

Contents

[Database Migration Service documentation](#)

[Overview](#)

[What is Azure Database Migration Service?](#)

[Quickstarts](#)

[Create service - Portal](#)

[Create service in hybrid mode - Portal](#)

[Tutorials](#)

[Migrate SQL Server to Azure SQL DB single database offline](#)

[Migrate SQL Server to Azure SQL DB single database online](#)

[Migrate SQL Server to Azure SQL DB managed instance offline](#)

[Migrate SQL Server to Azure SQL DB managed instance online](#)

[Migrate RDS SQL Server to Azure SQL DB single database or Azure SQL DB managed instance online](#)

[Migrate MySQL to Azure DB for MySQL online](#)

[Migrate RDS MySQL to Azure DB for MySQL online](#)

[Migrate PostgreSQL to Azure DB for PostgreSQL online via the portal](#)

[Migrate PostgreSQL to Azure DB for PostgreSQL online via the CLI](#)

[Migrate RDS PostgreSQL to Azure DB for PostgreSQL online](#)

[Migrate MongoDB to Azure Cosmos DB Mongo API offline](#)

[Migrate MongoDB to Azure Cosmos DB Mongo API online](#)

[Migrate Oracle to Azure DB for PostgreSQL—Single server online](#)

[How-to guides](#)

[Use PowerShell to migrate SQL Server to Azure SQL DB single database](#)

[Use PowerShell to migrate SQL Server to Azure SQL DB managed instance](#)

[Monitor migration activity](#)

[Redeploy SSIS packages to Azure SQL DB single database](#)

[Redeploy SSIS packages to an Azure SQL DB managed instance](#)

[Resources](#)

[Database migration scenario status](#)

[Overview of prerequisites](#)

[Network topologies for Azure SQL DB MI migrations](#)

[Custom roles for online migrations from SQL Server to Azure SQL DB MI](#)

[Troubleshooting and known issues](#)

[Troubleshoot source database connectivity issues](#)

[Troubleshoot common issues and errors](#)

[Known Issues - Online migration to Azure SQL DB single database](#)

[Known Issues - Online migration to Azure SQL DB managed instance](#)

[Known Issues - Online migration to Azure DB for MySQL](#)

[Known Issues - Online migration from PostgreSQL to Azure DB for PostgreSQL](#)

[Known Issues - Online migration from Oracle to Azure DB for PostgreSQL – Single server](#)

[Known Issues - Using hybrid mode](#)

[Videos](#)

[Use the Database Migration Guide](#)

[The migration process and recommended tools/services](#)

[Address prerequisites and create a DMS instance](#)

[Migrate SQL Server 2008 to Azure SQL DB managed instance](#)

[Migrate PostgreSQL to Azure DB for PostgreSQL](#)

[Monitor an online migration and perform cutover](#)

[Migrate Oracle to Azure SQL DB](#)

[Migrate MongoDB to Azure Cosmos DB](#)

[Tools and guidance](#)

[Services and tools available for data migration scenarios](#)

[Azure Database Migration Guide](#)

[Data Migration Assistant](#)

[SQL Server Migration Assistant](#)

[Database Experimentation Assistant](#)

[Data Access Migration Toolkit](#)

[Frequently Asked Questions](#)

[User Voice Feedback](#)

[Pricing](#)

[Service updates](#)

Azure Roadmap

What is Azure Database Migration Service?

2/26/2020 • 2 minutes to read • [Edit Online](#)

Azure Database Migration Service is a fully managed service designed to enable seamless migrations from multiple database sources to Azure data platforms with minimal downtime (online migrations).

Migrate databases to Azure with familiar tools

Azure Database Migration Service integrates some of the functionality of our existing tools and services. It provides customers with a comprehensive, highly available solution. The service uses the [Data Migration Assistant](#) to generate assessment reports that provide recommendations to guide you through the changes required prior to performing a migration. It's up to you to perform any remediation required. When you're ready to begin the migration process, Azure Database Migration Service performs all of the required steps. You can fire and forget your migration projects with peace of mind, knowing that the process takes advantage of best practices as determined by Microsoft.

NOTE

Using Azure Database Migration Service to perform an online migration requires creating an instance based on the Premium pricing tier.

Regional availability

For up-to-date info about regional availability of Azure Database Migration Service, see [Products available by region](#).

Pricing

For up-to-date info about Azure Database Migration Service pricing, see [Azure Database Migration Service pricing](#).

Next steps

- [Status of migration scenarios supported by Azure Database Migration Service](#).
- [Create an instance of Azure Database Migration Service by using the Azure portal](#).
- [Migrate SQL Server to Azure SQL Database](#).
- [Overview of prerequisites for using Azure Database Migration Service](#).
- [FAQ about using Azure Database Migration Service](#).
- [Services and tools available for data migration scenarios](#).

Quickstart: Create an instance of the Azure Database Migration Service by using the Azure portal

1/8/2020 • 2 minutes to read • [Edit Online](#)

In this Quickstart, you use the Azure portal to create an instance of Azure Database Migration Service. After you create the service, you can use it to migrate data from SQL Server on-premises to Azure SQL Database.

If you don't have an Azure subscription, create a [free](#) account before you begin.

Sign in to the Azure portal

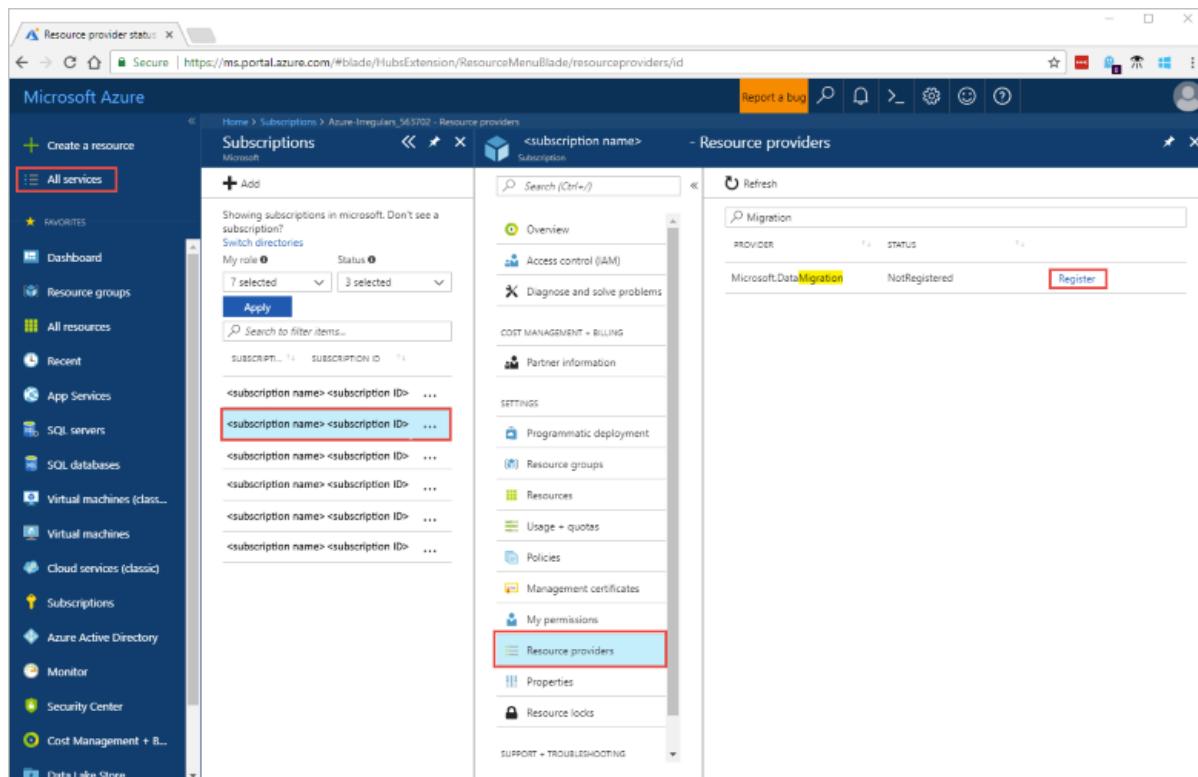
Open your web browser, navigate to the [Microsoft Azure portal](#), and then enter your credentials to sign in to the portal.

The default view is your service dashboard.

Register the resource provider

Register the Microsoft.DataMigration resource provider before you create your first instance of the Database Migration Service.

1. In the Azure portal, select **All services**, and then select **Subscriptions**.
2. Select the subscription in which you want to create the instance of Azure Database Migration Service, and then select **Resource providers**.
3. Search for migration, and then to the right of **Microsoft.DataMigration**, select **Register**.



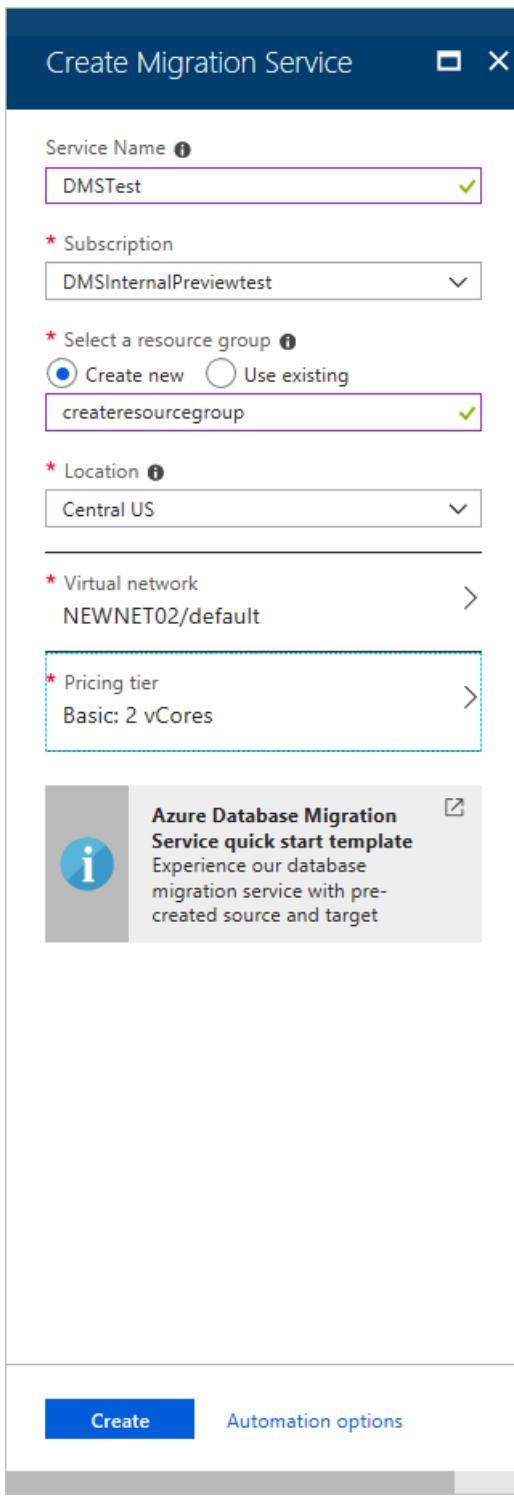
Create an instance of the service

1. Select +**Create a resource** to create an instance of Azure Database Migration Service.
2. Search the marketplace for "migration", select **Azure Database Migration Service**, and then on the **Azure Database Migration Service** screen, select **Create**.
3. On the **Create Migration Service** screen:
 - Choose a **Service Name** that is memorable and unique to identify your instance of Azure Database Migration Service.
 - Select the Azure **Subscription** in which you want to create the instance.
 - Select an existing **Resource Group** or create a new one.
 - Choose the **Location** that is closest to your source or target server.
 - Select an existing **Virtual network** or create one.

The virtual network provides Azure Database Migration Service with access to the source database and target environment.

For more information on how to create a virtual network in the Azure portal, see the article [Create a virtual network using the Azure portal](#).

- Select Basic: 1 vCore for the **Pricing tier**.



4. Select **Create**.

After a few moments, your instance of Azure Database Migration service is created and ready to use. Azure Database Migration Service displays as shown in the following image:

The screenshot shows the 'Overview' page of the Azure Database Migration Service. At the top, there's a search bar and several navigation links: '+ New Migration Project', 'Delete service', 'Refresh', 'Start Service', and 'Stop Service'. A green success message box says, 'Great job! Your database migration service was successfully created. You can create your first migration project now.' On the left, a sidebar has sections for 'Overview', 'Activity log', 'Access control (IAM)', 'Tags', 'SETTINGS' (with 'Properties', 'Locks', and 'Automation script'), and 'SUPPORT + TROUBLESHOOTING' (with 'New support request'). The main content area has two tabs: 'Essentials' (selected) and 'SQL DB Content'. Under 'Essentials', it shows a resource group 'edmacarg', status 'Online', location 'westus', subscription ID '<subscription id>', and service/UI version '3.4.4038.1/3.4.4038.1'. Under 'SQL DB Content', it shows 'SKU: Basic: 1 vCore'. Below this is a 'Migration Projects' section with a table:

NAME	SOURCE	TARGET	CREATED
No database migration projects to display			

At the bottom, a message says, 'Great job! Your database migration service was successfully created. You can create your first migration project now.' followed by a blue 'New migration project' button.

Clean up resources

You can clean up the resources created in this Quickstart by deleting the [Azure resource group](#). To delete the resource group, navigate to the instance of the Azure Database Migration Service that you created. Select the **Resource group** name, and then select **Delete resource group**. This action deletes all assets in the resource group as well as the group itself.

Next steps

[Migrate SQL Server on-premises to Azure SQL Database](#)

Quickstart: Create a hybrid mode instance with Azure portal & Azure Database Migration Service

1/19/2020 • 6 minutes to read • [Edit Online](#)

Azure Database Migration Service hybrid mode manages database migrations by using a migration worker that's hosted on-premises together with an instance of Azure Database Migration Service running in the cloud. Hybrid mode is especially useful for scenarios in which there's a lack of site-to-site connectivity between the on-premises network and Azure or if there's limited site-to-site connectivity bandwidth.

NOTE

Currently, Azure Database Migration Service running in hybrid mode supports SQL Server migrations to:

- Azure SQL Database managed instance with near zero downtime (online).
- Azure SQL Database single database with some downtime (offline).
- MongoDB to Azure CosmosDB with near zero downtime (online).
- MongoDB to Azure CosmosDB with some downtime (offline).

In this Quickstart, you use the Azure portal to create an instance of Azure Database Migration Service in hybrid mode. Afterwards, you download, install, and set up the hybrid worker in your on-premises network. During preview, you can use Azure Database Migration Service hybrid mode to migrate data from an on-premises instance of SQL Server to Azure SQL Database.

NOTE

The Azure Database Migration Service hybrid installer runs on Microsoft Windows Server 2012 R2, Windows Server 2016, Windows Server 2019, and Windows 10.

IMPORTANT

The Azure Database Migration Service hybrid installer requires .NET 4.7.2 or later. To find the latest versions of .NET, see the [Download .NET Framework](#) page.

If you don't have an Azure subscription, create a [free](#) account before you begin.

Sign in to the Azure portal

Open your web browser, navigate to the [Microsoft Azure portal](#), and then enter your credentials to sign in to the portal.

The default view is your service dashboard.

Register the resource provider

Register the Microsoft.DataMigration resource provider before you create your first instance of Azure Database Migration Service.

1. In the Azure portal, select **Subscriptions**, select the subscription in which you want to create the instance of Azure Database Migration Service, and then select **Resource providers**.

The screenshot shows the Microsoft Azure Subscriptions page. On the left, there's a list of subscriptions. A specific subscription is selected, indicated by a red box around its name. In the center, there's a sidebar with various management options. Under the 'Resource providers' section, which is also highlighted with a red box, there's a table showing one provider: 'Microsoft.DataMigration' with a status of 'Unregistered'. At the top right, there are tabs for 'Manage', 'Cancel subscription', 'Rename', and 'Change directory'.

2. Search for migration, and then to the right of **Microsoft.DataMigration**, select **Register**.

This screenshot is similar to the previous one, but the 'Microsoft.DataMigration' provider has been registered. The 'Status' column now shows 'Registered' instead of 'Unregistered'. The 'Register' button, which was previously highlighted with a red box, is now grayed out, indicating the action has been completed.

Create an instance of the service

1. Select +**Create a resource** to create an instance of Azure Database Migration Service.
2. Search the Marketplace for "migration", select **Azure Database Migration Service**, and then on the **Azure Database Migration Service** screen, select **Create**.
3. On the **Create Migration Service** screen:
 - Choose a **Service Name** that is memorable and unique to identify your instance of Azure Database Migration Service.
 - Select the Azure **Subscription** in which you want to create the instance.

- Select an existing **Resource Group** or create a new one.
- Choose the **Location** that is closest to your source or target server.
- For **Service mode**, select **Hybrid (Preview)**.

Create Migration Service

Basics Networking Tags Review + create

Azure Database Migration Service is designed to streamline the process of migrating on-premises databases to Azure. [Learn more.](#)

Project details

Select the subscription to manage deployed resources and constants. Use resource groups as you would folders, to organize and manage all of your resources.

Subscription * ⓘ	DMSInternalPreviewtest
Resource group * ⓘ	(New) NewResourcegroup
	Create new

Instance details

Migration service name * ⓘ	DemoHybridDMS
Location * ⓘ	East US
Service mode * ⓘ	Azure Hybrid (Preview)

All migration activities are run from an on-premises machine.

Tip: Use an Azure Database Migration Service quick start template with pre-created source and targets. [Learn more.](#)

Review + create **Next : Networking >>**

4. Select **Review + create**.
5. On the **Review + create** tab, review the Terms, verify the other information provided, and then select **Create**.

Create Migration Service

Basics Networking Tags **Review + create**

Project details

Database Migration Service

by Microsoft

[Terms of use](#) | [Privacy policy](#)

Terms

By clicking "Create", I (a) agree to the legal terms and privacy statement(s) associated with the Marketplace offering(s) listed above; (b) authorize Microsoft to bill my current payment method for the fees associated with the offering(s), with the same billing frequency as my Azure subscription; and (c) agree that Microsoft may share my contact, usage and transactional information with the provider(s) of the offering(s) for support, billing and other transactional activities. Microsoft does not provide rights for third-party offerings. For additional details see [Azure Marketplace Terms](#).

Basics

Subscription	DMSInternalPreviewtest
Resource group	NewResourcegroup
Migration service name	DemoHybridDMS
Region	East US
Location type	Hybrid (Preview)

Networking

Virtual network --

Tags

Create

<< Previous

Automation options

After a few moments, your instance of Azure Database Migration Service in hybrid mode is created and ready to set up. The Azure Database Migration Service instance displays as shown in the following image:

<Instance name>
Azure Database Migration Service

Search (Ctrl+ /)

+ New Migration Project Delete service Refresh Start Service Stop Service

✓ Great job! Your database migration service was successfully created. You can create your first migration project now.

Resource group : <resource group>	Status : Online
Virtual network & IP Ad... : ---	Location : East US
Subscription : <subscription name>	Subscription ID : <subscription ID>
SKU : ---	Service/UI Version : 4.4.4567.6/4.4.4581.11
Tags (change) : Click here to add tags	

Name ↑ Source ↓ Target ↑ ↓ Created

No database migration projects to display

Great job! Your database migration service was successfully created. You can create your first migration project now.

New migration project

6. After the service created, select **Properties**, and then copy the value displayed in the **Resource Id** box, which you'll use to install the Azure Database Migration Service hybrid worker.

Create Azure App registration ID

You need to create an Azure App registration ID that the on-premises hybrid worker can use to communicate with Azure Database Migration Service in the cloud.

1. In the Azure portal, select **Azure Active Directory**, select **App registrations**, and then select **New registration**.
2. Specify a name for the application, and then, under **Supported account types**, select the type of accounts to support to specify who can use the application.

Register an application

⚠️ If you are building an application for external users that will be distributed by Microsoft, you must register as a first party application to meet all security, privacy, and compliance policies. [Read our decision guide](#)

*** Name**
The user-facing display name for this application (this can be changed later).

Supported account types
Who can use this application or access this API?
 Accounts in this organizational directory only (Microsoft only - Single tenant)
 Accounts in any organizational directory (Any Azure AD directory - Multitenant)
 Accounts in any organizational directory (Any Azure AD directory - Multitenant) and personal Microsoft accounts (e.g. Skype, Xbox)
[Help me choose...](#)

Redirect URI (optional)
We'll return the authentication response to this URI after successfully authenticating the user. Providing this now is optional and it can be changed later, but a value is required for most authentication scenarios.

By proceeding, you agree to the Microsoft Platform Policies

3. Use the default values for the **Redirect URI (optional)** fields, and then select **Register**.
4. After App ID registration is completed, make a note of the **Application (client) ID**, which you'll use while

installing the hybrid worker.

5. In the Azure portal, navigate to Azure Database Migration Service, select **Access control (IAM)**, and then select **Add role assignment** to assign contributor access to the App ID.

The screenshot shows the 'Access control (IAM)' page for an Azure Database Migration Service resource named '<name>'. The left sidebar includes links for Overview, Activity log, Access control (IAM), Tags, Settings (Configuration, Hybrid, Properties, Locks, Export template), Support + troubleshooting (Resource health, New support request), and a search bar. The main area has tabs for Add role assignment, Deny assignments, Classic administrators, and Roles. A red box highlights the 'Add role assignment' tab. Below it, there's a 'Check access' section with a dropdown set to 'Azure AD user, group, or service principal' and a search bar. To the right are three cards: 'Add a role assignment' (Grant access by assigning a role to a user, group, service principal, or managed identity), 'View role assignments' (View users, groups, service principals, and managed identities with role assignments), and 'View deny assignments' (View users, groups, service principals, and managed identities denied access).

6. Select **Contributor** as the role, assign access to **Azure AD user, or service principal**, and then select the App ID name.

Add role assignment

Role ⓘ
Contributor

Assign access to ⓘ
Azure AD user, group, or service principal

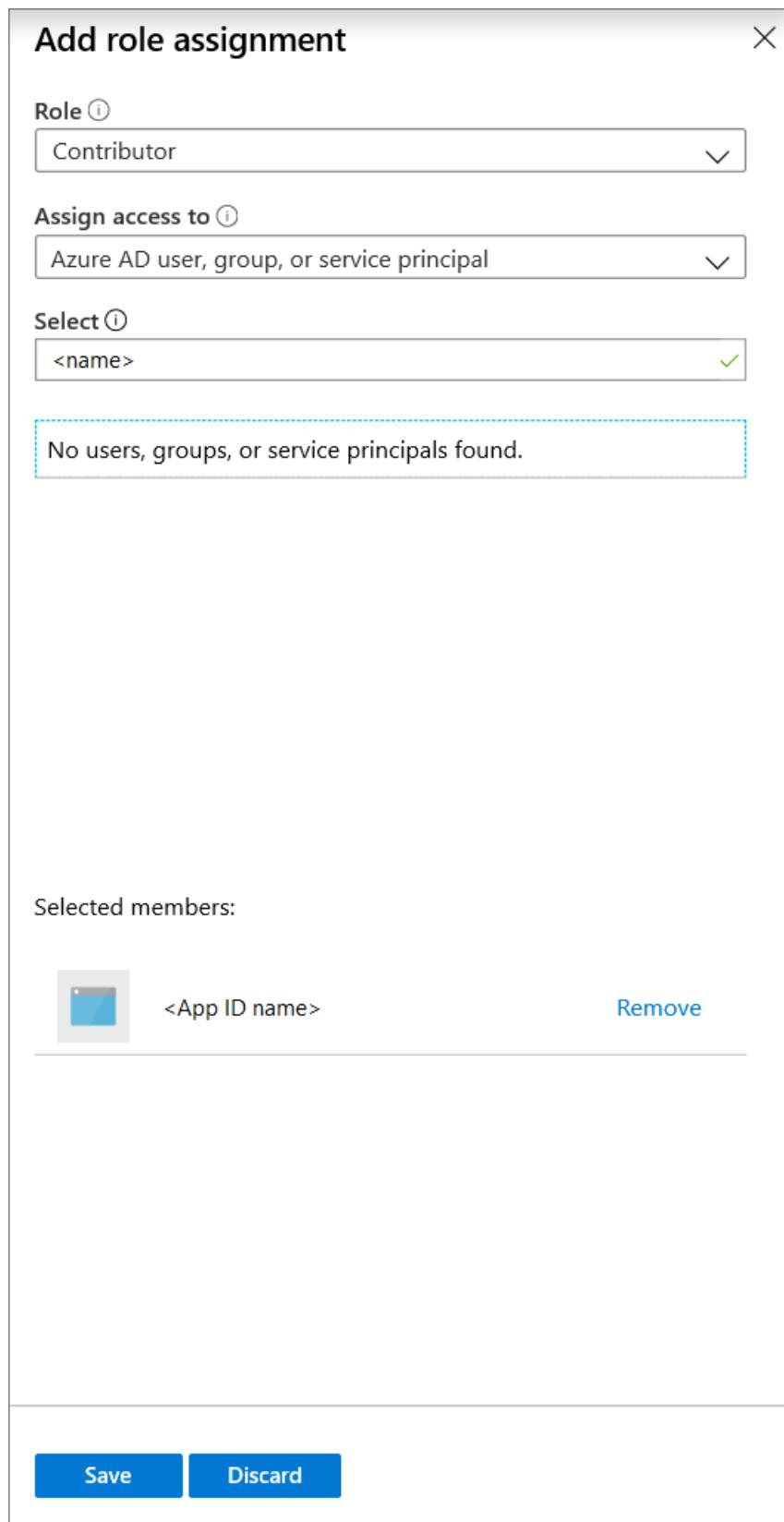
Select ⓘ
<name>

No users, groups, or service principals found.

Selected members:

	<App ID name>	Remove
-------------------------------------------------------------------------------------	---------------	--------

Save **Discard**



7. Select **Save** to save the role assignment for the App ID on the Azure Database Migration Service resource.

Download and install the hybrid worker

1. In the Azure portal, navigate to your instance of Azure Database Migration Service.
2. Under **Settings**, select **Hybrid**, and then select **Installer download** to download the hybrid worker.

The screenshot shows the Azure Database Migration Service hybrid worker configuration page. The left sidebar has a 'Hybrid' tab selected, indicated by a red box. At the top right, there is a link labeled 'Installer download' with a red box around it. Below this, a message says 'Please download the installer to register a hybrid worker.' The main area shows a table titled 'Hybrid workers' with one row: 'No hybrid workers to display.'

- Extract the ZIP file on the server that will be hosting the Azure Database Migration Service hybrid worker.

IMPORTANT

The Azure Database Migration Service hybrid installer requires .NET 4.7.2 or later. To find the latest versions of .NET, see the [Download .NET Framework](#) page.

- In the install folder, locate and open the **dmsSettings.json** file, specify the **ApplicationId** and **resourceId**, and then save the file.

The screenshot shows a Windows File Explorer window with the path 'Computer > Local Disk (C:) > Rajpo > HybridDMS > Install'. The 'dmsSettings' file is highlighted with a red box. In the foreground, a Notepad window displays the JSON content of the 'dmsSettings' file:

```
{  
  "Authentication": {  
    "isCertificateAuthentication": true,  
    "ApplicationId": "[Azure AD App Id]",  
    "CertificateThumbprint": "[placeholder]"  
  },  
  "DMSService": {  
    "resourceId": "[Cloud DMS resource id]",  
    "subscriptionId": "[placeholder]",  
    "resourceGroup": "[placeholder]",  
    "serviceName": "[placeholder]"  
  }  
}
```

- Generate a certificate that Azure Database Migration Service can use to authenticate the communication from the hybrid worker by using the following command.

```
<drive>:\<folder>\Install>DMSWorkerBootstrap.exe -a GenerateCert
```

A certificate is generated in the Install folder.

Name	Date modified	Type	Size
dbgshim.dll	9/12/2019 11:05 PM	Application extension	133 KB
DMS_Hybrid_App_Key.pfx	10/16/2019 8:49 PM	Security Certificate	2 KB
DMS_HYBRID_EULA	9/23/2019 4:05 PM	Text Document	1 KB
dmsSettings	10/16/2019 8:49 PM	JSON File	1 KB
DMSWorkerBootstrap.deps	10/15/2019 2:51 PM	JSON File	154 KB
DMSWorkerBootstrap.dll	10/15/2019 2:52 PM	Application extension	141 KB
DMSWorkerBootstrap.json	10/15/2019 2:52 PM	Application	156 KB
DMSWorkerBootstrap.pdb	10/15/2019 2:52 PM	PDB File	42 KB
DMSWorkerBootstrap.runtimeconfig.dev	10/15/2019 2:51 PM	JSON File	1 KB
DMSWorkerBootstrap.runtimeconfig	10/15/2019 2:51 PM	JSON File	1 KB
hostfxr.dll	9/13/2019 3:09 PM	Application extension	582 KB
hostpolicy.dll	9/13/2019 3:09 PM	Application extension	576 KB
hostSettings.json	9/12/2019 11:41 AM	JSON File	1 KB

6. In the Azure portal, navigate to the App ID, under **Manage**, select **Certified & secrets**, and then select **Upload certificate** to select the public certificate you generated.

Thumbprint	Start Date	Expires
<thumbprint value>	10/16/2019	10/16/2021

7. Install the Azure Database Migration Service hybrid worker on your on-premises server by running the following command:

```
<drive>:\<folder>\Install>DMSWorkerBootstrap.exe -a Install -IAcceptDMSLicenseTerms
```

NOTE

When running the install command, you can also use the following parameters:

- **-TelemetryOptOut** - Stops the worker from sending telemetry but continues to log locally minimally. The installer still sends telemetry.
- **-p {InstallLocation}**: Enables changing the installation path, which by default is "C:\Program Files\DatabaseMigrationServiceHybrid".

8. If the installer runs without error, then the service will show an online status in Azure Database Migration Service and you're ready to migrate your databases.

The screenshot shows the Azure Database Migration Service instance page. At the top, there's a search bar and several navigation links: Overview, Activity log, Access control (IAM), Tags, Configuration, Hybrid, Properties, Locks, Export template, Support + troubleshooting, Resource health, and New support request. A green banner at the top right says "Great job! Your database migration service was successfully created. You can create your first migration project now." Below this, there are sections for Resource group, Virtual network & IP Ad..., Subscription, SKU, and Tags (change). On the right, it shows Status: Online, Location: East US, Subscription ID: <subscription ID>, and Service/UI Version: 4.4.4567.6/4.4.4581.11. A table below lists migration projects with columns for Name, Source, Target, and Created. A message at the bottom says "Great job! Your database migration service was successfully created. You can create your first migration project now." and a "New migration project" button.

Uninstall Azure Database Migration Service hybrid mode

Currently, uninstalling Azure Database Migration Service hybrid mode is supported only via the Azure Database Migration Service hybrid worker installer on your on-premises server, by using the following command:

```
<drive>:\<folder>\Install>DMSWorkerBootstrap.exe -a uninstall
```

NOTE

When running the uninstall command, you can also use the "-ReuseCert" parameter, which keeps the AdApp cert generated by the generateCert workflow. This enables using the same cert that was previously generated and uploaded.

Set up the Azure Database Migration Service hybrid worker using PowerShell

In addition to installing the Azure Database Migration Service hybrid worker via the Azure portal, we provide a [PowerShell script](#) that you can use to automate the worker installation steps after you create a new instance of Azure Database Migration Service in hybrid mode. The script:

1. Creates a new AdApp.
2. Downloads the installer.
3. Runs the generateCert workflow.
4. Uploads the certificate.
5. Adds the AdApp as contributor to your Azure Database Migration Service instance.
6. Runs the install workflow.

This script is intended for quick prototyping when the user already has all the necessary permissions in the environment. Note that in your production environment, the AdApp and Cert may have different requirements, so the script could fail.

IMPORTANT

This script assumes that there is an existing instance of Azure Database Migration Service in hybrid mode and that the Azure account used has permissions to create AdApps in the tenant and to modify RBAC on the subscription.

Fill in the parameters at the top of the script, and then run the script from an Administrator PowerShell instance.

Next steps

[Migrate SQL Server to an Azure SQL Database managed instance online](#) [Migrate SQL Server to a single database or pooled database in Azure SQL Database offline](#)

Tutorial: Migrate SQL Server to a single database or pooled database in Azure SQL Database offline using DMS

1/8/2020 • 12 minutes to read • [Edit Online](#)

You can use Azure Database Migration Service to migrate the databases from an on-premises SQL Server instance to [Azure SQL Database](#). In this tutorial, you migrate the **Adventureworks2012** database restored to an on-premises instance of SQL Server 2016 (or later) to a single database or pooled database in Azure SQL Database by using Azure Database Migration Service.

