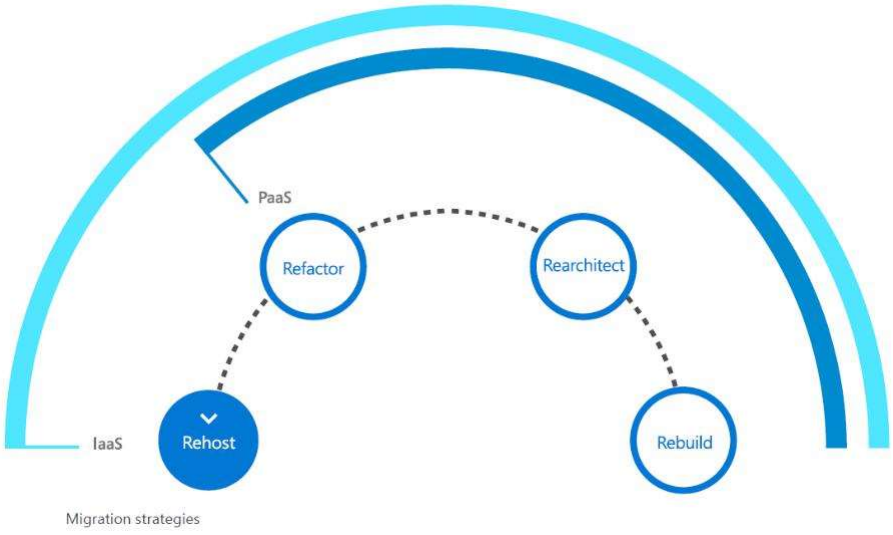
High-level plan for Migrating DB to Azure

Contents

Migration Strategies

Azure SQL offers several options for migrating and modernizing on-premises databases, ranging from rehosting IaaS approaches to various PaaS options, including support for fully managed data platforms. We have different strategies which are aligned with the different Azure Databases options like: Azure SQL Database, Managed Instance and VM Azure SQL.

Migration strategies can be boiled down to four main categories: Rehosting, refactoring, rearchitecting, and rebuilding.



The following section explains in detail the difference and best scenarios for the different strategies to be used.

Rehost- Move on-premises workloads to the cloud

The rehost strategy consists in migrating your physical servers and VMs to the cloud just as they are, without any changes to the code. You can also leverage SQL Server on Azure Virtual Machines, Microsoft's IaaS. The advantages of this strategy include moving quickly with no code changes, having the ability to have a cloud provider manage hardware and realizing lower total cost of ownership.

Refactor - Modernize applications for the cloud

The refactor strategy involves using additional cloud provider services to optimize reliability and performance by refactoring applications. The application can take advantage of IaaS and PaaS products such as Azure SQL Database, Azure SQL Managed Instance, and containers. The advantages of this option imply the current application as-is or with some minor code or configuration changes and connecting to new infrastructure services.

Rearchitect - Build new cloud-native SQL database apps

The rearchitect strategy is also known as "redesigning" an application to modernize it. Rearchitecting modifies or extends an existing application's code base to optimize it for a cloud platform and for better scalability. You can also take advantage of Azure SQL Database, Microsoft's PaaS offering to accelerate the process.

Rebuild - Take advantage of innovative cloud capabilities

The rebuild strategy revises the existing application by aggressively adopting PaaS or even SaaS architecture. The advantages of this strategy include building new applications using cloud-native technologies.

Migration decision tree

The following Microsoft's simple migration decision tree can help us to drive our decisions based on our priorities and requirements.

On the base of the objectives established for the migration, if we are migrating an SQL database or building new apps in the cloud, we should consider Azure SQL Database with a single database or elastic pools. Another option is Azure SQL Managed Instance, which provides full instance-level compatibility with the productivity benefits of a fully managed Azure service.

