documentation