In this tutorial, you learn how to:

- Assess your on-premises database by using the Data Migration Assistant.
- Migrate the sample schema by using the Data Migration Assistant.
- Create an instance of Azure Database Migration Service.
- Create a migration project by using Azure Database Migration Service.
- Run the migration.
- Monitor the migration.
- Download a migration report.

TIP

When you migrate databases to Azure by using Azure Database Migration Service, you can do an *offline* or an *online* migration. With an offline migration, application downtime starts when the migration starts. With an online migration, downtime is limited to the time to cut over at the end of migration. We suggest that you test an offline migration to determine whether the downtime is acceptable; if not, do an online migration.

This article describes an offline migration from SQL Server to a single database or pooled database in Azure SQL Database. For an online migration, see [Migrate SQL Server to Azure SQL Database online using DMS](#).

Prerequisites

To complete this tutorial, you need to:

- Download and install [SQL Server 2016 or later](#).
- Enable the TCP/IP protocol, which is disabled by default during SQL Server Express installation, by following the instructions in the article [Enable or Disable a Server Network Protocol](#).
- Create a single (or pooled) database in Azure SQL Database, which you do by following the detail in the article [Create a single database in Azure SQL Database using the Azure portal](#).

NOTE

If you use SQL Server Integration Services (SSIS) and want to migrate the catalog database for your SSIS projects/packages (SSISDB) from SQL Server to Azure SQL Database, the destination SSISDB will be created and managed automatically on your behalf when you provision SSIS in Azure Data Factory (ADF). For more information about migrating SSIS packages, see the article [Migrate SQL Server Integration Services packages to Azure](#).

- Download and install the [Data Migration Assistant](#) v3.3 or later.
- Create a Microsoft Azure Virtual Network for Azure Database Migration Service by using the Azure Resource Manager deployment model, which provides site-to-site connectivity to your on-premises source servers by using either [ExpressRoute](#) or [VPN](#). For more information about creating a virtual network, see the [Virtual Network Documentation](#), and especially the quickstart articles with step-by-step details.

NOTE

During virtual network setup, if you use ExpressRoute with network peering to Microsoft, add the following service endpoints to the subnet in which the service will be provisioned:

- Target database endpoint (for example, SQL endpoint, Cosmos DB endpoint, and so on)
- Storage endpoint
- Service bus endpoint

This configuration is necessary because Azure Database Migration Service lacks internet connectivity.

If you don't have site-to-site connectivity between the on-premises network and Azure or if there is limited site-to-site connectivity bandwidth, consider using Azure Database Migration Service in hybrid mode (Preview). Hybrid mode leverages an on-premises migration worker together with an instance of Azure Database Migration Service running in the cloud. To create an instance of Azure Database Migration Service in hybrid mode, see the article [Create an instance of Azure Database Migration Service in hybrid mode using the Azure portal](#).

- Ensure that your virtual network Network Security Group rules don't block the following inbound communication ports to Azure Database Migration Service: 443, 53, 9354, 445, 12000. For more detail on Azure virtual network NSG traffic filtering, see the article [Filter network traffic with network security groups](#).
- Configure your [Windows Firewall for database engine access](#).
- Open your Windows firewall to allow Azure Database Migration Service to access the source SQL Server, which by default is TCP port 1433.
- If you're running multiple named SQL Server instances using dynamic ports, you may wish to enable the SQL Browser Service and allow access to UDP port 1434 through your firewalls so that Azure Database Migration Service can connect to a named instance on your source server.
- When using a firewall appliance in front of your source database(s), you may need to add firewall rules to allow Azure Database Migration Service to access the source database(s) for migration.
- Create a server-level IP [firewall rule](#) for the Azure SQL Database server to allow Azure Database Migration Service access to the target databases. Provide the subnet range of the virtual network used for Azure Database Migration Service.
- Ensure that the credentials used to connect to source SQL Server instance have [CONTROL SERVER](#) permissions.
- Ensure that the credentials used to connect to target Azure SQL Database instance have [CONTROL DATABASE](#) permission on the target Azure SQL databases.

Assess your on-premises database

Before you can migrate data from an on-premises SQL Server instance to a single database or pooled database in Azure SQL Database, you need to assess the SQL Server database for any blocking issues that might prevent migration. Using the Data Migration Assistant v3.3 or later, follow the steps described in the article [Performing a SQL Server migration assessment](#) to complete the on-premises database assessment. A summary of the required steps follows:

1. In the Data Migration Assistant, select the New (+) icon, and then select the **Assessment** project type.
2. Specify a project name, in the **Source server type** text box, select **SQL Server**, in the **Target server type** text box, select **Azure SQL Database**, and then select **Create** to create the project.

When you're assessing the source SQL Server database migrating to a single database or pooled database in Azure SQL Database, you can choose one or both of the following assessment report types:

- Check database compatibility
- Check feature parity

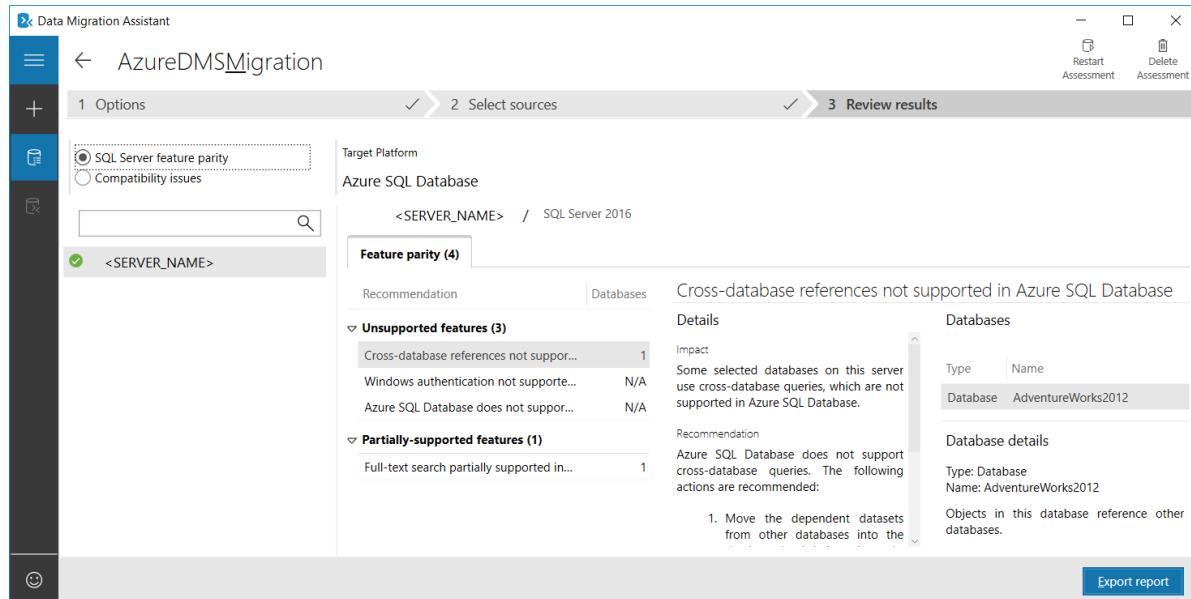
Both report types are selected by default.

3. In the Data Migration Assistant, on the **Options** screen, select **Next**.
4. On the **Select sources** screen, in the **Connect to a server** dialog box, provide the connection details to your SQL Server, and then select **Connect**.
5. In the **Add sources** dialog box, select **AdventureWorks2012**, select **Add**, and then select **Start Assessment**.

NOTE

If you use SSIS, DMA does not currently support the assessment of the source SSISDB. However, SSIS projects/packages will be assessed/validated as they are redeployed to the destination SSISDB hosted by Azure SQL Database. For more information about migrating SSIS packages, see the article [Migrate SQL Server Integration Services packages to Azure](#).

When the assessment is complete, the results display as shown in the following graphic:



For single databases or pooled databases in Azure SQL Database, the assessments identify feature parity issues and migration blocking issues for deploying to a single database or pooled database.

6. Review the assessment results for migration blocking issues and feature parity issues by selecting the
- The **SQL Server feature parity** category provides a comprehensive set of recommendations, alternative approaches available in Azure, and mitigating steps to help you plan the effort into your migration projects.
 - The **Compatibility issues** category identifies partially supported or unsupported features that reflect compatibility issues that might block migrating on-premises SQL Server database(s) to Azure SQL Database. Recommendations are also provided to help you address those issues.

specific options.

Migrate the sample schema

After you're comfortable with the assessment and satisfied that the selected database is a viable candidate for migration to a single database or pooled database in Azure SQL Database, use DMA to migrate the schema to Azure SQL Database.

NOTE

Before you create a migration project in Data Migration Assistant, be sure that you have already provisioned an Azure SQL database as mentioned in the prerequisites. For purposes of this tutorial, the name of the Azure SQL Database is assumed to be **AdventureWorksAzure**, but you can provide whatever name you wish.

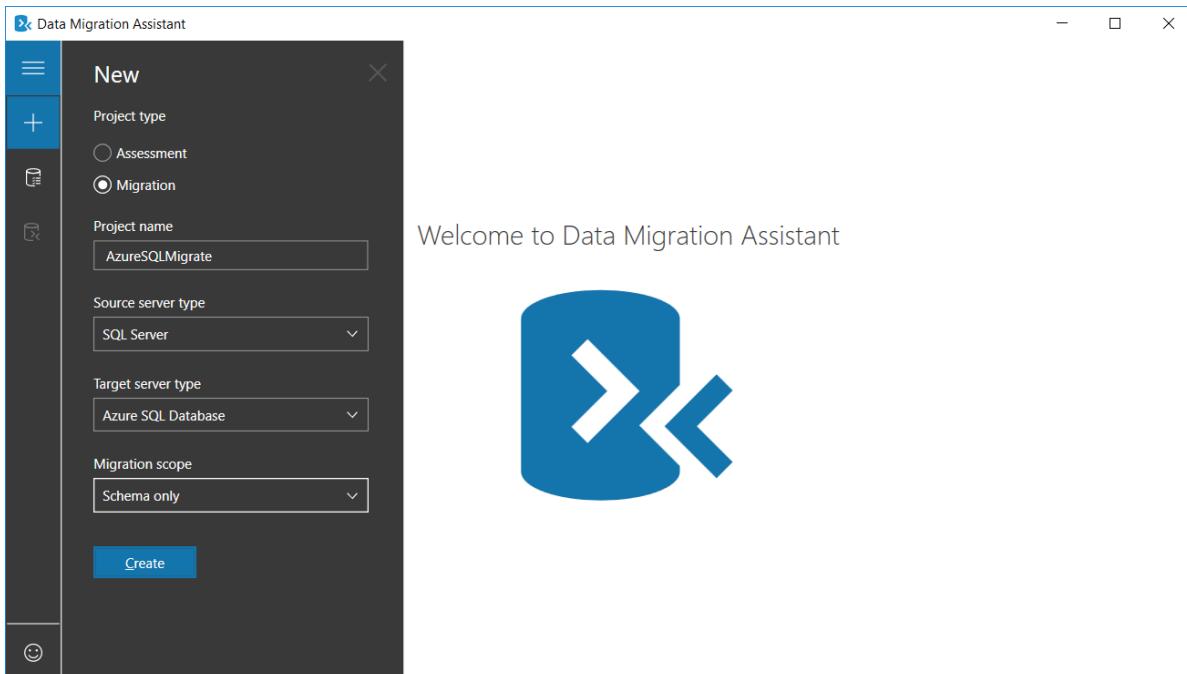
IMPORTANT

If you use SSIS, DMA does not currently support the migration of source SSISDB, but you can redeploy your SSIS projects/packages to the destination SSISDB hosted by Azure SQL Database. For more information about migrating SSIS packages, see the article [Migrate SQL Server Integration Services packages to Azure](#).

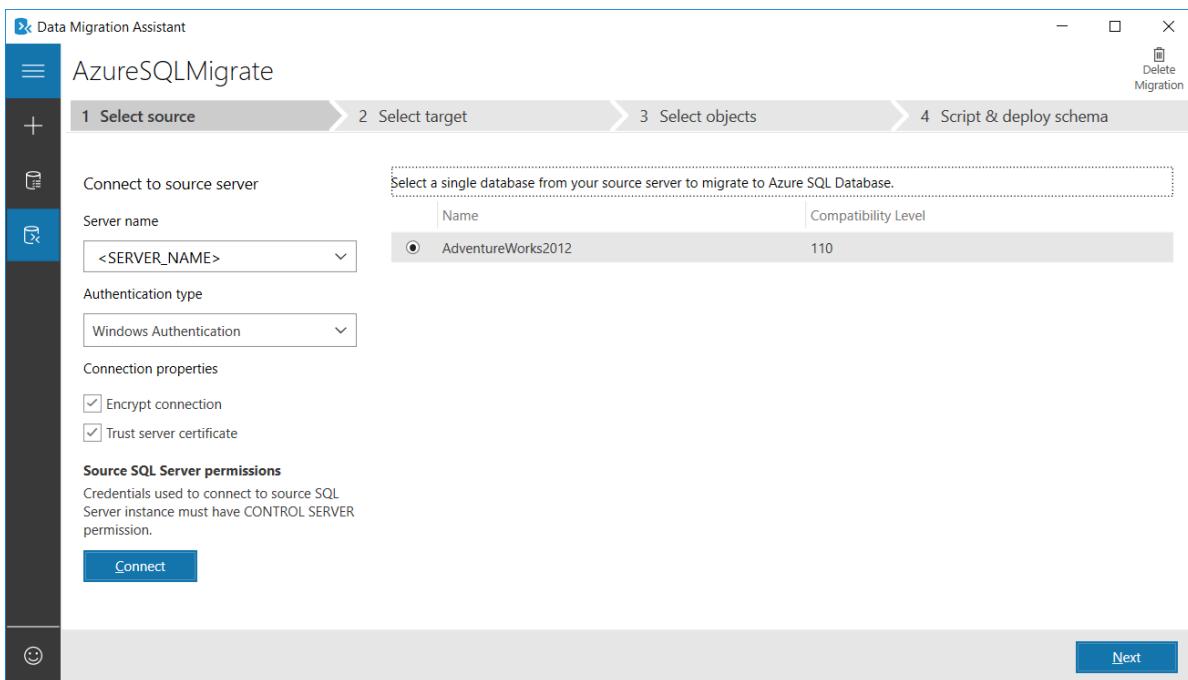
To migrate the **AdventureWorks2012** schema to a single database or pooled database Azure SQL Database, perform the following steps:

1. In the Data Migration Assistant, select the New (+) icon, and then under **Project type**, select **Migration**.
2. Specify a project name, in the **Source server type** text box, select **SQL Server**, and then in the **Target server type** text box, select **Azure SQL Database**.
3. Under **Migration Scope**, select **Schema only**.

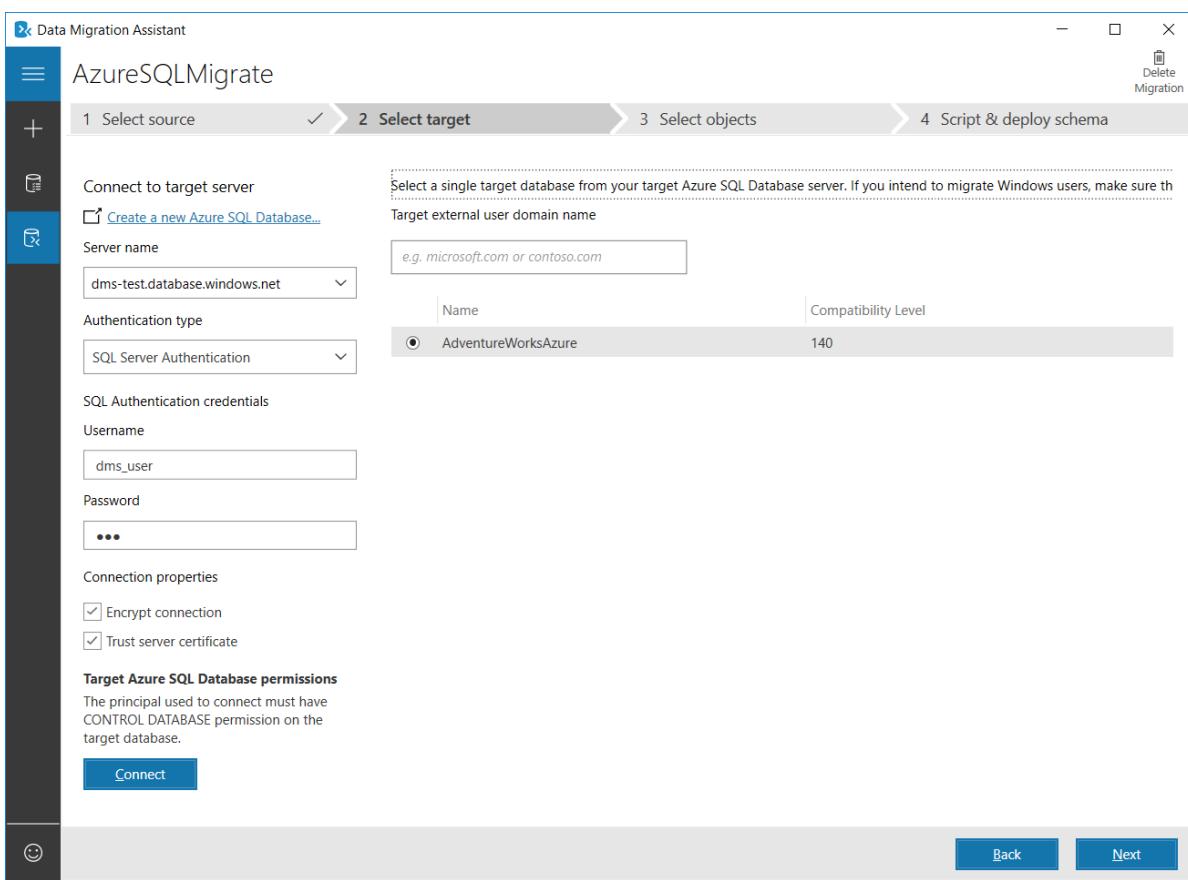
After performing the previous steps, the Data Migration Assistant interface should appear as shown in the following graphic:



4. Select **Create** to create the project.
5. In the Data Migration Assistant, specify the source connection details for your SQL Server, select **Connect**, and then select the **AdventureWorks2012** database.

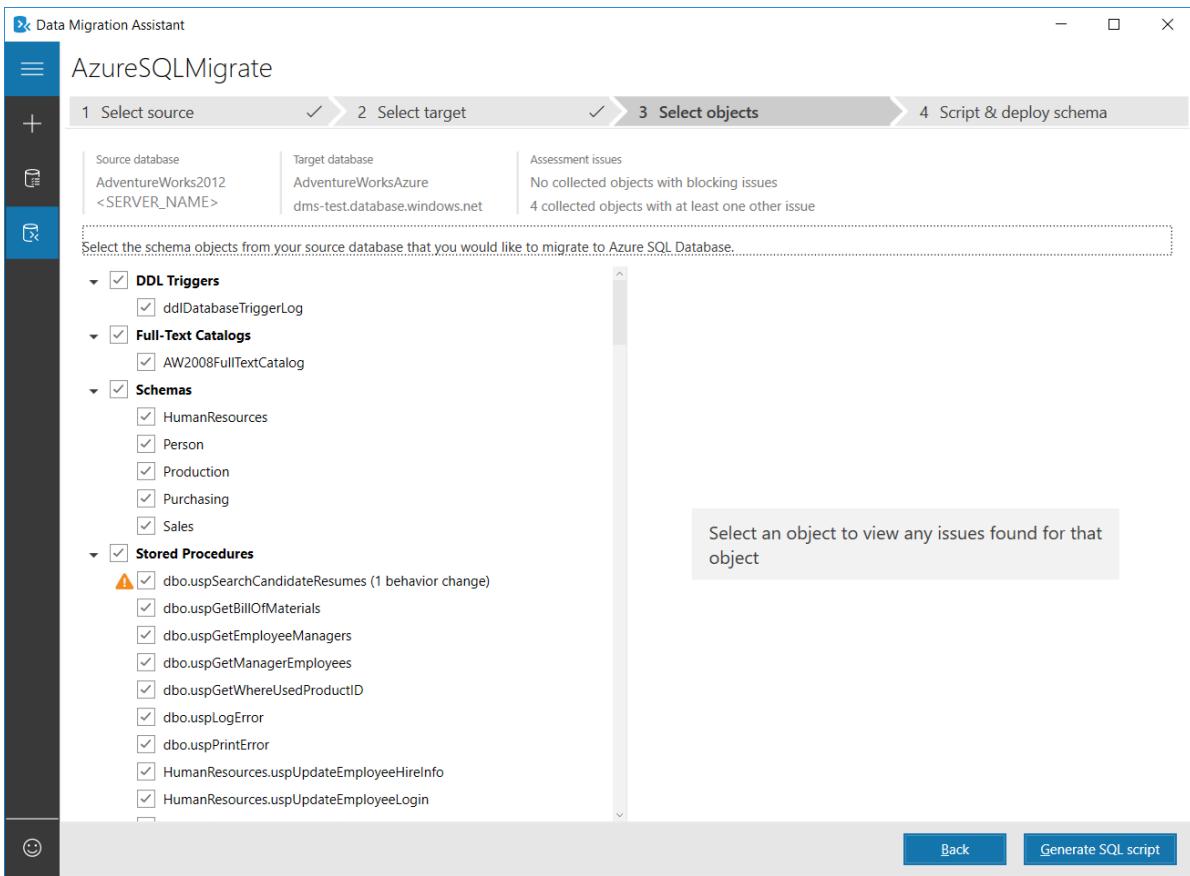


6. Select **Next**, under **Connect to target server**, specify the target connection details for the Azure SQL Database, select **Connect**, and then select the **AdventureWorksAzure** database you had pre-provisioned in Azure SQL Database.

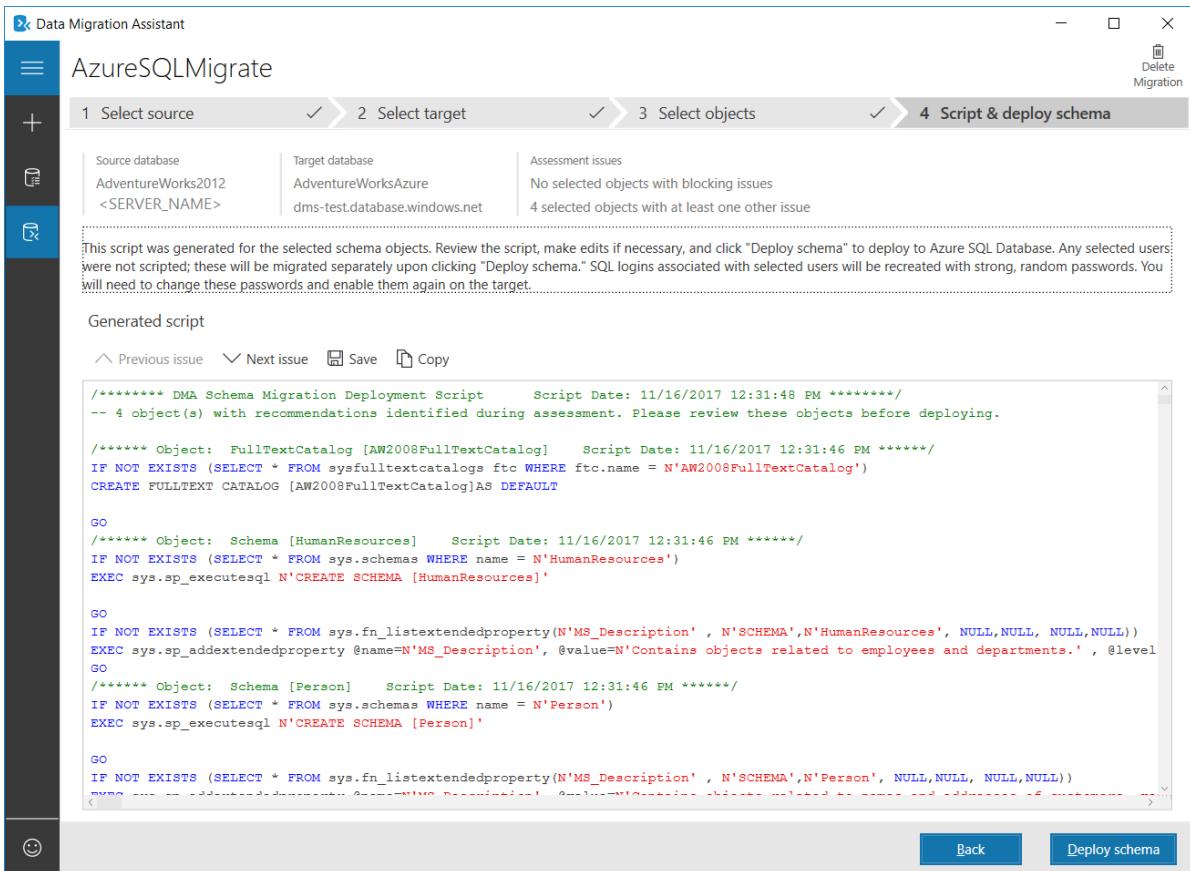


7. Select **Next** to advance to the **Select objects** screen, on which you can specify the schema objects in the **AdventureWorks2012** database that need to be deployed to Azure SQL Database.

By default, all objects are selected.



8. Select **Generate SQL script** to create the SQL scripts, and then review the scripts for any errors.



9. Select **Deploy schema** to deploy the schema to Azure SQL Database, and then after the schema is deployed, check the target server for any anomalies.

This screenshot shows the Microsoft Data Migration Assistant interface for migrating data from an on-premises database to Azure SQL Database. The project name is 'AzureSQLMigrate'. The process is divided into four main steps:

- 1 Select source**: Shows the source database as 'AdventureWorks2012' and the target database as 'AdventureWorksAzure' on '**<SERVER_NAME>**'.
- 2 Select target**: Shows the target database as 'AdventureWorksAzure' on '**<SERVER_NAME>**'.
- 3 Select objects**: Shows 'Assessment issues' with 'No selected objects with blocking issues' and '4 selected objects with at least one other issue'.
- 4 Script & deploy schema**: Displays the 'Generated script' and 'Deployment results'.

The 'Generated script' pane contains T-SQL code for creating a FullTextCatalog, schemas, and extended properties. The 'Deployment results' pane shows 2,236 commands executed successfully, including ALTER AUTHORIZATION, CREATE FULLTEXT CATALOG, and various IF NOT EXISTS statements.

Register the Microsoft.DataMigration resource provider

1. Sign in to the Azure portal. Search for and select **Subscriptions**.

This screenshot shows the Microsoft Azure portal's 'Subscriptions' blade. The user has signed in with 'admin@contoso.com' and 'CONTOSO'. The search bar at the top has 'Subscriptions' typed into it, which is highlighted with a red box. The left sidebar shows 'Azure services' like 'Create a resource' and 'Virtual machines'. The main area lists 'Services' such as 'Subscriptions', 'Event Grid Subscriptions', 'Resource groups', and 'Manage subscriptions in the Billing/Account Center'. A specific subscription named 'Contoso, Ltd. Subscription' is listed under 'Resources'. The right side of the blade includes sections for 'Marketplace', 'Documentation', and 'Recent resources'.

2. Select the subscription in which you want to create the instance of Azure Database Migration Service, and then select **Resource providers**.

Home > Subscriptions > Contoso, Ltd Subscription - Resource providers

Subscriptions

Contoso

+ Add

Showing subscriptions in Contoso. Don't see a subscription?
Switch directories

My role ⓘ Status ⓘ

8 selected 3 selected

Apply

Show only subscriptions selected in the global subscriptions filter ⓘ

Search to filter items...

Subscription...↑↓

Contoso - ... 01234567-89ab ... ⋮

Resource providers

Provider

- microsoft.insig
- Sendgrid.Email
- Microsoft.Cont
- Microsoft.Sql
- Microsoft.Data
- Microsoft.Servi
- Microsoft.Reco
- Microsoft.Devi
- Microsoft Servi

3. Search for migration, and then select **Register** for **Microsoft.DataMigration**.

Home > Subscriptions > Contoso, Ltd Subscription - Resource provider

Contoso, Ltd Subscription - Resource provider

Subscription

Search (Ctrl+ /)

Register Unregister Refresh

Migration

Provider	Status
Microsoft.DataMigration	Registering

Resource providers

Provider

- Resource groups
- Resources
- Usage + quotas
- Policies
- Management certificates
- My permissions
- Resource providers
- Deployments
- Properties
- Resource locks

Support + troubleshooting

New support request

Create an instance

1. In the Azure portal menu or on the **Home** page, select **Create a resource**. Search for and select **Azure Database Migration Service**.

Azure Marketplace

Popular

- Get started
- Recently created
- AI + Machine Learning
- Analytics
- Blockchain

Windows Server 2016 Datacenter
Ubuntu Server 18.04 LTS
Web App

2. On the **Azure Database Migration Service** screen, select **Create**.

Azure Database Migration Service

Microsoft

Azure Database Migration Service [Save for later](#)

Microsoft

Create

Overview [Plans](#)

Azure Database Migration Service (DMS) is designed to streamline the process of migrating on-premises databases to Azure. Get started with [step-by-step guidance](#).

Common scenarios:

- SQL Server → [Azure SQL Database](#)
- SQL Server → [Azure SQL Database Managed Instance](#)
- MongoDB → [Azure Cosmos DB](#)

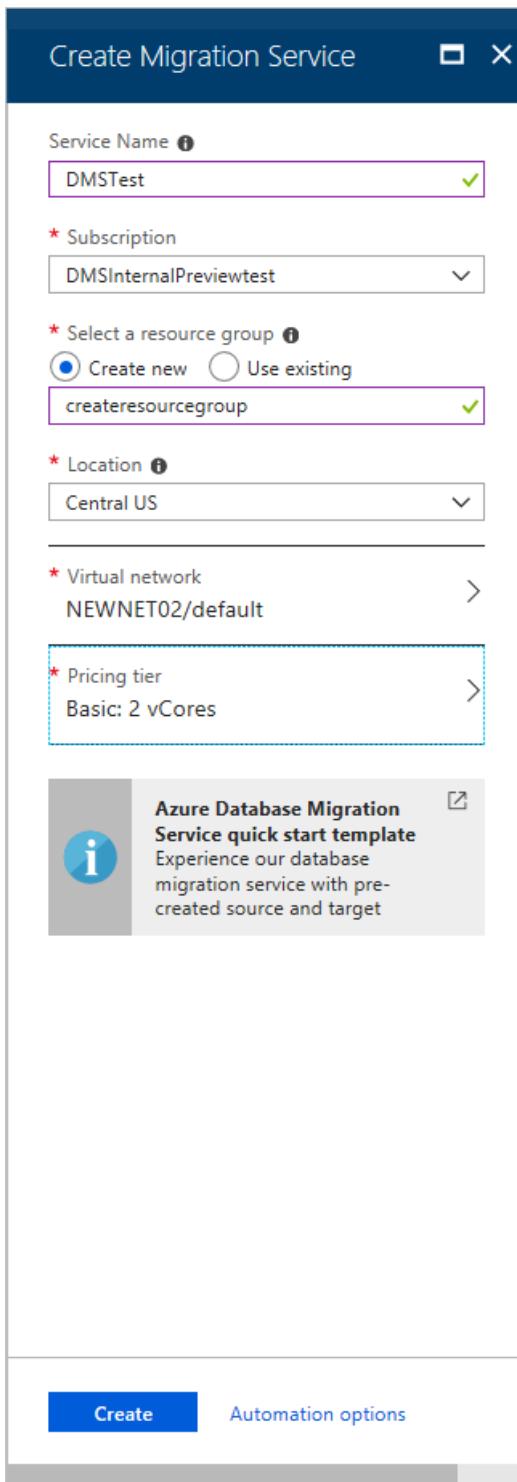
3. On the **Create Migration Service** screen, specify a name for the service, the subscription, and a new or existing resource group.
4. Select the location in which you want to create the instance of Azure Database Migration Service.
5. Select an existing virtual network or create a new one.

The virtual network provides Azure Database Migration Service with access to the source SQL Server and the target Azure SQL Database instance.

For more information about how to create a virtual network in the Azure portal, see the article [Create a virtual network using the Azure portal](#).

6. Select a pricing tier.

For more information on costs and pricing tiers, see the [pricing page](#).



7. Select **Create** to create the service.

Create a migration project

After the service is created, locate it within the Azure portal, open it, and then create a new migration project.