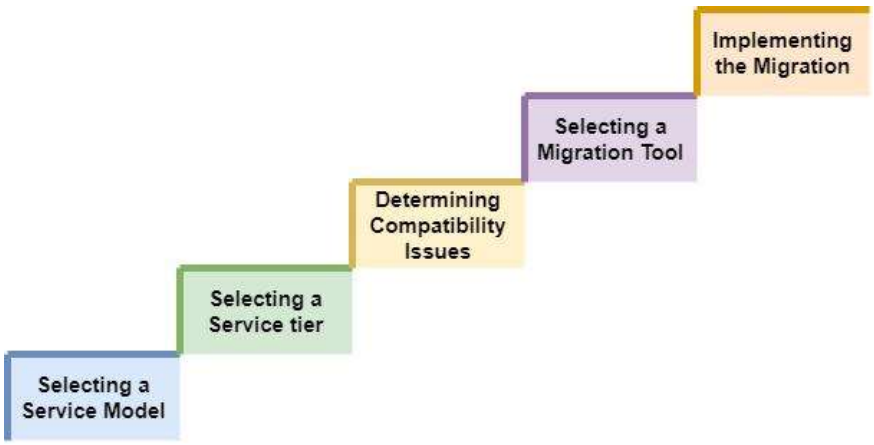
Discover and assess digital estate (DB Inventory)

The process of collecting all the DB inventory information is vital for doing the right planning, it includes the measuring and monitoring of workload. This part is so important for having the right inputs which help us to identify the best Azure Database option, starting with considering if is a PaaS or IaaS the most convenient.

Based on the DB Inventory, we should group and classify the Databases and prioritize which and what is the proposed Azure Database option for each of them, this would be based on dimensions such as business priority and complexity.

Migration Methodology

The following diagram shows an ideal methodology which can help us to organize in a right direction the migration steps.



Selecting a Service Model

Based on the Migration decision tree we have a first step in advance, in this step we must decide whether the database will be deployed individually, in an elastic pool, or as an SQL Managed Instance.

Selecting a Service Tier

In this step we must find an appropriate service tier and performance level for Azure SQL Database, we are going to use Data Migration Assistant tool to get recommendations for an appropriate starting tier to use when migrating an on-premises workload.

Determining Compatibility Issues

In this step we must find out the compatibility issues and re-write the application if required, for instance: cross database queries (not supported in Azure SQL Database).

The Data Migration Assistant (DMA) is a tool which helps to upgrade to a modern data platform by detecting compatibility issues that can impact database functionality in your new version of SQL Server or Azure SQL Database. DMA recommends performance and reliability improvements for the target environment and allows us to move the schema, data, and uncontained objects from your source server to your target server.

This tool must be used as an integral approach for identifying any potential issue and remediate it before it arrives at Cloud Production. In the following link you can find a detailed explanation about the use of DMA on target Production on premise environment.

Selecting a Migration Tool

There are different methods available for various scenarios. The selection largely depends on downtime, database size, and network speed/quality.

Here is a comparison of various migration methods.

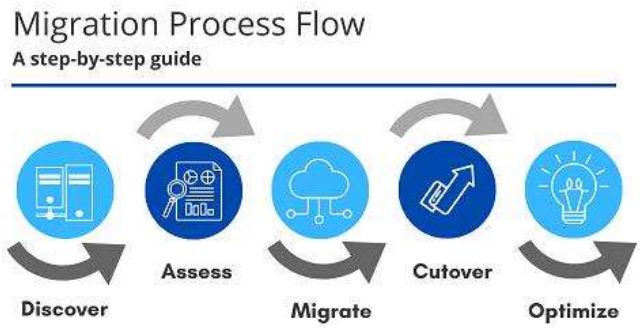
Migration Method	Description	Downtime	Recommendations
SQL Server Management Studio- deploy database to Azure SQL Database	Wizard-based GUI for exporting an on-premises database to bacpac and importing the bacpac into Azure SQL Database. This only applies to Azure SQL Database.	Yes	Best to quickly deploy test/development databases to Azure SQL Database.
SQLPackage.exe	Command-line utility for exporting an on-premises database to bacpac and importing the bacpac to Azure SQL Database. It can be used to migrate a database to Azure SQL Database and Managed Instance.	Yes	Uses parallel bcp when importing data into an Azure SQL Database. Suitable for production and development/test database migration.
Data Migration Assistant	Free wizard-based GUI tool from Microsoft. Detects and list	Yes	