1. In the Azure portal menu, select **All services**. Search for and select **Azure Database Migration Services**.

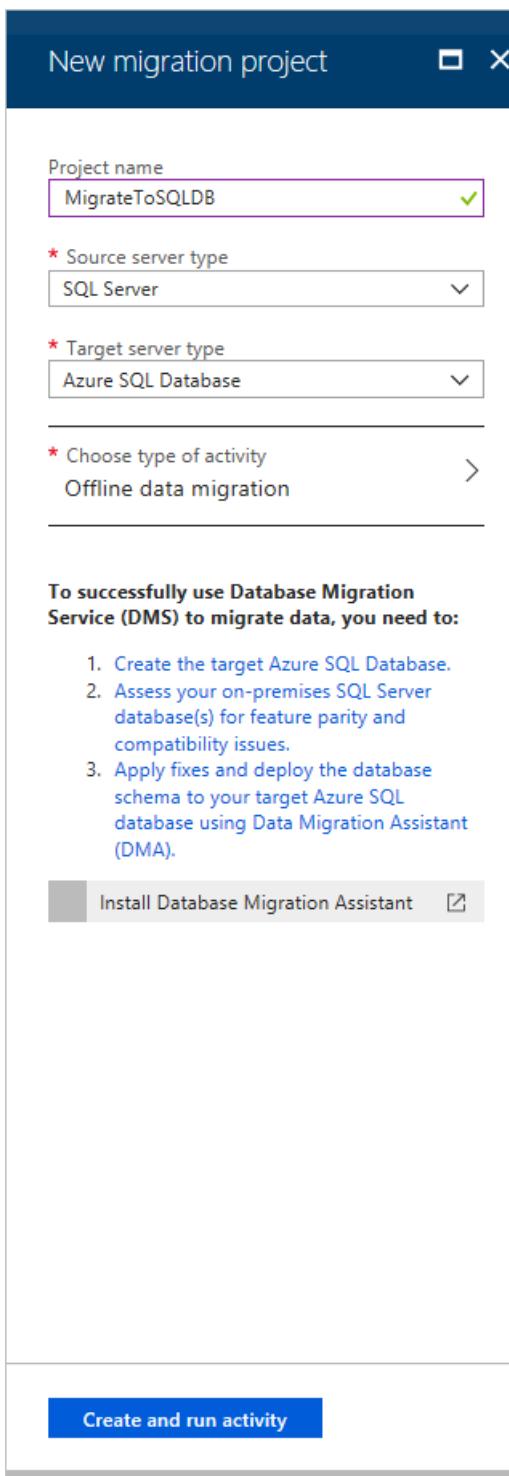
The screenshot shows the Azure portal's search interface. A red box highlights the search bar at the top containing the text "Azure Database Migration Service". Below the search bar, the results are displayed under the heading "Azure Database Migration Services". This section includes links for "SQL databases" (Keywords: Stretch Database), "Azure Database for MySQL servers", "Azure Cosmos DB" (Keywords: Database), "Service Health", "Azure Blockchain Service" (PREVIEW), "Analysis Services", and "App Services" (Keywords: mobile services).

2. On the **Azure Database Migration Services** screen, select the Azure Database Migration Service instance that you created.

3. Select **New Migration Project**.

The screenshot shows the details page for the "DMSTest" Azure Database Migration Service instance. A red box highlights the "New Migration Project" button in the top right corner. The left sidebar lists navigation options: "Overview" (highlighted with a red box), "Activity log", "Access control (IAM)", "Tags", "Configuration", "Hybrid", and "Properties". The main pane displays service details: "Resource group: creatoresourcegroup", "Status: Online", "Virtual network & IP Address: NEWNET02/subnets/default 10.0.0.4", "Location: Central US", "Subscription: Contoso, Ltd. Subscription", "SKU: Standard: 1 vCores", "Tags (change): Click here to add tags", and "Tags (change): Click here to add tags".

4. On the **New migration project** screen, specify a name for the project, in the **Source server type** text box, select **SQL Server**, in the **Target server type** text box, select **Azure SQL Database**, and then for **Choose type of activity**, select **Offline data migration**.



5. Select **Create and run activity** to create the project and run the migration activity.

Specify source details

1. On the **Migration source detail** screen, specify the connection details for the source SQL Server instance.

Make sure to use a Fully Qualified Domain Name (FQDN) for the source SQL Server instance name. You can also use the IP Address for situations in which DNS name resolution isn't possible.

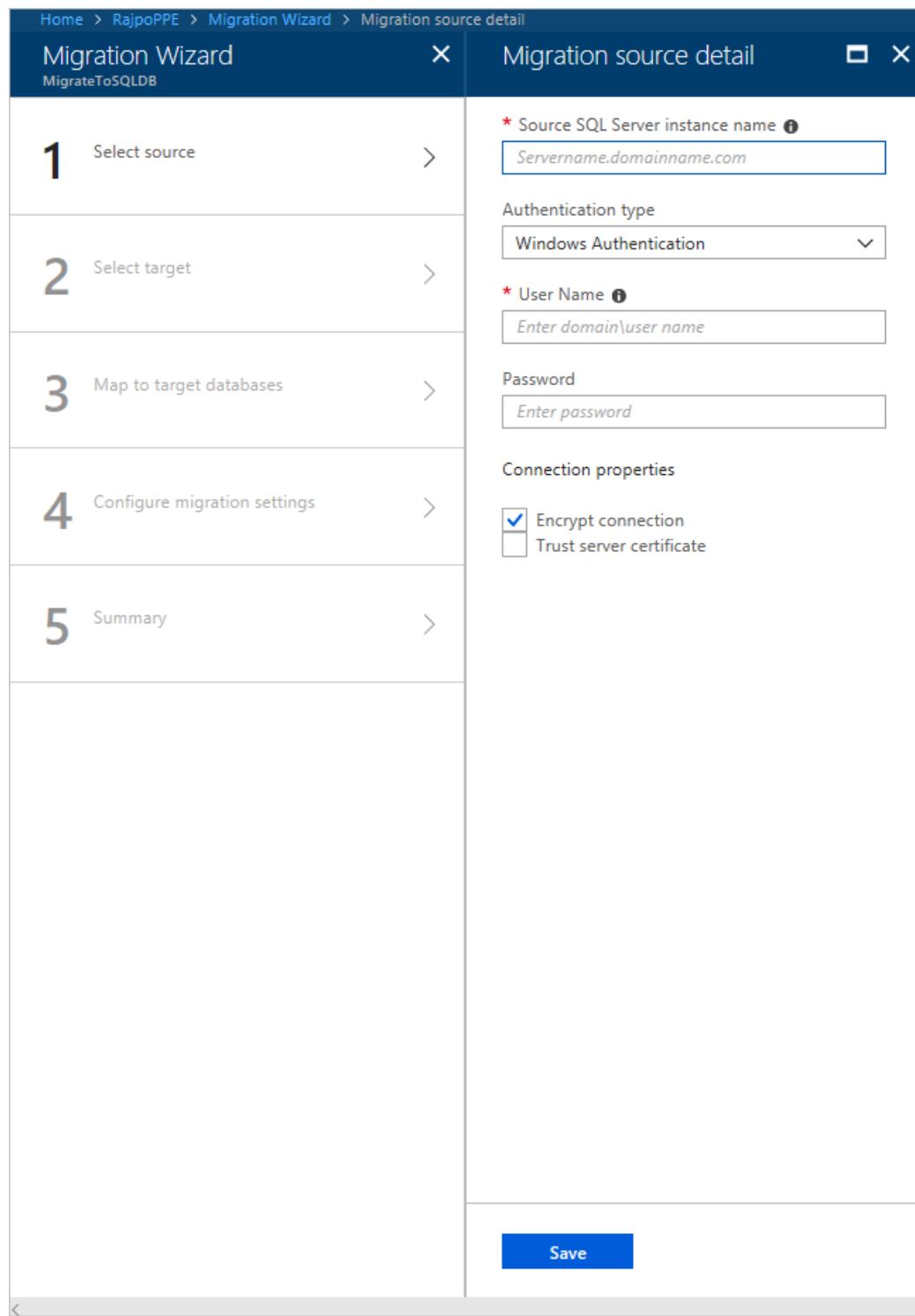
2. If you have not installed a trusted certificate on your source server, select the **Trust server certificate** check box.

When a trusted certificate is not installed, SQL Server generates a self-signed certificate when the instance is started. This certificate is used to encrypt the credentials for client connections.

Caution

SSL connections that are encrypted using a self-signed certificate do not provide strong security. They are

susceptible to man-in-the-middle attacks. You should not rely on SSL using self-signed certificates in a production environment or on servers that are connected to the internet.

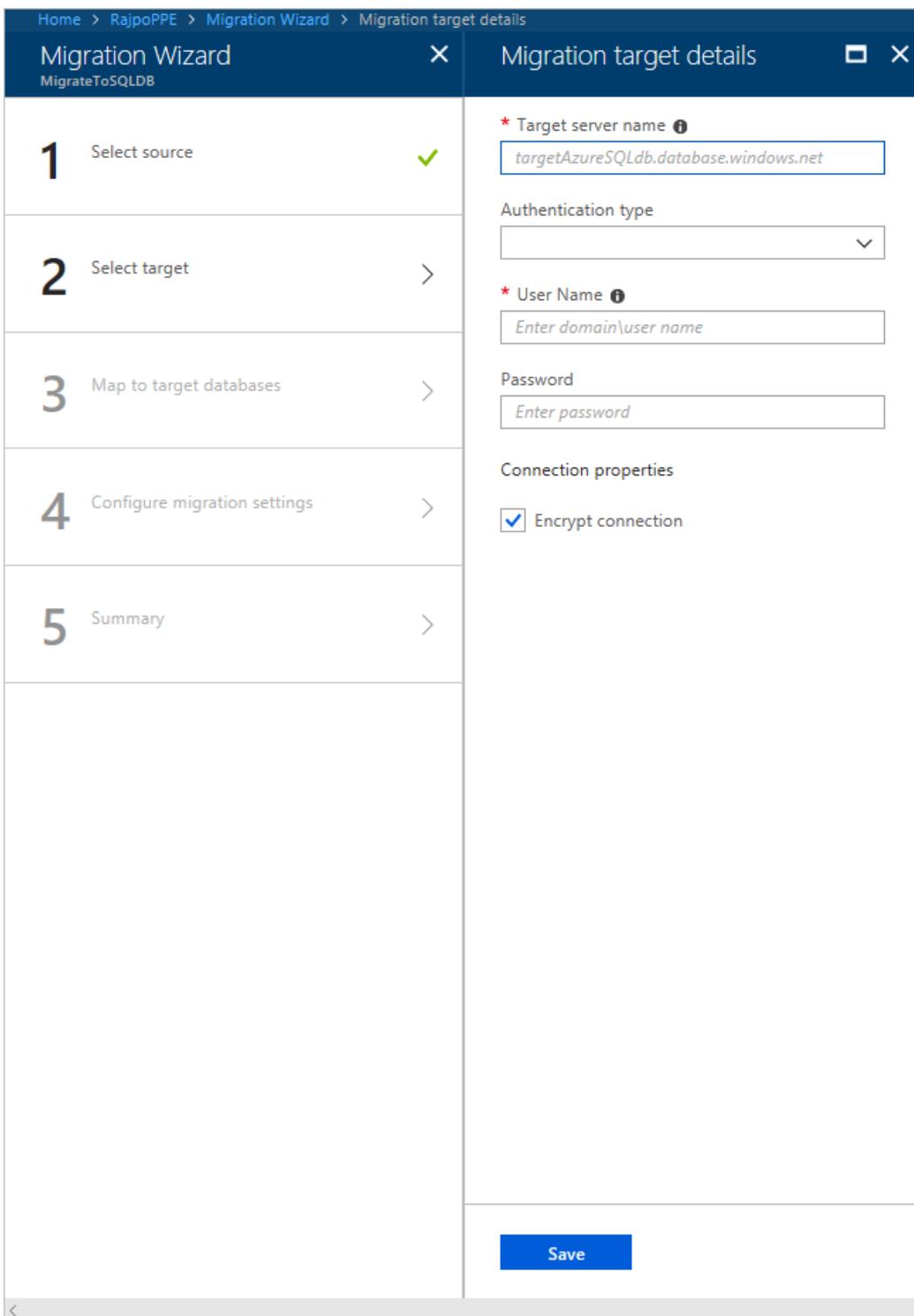


IMPORTANT

If you use SSIS, DMS does not currently support the migration of source SSISDB, but you can redeploy your SSIS projects/packages to the destination SSISDB hosted by Azure SQL Database. For more information about migrating SSIS packages, see the article [Migrate SQL Server Integration Services packages to Azure](#).

Specify target details

1. Select **Save**, and then on the **Migration target details** screen, specify the connection details for the target Azure SQL Database Server, which is the pre-provisioned Azure SQL Database to which the **AdventureWorks2012** schema was deployed by using the Data Migration Assistant.



2. Select **Save**, and then on the **Map to target databases** screen, map the source and the target database for migration.

If the target database contains the same database name as the source database, Azure Database Migration Service selects the target database by default.

Set the source database to read-only mode during production migrations, to preserve the data consistency and prevent modification of data during the migration. This operation will rollback any active transactions in the source database. The source databases remain in read-only mode after the migration.

SOURCE DATABASE	SIZE	TARGET DATABASE	SET SOURCE DB READ-ONLY
<input checked="" type="checkbox"/> AdventureWorks2012	230.06 MB	AdventureWorksAzure	<input type="checkbox"/>
<input type="checkbox"/> contosopayrolldb	5.00 MB		<input type="checkbox"/>
<input type="checkbox"/> HRMinDowntime	147.31 MB		<input type="checkbox"/>
<input type="checkbox"/> SitesEE	372.00 MB		<input type="checkbox"/>
<input type="checkbox"/> StackOverflowEE	486.00 MB		<input type="checkbox"/>
<input type="checkbox"/> test	21.00 MB		<input type="checkbox"/>

Save

3. Select **Save**, on the **Select tables** screen, expand the table listing, and then review the list of affected fields.

Azure Database Migration Service auto selects all the empty source tables that exist on the target Azure SQL Database instance. If you want to remigrate tables that already include data, you need to explicitly select the tables on this blade.

Database settings

AdventureWorks2012 71 of 89

NAME	DESCRIPTION
<input checked="" type="checkbox"/> Person.Address	The target table is not empty. If selected for migration, all the records in the table will be deleted...
<input checked="" type="checkbox"/> Person.AddressType	The target table is not empty. If selected for migration, all the records in the table will be deleted...
<input checked="" type="checkbox"/> dbo.AWBuildVersion	The target table is not empty. If selected for migration, all the records in the table will be deleted...
<input checked="" type="checkbox"/> Production.BillOfMat...	The target table is not empty. If selected for migration, all the records in the table will be deleted...
<input checked="" type="checkbox"/> Person.BusinessEntity	The target table is not empty. If selected for migration, all the records in the table will be deleted...
<input checked="" type="checkbox"/> Person.BusinessEntit...	The target table is not empty. If selected for migration, all the records in the table will be deleted...
<input checked="" type="checkbox"/> Person.BusinessEntit...	The target table is not empty. If selected for migration, all the records in the table will be deleted...
<input checked="" type="checkbox"/> Person.ContactType	The target table is not empty. If selected for migration, all the records in the table will be deleted...
<input checked="" type="checkbox"/> Person.CountryRegion	The target table is not empty. If selected for migration, all the records in the table will be deleted...
<input checked="" type="checkbox"/> Sales.CountryRegion...	The target table is not empty. If selected for migration, all the records in the table will be deleted...
<input checked="" type="checkbox"/> Sales.CreditCard	The target table is not empty. If selected for migration, all the records in the table will be deleted...
<input checked="" type="checkbox"/> Production.Culture	The target table is not empty. If selected for migration, all the records in the table will be deleted...
<input checked="" type="checkbox"/> Sales.Currency	The target table is not empty. If selected for migration, all the records in the table will be deleted...
<input checked="" type="checkbox"/> Sales.CurrencyRate	The target table is not empty. If selected for migration, all the records in the table will be deleted...
<input checked="" type="checkbox"/> Sales.Customer	The target table is not empty. If selected for migration, all the records in the table will be deleted...
<input checked="" type="checkbox"/> dbo.DatabaseLog	The target table is not empty. If selected for migration, all the records in the table will be deleted...
<input checked="" type="checkbox"/> HumanResources.De...	The target table is not empty. If selected for migration, all the records in the table will be deleted...
<input checked="" type="checkbox"/> Production.Document	

Save

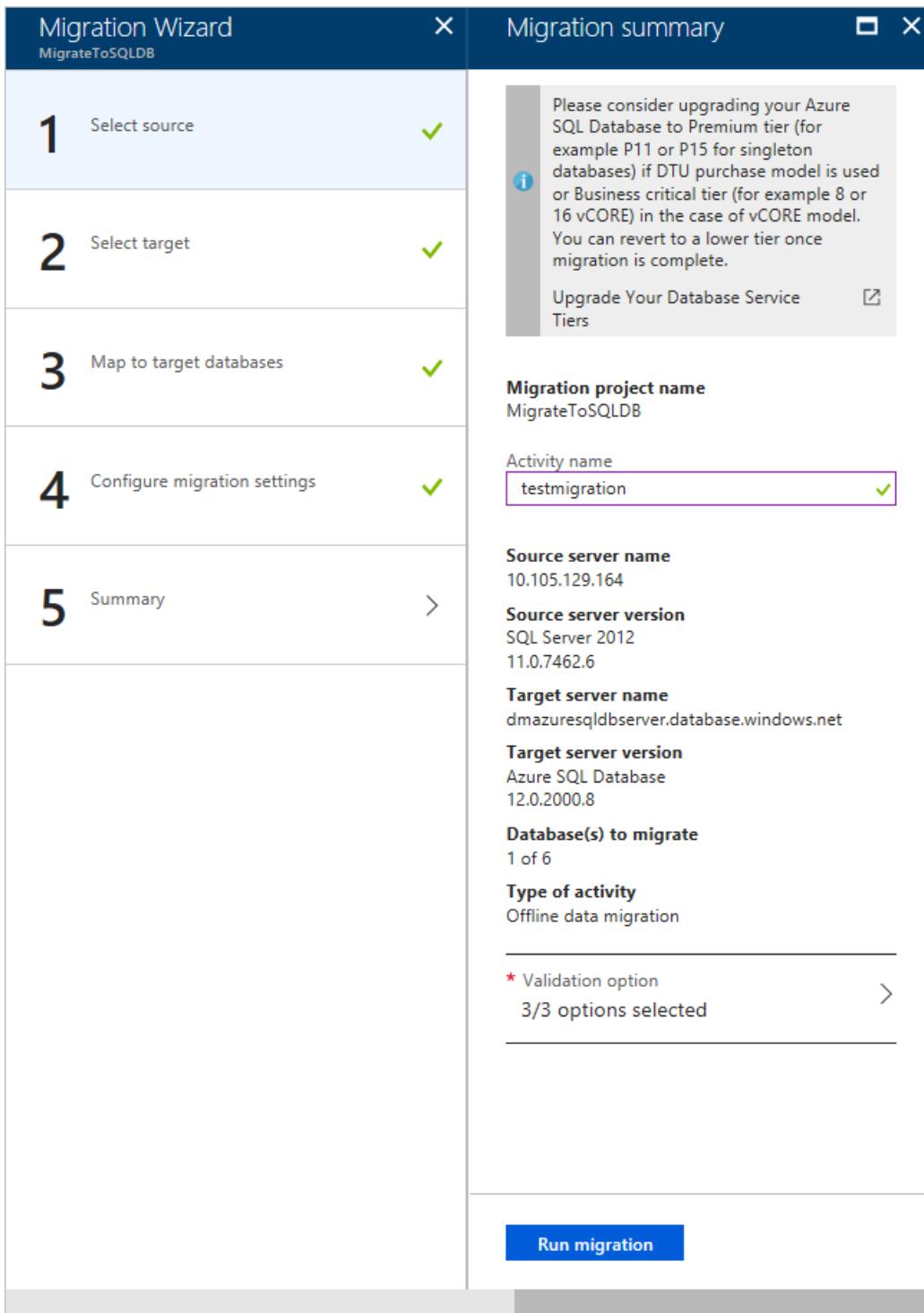
- Select **Save**, on the **Migration summary** screen, in the **Activity name** text box, specify a name for the migration activity.
- Expand the **Validation option** section to display the **Choose validation option** screen, and then specify whether to validate the migrated databases for **Schema comparison**, **Data consistency**, and **Query correctness**.

The screenshot shows the Microsoft Azure Migrate Wizard interface. On the left, the 'Migration Wizard' sidebar lists five steps: 1. Select source (green checkmark), 2. Select target (green checkmark), 3. Map to target databases (green checkmark), 4. Configure migration settings (green checkmark), and 5. Summary (grey arrow). The 'Migration summary' screen is the active tab, showing the following details:

- Migration project name:** MigrateToSQLDB
- Activity name:** testmigration (highlighted with a purple border)
- Source server name:** 10.105.129.164
- Source server version:** SQL Server 2012
11.0.7462.6
- Target server name:** dmazuresqlserver.database.windows.net
- Target server version:** Azure SQL Database
12.0.2000.8
- Database(s) to migrate:** 1 of 6
- Type of activity:** Offline data migration
- Validation option:** Schema comparison, Data consistency, Query correctness (all checked)

At the bottom of the 'Migration summary' screen are 'Run migration' and 'Save' buttons. The 'Choose validation option' screen is also visible on the right, showing the validation options selected.

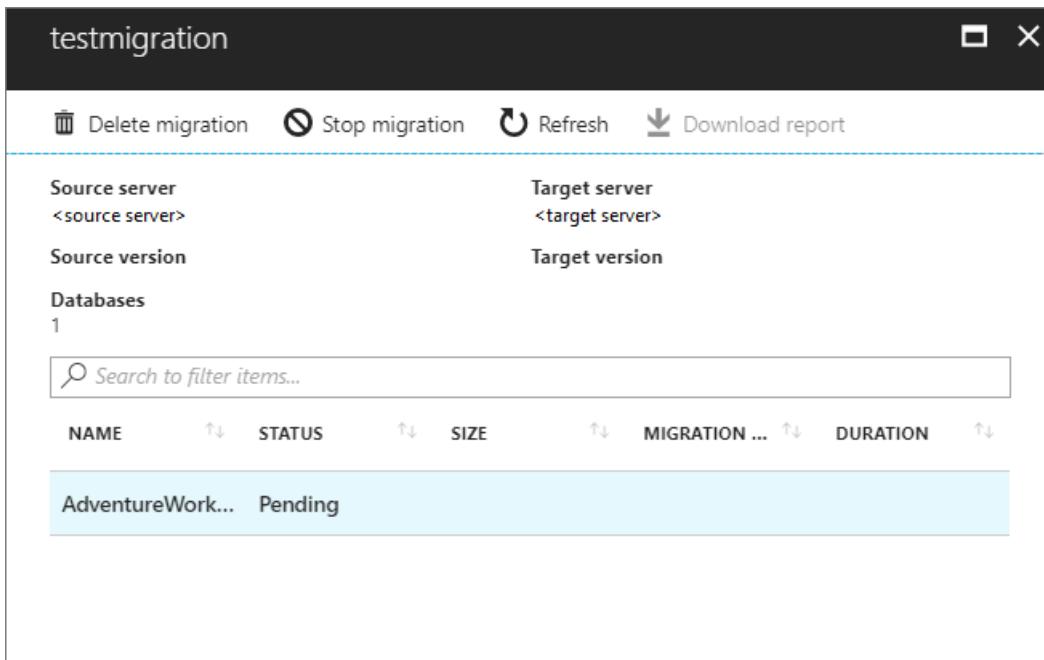
- Select **Save**, review the summary to ensure that the source and target details match what you previously specified.



Run the migration

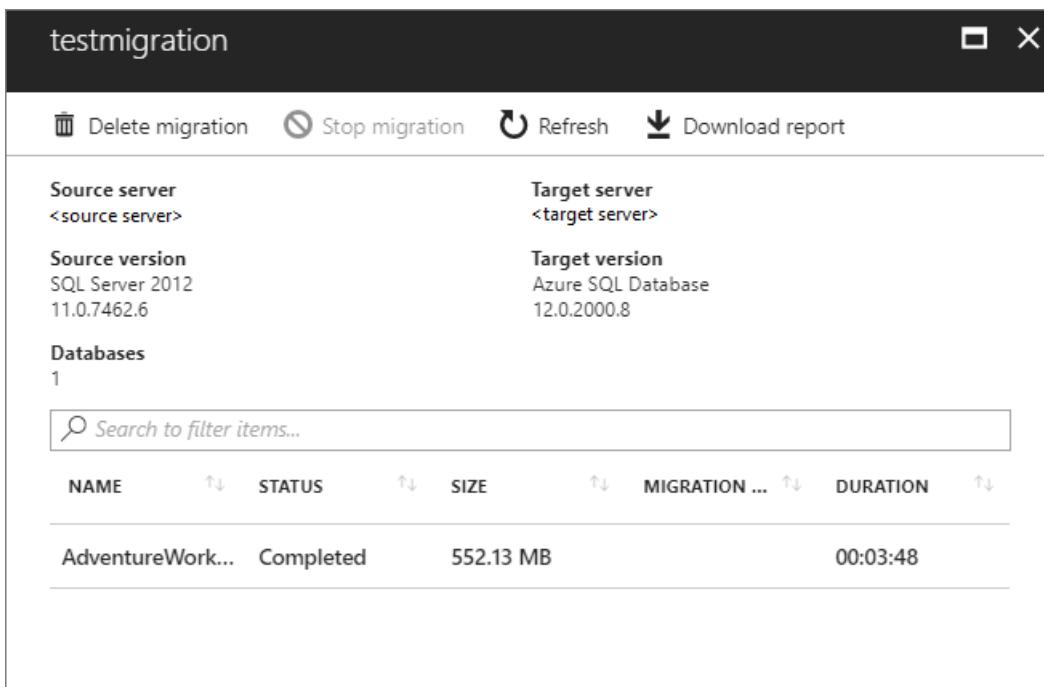
- Select **Run migration**.

The migration activity window appears, and the **Status** of the activity is **Pending**.



Monitor the migration

1. On the migration activity screen, select **Refresh** to update the display until the **Status** of the migration shows as **Completed**.



2. After the migration completes, select **Download report** to get a report listing the details associated with the migration process.
3. Verify the target database(s) on the target Azure SQL Database server.

Additional resources

- [SQL migration using Azure Data Migration Service](#) hands-on lab.
- For information about known issues and limitations when performing online migrations to Azure SQL Database, see the article [Known issues and workarounds with Azure SQL Database online migrations](#).
- For information about Azure Database Migration Service, see the article [What is Azure Database Migration Service?](#).
- For information about Azure SQL Database, see the article [What is the Azure SQL Database service?](#).

Tutorial: Migrate SQL Server to a single database or pooled database in Azure SQL Database online using DMS

1/21/2020 • 14 minutes to read • [Edit Online](#)

You can use the Azure Database Migration Service to migrate the databases from an on-premises SQL Server instance to [Azure SQL Database](#) with minimal downtime. In this tutorial, you migrate the **Adventureworks2012** database restored to an on-premises instance of SQL Server 2016 (or later) to a single database or pooled database in Azure SQL Database by using the Azure Database Migration Service.

In this tutorial, you learn how to:

- Assess your on-premises database by using the Data Migration Assistant.
- Migrate the sample schema by using the Data Migration Assistant.
- Create an instance of the Azure Database Migration Service.
- Create a migration project by using the Azure Database Migration Service.
- Run the migration.
- Monitor the migration.
- Download a migration report.

NOTE

Using the Azure Database Migration Service to perform an online migration requires creating an instance based on the Premium pricing tier. For more information, see the Azure Database Migration Service [pricing](#) page.

IMPORTANT

For an optimal migration experience, Microsoft recommends creating an instance of the Azure Database Migration Service in the same Azure region as the target database. Moving data across regions or geographies can slow down the migration process and introduce errors.

TIP

When you migrate databases to Azure by using Azure Database Migration Service, you can do an *offline* or an *online* migration. With an offline migration, application downtime starts when the migration starts. With an online migration, downtime is limited to the time to cut over at the end of migration. We suggest that you test an offline migration to determine whether the downtime is acceptable; if not, do an online migration.

This article describes an online migration from SQL Server to a single database or pooled database in Azure SQL Database. For an offline migration, see [Migrate SQL Server to Azure SQL Database offline using DMS](#).

Prerequisites

To complete this tutorial, you need to:

- Download and install [SQL Server 2012 or later](#).

- Enable the TCP/IP protocol, which is disabled by default during SQL Server Express installation, by following the instructions in the article [Enable or Disable a Server Network Protocol](#).
- Create a single (or pooled) database in Azure SQL Database, which you do by following the detail in the article [Create a single database in Azure SQL Database using the Azure portal](#).

NOTE

If you use SQL Server Integration Services (SSIS) and want to migrate the catalog database for your SSIS projects/packages (SSISDB) from SQL Server to Azure SQL Database, the destination SSISDB will be created and managed automatically on your behalf when you provision SSIS in Azure Data Factory (ADF). For more information about migrating SSIS packages, see the article [Migrate SQL Server Integration Services packages to Azure](#).

- Download and install the [Data Migration Assistant](#) (DMA) v3.3 or later.
- Create a Microsoft Azure Virtual Network for the Azure Database Migration Service by using the Azure Resource Manager deployment model, which provides site-to-site connectivity to your on-premises source servers by using either [ExpressRoute](#) or [VPN](#). For more information about creating a virtual network, see the [Virtual Network Documentation](#), and especially the quickstart articles with step-by-step details.

NOTE

During virtual network setup, if you use ExpressRoute with network peering to Microsoft, add the following service endpoints to the subnet in which the service will be provisioned:

- Target database endpoint (for example, SQL endpoint, Cosmos DB endpoint, and so on)
- Storage endpoint
- Service bus endpoint

This configuration is necessary because the Azure Database Migration Service lacks internet connectivity.

- Ensure that your virtual network Network Security Group rules don't block the following inbound communication ports to Azure Database Migration Service: 443, 53, 9354, 445, 12000. For more detail on Azure Virtual Network NSG traffic filtering, see the article [Filter network traffic with network security groups](#).
- Configure your [Windows Firewall for database engine access](#).
- Open your Windows firewall to allow the Azure Database Migration Service to access the source SQL Server, which by default is TCP port 1433.
- If you're running multiple named SQL Server instances using dynamic ports, you may wish to enable the SQL Browser Service and allow access to UDP port 1434 through your firewalls so that the Azure Database Migration Service can connect to a named instance on your source server.
- When using a firewall appliance in front of your source database(s), you may need to add firewall rules to allow the Azure Database Migration Service to access the source database(s) for migration.
- Create a server-level [firewall rule](#) for the Azure SQL Database server to allow the Azure Database Migration Service access to the target databases. Provide the subnet range of the virtual network used for the Azure Database Migration Service.
- Ensure that the credentials used to connect to source SQL Server instance have [CONTROL SERVER](#) permissions.
- Ensure that the credentials used to connect to the target Azure SQL Database instance have [CONTROL DATABASE](#) permission on the target Azure SQL Database instances.

- The source SQL Server version must be SQL Server 2005 and above. To determine the version that your SQL Server instance is running, see the article [How to determine the version, edition, and update level of SQL Server and its components](#).
- Database(s) must be in either Bulk-logged or Full recovery mode. To determine the recovery model configured for your SQL Server instance, see the article [View or Change the Recovery Model of a Database \(SQL Server\)](#).
- Make sure to take the Full database backups for the databases. To create a Full database backup, see the article [How to: Create a Full Database Backup \(Transact-SQL\)](#).
- If any of the tables don't have a primary key, enable Change Data Capture (CDC) on the database and specific table(s).

NOTE

You can use the script below to find any tables that do not have primary keys.

```
USE <DBName>;
go
SELECT is_tracked_by_cdc, name AS TableName
FROM sys.tables WHERE type = 'U' and is_ms_shipped = 0 AND
OBJECTPROPERTY(OBJECT_ID, 'TableHasPrimaryKey') = 0;
```

If the results show one or more tables with 'is_tracked_by_cdc' as '0', enable change capture for the database and for the specific tables by using the process described in the article [Enable and Disable Change Data Capture \(SQL Server\)](#).

- Configure the distributor role for source SQL Server.

NOTE

You can determine if replication components are installed by using the query below.

```
USE master;
DECLARE @installed int;
EXEC @installed = sys.sp_MS_rePLICATION_installed;
SELECT @installed as installed;
```

If the result returns an error message suggesting to install replication components, install SQL Server replication components by using the process in the article [Install SQL Server replication](#).

If the replication is already installed, check if the distribution role is configured on the source SQL Server using the T-SQL command below.

```
EXEC sp_get_distributor;
```

If the distribution isn't set up, where the distribution server shows NULL for above command output, configure the distribution using the guidance provided in the article [Configure Distribution](#).

- Disable database triggers on the target Azure SQL Database.

NOTE

You can find the database triggers on the target Azure SQL Database by using the following query:

```
Use <Database name>
select * from sys.triggers
DISABLE TRIGGER (Transact-SQL)
```

For more information, see the article [DISABLE TRIGGER \(Transact-SQL\)](#).

Assess your on-premises database

Before you can migrate data from an on-premises SQL Server instance to a single database or pooled database in Azure SQL Database, you need to assess the SQL Server database for any blocking issues that might prevent migration. Using the Data Migration Assistant v3.3 or later, follow the steps described in the article [Performing a SQL Server migration assessment](#) to complete the on-premises database assessment.

To assess an on-premises database, perform the following steps:

1. In DMA, select the New (+) icon, and then select the **Assessment** project type.
2. Specify a project name, in the **Source server type** text box, select **SQL Server**, in the **Target server type** text box, select **Azure SQL Database**, and then select **Create** to create the project.

When you're assessing the source SQL Server database migrating to a single database or pooled database in Azure SQL Database, you can choose one or both of the following assessment report types:

- Check database compatibility
- Check feature parity

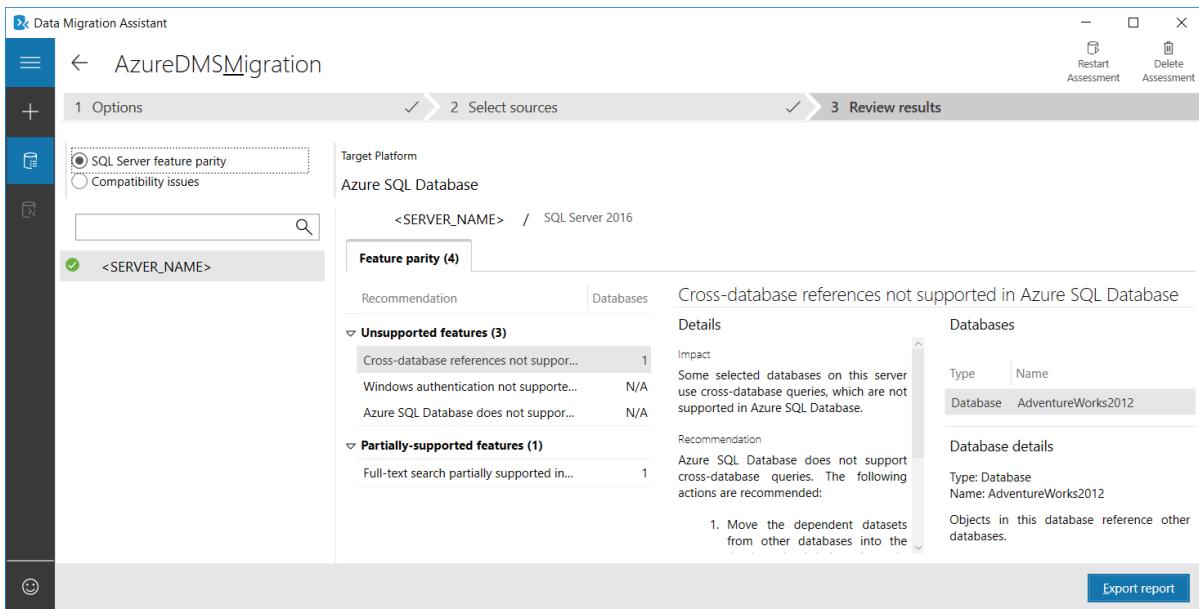
Both report types are selected by default.

3. In DMA, on the **Options** screen, select **Next**.
4. On the **Select sources** screen, in the **Connect to a server** dialog box, provide the connection details to your SQL Server, and then select **Connect**.
5. In the **Add sources** dialog box, select **AdventureWorks2012**, select **Add**, and then select **Start Assessment**.

NOTE

If you use SSIS, DMA does not currently support the assessment of the source SSISDB. However, SSIS projects/packages will be assessed/validated as they are redeployed to the destination SSISDB hosted by Azure SQL Database. For more information about migrating SSIS packages, see the article [Migrate SQL Server Integration Services packages to Azure](#).

When the assessment is complete, the results display as shown in the following graphic:



For single databases or pooled databases in Azure SQL Database, the assessments identify feature parity issues and migration blocking issues for deploying to a single database or pooled database.

- The **SQL Server feature parity** category provides a comprehensive set of recommendations, alternative approaches available in Azure, and mitigating steps to help you plan the effort into your migration projects.
- The **Compatibility issues** category identifies partially supported or unsupported features that reflect compatibility issues that might block migrating on-premises SQL Server database(s) to Azure SQL Database. Recommendations are also provided to help you address those issues.

6. Review the assessment results for migration blocking issues and feature parity issues by selecting the specific options.

Migrate the sample schema

After you're comfortable with the assessment and satisfied that the selected database is a viable candidate for migration to a single database or pooled database in Azure SQL Database, use DMA to migrate the schema to Azure SQL Database.

NOTE

Before you create a migration project in DMA, be sure that you have already provisioned an Azure SQL database as mentioned in the prerequisites. For purposes of this tutorial, the name of the Azure SQL Database is assumed to be **AdventureWorksAzure**, but you can provide whatever name you wish.

IMPORTANT

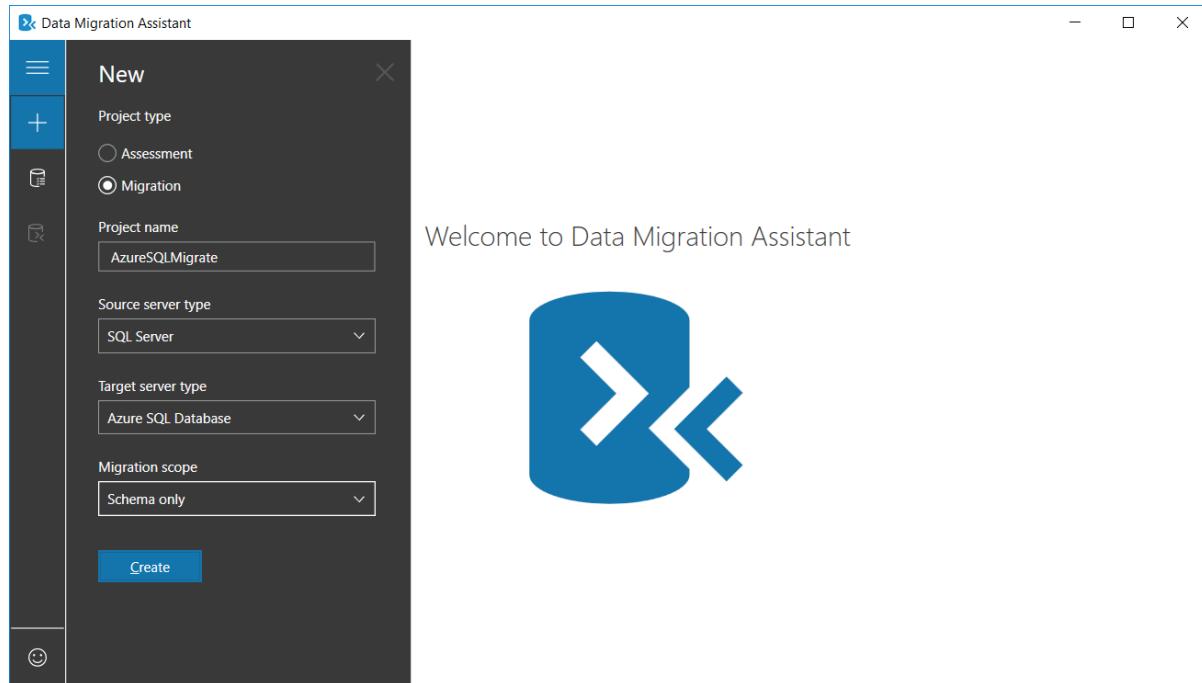
If you use SSIS, DMA does not currently support the migration of source SSISDB, but you can redeploy your SSIS projects/packages to the destination SSISDB hosted by Azure SQL Database. For more information about migrating SSIS packages, see the article [Migrate SQL Server Integration Services packages to Azure](#).

To migrate the **AdventureWorks2012** schema to a single database or pooled database Azure SQL Database, perform the following steps:

1. In the Data Migration Assistant, select the New (+) icon, and then under **Project type**, select **Migration**.
2. Specify a project name, in the **Source server type** text box, select **SQL Server**, and then in the **Target server type** text box, select **Azure SQL Database**.

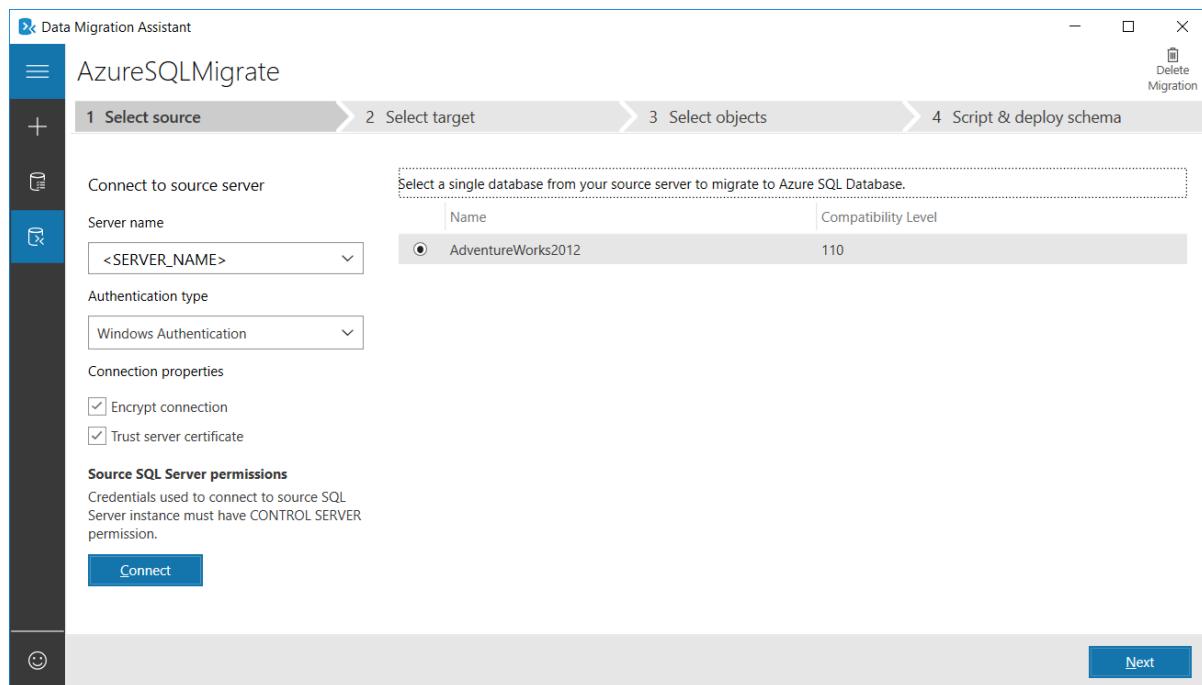
3. Under **Migration Scope**, select **Schema only**.

After performing the previous steps, the DMA interface should appear as shown in the following graphic:

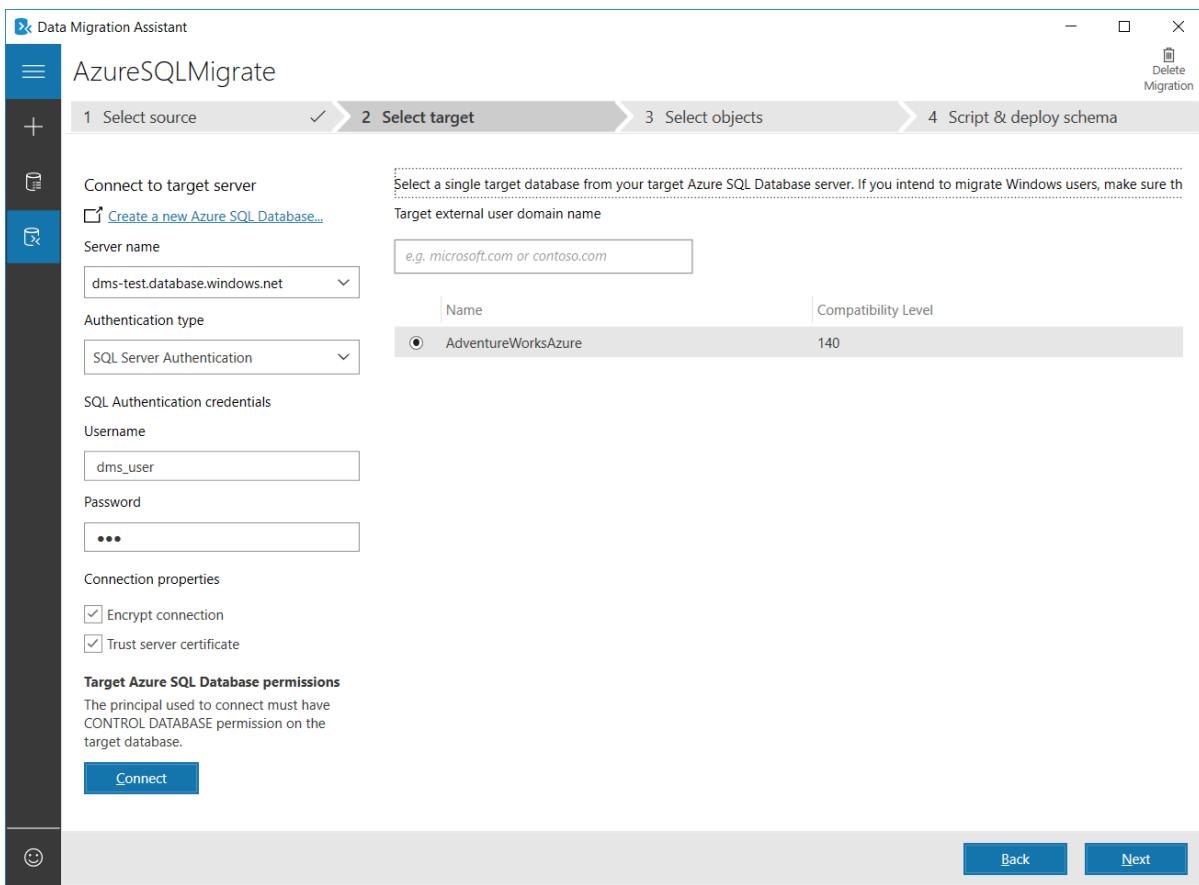


4. Select **Create** to create the project.

5. In DMA, specify the source connection details for your SQL Server, select **Connect**, and then select the **AdventureWorks2012** database.

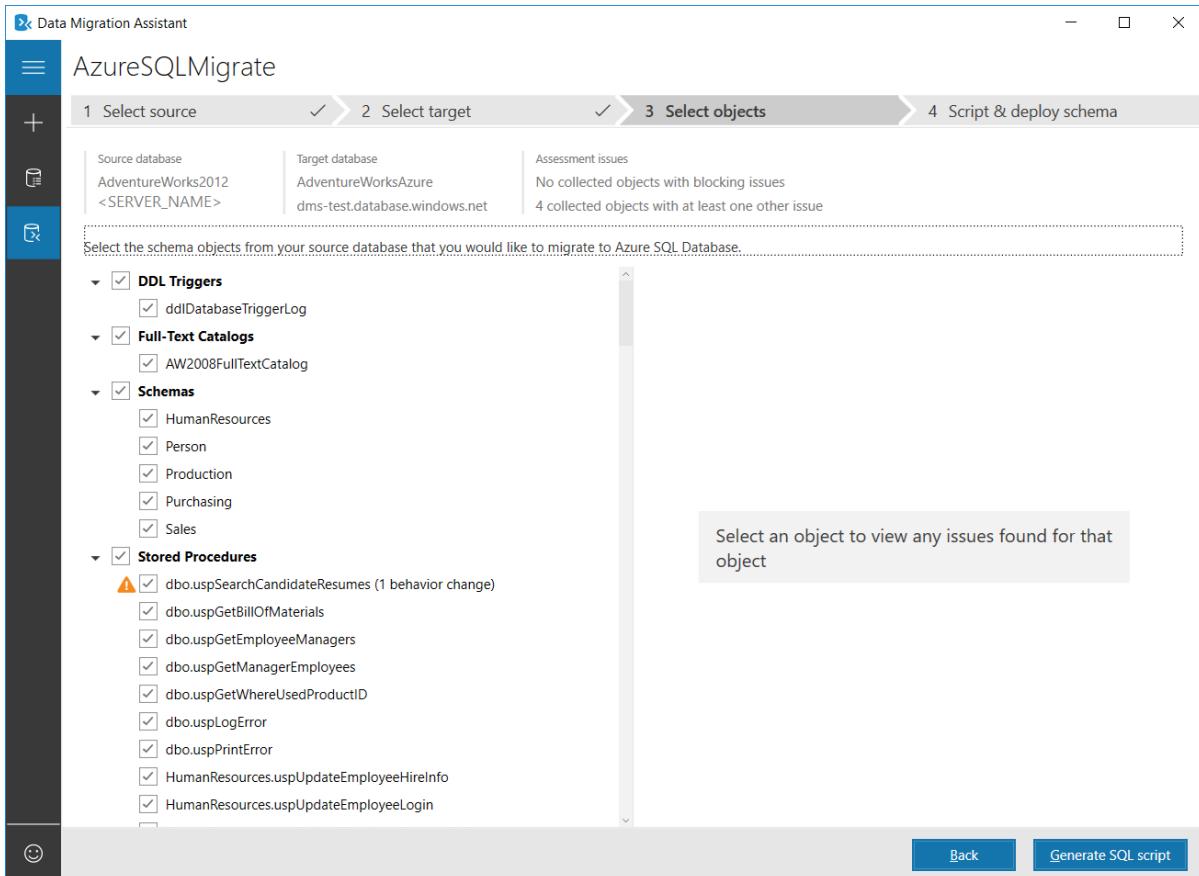


6. Select **Next**, under **Connect to target server**, specify the target connection details for the Azure SQL database, select **Connect**, and then select the **AdventureWorksAzure** database you had pre-provisioned in Azure SQL Database.



7. Select **Next** to advance to the **Select objects** screen, on which you can specify the schema objects in the **AdventureWorks2012** database that need to be deployed to Azure SQL Database.

By default, all objects are selected.



8. Select **Generate SQL script** to create the SQL scripts, and then review the scripts for any errors.

9. Select **Deploy schema** to deploy the schema to Azure SQL Database, and then after the schema is deployed, check the target server for any anomalies.

Register the Microsoft.DataMigration resource provider

1. Sign in to the Azure portal, select **All services**, and then select **Subscriptions**.

All services Filter By category ▾

GENERAL (14) —

- Dashboard
- Management Groups PREVIEW
- Cost Management + Billing PREVIEW
- Help + support
- Tags
- All resources
- Subscriptions
- Reservations
- Service Health
- What's new

COMPUTE (20) —

- Virtual machines
- Virtual machines (classic)
- Container services
- Function Apps

2. Select the subscription in which you want to create the instance of the Azure Database Migration Service, and then select **Resource providers**.

Home > Subscriptions > <subscription>

Subscriptions Microsoft

Add

My role: Owner Status: 3 selected

Apply

Search to filter items...

SUBSCRIPTI... ↑↓ SUBSCRIPTION ID ↑↓

<subscription> <subscription ID>

Overview

Access control (IAM)

Diagnose and solve problems

COST MANAGEMENT + BILLING

Partner information

SETTINGS

Programmatic deployment

Resource groups

Resources

Usage + quotas

Policies

Management certificates

My permissions

Resource providers

Properties

Resource locks

SUPPORT + TROUBLESHOOTING

New support request

3. Search for migration, and then to the right of **Microsoft.DataMigration**, select **Register**.

The screenshot shows the Azure portal interface. On the left, the navigation menu includes 'Create a resource', 'All services', 'Dashboard', 'Resource groups', 'App Services', 'Function Apps', 'SQL databases', 'Azure Cosmos DB', 'Virtual machines', 'Load balancers', 'Storage accounts', 'Virtual networks', 'Azure Active Directory', 'Monitor', and 'Advisor'. The main content area is titled '<Subscription>' and shows a list of registered providers. One provider, 'Microsoft.DataMigration', is listed with the status 'NotRegistered'. A red box highlights the 'Register' button next to it.

Create an instance

1. In the Azure portal, select + **Create a resource**, search for Azure Database Migration Service, and then select **Azure Database Migration Service** from the drop-down list.

The screenshot shows the Azure portal's 'New' blade. The search bar at the top contains the text 'Azure Database Migration Service'. Below the search bar, there is a list of service categories: Networking, Storage, Web + Mobile, Containers, Databases, Data + Analytics, AI + Cognitive Services, Internet of Things, Enterprise Integration, Security + Identity, Developer tools, Monitoring + Management, Add-ons, and Blockchain. Under the 'Databases' category, 'Azure Database Migration Service' is listed with a 'PREVIEW' badge and a 'Quickstart tutorial' link. A red box highlights the search bar text.

2. On the **Azure Database Migration Service** screen, select **Create**.

The screenshot shows the Azure Database Migration Service (DMS) page. On the left, there's a sidebar with various service icons and names: Create a resource, All services, Favorites, Dashboard, Subscriptions, All resources, Resource groups, App Services, Function Apps, SQL databases, Azure Cosmos DB, Virtual machines, Load balancers, Storage accounts, Virtual networks, Azure Active Directory, Monitor, Advisor, and Security Center. The 'Create a resource' icon is at the top of the list. The main content area has a title 'Azure Database Migration Service' with a Microsoft logo. Below the title, a paragraph explains DMS's purpose: 'The Azure Database Migration Service (DMS) is designed to streamline the process of migrating on-premises databases to Azure. DMS will simplify the migration of existing on-premises SQL Server and Oracle databases to Azure SQL Database, Azure SQL Managed Instance or Microsoft SQL Server in an Azure Virtual Machine. Learn [more](#)'. A section titled 'Before using DMS we recommend you complete the following three steps:' lists three items. Below this, another section says 'Once your assessments are complete, fixes are applied and schema is deployed, proceed with creating a migration service by clicking **Create** below to migrate the data from your source database to the target.' At the bottom of the page, there's a 'Save for later' button, followed by sections for 'PUBLISHER' (Microsoft), 'USEFUL LINKS' (Documentation, Privacy Statement), and a large blue 'Create' button which is highlighted with a red border.

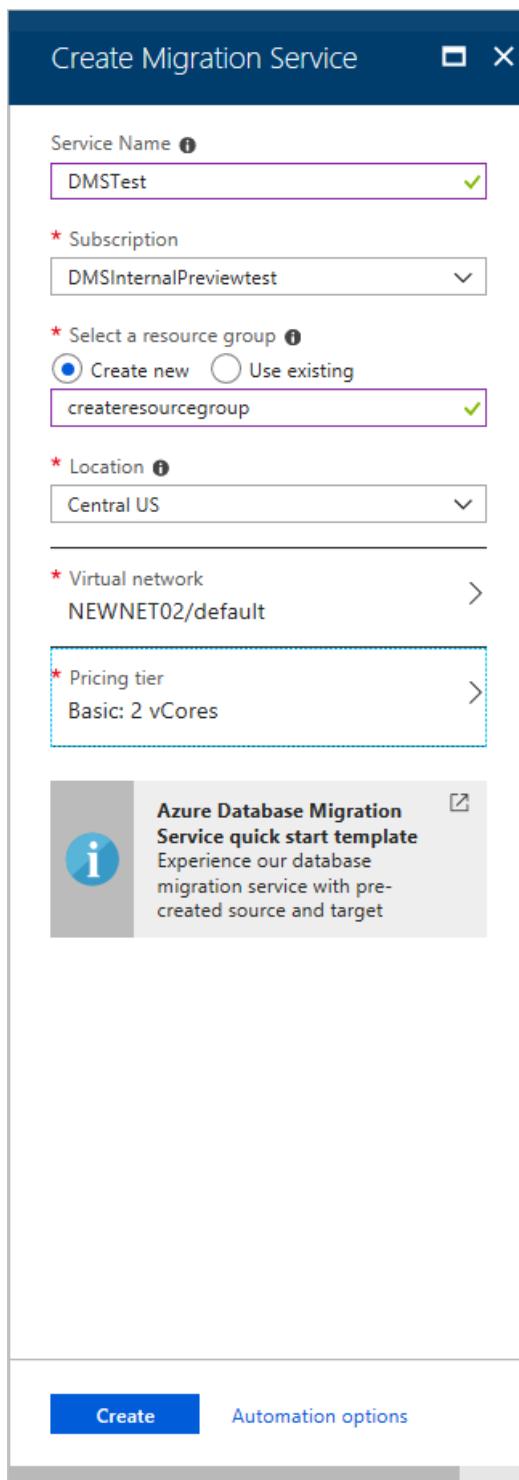
3. On the **Create Migration Service** screen, specify a name for the service, the subscription, and a new or existing resource group.
4. Select the location in which you want to create the instance of the Azure Database Migration Service.
5. Select an existing virtual network or create a new one.

The virtual network provides the Azure Database Migration Service with access to the source SQL Server and the target Azure SQL Database instance.

For more information about how to create a virtual network in the Azure portal, see the article [Create a virtual network using the Azure portal](#).

6. Select a pricing tier.

For more information on costs and pricing tiers, see the [pricing page](#).



7. Select **Create** to create the service.

Create a migration project

After the service is created, locate it within the Azure portal, open it, and then create a new migration project.

1. In the Azure portal, select **All services**, search for Azure Database Migration Service, and then select **Azure Database Migration Services**.

The screenshot shows the Azure portal's left sidebar with various service icons and names. At the top, there is a search bar with the placeholder "Shift+Space to toggle favorites" and the text "Azure Database Migration Service". Below the search bar, the "Azure Database Migration Services" service is listed under "FAVORITES". The main content area displays the "Azure Database Migration Services" page, which includes a "Help improve the service menu!" link at the bottom.

2. On the **Azure Database Migration Services** screen, search for the name of the Azure Database Migration Service instance that you created, and then select the instance.

The screenshot shows the "SQL_Migrate" instance details page within the Azure Database Migration Services. The page includes sections for "Essentials" (Resource group: SQL_Migrate, Network: <Network>, Subscription name: DMSInternalPreviewtest, Pricing tier: 0) and "Migration Projects" (No database migration projects to display).

3. Select + **New Migration Project**.
4. On the **New migration project** screen, specify a name for the project, in the **Source server type** text box, select **SQL Server**, in the **Target server type** text box, select **Azure SQL Database**.
5. In the **Choose type of activity** section, select **Online data migration**.

Choose type of activity

Online data migration

Use this option to migrate databases that must be accessible and continuously updated during migration.

To continuously replicate data changes from your source to your target, please implement the steps below on your source server. These steps can be implemented anytime between the moment you create a project and the moment you start a migration activity. After your migration is complete, you will reverse those preparation steps.

- The source SQL Server version must be SQL Server 2005 or above.
- Database(s) must be in either Bulk-logged or Full recovery mode.
- Make sure to take full database backups.
- If tables do not have primary key, enable CDC on the database and specific tables.
- The distributor role must be configured for the source database.

To successfully use Database Migration Service (DMS) to migrate data, you need to:

1. Create the target Azure SQL Database.
2. Assess your on-premises SQL Server database(s) for feature parity and compatibility issues.
3. Apply fixes and deploy the database schema to your target Azure SQL database using Data Migration Assistant (DMA).

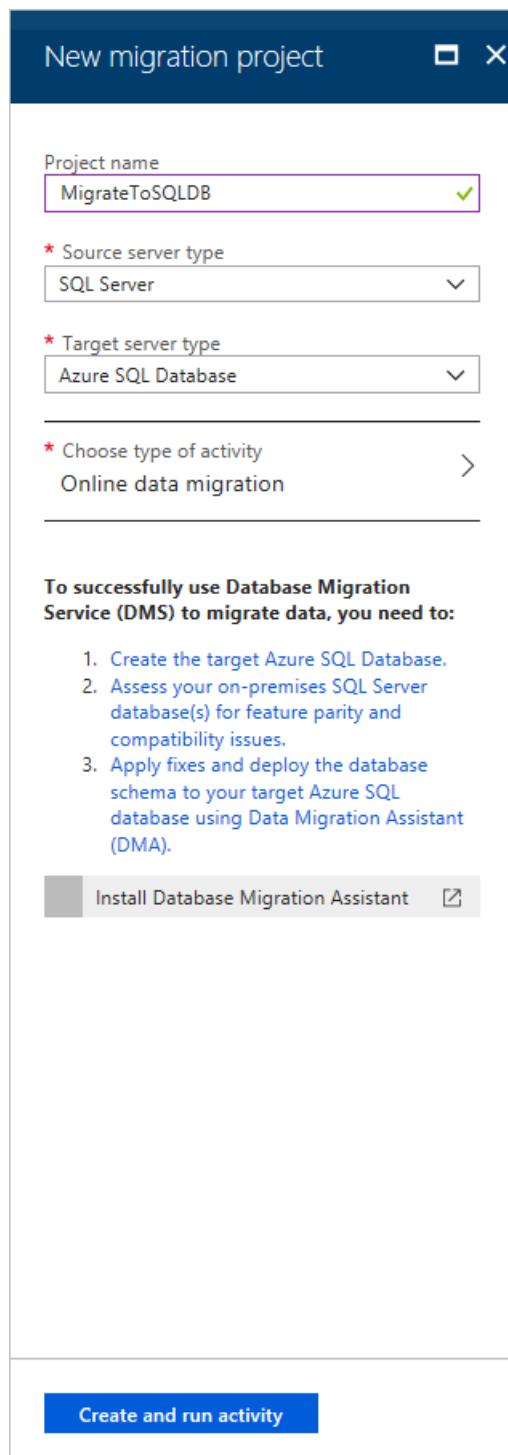
Install Database Migration Assistant

Create and run activity	Save
--------------------------------	-------------

NOTE

Alternately, you can choose **Create project only** to create the migration project now and execute the migration later.

6. Select **Save**.
7. Select **Create and run activity** to create the project and run the migration activity.



Specify source details

1. On the **Migration source detail** screen, specify the connection details for the source SQL Server instance.

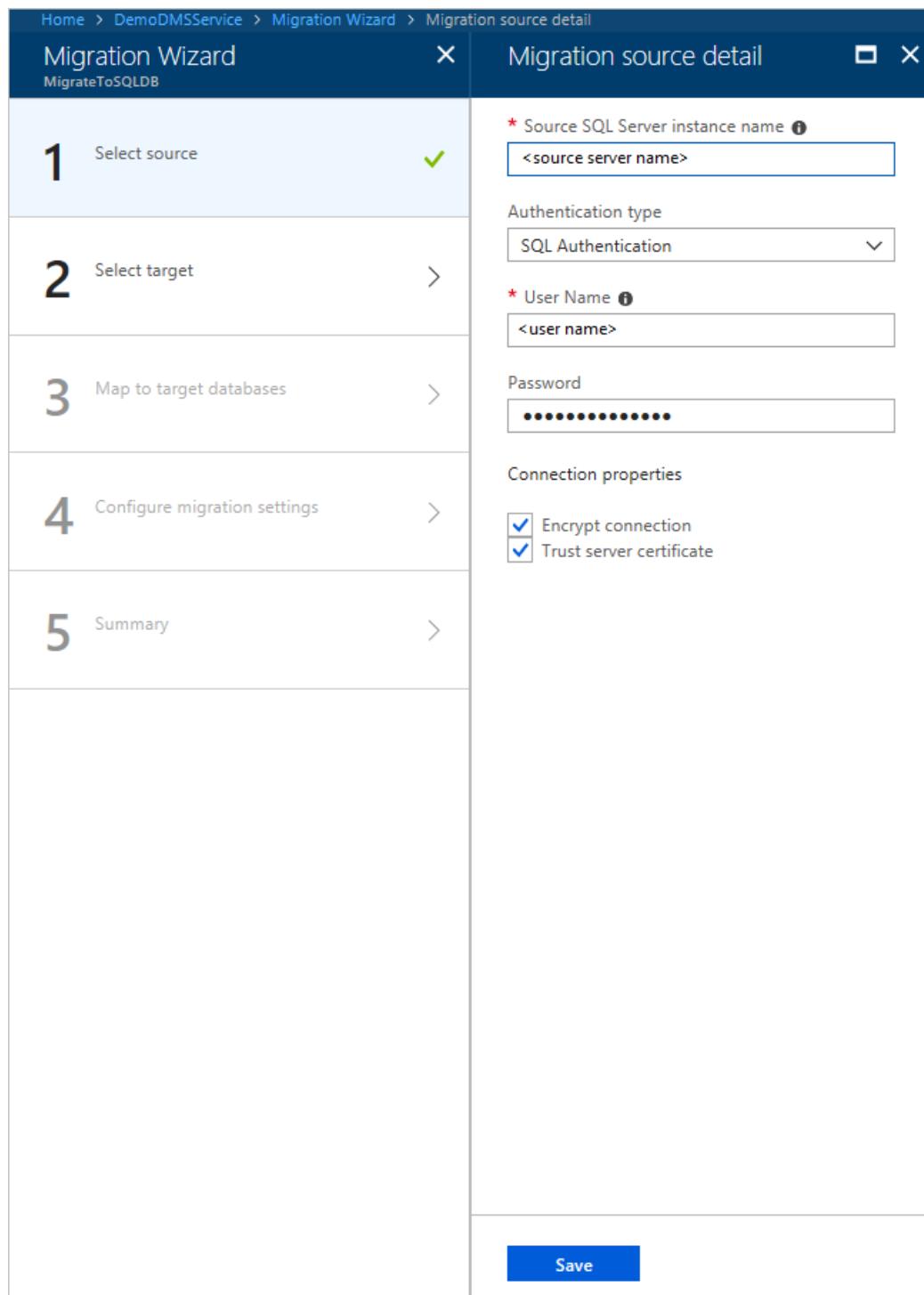
Make sure to use a Fully Qualified Domain Name (FQDN) for the source SQL Server instance name. You can also use the IP Address for situations in which DNS name resolution isn't possible.