	compatibility issues. Uses T-SQL scripts for schema migration and bcp to copy data. Option to choose database objects to migrate. It can be used to migrate a database to Azure SQL database and Managed Instance.		Recommended for production, development, and test database migration.
Transactional replication	Azure SQL Database as a subscriber to an on-premises (SQL Server 2012 onward) SQL Server Publisher. Use the transaction replication filtering feature to migrate selective data. It can be used to migrate a database to Azure SQL Database and Managed Instance.	Near-zero downtime. In-process transactions may be affected when switching the application connection from on-premises to Azure SQL Databases. However, this can be managed using retry logic in the application.	Recommended for zero-downtime production migration.
Azure Database Migration Service	Fully managed service to migrate schema and data. Incurs additional costs. Supports migration from heterogeneous databases. It can be used to migrate a database to Azure SQL Database and Managed Instance.	Near-zero downtime. In-process transactions may be affected when switching the application connection from on-premises to Azure SQL Databases. However, this can be managed using retry logic in the application.	Recommended for zero-downtime production migration.
Log Replay Service	Azure Database Migration Service and LRS use the same underlying migration technology and the same APIs. By releasing LRS, we are further enabling complex custom migrations and hybrid architecture between on-premises SQL Server and SQL Managed Instance.	Near-zero downtime. LRS can support up to 100 simultaneous restore processes per single managed instance.	Recommended for zero-downtime production migration. Only apply to Managed Instance.
Azure SQL Migration extension for Azure Data Studio	The Azure SQL Migration extension for Azure Data Studio enables you to use the new SQL Server assessment and migration capability in Azure Data Studio.	Yes	Recommended for production, development, and test database migration. Only apply for SQL Managed Instance and SQL Server on Azure VM

Implementing the Migration

It is important to be clear about the iterative nature of the migration methodology. This stage can be composed of three substages, these are: pre-migration (preparation tasks), migration execution, and post migration.

We could analyze the typical Migration Process Flow according to the following image which is a straightforward way to summarize it.



Source: Microsoft

Pre-Migration

It is important to prepare a list of tasks which should be executed in order and completed without issues, in case of any problem detected, we should fix before to advance to the migration, here is a list of proposed tasks that need to be part of this stage.

Additionally, as part of the Load Tests, we have the option to use a tool called Microsoft® Database Experimentation Assistant (DEA) that can perfectly fit to a standard assess process after that you have prepared executed the migration rehearse, what exactly it does mean? We can test our Migration process in a PRE environment or even on PROD but without pointing out our application to new Azure DB, it can involve more work, but give us more confidence.

More details about it are in the following links:

https://cutt.ly/EElF93E

https://cutt.ly/kElF8D8

An additional tool that can be used for evaluating the proper Service Tier features is the DTO Calculator, which helps us to evaluate the tolerance to the new server on the base of the workload, it is measured through the PerfMon counter. The translation of the results toward SQL Manage Instance option will be does it by Microsoft Fastrack engineer as part of her compromise during the migration process. More about the DTO Calculator in this link:

https://dtucalculator.azurewebsites.net/

Migration Execution

On the base of the feedback that we have gotten during the pre-migration, we are able to add any additional step, taking account of the stats, time, and other important components. For the migration execution we should include some tasks like:

- Progressing on Execution Plan tasks: It should include a proper definition and interaction between the Database side and the application side plus DevOps.
- Measuring the progress on the execution plan.
- Test the interaction between the application and database behind the curtains.
- Open access to customers for the application.

Post Migration

It is important to have a proper monitoring tool for evaluating the post migration stage. In our case, we are going to use Red-Gate Monitoring tool together with Query Store. It is important to pay attention to a group of standards SQL Server metric which includes waits, CPU and Memory consumption, so based on the threshold defined during the Load Tests executed in the Pre-Migration stage we are going to have a realistic expectation to compare with. In case of detecting some problems during this stage, we should take the decision about the path to follow, it will depend on the kind of problem experienced, but always having in mind that increase the resource (and billing) should not be the first option.