2. If you haven't installed a trusted certificate on your source server, select the **Trust server certificate** check box.

When a trusted certificate isn't installed, SQL Server generates a self-signed certificate when the instance is started. This certificate is used to encrypt the credentials for client connections.

Caution

SSL connections that are encrypted using a self-signed certificate do not provide strong security. They are susceptible to man-in-the-middle attacks. You should not rely on SSL using self-signed certificates in a production environment or on servers that are connected to the internet.

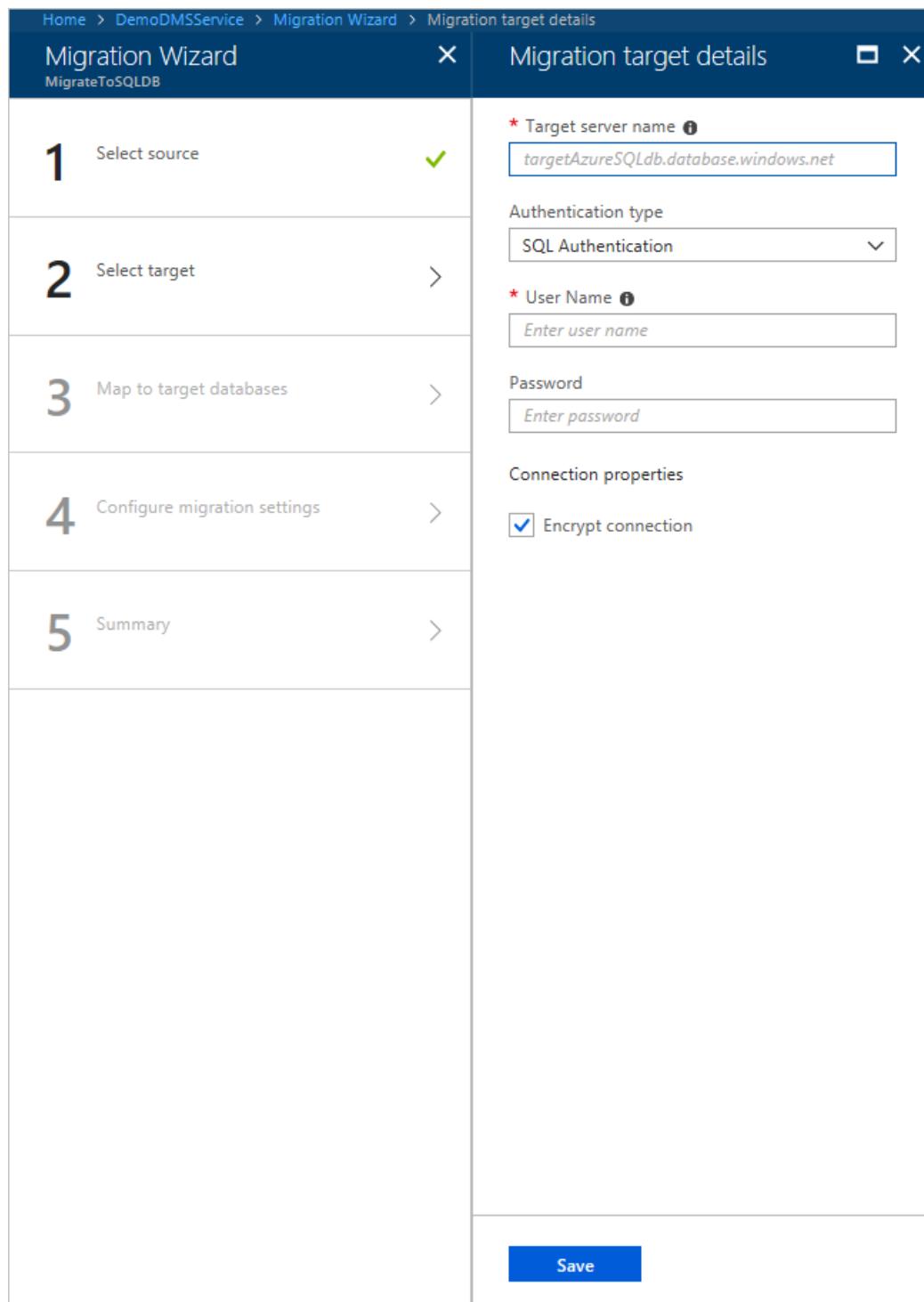


IMPORTANT

If you use SSIS, DMS does not currently support the migration of source SSISDB, but you can redeploy your SSIS projects/packages to the destination SSISDB hosted by Azure SQL Database. For more information about migrating SSIS packages, see the article [Migrate SQL Server Integration Services packages to Azure](#).

Specify target details

1. Select **Save**, and then on the **Migration target details** screen, specify the connection details for the target Azure SQL Database server, which is the pre-provisioned Azure SQL Database to which the **AdventureWorks2012** schema was deployed by using the DMA.



2. Select **Save**, and then on the **Map to target databases** screen, map the source and the target database for migration.

If the target database contains the same database name as the source database, the Azure Database Migration Service selects the target database by default.

Migration Wizard

MigrateToSQLDB

1 Select source ✓

2 Select target ✓

3 Map to target databases >

4 Configure migration settings >

5 Summary >

Map to target databases

Search to filter items... All Page 1 of 1

SOURCE DATABASE	SIZE	TARGET DATABASE
<input checked="" type="checkbox"/> AdventureWorks2012	229.06 MB	AdventureWorksAzure
<input type="checkbox"/> contosopayrolldb	5.00 MB	
<input type="checkbox"/> HRMinDowntime	147.31 MB	
<input type="checkbox"/> SitesEE	372.00 MB	
<input type="checkbox"/> StackOverflowEE	486.00 MB	
<input type="checkbox"/> test	21.00 MB	

Save

3. Select **Save**, on the **Select tables** screen, expand the table listing, and then review the list of affected fields.

The Azure Database Migration Service auto selects all the empty source tables that exist on the target Azure SQL Database instance. If you want to remigrate tables that already include data, you need to explicitly select the tables on this blade.

Migration Wizard

MigrateToSQLDB

1 Select source ✓

2 Select target ✓

3 Map to target databases ✓

4 Configure migration settings ✓

5 Summary >

Select tables

Database settings

AdventureWorks2012 71 of 71

Search to filter items... All Page 1 of 2

NAME	Notes
<input checked="" type="checkbox"/> dbo.AWBuildVersion	The target table is not empty. If selected for migration, all the records in the table will be deleted...
<input checked="" type="checkbox"/> dbo.DatabaseLog	The target table is not empty. If selected for migration, all the records in the table will be deleted...
<input checked="" type="checkbox"/> dbo.ErrorLog	
<input checked="" type="checkbox"/> HumanResources.De...	
<input checked="" type="checkbox"/> HumanResources.E...	
<input checked="" type="checkbox"/> HumanResources.E...	
<input checked="" type="checkbox"/> HumanResources.E...	
<input checked="" type="checkbox"/> HumanResources.Jo...	
<input checked="" type="checkbox"/> HumanResources.Shift	
<input checked="" type="checkbox"/> Person.Address	
<input checked="" type="checkbox"/> Person.AddressType	
<input checked="" type="checkbox"/> Person.BusinessEntity	
<input checked="" type="checkbox"/> Person.BusinessEntit...	
<input checked="" type="checkbox"/> Person.ContactType	
<input checked="" type="checkbox"/> Person.CountryRegion	The target table is not empty. If selected for migration, all the records in the table will be deleted...

Save

4. Select **Save**, on the **Migration summary** screen, in the **Activity name** text box, specify a name for the migration activity, and then review the summary to ensure that the source and target details match what you previously specified.

The screenshot shows two windows side-by-side. On the left is the 'Migration Wizard' window, which has five steps listed vertically: 1. Select source, 2. Select target, 3. Map to target databases, 4. Configure migration settings, and 5. Summary. Step 1 has a green checkmark, while steps 2, 3, and 4 have yellow checkmarks. Step 5 has a right-pointing arrow. On the right is the 'Migration summary' window, which contains the following information:

- Migration project name:** MigrateToSQLDB
- Activity name:** Mymigration (highlighted with a purple border)
- Source server name:** <source server name>
- Source server version:** SQL Server 2012
11.0.7462.6
- Target server name:** <target server name>
- Target server version:** Azure SQL Database
12.0.2000.8
- Database(s) to migrate:** 1 of 6
- Type of activity:** Online data migration

A note in the summary window suggests upgrading the Azure SQL Database to Premium tier if DTU purchase model is used or Business critical tier (for example 8 or 16 vCORE) in the case of vCORE model. It also includes a link to 'Upgrade Your Database Service Tiers'.

Run the migration

- Select **Run migration**.

The migration activity window appears, and the **Status** of the activity is **Initializing**.

Home > <service instance> > Mymigration

Mymigration

Refresh Stop migration Delete activity

Source server	Source version	Source databases			
Target server	Target version	Type of activity			
Activity status		Duration			
10.105.129.164	SQL Server 2012	1			
dmazuresqldbserver.database.windows.net	Azure SQL Database	Online			
Running		---			
Database Name	Status	Migration Details	Duration	Estimated Application Downtime	Finish Date
AdventureWorks2012	Initializing	Initializing the migration pipeline	---	---	---

Monitor the migration

1. On the migration activity screen, select **Refresh** to update the display until the **Status** of the migration shows as **Running**.
2. Click on a specific database to get to the migration status for **Full data load** and **Incremental data sync** operations.

Home > <service instance> > MigrateToSQLDB > MyMigration1 > AdventureWorks2012

AdventureWorks2012

Refresh Start Cutover

Source database name	Full load completed	Incremental updates	Pending changes
AdventureWorks2012	46	0	0
Target database name	Full load queued	Incremental inserts	Applied changes
AdventureWorksAzure	0	0	0
Database status	Full load loading	Incremental deletes	Tables in error state ⓘ
Running	1	0	0
Migration details	Full load failed		
Full data load in progress	0		

Full load Incremental data sync

47 item(s) Page 1 of 5 next →

TABLE NAME	STATUS	COMPLETED	ROWS	VOLUME (MB)	DURATION
HumanResources.Department	Completed	7/23/2018 6:33:55 PM	16	1	00:00:03
HumanResources.Employee	Completed	7/23/2018 6:33:56 PM	290	1	00:00:04
HumanResources.EmployeeDe...	Completed	7/23/2018 6:33:56 PM	296	1	00:00:04
HumanResources.EmployeePa...	Completed	7/23/2018 6:34:00 PM	316	1	00:00:05
HumanResources.JobCandidate	Completed	7/23/2018 6:34:01 PM	13	1	00:00:05
HumanResources.Shift	Completed	7/23/2018 6:34:01 PM	3	1	00:00:05
Person.Address	Completed	7/23/2018 6:34:09 PM	19614	13.9	00:00:12
Person.AddressType	Completed	7/23/2018 6:34:05 PM	6	1	00:00:06
Person.BusinessEntity	Completed	7/23/2018 6:34:07 PM	20777	8.4	00:00:07
Person.BusinessEntityAddress	Completed	7/23/2018 6:34:07 PM	19614	9.9	00:00:06

Perform migration cutover

After the initial Full load is completed, the databases are marked **Ready to cutover**.

- When you're ready to complete the database migration, select **Start Cutover**.

Home > <service instance> > MigrateToSQLDB > MyMigration1 > AdventureWorks2012 > Complete cutover

AdventureWorks2012

Refresh Start Cutover

Source database name	Full load completed	Incremental updates	Pending changes
AdventureWorks2012	47	3	0
Target database name	Full load queued	Incremental inserts	Applied changes
AdventureWorksAzure	0	0	3
Database status	Full load loading	Incremental deletes	Tables in error state ⓘ
Running	0	0	0
Migration details	Full load failed		
Ready to cutover	0		

Complete cutover
AdventureWorks2012

When you are ready to do the migration cutover, perform the following steps to complete the database migration. Please note that the database is ready for cutover only after the full data load is completed.

- Stop all the incoming transactions coming to the source database.
- Wait until all the pending transactions have been applied to the target database. At that time the pending changes counter will set to 0.

Pending changes 0

Confirm

3. Reconnect your applications to the new Azure target database.

47 item(s) Page 4 of 5 next →

TABLE NAME	STATUS	COMPLETED	ROWS	VOLUME (MB)	DURATION
Sales.PersonCreditCard	Completed	7/23/2018 6:34:39 PM	19118	7.4	00:00:06
Sales.SalesOrderDetail	Completed	7/23/2018 6:48:12 PM	121317	86.5	00:13:35
Sales.SalesOrderHeader	Completed	7/23/2018 6:34:53 PM	31465	37.7	00:00:15
Sales.SalesOrderHeaderSalesR...	Completed	7/23/2018 6:34:50 PM	27647	10.8	00:00:11
Sales.SalesPerson	Completed	7/23/2018 6:34:50 PM	17	1	00:00:11
Sales.SalesPersonQuotaHistory	Completed	7/23/2018 6:34:51 PM	163	1	00:00:10
Sales.SalesReason	Completed	7/23/2018 6:34:54 PM	10	1	00:00:04
Sales.SalesTaxRate	Completed	7/23/2018 6:34:55 PM	29	1	00:00:04
Sales.SalesTerritory	Completed	7/23/2018 6:35:39 PM	10	1	00:00:47
Sales.SalesTerritoryHistory	Completed	7/23/2018 6:34:59 PM	17	1	00:00:05

- Make sure to stop all the incoming transactions to the source database; wait until the **Pending changes** counter shows **0**.

3. Select **Confirm**, and the select **Apply**.

4. When the database migration status shows **Completed**, connect your applications to the new target Azure SQL Database.

The screenshot shows the 'Complete cutover' page for the 'AdventureWorks2012' service instance. It displays migration statistics and a step-by-step guide for finalizing the migration.

Migration Statistics:

Source database name	Full load completed	Incremental updates	Pending changes
AdventureWorks2012	47	3	0
Target database name	Full load queued	Incremental inserts	Applied changes
AdventureWorksAzure	0	0	3
Database status	Full load loading	Incremental deletes	Tables in error state ⓘ
Complete	0	0	0
Migration details	Full load failed		
All changes applied	0		

Migration Details Table:

TABLE NAME	STATUS	COMPLETED	ROWS	VOLUME (MB)	DURATION
Sales.PersonCreditCard	Completed	7/23/2018 6:34:39 PM	19118	7.4	00:00:06
Sales.SalesOrderDetail	Completed	7/23/2018 6:48:12 PM	121317	86.5	00:13:35
Sales.SalesOrderHeader	Completed	7/23/2018 6:34:53 PM	31465	37.7	00:00:15
Sales.SalesOrderHeaderSalesR...	Completed	7/23/2018 6:34:50 PM	27647	10.8	00:00:11
Sales.SalesPerson	Completed	7/23/2018 6:34:50 PM	17	1	00:00:11
Sales.SalesPersonQuotaHistory	Completed	7/23/2018 6:34:51 PM	163	1	00:00:10
Sales.SalesReason	Completed	7/23/2018 6:34:54 PM	10	1	00:00:04
Sales.SalesTaxRate	Completed	7/23/2018 6:34:55 PM	29	1	00:00:04
Sales.SalesTerritory	Completed	7/23/2018 6:35:39 PM	10	1	00:00:47
Sales.SalesTerritoryHistory	Completed	7/23/2018 6:34:59 PM	17	1	00:00:05

Right Panel - Step-by-Step Guide:

1. Stop all the incoming transactions coming to the source database.
2. Wait until all the pending transactions have been applied to the target database. At that time the pending changes counter will set to 0.
3. Reconnect your applications to the new Azure target database.

Buttons:

- Pending changes: 0
- Confirm
- Apply

Next steps

- For information about known issues and limitations when performing online migrations to Azure SQL Database, see the article [Known issues and workarounds with Azure SQL Database online migrations](#).
- For information about the Azure Database Migration Service, see the article [What is the Azure Database Migration Service?](#).
- For information about Azure SQL Database, see the article [What is the Azure SQL Database service?](#).

Tutorial: Migrate SQL Server to an Azure SQL Database managed instance offline using DMS

1/8/2020 • 10 minutes to read • [Edit Online](#)

You can use Azure Database Migration Service to migrate the databases from an on-premises SQL Server instance to an [Azure SQL Database managed instance](#). For additional methods that may require some manual effort, see the article [SQL Server instance migration to Azure SQL Database managed instance](#).

In this tutorial, you migrate the **Adventureworks2012** database from an on-premises instance of SQL Server to a SQL Database managed instance by using Azure Database Migration Service.

In this tutorial, you learn how to:

- Create an instance of Azure Database Migration Service.
- Create a migration project by using Azure Database Migration Service.
- Run the migration.
- Monitor the migration.
- Download a migration report.

IMPORTANT

For offline migrations from SQL Server to SQL Database managed instance, Azure Database Migration Service can create the backup files for you. Alternately, you can provide the latest full database backup in the SMB network share that the service will use to migrate your databases. Do not append multiple backups into a single backup media; take each backup on a separate backup file. Note that you can use compressed backups as well, to reduce the likelihood of experiencing potential issues with migrating large backups.

TIP

When you migrate databases to Azure by using Azure Database Migration Service, you can do an *offline* or an *online* migration. With an offline migration, application downtime starts when the migration starts. With an online migration, downtime is limited to the time to cut over at the end of migration. We suggest that you test an offline migration to determine whether the downtime is acceptable; if not, do an online migration.

This article describes an offline migration from SQL Server to a SQL Database managed instance. For an online migration, see [Migrate SQL Server to an Azure SQL Database managed instance online using DMS](#).

Prerequisites

To complete this tutorial, you need to:

- Create a Microsoft Azure Virtual Network for Azure Database Migration Service by using the Azure Resource Manager deployment model, which provides site-to-site connectivity to your on-premises source servers by using either [ExpressRoute](#) or [VPN](#). [Learn network topologies for Azure SQL Database managed instance migrations using Azure Database Migration Service](#). For more information about creating a virtual network, see the [Virtual Network Documentation](#), and especially the quickstart articles with step-by-step details.

NOTE

During virtual network setup, if you use ExpressRoute with network peering to Microsoft, add the following service endpoints to the subnet in which the service will be provisioned:

- Target database endpoint (for example, SQL endpoint, Cosmos DB endpoint, and so on)
- Storage endpoint
- Service bus endpoint

This configuration is necessary because Azure Database Migration Service lacks internet connectivity.

- Ensure that your virtual network Network Security Group rules don't block the following inbound communication ports to Azure Database Migration Service: 443, 53, 9354, 445, 12000. For more detail on virtual network NSG traffic filtering, see the article [Filter network traffic with network security groups](#).
- Configure your [Windows Firewall for source database engine access](#).
- Open your Windows Firewall to allow Azure Database Migration Service to access the source SQL Server, which by default is TCP port 1433.
- If you're running multiple named SQL Server instances using dynamic ports, you may wish to enable the SQL Browser Service and allow access to UDP port 1434 through your firewalls so that Azure Database Migration Service can connect to a named instance on your source server.
- If you're using a firewall appliance in front of your source databases, you may need to add firewall rules to allow Azure Database Migration Service to access the source database(s) for migration, as well as files via SMB port 445.
- Create an Azure SQL Database managed instance by following the detail in the article [Create an Azure SQL Database managed instance in the Azure portal](#).
- Ensure that the logins used to connect the source SQL Server and target managed instance are members of the sysadmin server role.

NOTE

By default, Azure Database Migration Service only supports migrating SQL logins. However, you can enable the ability to migrate Windows logins by:

- Ensuring that the target SQL Database managed instance has AAD read access, which can be configured via the Azure portal by a user with the **Company Administrator** or a **Global Administrator** role.
- Configuring your Azure Database Migration Service instance to enable Windows user/group login migrations, which is set up via the Azure portal, on the Configuration page. After enabling this setting, restart the service for the changes to take effect.

After restarting the service, Windows user/group logins appear in the list of logins available for migration. For any Windows user/group logins you migrate, you are prompted to provide the associated domain name. Service user accounts (account with domain name NT AUTHORITY) and virtual user accounts (account name with domain name NT SERVICE) are not supported.

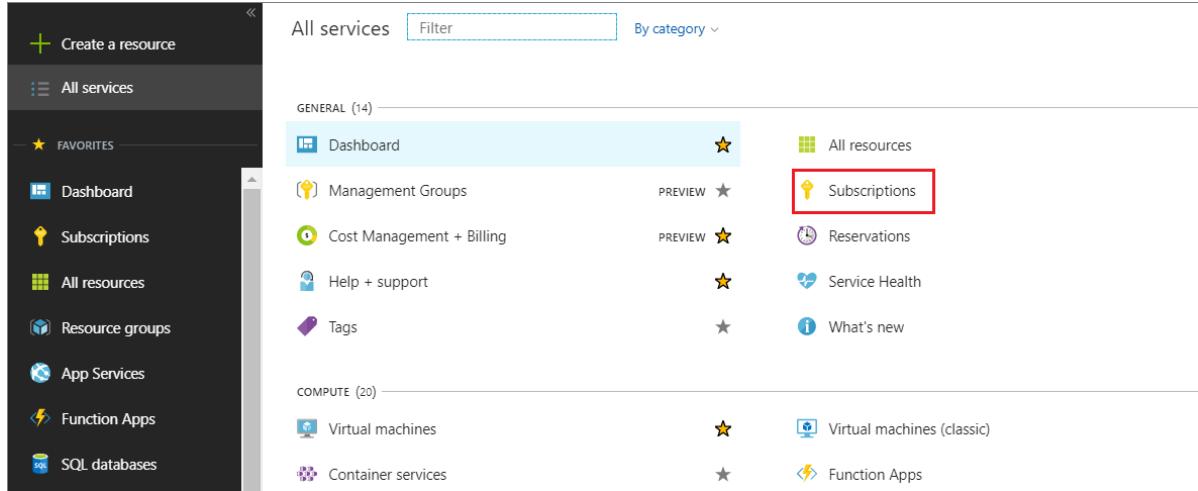
- Create a network share that Azure Database Migration Service can use to back up the source database.
- Ensure that the service account running the source SQL Server instance has write privileges on the network share that you created and that the computer account for the source server has read/write access to the same share.
- Make a note of a Windows user (and password) that has full control privilege on the network share that you previously created. Azure Database Migration Service impersonates the user credential to upload the

backup files to Azure Storage container for restore operation.

- Create a blob container and retrieve its SAS URI by using the steps in the article [Manage Azure Blob Storage resources with Storage Explorer](#), be sure to select all permissions (Read, Write, Delete, List) on the policy window while creating the SAS URI. This detail provides Azure Database Migration Service with access to your storage account container for uploading the backup files used for migrating databases to Azure SQL Database managed instance.

Register the Microsoft.DataMigration resource provider

1. Sign in to the Azure portal, select **All services**, and then select **Subscriptions**.



The screenshot shows the Azure portal's 'All services' blade. On the left, there's a sidebar with 'Create a resource', 'All services', 'FAVORITES' (Dashboard, Subscriptions, All resources, Resource groups, App Services, Function Apps, SQL databases), and a 'Filter' input field. The main area is titled 'All services' with a 'Filter' button and a 'By category' dropdown. It lists services under 'GENERAL' (14) and 'COMPUTE' (20). The 'Subscriptions' service is listed under GENERAL, with a yellow key icon and the text 'Subscriptions'. A red box highlights this 'Subscriptions' link.

2. Select the subscription in which you want to create the instance of Azure Database Migration Service, and then select **Resource providers**.

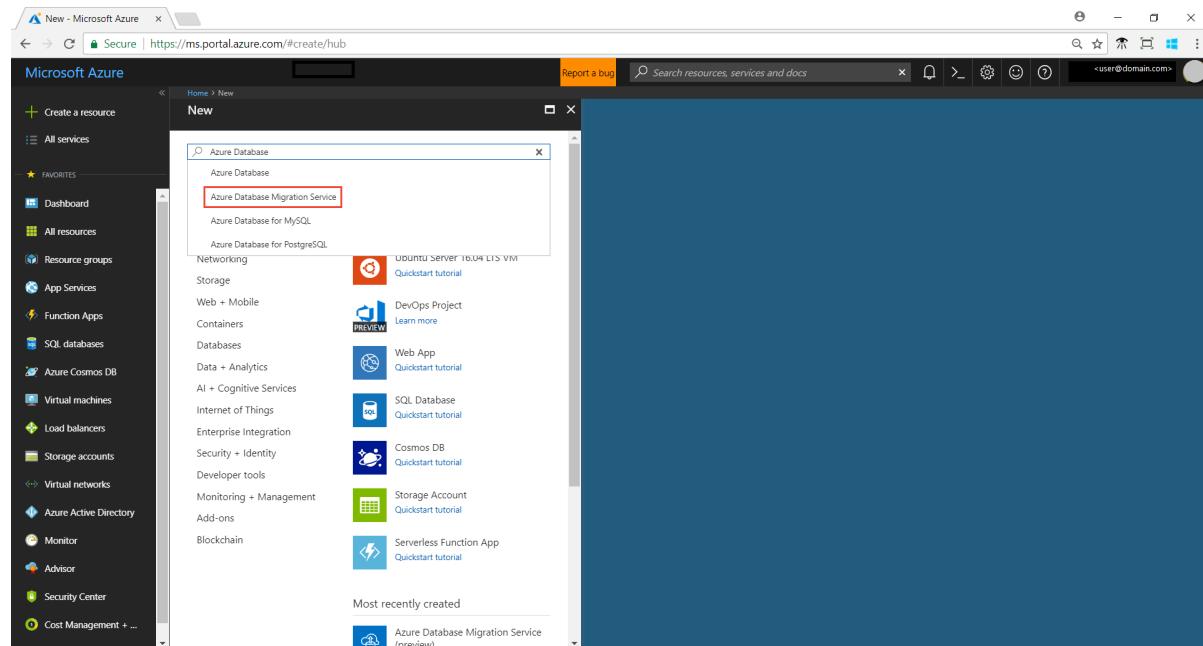
3. Search for migration, and then to the right of **Microsoft.DataMigration**, select **Register**.

Provider	Status	Action
Microsoft.DataMigration	NotRegistered	Register

Create an Azure Database Migration Service instance

1. In the Azure portal, select **+ Create a resource**, search for **Azure Database Migration Service**, and then

select **Azure Database Migration Service** from the drop-down list.



2. On the **Azure Database Migration Service** screen, select **Create**.

3. On the **Create Migration Service** screen, specify a name for the service, the subscription, and a new or existing resource group.

4. Select the location in which you want to create the instance of DMS.

5. Select an existing virtual network or create one.

The virtual network provides Azure Database Migration Service with access to the source SQL Server and target Azure SQL Database managed instance.

For more information on how to create a virtual network in Azure portal, see the article [Create a virtual network using the Azure portal](#).

For additional detail, see the article [Network topologies for Azure SQL DB managed instance migrations using Azure Database Migration Service](#).

6. Select a pricing tier.

For more information on costs and pricing tiers, see the [pricing page](#).

Create Migration Service

Service Name i
DMSTest ✓

* Subscription
DMSInternalPreviewtest

* Select a resource group i
 Create new Use existing
createresourcegroup ✓

* Location i
Central US

* Virtual network
NEWNET02/default

* Pricing tier
Basic: 2 vCores

Azure Database Migration Service quick start template
Experience our database migration service with pre-created source and target

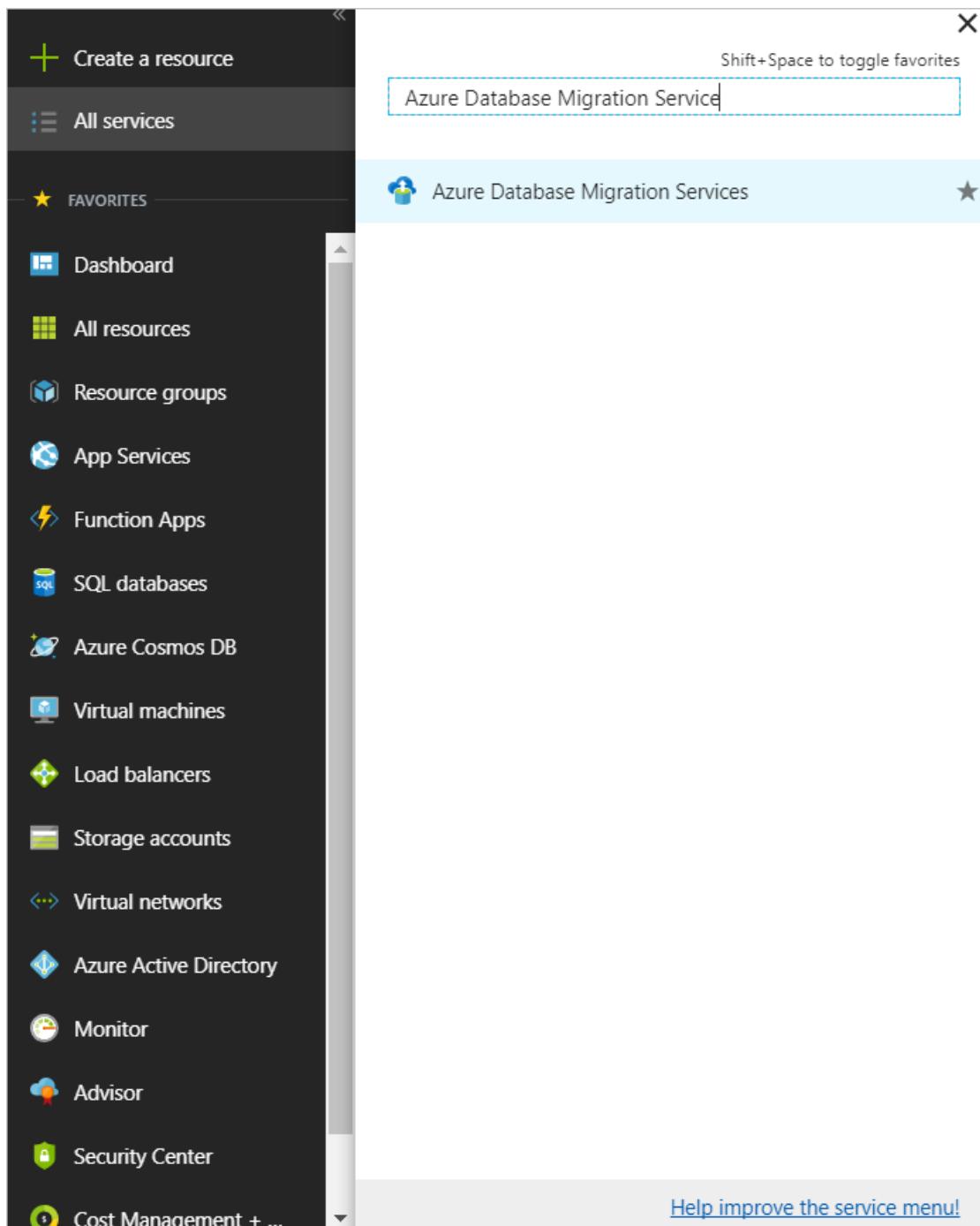
Create [Automation options](#)

7. Select **Create** to create the service.

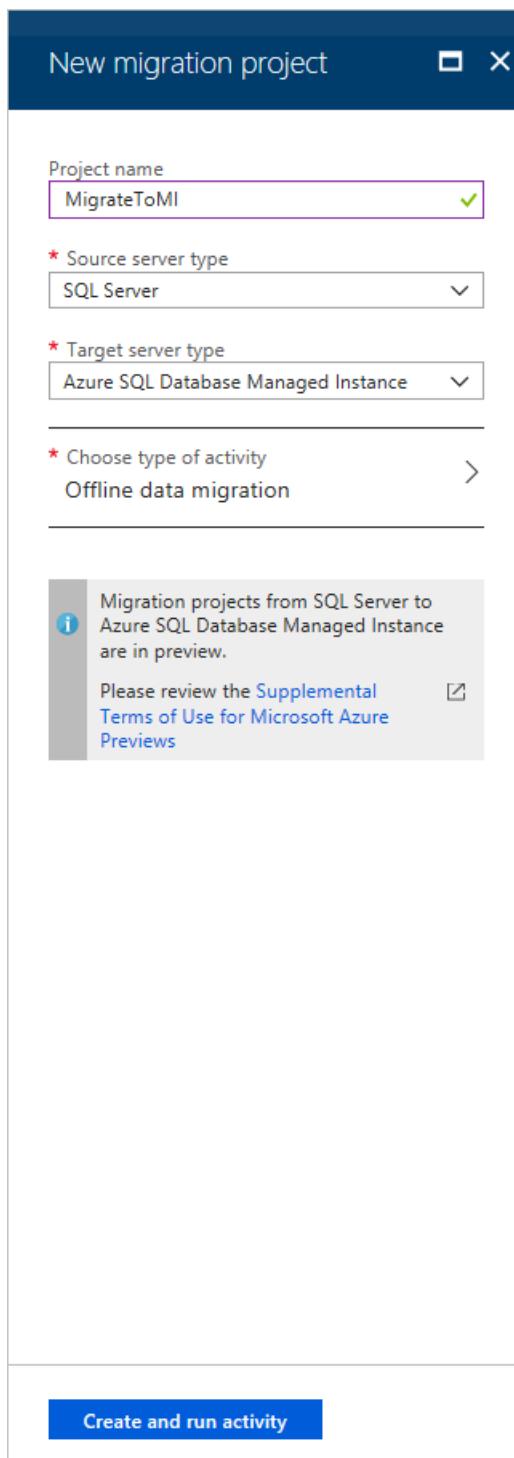
Create a migration project

After an instance of the service is created, locate it within the Azure portal, open it, and then create a new migration project.

1. In the Azure portal, select **All services**, search for Azure Database Migration Service, and then select **Azure Database Migration Services**.



2. On the **Azure Database Migration Service** screen, search for the name of the instance that you created, and then select the instance.
3. Select + **New Migration Project**.
4. On the **New migration project** screen, specify a name for the project, in the **Source server type** text box, select **SQL Server**, in the **Target server type** text box, select **Azure SQL Database Managed Instance**, and then for **Choose type of activity**, select **Offline data migration**.



5. Select **Create** to create the project.

Specify source details

1. On the **Migration source detail** screen, specify the connection details for the source SQL Server.
2. If you haven't installed a trusted certificate on your server, select the **Trust server certificate** check box.

When a trusted certificate isn't installed, SQL Server generates a self-signed certificate when the instance is started. This certificate is used to encrypt the credentials for client connections.

Caution

SSL connections that are encrypted using a self-signed certificate does not provide strong security. They are susceptible to man-in-the-middle attacks. You should not rely on SSL using self-signed certificates in a production environment or on servers that are connected to the internet.

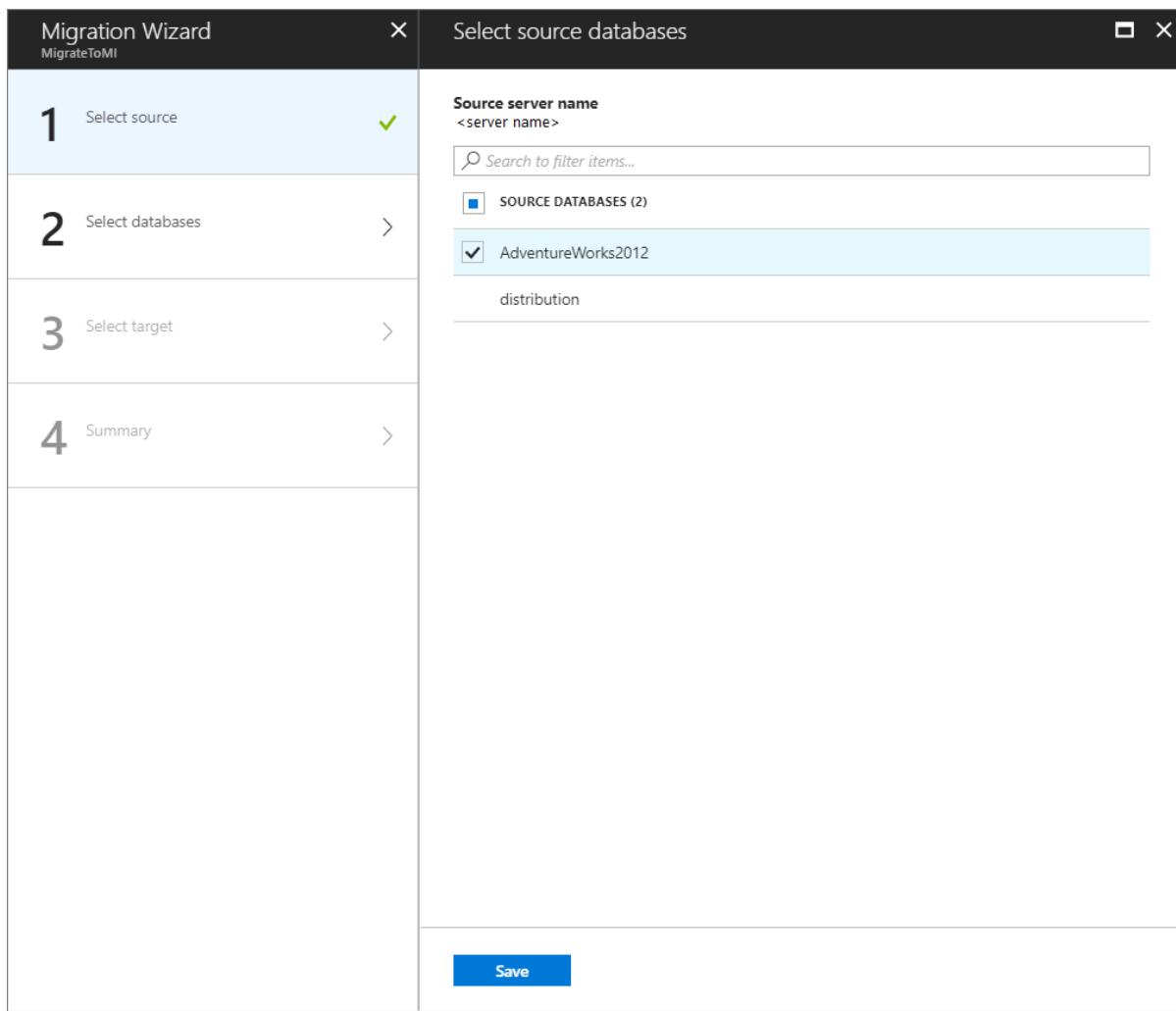
Migration Wizard

MigrateToMI

Source details

1 Select source >	* Source SQL Server instance name <input type="text" value="Servername.domainname.com"/>
2 Select databases >	Authentication type SQL Authentication
3 Select target >	* User Name <input type="text" value="Enter user name"/> Password <input type="text" value="Enter password"/>
4 Summary >	Connection properties <input checked="" type="checkbox"/> Encrypt connection <input type="checkbox"/> Trust server certificate
Save	

3. Select **Save**.
4. On the **Select source databases** screen, select the **Adventureworks2012** database for migration.



IMPORTANT

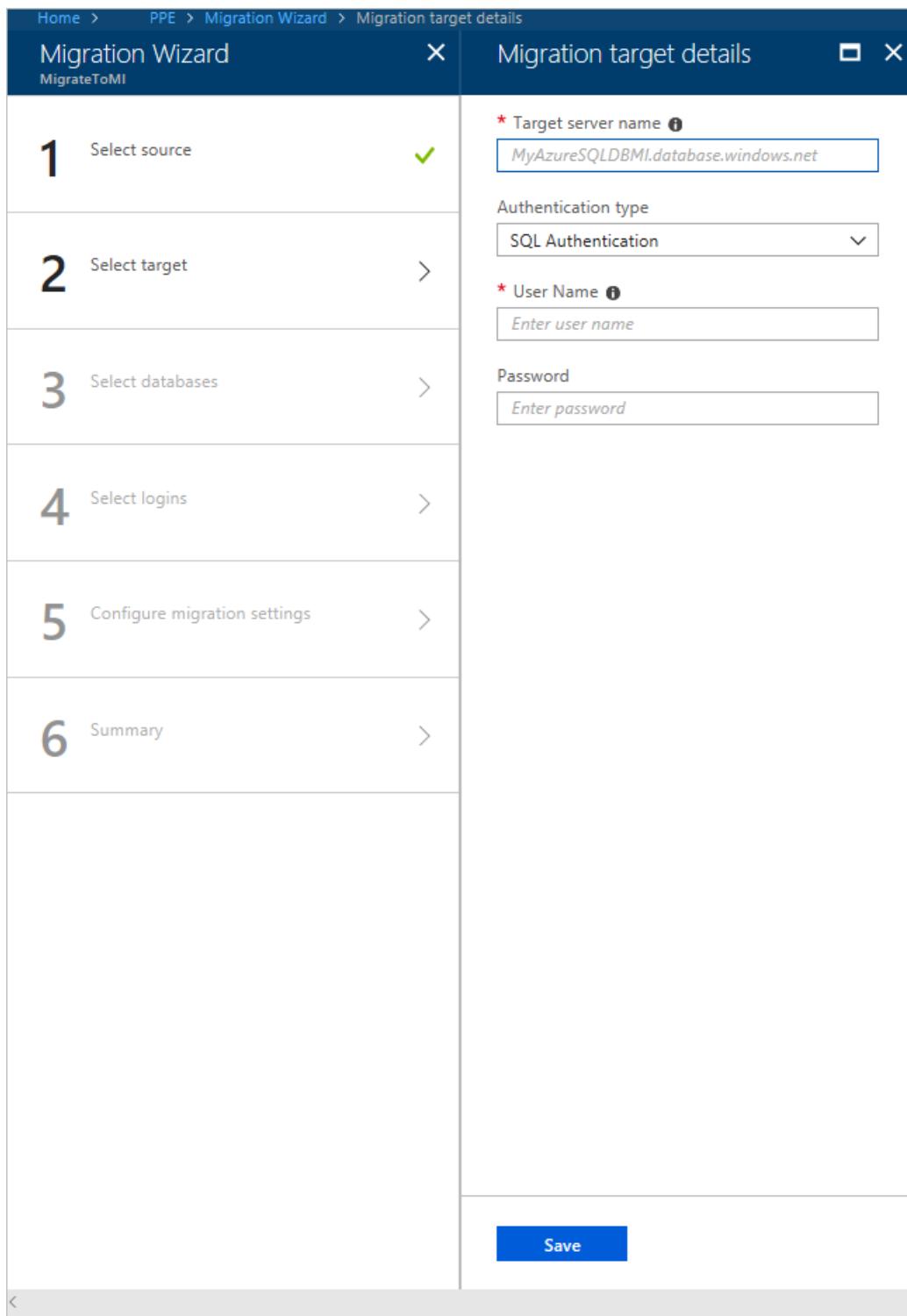
If you use SQL Server Integration Services (SSIS), DMS does not currently support migrating the catalog database for your SSIS projects/packages (SSISDB) from SQL Server to Azure SQL Database managed instance. However, you can provision SSIS in Azure Data Factory (ADF) and redeploy your SSIS projects/packages to the destination SSISDB hosted by Azure SQL Database managed instance. For more information about migrating SSIS packages, see the article [Migrate SQL Server Integration Services packages to Azure](#).

5. Select **Save**.

Specify target details

1. On the **Migration target details** screen, specify the connection details for the target, which is the pre-provisioned Azure SQL Database managed instance to which you're migrating the **AdventureWorks2012** database.

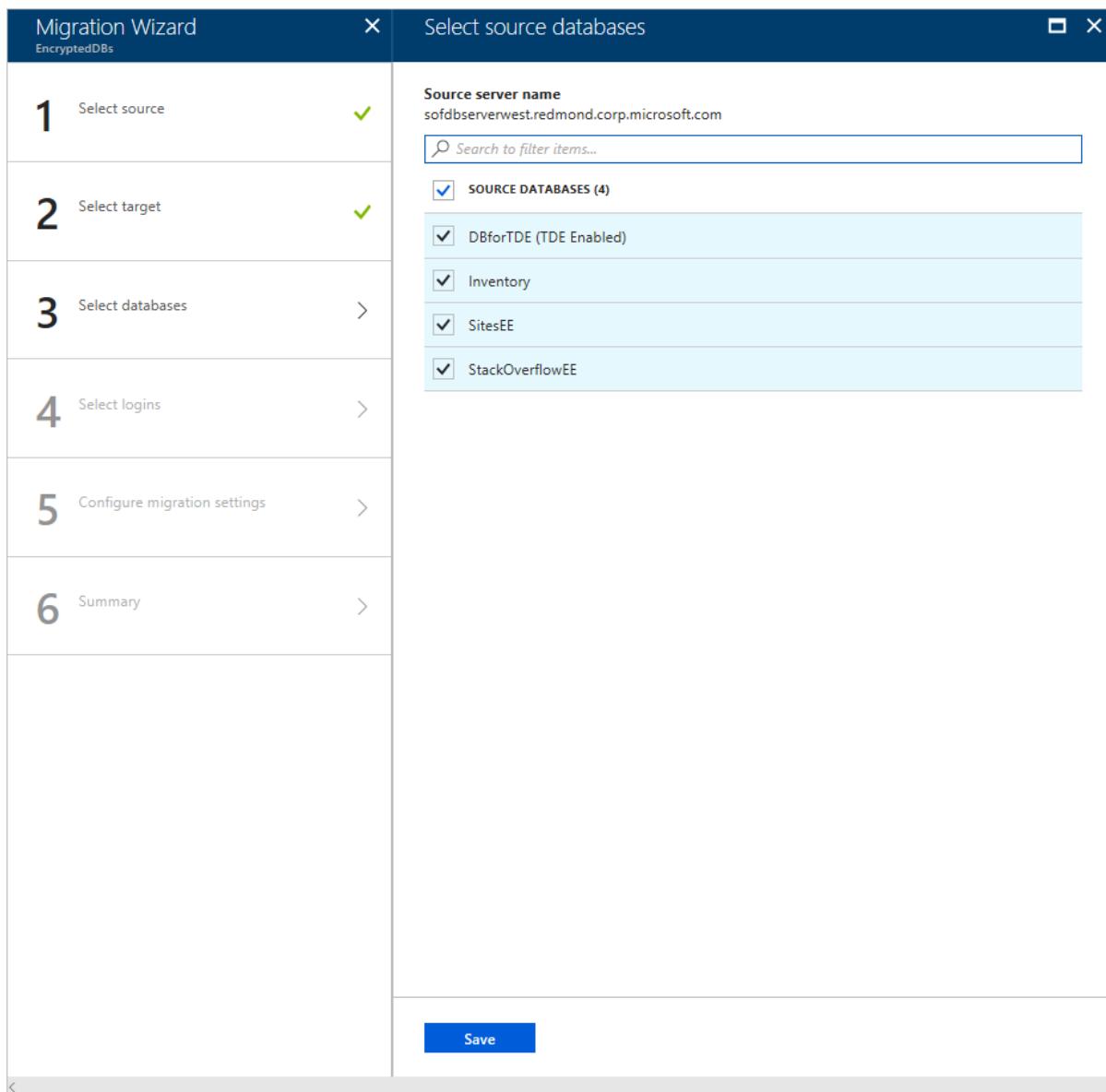
If you haven't already provisioned the SQL Database managed instance, select the [link](#) to help you provision the instance. You can still continue with project creation and then, when the Azure SQL Database managed instance is ready, return to this specific project to execute the migration.



2. Select **Save**.

Select source databases

1. On the **Select source databases** screen, select the source database that you want to migrate.



2. Select **Save**.

Select logins

1. On the **Select logins** screen, select the logins that you want to migrate.

NOTE

By default, Azure Database Migration Service only supports migrating SQL logins. To enable support for migrating Windows logins, see the **Prerequisites** section of this tutorial.

Migration Wizard

Select logins

Source server name
10.105.129.164

SOURCE LOGINS (28)	LOGIN TYPE	DEFAULT DATABASE	STATUS	READY TO MOVE
NT SERVICE\Winmgmt	Windows	master	Enabled	LoginType 'WindowsUser' is not supported for migration.
NT Service\SQLLaaSExten...	Windows	master	Enabled	LoginType 'WindowsUser' is not supported for migration.
NT SERVICE\SQLSERVER...	Windows	master	Enabled	LoginType 'WindowsUser' is not supported for migration.
NT Service\MSSQLSERVER...	Windows	master	Enabled	LoginType 'WindowsUser' is not supported for migration.
DemoSQLServer\demouser	Windows	master	Enabled	LoginType 'WindowsUser' is not supported for migration.
NT AUTHORITY\SYSTEM	Windows	master	Enabled	LoginType 'WindowsUser' is not supported for migration.
##MS_PolicyTsqlExecution...	SQL	master	Disabled	Login '##MS_PolicyTsqlExecutionLogin##' created by SQL component is not supported for migration.
sa	SQL	master	Disabled	Login 'sa' is not migrated as it is a system login.
<input checked="" type="checkbox"/> sqluser	SQL	master	Enabled	
<input checked="" type="checkbox"/> distributor_admin	SQL	master	Enabled	
##MS_PolicyEventProcess...	SQL	master	Disabled	Login '##MS_PolicyEventProcessingLogin##' created by SQL component is not supported for migration.
NT SERVICE\SQLWriter	Windows	master	Enabled	LoginType 'WindowsUser' is not supported for migration.
DEMOSSQLSERVER\rajpo	Windows	master	Enabled	LoginType 'WindowsUser' is not supported for migration.
demouser	SQL	master	Enabled	⚠ Login already exists, only securables will be migrated and orphan users mapped.
NT SERVICE\Winmgmt	Windows	master	Enabled	LoginType 'WindowsUser' is not supported for migration.
DemoSQLServer\demouser	Windows	master	Enabled	LoginType 'WindowsUser' is not supported for migration.
NT SERVICE\SQLSERVER...	Windows	master	Enabled	LoginType 'WindowsUser' is not supported for migration.
NT SERVICE\SQLWriter	Windows	master	Enabled	LoginType 'WindowsUser' is not supported for migration.
NT AUTHORITY\SYSTEM	Windows	master	Enabled	LoginType 'WindowsUser' is not supported for migration.
##MS_PolicyTsqlExecution...	SQL	master	Disabled	Login '##MS_PolicyTsqlExecutionLogin##' created by SQL component is not supported for migration.

Save

2. Select **Save**.

Configure migration settings

- On the **Configure migration settings** screen, provide the following detail:

Choose source backup option	Choose the option I will provide latest backup files when you already have full backup files available for DMS to use for database migration. Choose the option I will let Azure Database Migration Service create backup files when you want DMS to take the source database full backup at first and use it for migration.
Network location share	The local SMB network share that Azure Database Migration Service can take the source database backups to. The service account running source SQL Server instance must have write privileges on this network share. Provide an FQDN or IP addresses of the server in the network share, for example, '\\servername.domainname.com\\backupfolder' or '\\IP address\\backupfolder'.
User name	Make sure that the Windows user has full control privilege on the network share that you provided above. Azure Database Migration Service will impersonate the user credential to upload the backup files to Azure Storage container for restore operation. If TDE-enabled databases are selected for migration, the above windows user must be the built-in administrator account and User Account Control must be disabled for Azure Database Migration Service to upload and delete the certificates files.)
Password	Password for the user.

Storage account settings	The SAS URI that provides Azure Database Migration Service with access to your storage account container to which the service uploads the backup files and that is used for migrating databases to Azure SQL Database managed instance. Learn how to get the SAS URI for blob container.
TDE Settings	If you're migrating the source databases with Transparent Data Encryption (TDE) enabled, you need to have write privileges on the target Azure SQL Database managed instance. Select the subscription in which the Azure SQL Database managed instance provisioned from the drop-down menu. Select the target Azure SQL Database Managed Instance in the drop-down menu.

The screenshot shows the Migration Wizard on the left and the Configure migration settings page on the right. The Migration Wizard has six steps: 1. Select source (done), 2. Select target (done), 3. Select databases (done), 4. Select logins (done), 5. Configure migration settings (in progress), and 6. Summary (not yet started). The Configure migration settings page contains several sections: Choose source backup option (set to 'I will let Azure Database Migration Service create backup files.'), Backup settings (warning about write privileges on the network share), Storage account settings (SAS URI input field containing a placeholder SAS URL), and TDE Settings (subscription dropdown set to 'DMSINTERNALDEMO' and target instance dropdown set to 'demomi').

2. Select **Save**.

Review the migration summary

- On the **Migration summary** screen, in the **Activity name** text box, specify a name for the migration activity.
- Expand the **Validation option** section to display the **Choose validation option** screen, specify whether to validate the migrated database for query correctness, and then select **Save**.
- Review and verify the details associated with the migration project.

Migration Wizard
ReadyMI

Migration summary

Activity name
demomigration

Target server name
demomi.scus1b3fba4c2acae.database.windows.net

Target server version
Azure SQL Database Managed Instance
12.0.2000.8

Source server name
10.105.129.164

Source server version
SQL Server 2012
11.0.7462.6

Databases to migrate
2 of 5

Login(s) to migrate
6/28

* Validation option
1/1 options selected

Run migration

4. Select **Save**.

Run the migration

- Select **Run migration**.

The migration activity window appears, and the status of the activity is **Pending**.

Monitor the migration

- In the migration activity screen, select **Refresh** to update the display.

Home > DemoDMSService > ReadyMI > MyMigrateToMI

MyMigrateToMI

Delete migration Stop migration Refresh Download report

Source server	Target server						
10.105.129.164	demomi.scus1b3fba4c2acae.database.windows.net						
Source version	Target version						
SQL Server 2012	Azure SQL Database Managed Instance						
11.0.7462.6	12.0.2000.8						
Server objects							
3							
<input type="text"/> Search to filter items...							
SERVER OBJ...	NOT START...	IN PROGRESS	COMPLETED	WARNING	FAILED	STOPPED	SKIPPED
Databases	0	0	2	0	0	0	0
Logins	0	0	1	0	0	0	0

You can further expand the databases and logins categories to monitor the migration status of the respective server objects.

MyMigrateToMI

Delete migration Stop migration Refresh Download report

Source server	Target server						
10.105.129.164	demomi.scus1b3fba4c2acae.database.windows.net						
Source version	Target version						
SQL Server 2012	Azure SQL Database Managed Instance						
11.0.7462.6	12.0.2000.8						
Server objects							
3							
<input type="text"/> Search to filter items...							
SERVER OBJ...	NOT START...	IN PROGRESS	COMPLETED	WARNING	FAILED	STOPPED	SKIPPED
Databases	0	0	2	0	0	0	0
Logins	0	0	1	0	0	0	0

MyMigrateToMI

Refresh

Source server	Target server		
10.105.129.164	demomi.scus1b3fba4c2acae.database.windows.net		
Source version	Target version		
SQL Server 2012	Azure SQL Database Managed Instance		
11.0.7462.6	12.0.2000.8		
Logins			
1			
<input type="text"/> Search to filter items...			
NAME	STATUS	MESSAGE	DURATION
demouser	Completed		00:00:01

- After the migration completes, select **Download report** to get a report listing the details associated with

the migration process.

3. Verify that the target database on the target Azure SQL Database managed instance environment.

Next steps

- For a tutorial showing you how to migrate a database to a managed instance using the T-SQL RESTORE command, see [Restore a backup to a managed instance using the restore command](#).
- For information about managed instance, see [What is a managed instance](#).
- For information about connecting apps to a managed instance, see [Connect applications](#).

Tutorial: Migrate SQL Server to an Azure SQL Database managed instance online using DMS

1/10/2020 • 12 minutes to read • [Edit Online](#)

You can use Azure Database Migration Service to migrate the databases from an on-premises SQL Server instance to an [Azure SQL Database managed instance](#) with minimal downtime. For additional methods that may require some manual effort, see the article [SQL Server instance migration to Azure SQL Database managed instance](#).

In this tutorial, you migrate the **Adventureworks2012** database from an on-premises instance of SQL Server to a SQL Database managed instance with minimal downtime by using Azure Database Migration Service.

In this tutorial, you learn how to:

- Create an instance of Azure Database Migration Service.
- Create a migration project and start online migration by using Azure Database Migration Service.
- Monitor the migration.
- Perform the migration cutover when you are ready.

IMPORTANT

For online migrations from SQL Server to SQL Database managed instance using Azure Database Migration Service, you must provide the full database backup and subsequent log backups in the SMB network share that the service can use to migrate your databases. Azure Database Migration Service does not initiate any backups, and instead uses existing backups, which you may already have as part of your disaster recovery plan, for the migration. Be sure that you take [backups using the WITH CHECKSUM option](#). Also, make sure not to append multiple backups (i.e. full and t-log) into a single backup media; take each backup on a separate backup file. Finally, you can use compressed backups to reduce the likelihood of experiencing potential issues associated with migrating large backups.

NOTE

Using Azure Database Migration Service to perform an online migration requires creating an instance based on the Premium pricing tier.

IMPORTANT

For an optimal migration experience, Microsoft recommends creating an instance of Azure Database Migration Service in the same Azure region as the target database. Moving data across regions or geographies can slow down the migration process and introduce errors.

IMPORTANT

Reduce the duration of the online migration process as much as possible to minimize the risk of interruption caused by instance reconfiguration or planned maintenance. In case of such an event, migration process will start from the beginning. In case of planned maintenance, there is a grace period of 36 hours before migration process is restarted.

TIP

When you migrate databases to Azure by using Azure Database Migration Service, you can do an *offline* or an *online* migration. With an offline migration, application downtime starts when the migration starts. With an online migration, downtime is limited to the time to cut over at the end of migration. We suggest that you test an offline migration to determine whether the downtime is acceptable; if not, do an online migration.

This article describes an online migration from SQL Server to a SQL Database managed instance. For an offline migration, see [Migrate SQL Server to a SQL Database managed instance offline using DMS](#).

Prerequisites

To complete this tutorial, you need to:

- Create a Microsoft Azure Virtual Network for Azure Database Migration Service by using the Azure Resource Manager deployment model, which provides site-to-site connectivity to your on-premises source servers by using either [ExpressRoute](#) or [VPN](#). [Learn network topologies for Azure SQL Database managed instance migrations using Azure Database Migration Service](#). For more information about creating a virtual network, see the [Virtual Network Documentation](#), and especially the quickstart articles with step-by-step details.

NOTE

During virtual network setup, if you use ExpressRoute with network peering to Microsoft, add the following service endpoints to the subnet in which the service will be provisioned:

- Target database endpoint (for example, SQL endpoint, Cosmos DB endpoint, and so on)
- Storage endpoint
- Service bus endpoint

This configuration is necessary because Azure Database Migration Service lacks internet connectivity.

If you don't have site-to-site connectivity between the on-premises network and Azure or if there is limited site-to-site connectivity bandwidth, consider using Azure Database Migration Service in hybrid mode (Preview). Hybrid mode leverages an on-premises migration worker together with an instance of Azure Database Migration Service running in the cloud. To create an instance of Azure Database Migration Service in hybrid mode, see the article [Create an instance of Azure Database Migration Service in hybrid mode using the Azure portal](#).

IMPORTANT

Regarding the storage account used as part of the migration, you must either:

- Choose to allow all network to access the storage account.
- Turn on [subnet delegation](#) on MI subnet and update the Storage Account firewall rules to allow this subnet.

- Ensure that your virtual network Network Security Group rules don't block the following inbound communication ports to Azure Database Migration Service: 443, 53, 9354, 445, 12000. For more detail on virtual network NSG traffic filtering, see the article [Filter network traffic with network security groups](#).
- Configure your [Windows Firewall for source database engine access](#).
- Open your Windows Firewall to allow Azure Database Migration Service to access the source SQL Server, which by default is TCP port 1433.
- If you're running multiple named SQL Server instances using dynamic ports, you may wish to enable the

SQL Browser Service and allow access to UDP port 1434 through your firewalls so that Azure Database Migration Service can connect to a named instance on your source server.

- If you're using a firewall appliance in front of your source databases, you may need to add firewall rules to allow Azure Database Migration Service to access the source database(s) for migration, as well as files via SMB port 445.
- Create a SQL Database managed instance by following the detail in the article [Create an Azure SQL Database managed instance in the Azure portal](#).
- Ensure that the logins used to connect the source SQL Server and the target managed instance are members of the sysadmin server role.
- Provide an SMB network share that contains all your database full database backup files and subsequent transaction log backup files, which Azure Database Migration Service can use for database migration.
- Ensure that the service account running the source SQL Server instance has write privileges on the network share that you created and that the computer account for the source server has read/write access to the same share.
- Make a note of a Windows user (and password) that has full control privilege on the network share that you previously created. Azure Database Migration Service impersonates the user credential to upload the backup files to Azure Storage container for restore operation.
- Create an Azure Active Directory Application ID that generates the Application ID key that Azure Database Migration Service can use to connect to target Azure Database managed instance and Azure Storage Container. For more information, see the article [Use portal to create an Azure Active Directory application and service principal that can access resources](#).

NOTE

Azure Database Migration Service requires the Contributor permission on the subscription for the specified Application ID. Alternatively, you can create custom roles that grant the specific permissions that Azure Database Migration Service requires. For step-by-step guidance about using custom roles, see the article [Custom roles for SQL Server to SQL Database managed instance online migrations](#).

- Create or make a note of **Standard Performance tier**, Azure Storage Account, that allows DMS service to upload the database backup files to and use for migrating databases. Make sure to create the Azure Storage Account in the same region as the Azure Database Migration Service instance is created.

Register the Microsoft.DataMigration resource provider

1. Sign in to the Azure portal, select **All services**, and then select **Subscriptions**.

The screenshot shows the Azure portal's 'All services' blade. On the left, there's a sidebar with various service icons and names. The 'Subscriptions' icon is highlighted with a red box. The main area lists services under 'GENERAL' and 'COMPUTE' categories. In the 'GENERAL' section, 'Subscriptions' is listed under 'PREVIEW'. In the 'COMPUTE' section, 'Virtual machines' and 'Container services' are listed.

2. Select the subscription in which you want to create the instance of Azure Database Migration Service, and then select **Resource providers**.

The screenshot shows the 'Subscriptions' blade in the Azure portal. The left sidebar has the same list of services as the previous screenshot. The main area shows a 'Subscriptions' list with one item: '<subscription> <subscription ID>'. On the right, a navigation menu is open, showing options like 'Overview', 'Access control (IAM)', 'Diagnose and solve problems', 'COST MANAGEMENT + BILLING', 'Partner information', 'SETTINGS', 'Programmatic deployment', 'Resource groups', 'Resources', 'Usage + quotas', 'Policies', 'Management certificates', 'My permissions', 'Resource providers' (which is highlighted with a red box), 'Properties', 'Resource locks', 'SUPPORT + TROUBLESHOOTING', and 'New support request'.

3. Search for migration, and then to the right of **Microsoft.DataMigration**, select **Register**.

The screenshot shows the Azure portal interface. On the left, there's a sidebar with various service icons like Dashboard, All resources, App Services, etc. The main area shows a 'Subscriptions' blade with a search bar and a list of selected subscriptions. To the right, a 'Subscription' blade is open for a specific subscription, showing sections for Overview, Access control (IAM), Diagnose and solve problems, Cost management + Billing, Partner information, Settings (Programmatic deployment, Resource groups, Resources, Usage + quotas, Policies, Management certificates, My permissions, Resource providers, Properties, Resource locks), and a 'Resource providers' table. In the 'Resource providers' table, there's a row for 'Microsoft.DataMigration' with a status of 'NotRegistered'. A red box highlights the 'Register' button.

Create an Azure Database Migration Service instance

1. In the Azure portal, select **+ Create a resource**, search for **Azure Database Migration Service**, and then select **Azure Database Migration Service** from the drop-down list.

The screenshot shows the Azure portal 'New' blade. The search bar at the top has 'Azure Database Migration Service' typed into it and is highlighted with a red box. Below the search bar, there's a list of service categories and individual services. Under 'Databases', the 'Azure Database Migration Service' option is visible. The rest of the blade shows other service categories like Networking, Storage, Web + Mobile, etc., each with their respective icons and quickstart tutorials.

2. On the **Azure Database Migration Service** screen, select **Create**.

The Azure Database Migration Service (DMS) is designed to streamline the process of migrating on-premises databases to Azure. DMS will simplify the migration of existing on-premises SQL Server and Oracle databases to Azure SQL Database, Azure SQL Managed Instance or Microsoft SQL Server in an Azure Virtual Machine. Learn [more](#)

Before using DMS we recommend you complete the following three steps:

1. Open the [Azure Database Migration Guide](#) for step by step guidance through the migration process
2. Assess your SQL Server on-premises database(s) for feature parity and potential compatibility issues by using [Data Migration Assistant \(DMA\)](#). Alternatively, if you are migrating from Oracle, use the [SQL Server Migration Assistant \(SSMA\)](#)
3. Based on your database needs, create a target database using one of the database services: Azure SQL Database, Azure SQL Managed Instance or SQL Server in an Azure Virtual Machine.

Once your assessments are complete, fixes are applied and schema is deployed, proceed with creating a migration service by clicking **Create** below to migrate the data from your source database to the target.

[Save for later](#)

PUBLISHER	Microsoft
USEFUL LINKS	Documentation Privacy Statement

Create

3. On the **Create Migration Service** screen, specify a name for the service, the subscription, and a new or existing resource group.
4. Select the location in which you want to create the instance of DMS.
5. Select an existing virtual network or create one.

The virtual network provides Azure Database Migration Service with access to the source SQL Server and target SQL Database managed instance.

For more information on how to create a virtual network in Azure portal, see the article [Create a virtual network using the Azure portal](#).

For additional detail, see the article [Network topologies for Azure SQL Database managed instance migrations using Azure Database Migration Service](#).

6. Select a SKU from the Premium pricing tier.

NOTE

Online migrations are supported only when using the Premium tier.

For more information on costs and pricing tiers, see the [pricing page](#).

Create Migration Service

Service Name i
DMSTest ✓

* Subscription
DMSMigrationDemo2

* Select a resource group i
DemoDMSService ✓
[Create new](#)

* Location i
West Europe

* Virtual network
ERWestEurope-PvtApp-WEU-VNET-9048/ >

* Pricing tier
Business Critical: 4 vCores >

Azure Database Migration Service quick start template i
Experience our database migration service with pre-created source and target

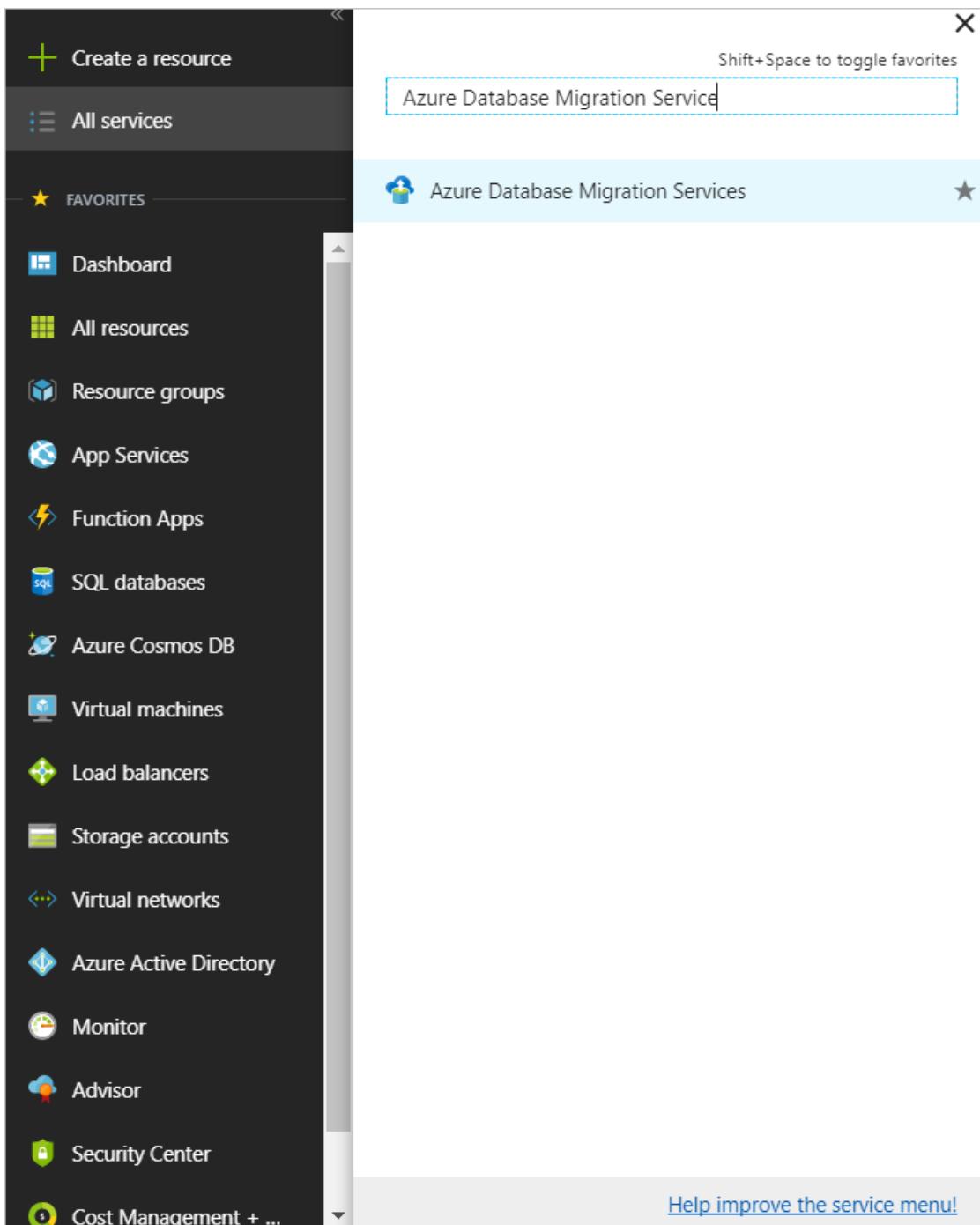
Create [Automation options](#)

7. Select **Create** to create the service.

Create a migration project

After an instance of the service is created, locate it within the Azure portal, open it, and then create a new migration project.

1. In the Azure portal, select **All services**, search for Azure Database Migration Service, and then select **Azure Database Migration Services**.



2. On the **Azure Database Migration Service** screen, search for the name of the instance that you created, and then select the instance.
3. Select + **New Migration Project**.
4. On the **New migration project** screen, specify a name for the project, in the **Source server type** text box, select **SQL Server**, in the **Target server type** text box, select **Azure SQL Database Managed Instance**, and then for **Choose type of activity**, select **Online data migration**.

New migration project

Project name: OnlineMiMigration ✓

* Source server type: SQL Server

* Target server type: Azure SQL Database Managed Instance

* Choose type of activity: Online data migration [preview] >

To successfully use Database Migration Service (DMS) to migrate data, you need to:

1. Create the target Azure SQL Database Managed Instance.
2. Use DMA to assess your on-premises SQL Server database(s) for feature parity and compatibility issues.
3. Apply the fixes to target Azure Database Managed Instance as recommended by DMA after the migration.

Online migration projects from SQL Server to Azure SQL Database Managed Instance are in preview and subject to the [Supplemental Terms of Use for Microsoft Azure Previews](#).

Create and run activity

5. Select **Create and run activity** to create the project.

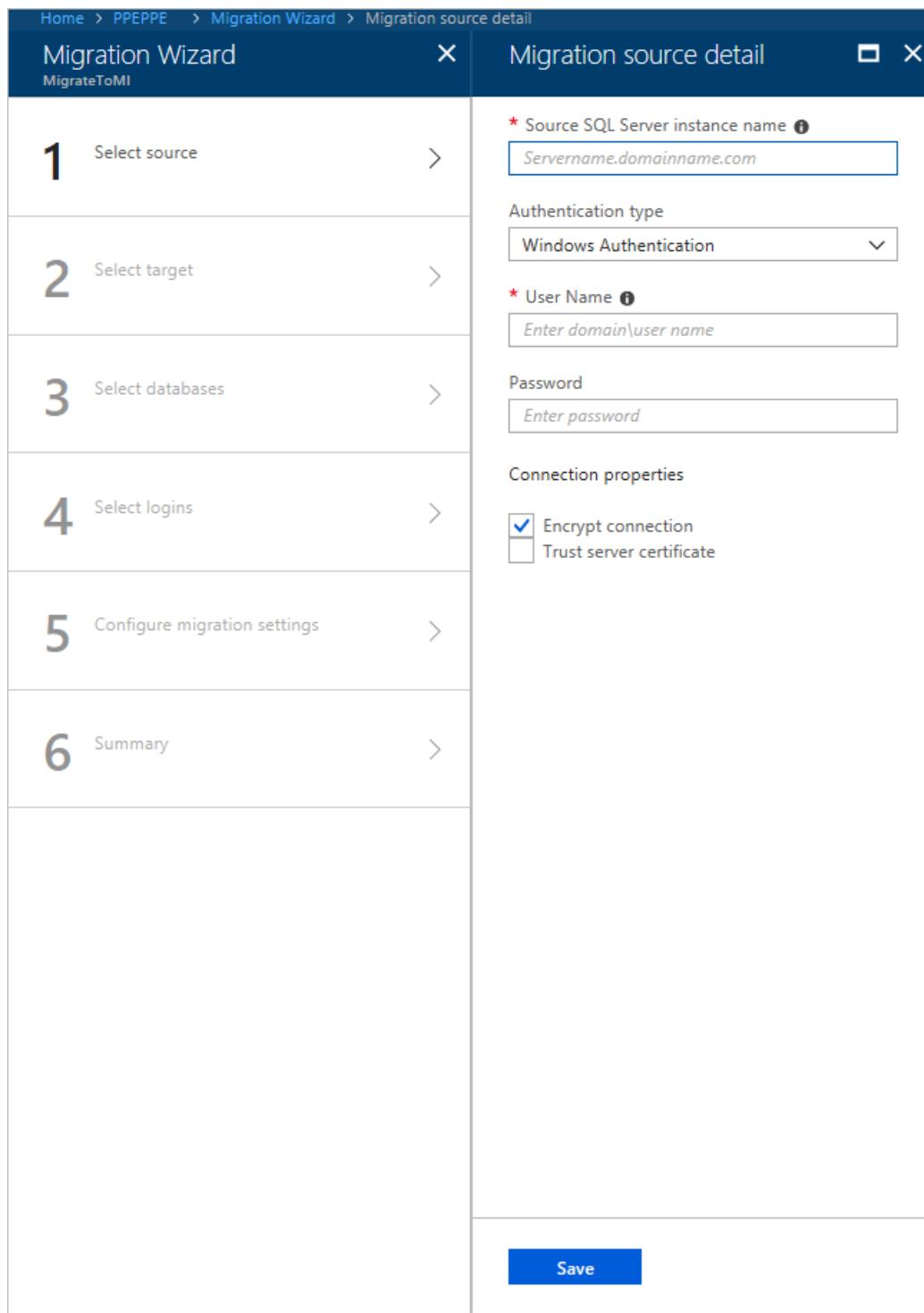
Specify source details

1. On the **Migration source detail** screen, specify the connection details for the source SQL Server.
2. If you haven't installed a trusted certificate on your server, select the **Trust server certificate** check box.

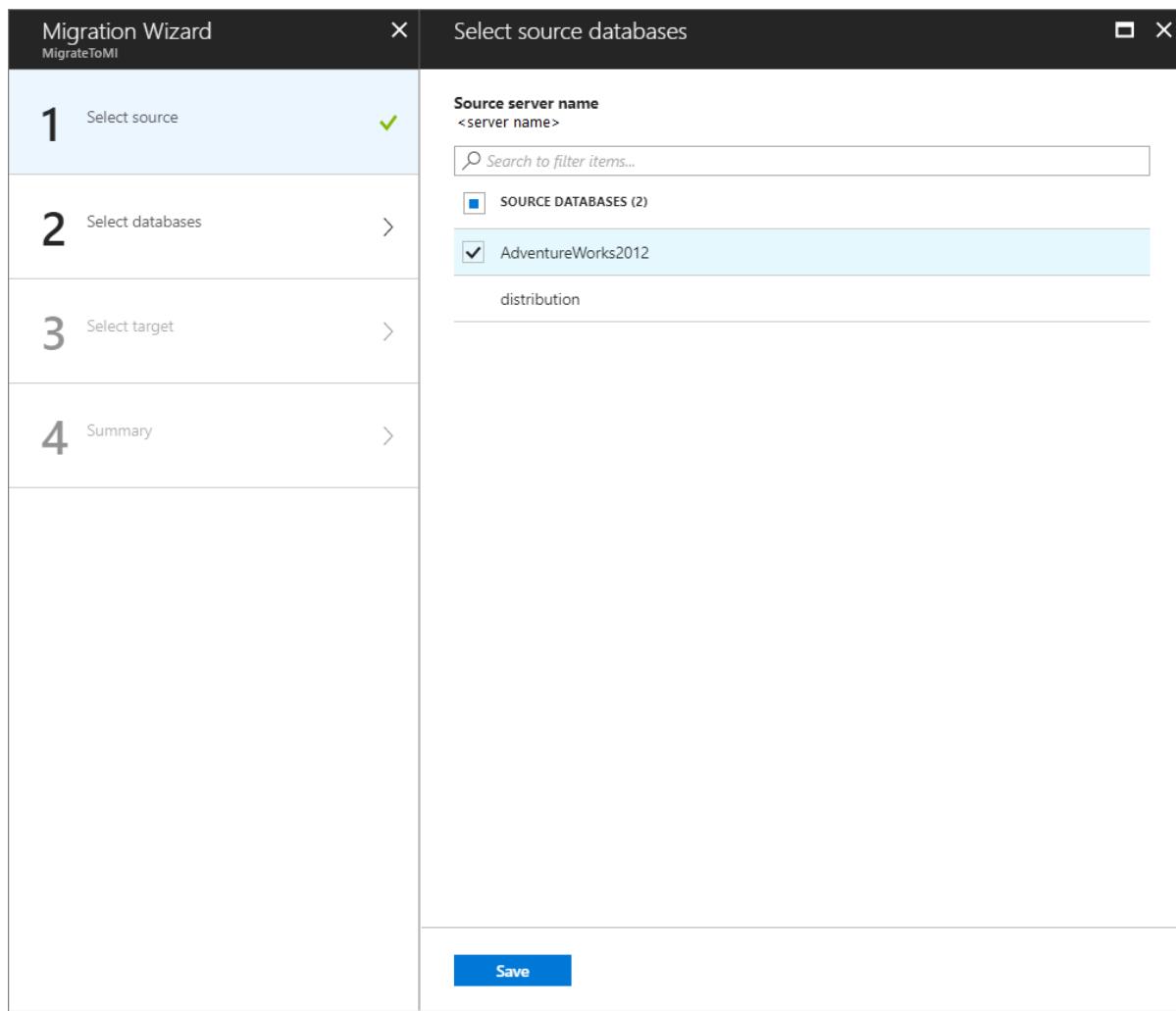
When a trusted certificate isn't installed, SQL Server generates a self-signed certificate when the instance is started. This certificate is used to encrypt the credentials for client connections.

Caution

SSL connections that are encrypted using a self-signed certificate does not provide strong security. They are susceptible to man-in-the-middle attacks. You should not rely on SSL using self-signed certificates in a production environment or on servers that are connected to the internet.



3. Select **Save**.
4. On the **Select source databases** screen, select the **Adventureworks2012** database for migration.



IMPORTANT

If you use SQL Server Integration Services (SSIS), DMS does not currently support migrating the catalog database for your SSIS projects/packages (SSISDB) from SQL Server to Azure SQL Database managed instance. However, you can provision SSIS in Azure Data Factory (ADF) and redeploy your SSIS projects/packages to the destination SSISDB hosted by Azure SQL Database managed instance. For more information about migrating SSIS packages, see the article [Migrate SQL Server Integration Services packages to Azure](#).

5. Select **Save**.

Specify target details

1. On the **Migration target details** screen, specify the **Application ID** and **Key** that the DMS instance can use to connect to the target instance of Azure SQL Database managed instance and the Azure Storage Account.

For more information, see the article [Use portal to create an Azure Active Directory application and service principal that can access resources](#).

2. Select the **Subscription** containing the target instance of Azure SQL Database managed instance, and then select the target instance.

If you haven't already provisioned the Azure SQL Database managed instance, select the [link](#) to help you provision the instance. When the Azure SQL Database managed instance is ready, return to this specific project to execute the migration.

3. Provide **SQL User** and **Password** to connect to the Azure SQL Database managed instance.

Home > DemoDMSService > Migration Wizard > Migration target details

Migration Wizard		X
1	Select source	✓
2	Select target	>
3	Select databases	>
4	Configure migration settings	>
5	Summary	>

Migration target details

* Application ID that Azure Database Migration Service will use to call restore service ⓘ
Please enter an application ID

* Key
Enter application key
[Learn how to create](#)

* Select subscription containing the target Azure SQL Database Managed Instance server
<subscription ID>

* Select target Azure SQL Database Managed Instance ⓘ

[Learn how to create](#)

* Sql User Name ⓘ
Enter user name

Password
Enter password

Save

4. Select **Save**.

Select source databases

1. On the **Select source databases** screen, select the source database that you want to migrate.

Migration Wizard X

OnlineMiMigration

1	Select source	✓
2	Select target	✓
3	Select databases	>
4	Configure migration settings	>
5	Summary	>

Select source databases X

Source server name
10.139.160.117

Search to filter items...

SOURCE DATABASES (9)

AdventureWorks2012

contosopayrolldb

DBforTDE

HR

HR_new

HRMinDowntime

HRMinforVM

HRnew

Inventory

Save

2. Select **Save**.

Configure migration settings

1. On the **Configure migration settings** screen, provide the following detail:

SMB Network location share	The local SMB network share or Azure file share that contains the Full database backup files and transaction log backup files that Azure Database Migration Service can use for migration. The service account running the source SQL Server instance must have read\write privileges on this network share. Provide an FQDN or IP addresses of the server in the network share, for example, '\\servername.domainname.com\\backupfolder' or '\\IP address\\backupfolder'. For improved performance, it's recommended to use separate folder for each database to be migrated. You can provide the database level file share path by using the Advanced Settings option.
-----------------------------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

User name	Make sure that the Windows user has full control privilege on the network share that you provided above. Azure Database Migration Service will impersonate the user credential to upload the backup files to Azure Storage container for restore operation. If using Azure File share, use the storage account name pre-pended with AZURE\ as the username.
Password	Password for the user. If using Azure file share, use a storage account key as the password.
Subscription of the Azure Storage Account	Select the subscription that contains the Azure Storage Account.
Azure Storage Account	Select the Azure Storage Account that DMS can upload the backup files from the SMB network share to and use for database migration. We recommend selecting the Storage Account in the same region as the DMS service for optimal file upload performance.

Home > DemoDMSService > Migration Wizard > Configure migration settings

Migration Wizard
OnlineMigration

- 1 Select source** ✓
- 2 Select target** ✓
- 3 Select databases** ✓
- 4 Configure migration settings** >
- 5 Summary** >

Configure migration settings

Backup settings

⚠ Ensure that the service account running the source SQL Server instance has write privileges on the network location share that you created.

* Network share location that Azure Database Migration Service can take database backups to
\\Servername.domainname.com\Backupfolder

Storage account settings

⚠ Select the subscription containing the desired storage account
DMSInternalPreviewtest

ℹ Select a Storage account created in location 'Central US' that allows Azure Database Migration Service to upload database backup files to and use for migrating databases to a Azure Sql Managed Instance. Use this [link](#) to learn more about creating a Storage account

⚠ Storage account that Azure Database Migration Service will upload the files to
Select a storage account

[Advanced settings ▾](#)

Save

NOTE

If Azure Database Migration Service shows error 'System Error 53' or 'System Error 57', the cause might result from an inability of Azure Database Migration Service to access Azure file share. If you encounter one of these errors, please grant access to the storage account from the virtual network using the instructions [here](#).

IMPORTANT

If loopback check functionality is enabled and the source SQL Server and file share are on the same computer, then source won't be able to access the files here using FQDN. To fix this issue, disable loopback check functionality using the instructions [here](#).

2. Select **Save**.

Review the migration summary

1. On the **Migration summary** screen, in the **Activity name** text box, specify a name for the migration activity.
2. Review and verify the details associated with the migration project.

Home > DemoDMSService > Migration Wizard > Migration summary

Migration Wizard		X	Migration summary		□ X
OnlineMiMigration			Activity name		
1	Select source	✓	Onlinemigration		✓
2	Select target	✓	Target server name		
			Target server version		
			Azure SQL Database Managed Instance		
			12.0.2000.8		
3	Select databases	✓	Source server name		
			10.139.160.117		
4	Configure migration settings	✓	Source server version		
			SQL Server 2012		
			11.0.7462.6		
5	Summary	>	Database(s) to migrate		
			1 of 9		
			Type of activity		
			Online data migration [preview]		

Run migration

Run and monitor the migration

1. Select **Run migration**.
2. On the migration activity screen, select **Refresh** to update the display.

The screenshot shows the 'Onlinemigration' activity screen. At the top, there are three buttons: 'Refresh' (highlighted with a blue border), 'Stop migration', and 'Delete activity'. Below these are two columns of migration details:

Source server	Target server
<source server>	<target server>
Source version 11.0.7462.6 SQL Server 2012	Target version 12.0.2000.8 Azure SQL Database Managed Instance
Databases 1	Type of activity Online
Application ID <application ID>	Activity status Running

Below this is a search bar labeled 'Search' and a pagination area showing '1 item(s)', 'Page 1 of 1', and 'next →'.

DATABASE NAME	STATUS	DURATION	FINISH DATE
AdventureWorks2012	Full backup uploading	00:00:13	---

You can further expand the databases and logins categories to monitor the migration status of the respective server objects.

The screenshot shows the 'AdventureWorks2012' migration status page. At the top, it displays the migration path: Home > DemoDMSService > Onlinemigration > AdventureWorks2012. Below this, the title 'AdventureWorks2012' is shown, along with a refresh button and a 'Start Cutover' button.

The main content area contains two tables:

Source server	Target server	Database status	Last applied LSN
<source server>	<target server>	Log files uploading	---

Source version	Target version	Full backup file(s)	Last applied backup file(s)
11.0.7462.6 SQL Server 2012	12.0.2000.8 Azure SQL Database Managed Instance	ADWorksFULL.bak	---

Below these tables is a search bar labeled 'Search' and a table showing the status of uploaded backup files:

NAME	TYPE	STATUS
ADWorksFULL.bak	Database	Arrived
ADWorksTR1.bak	Transaction log	Uploaded

At the bottom right of the page, there are navigation links: 'prev' (disabled), 'Page 1 of 1', and 'next'.

Performing migration cutover

After the full database backup is restored on the target instance of SQL Database managed instance, the database is available for performing a migration cutover.

1. When you're ready to complete the online database migration, select **Start Cutover**.
2. Stop all the incoming traffic to source databases.
3. Take the [tail-log backup], make the backup file available in the SMB network share, and then wait until this final transaction log backup is restored.

At that point, you'll see **Pending changes** set to 0.

4. Select **Confirm**, and then select **Apply**.

Home > DemoDMService > Onlinemigration > AdventureWorks2012 > Complete cutover

AdventureWorks2012

Refresh Start Cutover

Source server	Target server	Database status	Last applied LSN
<source server>	<target server>	Log files uploading	465000000043200000
Source version	Target version	Full backup file(s)	Last applied backup file(s)
11.0.7462.6	12.0.2000.8	ADWorksFULL.bak	ADWorksTR1.bak
SQL Server 2012	Azure SQL Database Managed Instance		

Search: 0 item(s) Page 0 of 0

NAME	TYPE	STATUS

Pending changes 0 Confirm Apply

When you are ready to do the migration cutover, perform the following steps to complete the database migration. Please note that the database is ready for cutover only after a full backup has been restored on the target Azure SQL Database Managed Instance.

1. Stop all the incoming transactions coming to the source database.
2. Take the final tail transaction log backup and provide backup file in the SMB network share.
3. Confirm the above and click "Apply" to initiate the migration cutover.

5. When the database migration status shows **Completed**, connect your applications to the new target instance of Azure SQL Database managed instance.

Home > DemoDMService > Onlinemigration > AdventureWorks2012 > Complete cutover

AdventureWorks2012

Refresh Start Cutover

Source server	Target server	Database status	Last applied LSN
<source server>	<target server>	Completed	465000000043200000
Source version	Target version	Full backup file(s)	Last applied backup file(s)
11.0.7462.6	12.0.2000.8	ADWorksFULL.bak	ADWorksTR1.bak
SQL Server 2012	Azure SQL Database Managed Instance		

Search: 0 item(s) Page 0 of 0

NAME	TYPE	STATUS

Pending changes 0 Confirm Apply

When you are ready to do the migration cutover, perform the following steps to complete the database migration. Please note that the database is ready for cutover only after a full backup has been restored on the target Azure SQL Database Managed Instance.

1. Stop all the incoming transactions coming to the source database.
2. Take the final tail transaction log backup and provide backup file in the SMB network share.
3. Confirm the above and click "Apply" to initiate the migration cutover.

Completed

Next steps

- For a tutorial showing you how to migrate a database to a managed instance using the T-SQL RESTORE command, see [Restore a backup to a managed instance using the restore command](#).
- For information about managed instance, see [What is a managed instance](#).
- For information about connecting apps to a managed instance, see [Connect applications](#).

Tutorial: Migrate RDS SQL Server to Azure SQL Database or an Azure SQL Database managed instance online using DMS

1/8/2020 • 11 minutes to read • [Edit Online](#)

You can use the Azure Database Migration Service to migrate the databases from an RDS SQL Server instance to [Azure SQL Database](#) or an [Azure SQL Database managed instance](#) with minimal downtime. In this tutorial, you migrate the **Adventureworks2012** database restored to an RDS SQL Server instance of SQL Server 2012 (or later) to Azure SQL Database or an Azure SQL Database managed instance by using the Azure Database Migration Service.

In this tutorial, you learn how to:

- Create an instance of Azure SQL Database or an Azure SQL Database managed instance.
- Migrate the sample schema by using the Data Migration Assistant.
- Create an instance of the Azure Database Migration Service.
- Create a migration project by using the Azure Database Migration Service.
- Run the migration.
- Monitor the migration.
- Download a migration report.

NOTE

Using the Azure Database Migration Service to perform an online migration requires creating an instance based on the Premium pricing tier. For more information, see the Azure Database Migration Service [pricing](#) page.

IMPORTANT

For an optimal migration experience, Microsoft recommends creating an instance of the Azure Database Migration Service in the same Azure region as the target database. Moving data across regions or geographies can slow down the migration process and introduce errors.

TIP

When you migrate databases to Azure by using Azure Database Migration Service, you can do an *offline* or an *online* migration. With an offline migration, application downtime starts when the migration starts. With an online migration, downtime is limited to the time to cut over at the end of migration. We suggest that you test an offline migration to determine whether the downtime is acceptable; if not, do an online migration.

This article describes an online migration from RDS SQL Server to Azure SQL Database or an Azure SQL Database managed instance.

Prerequisites

To complete this tutorial, you need to:

- Create an [RDS SQL Server database](#).

- Create an instance of Azure SQL Database, which you do by following the detail in the article [Create an Azure SQL database in the Azure portal](#).

NOTE

If you are migrating to an Azure SQL Database managed instance, follow the detail in the article [Create an Azure SQL Database managed instance](#), and then create an empty database named **AdventureWorks2012**.

- Download and install the [Data Migration Assistant](#) (DMA) v3.3 or later.
- Create a Microsoft Azure Virtual Network for Azure Database Migration Service by using the Azure Resource Manager deployment model. If you're migrating to an Azure SQL Database managed instance, make sure to create the DMS instance in the same virtual network used for the Azure SQL Database managed instance, but in a different subnet. Alternately, if you use a different virtual network for DMS, you need to create a virtual network peering between the two virtual networks. For more information about creating a virtual network, see the [Virtual Network Documentation](#), and especially the quickstart articles with step-by-step details.

NOTE

During virtual network setup, if you use ExpressRoute with network peering to Microsoft, add the following service endpoints to the subnet in which the service will be provisioned:

- Target database endpoint (for example, SQL endpoint, Cosmos DB endpoint, and so on)
- Storage endpoint
- Service bus endpoint

This configuration is necessary because the Azure Database Migration Service lacks internet connectivity.

- Ensure that your virtual network Network Security Group rules don't block the following inbound communication ports to Azure Database Migration Service: 443, 53, 9354, 445, 12000. For more detail on virtual network NSG traffic filtering, see the article [Filter network traffic with network security groups](#).
- Configure your [Windows Firewall for database engine access](#).
- Open your Windows firewall to allow the Azure Database Migration Service to access the source SQL Server, which by default is TCP port 1433.
- Create a server-level [firewall rule](#) for the Azure SQL Database server to allow the Azure Database Migration Service access to the target databases. Provide the subnet range of the virtual network used for the Azure Database Migration Service.
- Ensure that the credentials used to connect to the source RDS SQL Server instance are associated with an account that is a member of "Processadmin" server role and a member of the "db_owner" database roles on all databases that are to be migrated.
- Ensure that the credentials used to connect to target Azure SQL Database instance have CONTROL DATABASE permission on the target Azure SQL databases and a member of the sysadmin role if migrating to an Azure SQL Database managed instance.
- The source RDS SQL Server version must be SQL Server 2012 and above. To determine the version that your SQL Server instance is running, see the article [How to determine the version, edition, and update level of SQL Server and its components](#).
- Enable Change Data Capture (CDC) on the RDS SQL Server database and all user table(s) selected for migration.

NOTE

You can use the script below to enable CDC on an RDS SQL Server database.

```
exec msdb.dbo.rds_cdc_enable_db 'AdventureWorks2012'
```

You can use the script below to enable CDC on all tables.

```
use <Database name>
go
exec sys.sp_cdc_enable_table
@source_schema = N'Schema name',
@source_name = N'table name',
@role_name = NULL,
@supports_net_changes = 1 --for PK table 1, non PK tables 0
GO
```

- Disable database triggers on the target Azure SQL Database.

NOTE

You can find the database triggers on the target Azure SQL Database by using the following query:

```
Use <Database name>
select * from sys.triggers
DISABLE TRIGGER (Transact-SQL)
```

For more information, see the article [DISABLE TRIGGER \(Transact-SQL\)](#).

Migrate the sample schema

Use DMA to migrate the schema to Azure SQL Database.

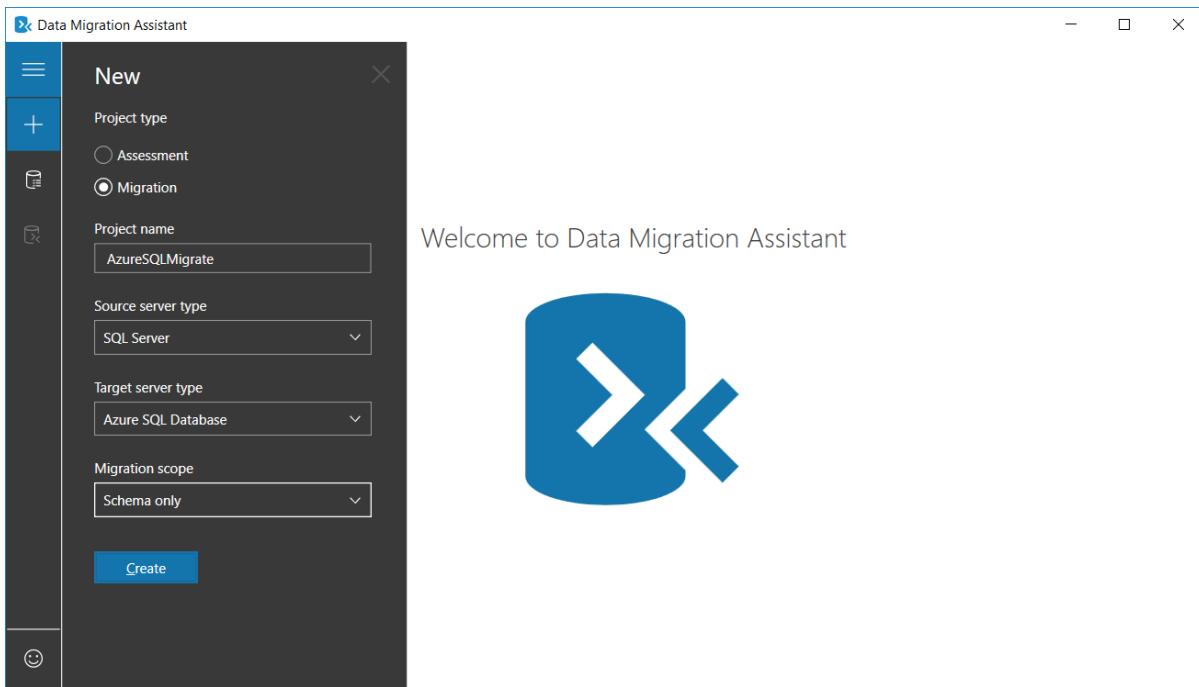
NOTE

Before you create a migration project in DMA, be sure that you have already provisioned an Azure SQL database as mentioned in the prerequisites. For purposes of this tutorial, the name of the Azure SQL Database is assumed to be **AdventureWorks2012**, but you can provide whatever name you wish.

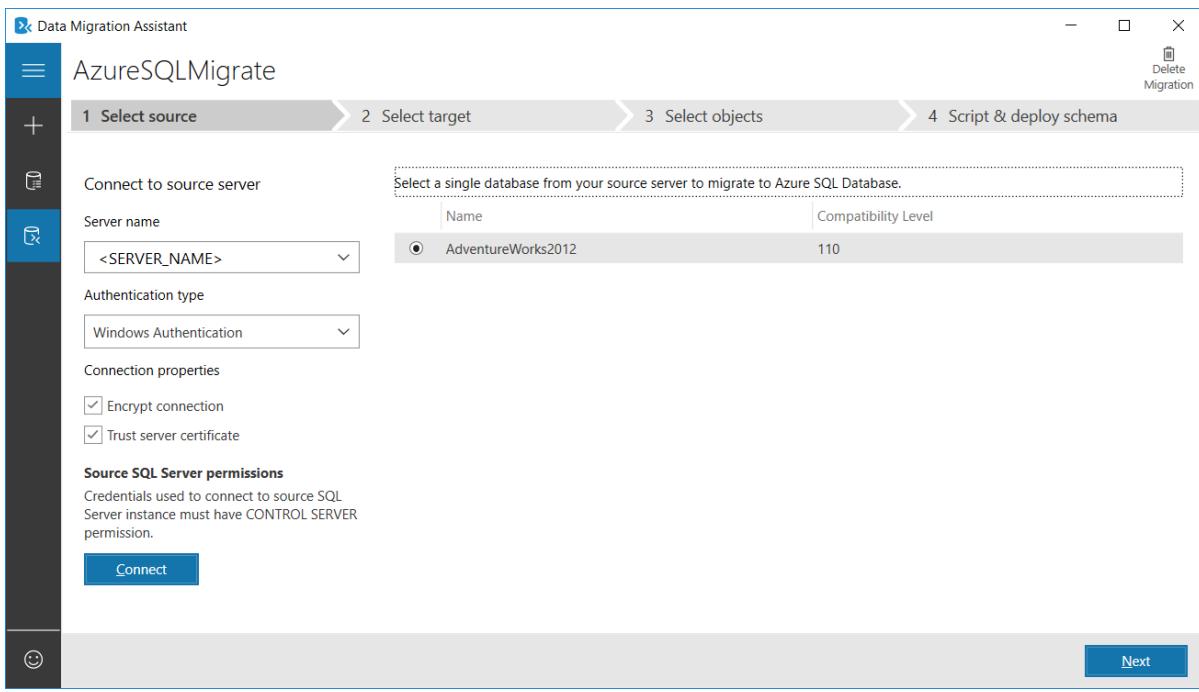
To migrate the **AdventureWorks2012** schema to Azure SQL Database, perform the following steps:

1. In the Data Migration Assistant, select the New (+) icon, and then under **Project type**, select **Migration**.
2. Specify a project name, in the **Source server type** text box, select **SQL Server**, and then in the **Target server type** text box, select **Azure SQL Database**.
3. Under **Migration Scope**, select **Schema only**.

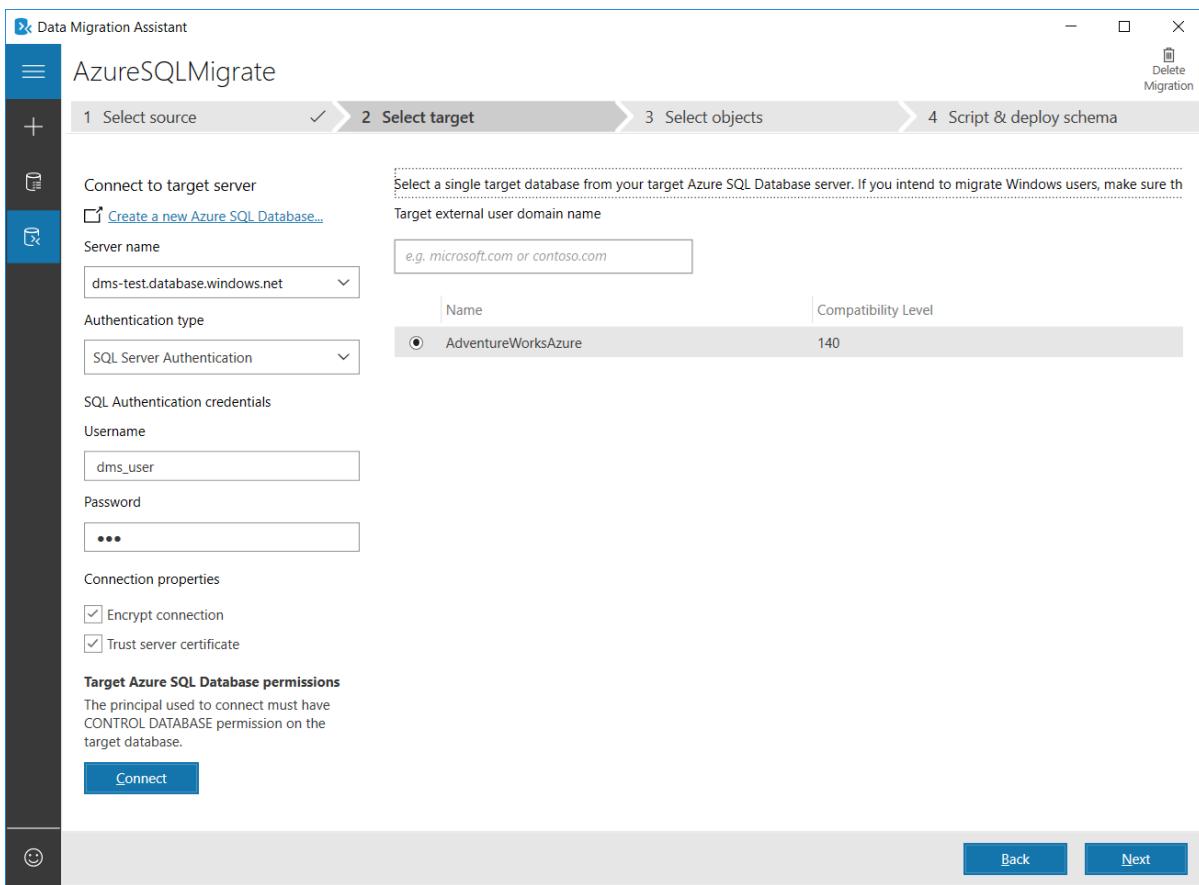
After performing the previous steps, the DMA interface should appear as shown in the following graphic:



4. Select **Create** to create the project.
5. In DMA, specify the source connection details for your SQL Server, select **Connect**, and then select the **AdventureWorks2012** database.

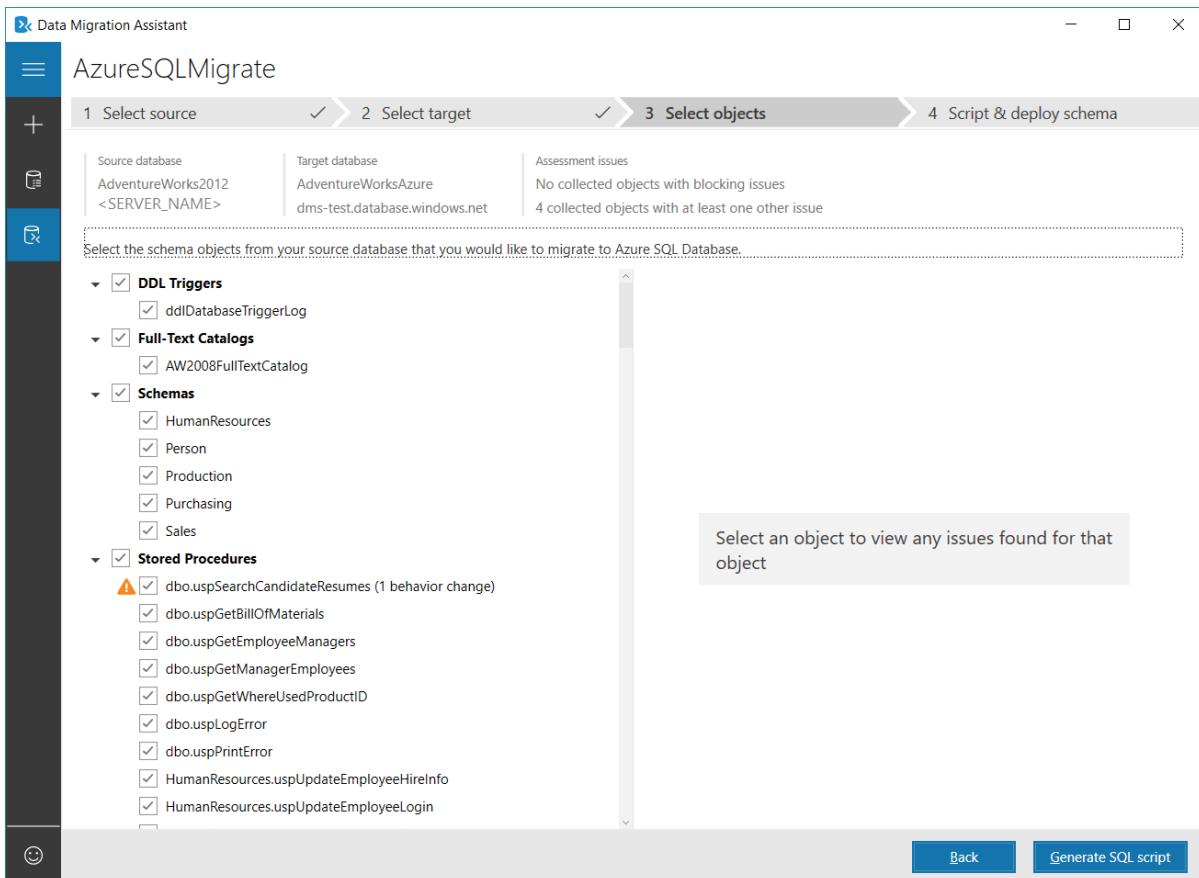


6. Select **Next**, under **Connect to target server**, specify the target connection details for the Azure SQL database, select **Connect**, and then select the **AdventureWorksAzure** database you pre-provisioned in Azure SQL Database.



7. Select **Next** to advance to the **Select objects** screen, on which you can specify the schema objects in the **AdventureWorks2012** database that need to be deployed to Azure SQL Database.

By default, all objects are selected.



8. Select **Generate SQL script** to create the SQL scripts, and then review the scripts for any errors.

This script was generated for the selected schema objects. Review the script, make edits if necessary, and click "Deploy schema" to deploy to Azure SQL Database. Any selected users were not scripted; these will be migrated separately upon clicking "Deploy schema." SQL logins associated with selected users will be recreated with strong, random passwords. You will need to change these passwords and enable them again on the target.

Generated script

```
***** DMA Schema Migration Deployment Script      Script Date: 11/16/2017 12:31:48 PM *****/
-- 4 object(s) with recommendations identified during assessment. Please review these objects before deploying.

***** Object: FullTextCatalog [AW2008FullTextCatalog]  Script Date: 11/16/2017 12:31:46 PM *****/
IF NOT EXISTS (SELECT * FROM sysfulltextcatalogs ftc WHERE ftc.name = N'AW2008FullTextCatalog')
CREATE FULLTEXT CATALOG [AW2008FullTextCatalog]AS DEFAULT

GO
***** Object: Schema [HumanResources]  Script Date: 11/16/2017 12:31:46 PM *****/
IF NOT EXISTS (SELECT * FROM sys.schemas WHERE name = N'HumanResources')
EXEC sys.sp_executesql N'CREATE SCHEMA [HumanResources]'

GO
IF NOT EXISTS (SELECT * FROM sys.fn_listextendedproperty(N'MS_Description', N'SCHEMA',N'HumanResources', NULL,NULL, NULL))
EXEC sys.sp_addextendedproperty @name=N'MS_Description', @value=N'Contains objects related to employees and departments.', @level GO
***** Object: Schema [Person]  Script Date: 11/16/2017 12:31:46 PM *****/
IF NOT EXISTS (SELECT * FROM sys.schemas WHERE name = N'Person')
EXEC sys.sp_executesql N'CREATE SCHEMA [Person]'

GO
IF NOT EXISTS (SELECT * FROM sys.fn_listextendedproperty(N'MS_Description', N'SCHEMA',N'Person', NULL,NULL, NULL))
EXEC sys.sp_addextendedproperty @name=N'MS_Description', @value=N'Contains objects related to employees and departments.', @level GO
```

Back Deploy schema

- Select **Deploy schema** to deploy the schema to Azure SQL Database, and then after the schema is deployed, check the target server for any anomalies.

This script was generated for the selected schema objects. Review the script, make edits if necessary, and click "Deploy schema" to deploy to Azure SQL Database. Any selected users were not scripted; these will be migrated separately upon clicking "Deploy schema." SQL logins associated with selected users will be recreated with strong, random passwords. You will need to change these passwords and enable them again on the target.

Generated script

```
***** DMA Schema Migration Deployment Script      Script Date: 11/16/2017 12:31:48 PM *****/
-- 4 object(s) with recommendations identified during assessment. Please review these objects before deploying.

***** Object: FullTextCatalog [AW2008FullTextCatalog]  Script Date: 11/16/2017 12:31:46 PM *****/
IF NOT EXISTS (SELECT * FROM sysfulltextcatalogs ftc WHERE ftc.name = N'AW2008FullTextCatalog')
CREATE FULLTEXT CATALOG [AW2008FullTextCatalog]AS DEFAULT

GO
***** Object: Schema [HumanResources]  Script Date: 11/16/2017 12:31:46 PM *****/
IF NOT EXISTS (SELECT * FROM sys.schemas WHERE name = N'HumanResources')
EXEC sys.sp_executesql N'CREATE SCHEMA [HumanResources]'

GO
IF NOT EXISTS (SELECT * FROM sys.fn_listextendedproperty(N'MS_Description', N'SCHEMA',N'HumanResources', NULL,NULL, NULL))
EXEC sys.sp_addextendedproperty @name=N'MS_Description', @value=N'Contains objects related to employees and departments.', @level GO
***** Object: Schema [Person]  Script Date: 11/16/2017 12:31:46 PM *****/
IF NOT EXISTS (SELECT * FROM sys.schemas WHERE name = N'Person')
EXEC sys.sp_executesql N'CREATE SCHEMA [Person]'

GO
IF NOT EXISTS (SELECT * FROM sys.fn_listextendedproperty(N'MS_Description', N'SCHEMA',N'Person', NULL,NULL, NULL))
EXEC sys.sp_addextendedproperty @name=N'MS_Description', @value=N'Contains objects related to employees and departments.', @level GO
```

Deployment results (2,236 commands executed...)

	Execution Status	Command Description
1	Success	Executed 2,230 of 2,236: ALTER AUTHORIZATION...
2	Success	Command executed successfully.
3	Success	Executed 2,231 of 2,236: /***** Indexes For Vie...
4	Success	Command executed successfully.
5	Success	Executed 2,232 of 2,236: IF NOT EXISTS (SELEC...
6	Success	Command executed successfully.
7	Success	Executed 2,233 of 2,236: IF NOT EXISTS (SELEC...
8	Success	Command executed successfully.
9	Success	Executed 2,234 of 2,236: SET ARITHABORT ON...
10	Success	Command executed successfully.
11	Success	Executed 2,235 of 2,236: IF NOT EXISTS (SELEC...
12	Success	Command executed successfully.
13	Success	Executed 2,236 of 2,236: IF NOT EXISTS (SELEC...
14	Success	Command executed successfully.

Schema migration completed with errors: Duration: 0h 2m 23s

Back Redeploy schema

Register the Microsoft.DataMigration resource provider

- Sign in to the Azure portal, select **All services**, and then select **Subscriptions**.

All services Filter By category ▾

GENERAL (14) —

- Dashboard
- Management Groups PREVIEW
- Cost Management + Billing PREVIEW
- Help + support
- Tags
- All resources
- Subscriptions
- Reservations
- Service Health
- What's new

COMPUTE (20) —

- Virtual machines
- Virtual machines (classic)
- Container services
- Function Apps

2. Select the subscription in which you want to create the instance of the Azure Database Migration Service, and then select **Resource providers**.

Home > Subscriptions > <subscription>

Subscriptions Microsoft

Add

My role: Owner Status: 3 selected

Apply

Search to filter items...

SUBSCRIPTI... ↑↓ SUBSCRIPTION ID ↑↓

<subscription> <subscription ID>

Overview

Access control (IAM)

Diagnose and solve problems

COST MANAGEMENT + BILLING

Partner information

SETTINGS

Programmatic deployment

Resource groups

Resources

Usage + quotas

Policies

Management certificates

My permissions

Resource providers

Properties

Resource locks

SUPPORT + TROUBLESHOOTING

New support request

3. Search for migration, and then to the right of **Microsoft.DataMigration**, select **Register**.

The screenshot shows the Azure portal interface. On the left, the navigation menu includes 'Create a resource', 'All services', 'Dashboard', 'Resource groups', 'App Services', 'Function Apps', 'SQL databases', 'Azure Cosmos DB', 'Virtual machines', 'Load balancers', 'Storage accounts', 'Virtual networks', 'Azure Active Directory', 'Monitor', and 'Advisor'. The main content area is titled '<Subscription>' and shows a list of registered providers. One provider, 'Microsoft.DataMigration', is listed with the status 'NotRegistered'. A red box highlights the 'Register' button next to it.

Create an instance

1. In the Azure portal, select + **Create a resource**, search for Azure Database Migration Service, and then select **Azure Database Migration Service** from the drop-down list.

The screenshot shows the Azure portal's 'New' blade. The search bar at the top contains the text 'Azure Database Migration Service', which is highlighted with a red box. Below the search bar, there is a list of service categories: Networking, Storage, Web + Mobile, Containers, Databases, Data + Analytics, AI + Cognitive Services, Internet of Things, Enterprise Integration, Security + Identity, Developer tools, Monitoring + Management, Add-ons, and Blockchain. Under the 'Databases' category, 'Azure Database Migration Service' is listed with a 'PREVIEW' badge and a 'Quickstart tutorial' link. The rest of the blade shows other service options like DevOps Project, Web App, SQL Database, Cosmos DB, Storage Account, and Serverless Function App, along with their respective quickstart tutorials.

2. On the **Azure Database Migration Service** screen, select **Create**.

The screenshot shows the Azure Database Migration Service page. On the left, there's a sidebar with various service icons: Create a resource, All services, Favorites, Dashboard, Subscriptions, All resources, Resource groups, App Services, Function Apps, SQL databases, Azure Cosmos DB, Virtual machines, Load balancers, Storage accounts, Virtual networks, Azure Active Directory, Monitor, Advisor, and Security Center. The 'Create a resource' icon is at the top of the list. The main content area has a title 'Azure Database Migration Service' with a Microsoft logo. It contains text about the service's purpose and migration steps, followed by a 'Save for later' button. Below that is a table with two rows: 'PUBLISHER' (Microsoft) and 'USEFUL LINKS' (Documentation, Privacy Statement). At the bottom is a large blue 'Create' button.

3. On the **Create Migration Service** screen, specify a name for the service, the subscription, and a new or existing resource group.
4. Select the location in which you want to create the instance of the Azure Database Migration Service.
5. Select an existing virtual network or create a new one.

The virtual network provides Azure Database Migration Service with access to the source SQL Server and the target Azure SQL Database instance.

For more information about how to create a virtual network in the Azure portal, see the article [Create a virtual network using the Azure portal](#).

6. Select a pricing tier; for this online migration, be sure to select the Premium pricing tier.

For more information on costs and pricing tiers, see the [pricing page](#).

Home > New > Azure Database Migration Service

Create Migration Service

Service Name ✓
DMSTest

* Subscription
<subscription ID>

* Select a resource group ✓
ADFHive

Create new

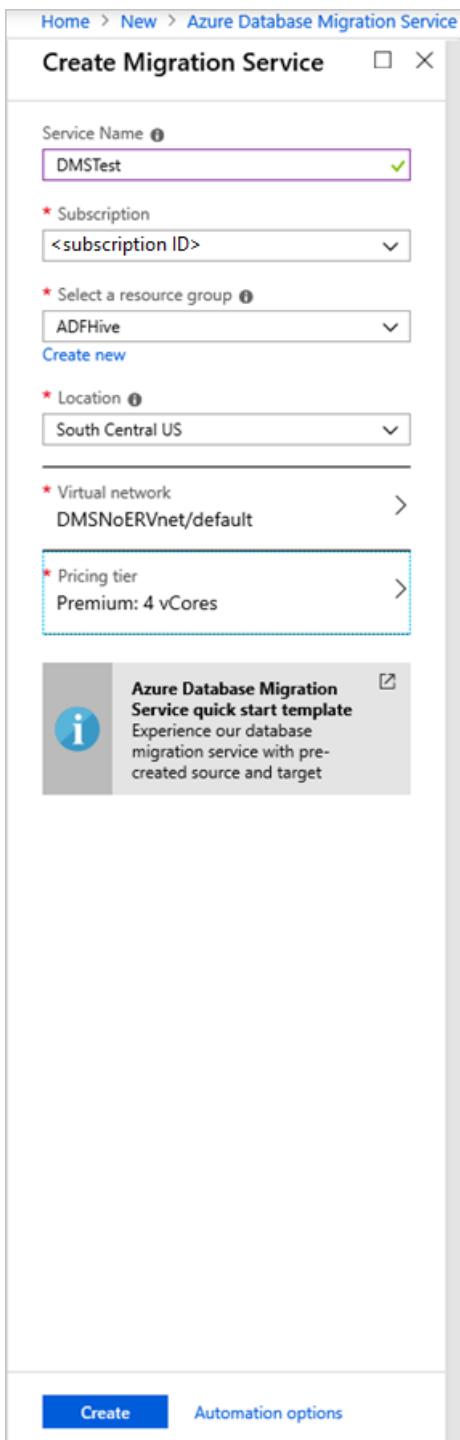
* Location ✓
South Central US

* Virtual network >
DMSNoERVnet/default

* Pricing tier >
Premium: 4 vCores

i Azure Database Migration Service quick start template
Experience our database migration service with pre-created source and target

Create [Automation options](#)



7. Select **Create** to create the service.

Create a migration project

After the service is created, locate it within the Azure portal, open it, and then create a new migration project.

1. In the Azure portal, select **All services**, search for Azure Database Migration Service, and then select **Azure Database Migration Services**.

The screenshot shows the Azure portal interface. On the left, there's a sidebar with various service icons and names. At the top, a search bar contains the text "Azure Database Migration Service". Below the search bar, the "Azure Database Migration Services" service is listed with a star icon for favoriting. A message at the bottom right of the main area says "Help improve the service menu!".

2. On the **Azure Database Migration Services** screen, search for the name of the Azure Database Migration Service instance that you created, and then select the instance.

The screenshot shows the "SQL_Migrate" instance details page within the Azure Database Migration Services. The left sidebar shows the "All services" list. The main pane displays the "Essentials" section for the "SQL_Migrate" instance, including its resource group, network settings, subscription name, and pricing tier. Below this, the "Migration Projects" section is shown, which is currently empty.

3. Select + **New Migration Project**.
4. On the **New migration project** screen, specify a name for the project, in the **Source server type** text box, select **AWS RDS for SQL Server**, in the **Target server type** text box, select **Azure SQL Database**.

NOTE

For Target server type, select **Azure SQL Database** for migrating to both an Azure SQL Database singleton database and as well as to an Azure SQL Database managed instance.

5. In the **Choose type of activity** section, select **Online data migration**.

IMPORTANT

Be sure to select **Online data migration**; offline migrations are not supported for this scenario.

Home > DemoDMSOnlineWestUS2 > New migration project > Type of activity

New migration project

Type of activity

Project name: MigratefromRDStoSQLDB

* Source server type: AWS RDS for SQL Server

* Target server type: Azure SQL Database

* Choose type of activity: Online data migration

Choose type of activity: Online data migration

Use this option to migrate databases that must be accessible and continuously updated during migration.

To continuously replicate data changes from your source to your target, please implement the steps below on your source server. These steps can be implemented anytime between the moment you create a project and the moment you start a migration activity. After your migration is complete, you will reverse those preparation steps.

- The source SQL Server version must be SQL Server 2005 or above.
- Database(s) must be in either Bulk-logged or Full recovery mode.
- Make sure to take full database backups.
- If tables do not have primary key, enable CDC on the database and specific tables.
- The distributor role must be configured for the source database.

To successfully use Database Migration Service (DMS) to migrate data, you need to:

- Create the target Azure SQL Database.
- Use DMA to assess your on-premises SQL Server database(s) for feature parity and compatibility issues.
- Apply fixes and deploy the database schema to your target Azure SQL database using Data Migration Assistant (DMA).

Install Database Migration Assistant

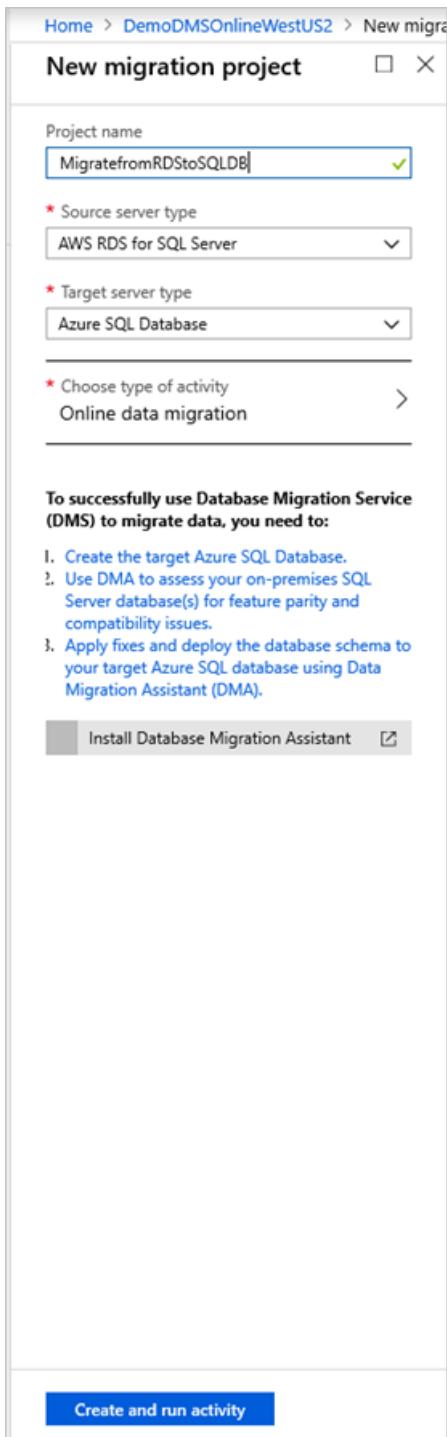
Create and run activity

Save

NOTE

Alternately, you can choose **Create project only** to create the migration project now and execute the migration later.

6. Select **Save**.
7. Select **Create and run activity** to create the project and run the migration activity.



Specify source details

1. On the **Migration source detail** screen, specify the connection details for the source SQL Server instance. Make sure to use a Fully Qualified Domain Name (FQDN) for the source SQL Server instance name.
2. If you haven't installed a trusted certificate on your source server, select the **Trust server certificate** check box.

When a trusted certificate isn't installed, SQL Server generates a self-signed certificate when the instance is started. This certificate is used to encrypt the credentials for client connections.

Caution

SSL connections that are encrypted using a self-signed certificate do not provide strong security. They are

susceptible to man-in-the-middle attacks. You should not rely on SSL using self-signed certificates in a production environment or on servers that are connected to the internet.

The screenshot shows the 'Migration Wizard' interface with the title 'MigrateToSQLDB'. The left sidebar lists five steps: 1. Select source (completed), 2. Select target, 3. Map to target databases, 4. Configure migration settings, and 5. Summary. The 'Migration source detail' screen is open on the right, containing fields for Source SQL Server instance name, Authentication type (SQL Authentication selected), User Name, Password, and Connection properties (Encrypt connection and Trust server certificate checked). A 'Save' button is at the bottom.

Migration Wizard

MigrateToSQLDB

Migration source detail

1 Select source ✓

2 Select target >

3 Map to target databases >

4 Configure migration settings >

5 Summary >

* Source SQL Server instance name

Authentication type

* User Name

Password

Connection properties

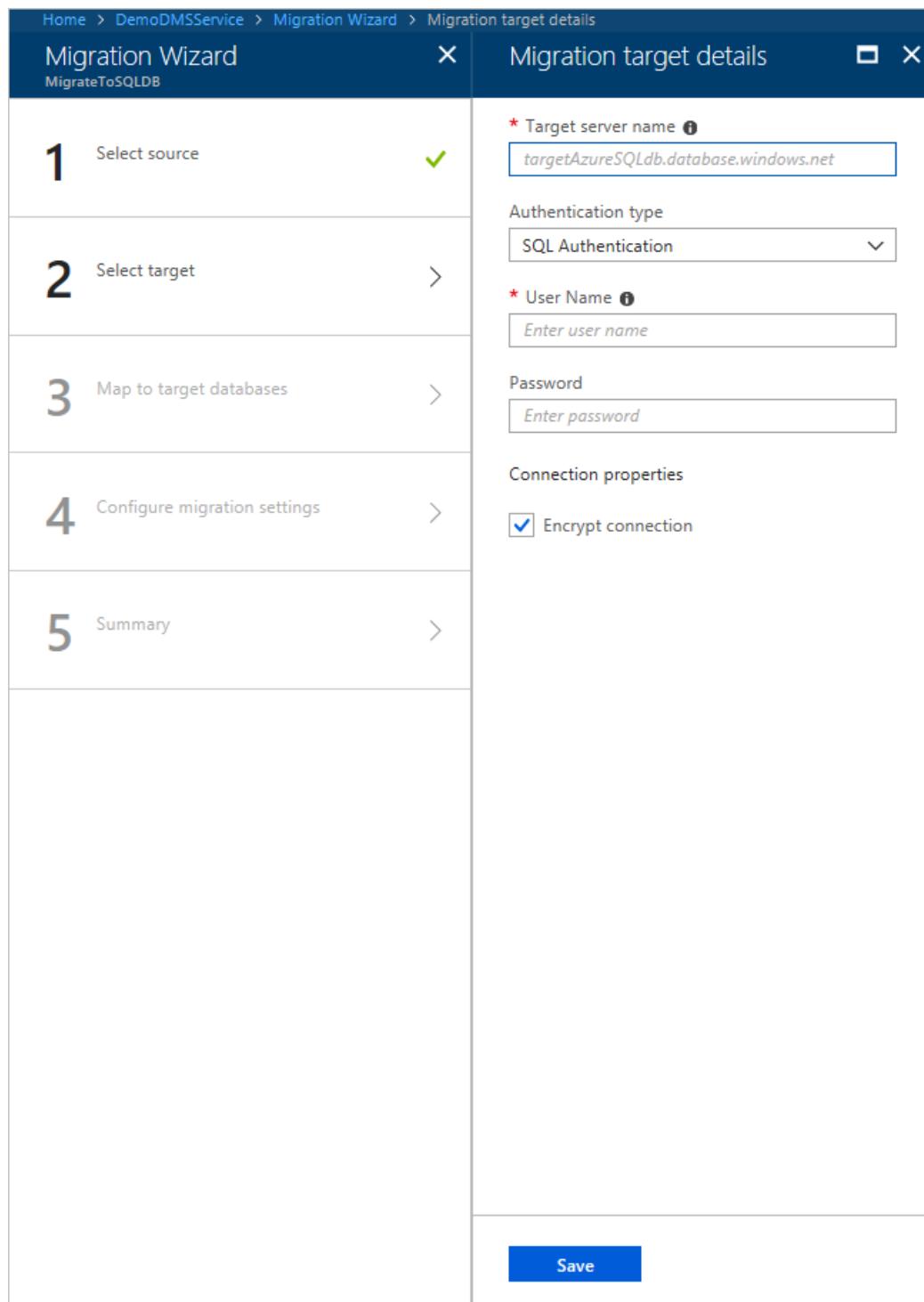
Encrypt connection

Trust server certificate

Save

Specify target details

1. Select **Save**, and then on the **Migration target details** screen, specify the connection details for the target Azure SQL Database server, which is the pre-provisioned Azure SQL Database to which the **AdventureWorks2012** schema was deployed by using the DMA.



2. Select **Save**, and then on the **Map to target databases** screen, map the source and the target database for migration.

If the target database contains the same database name as the source database, the Azure Database Migration Service selects the target database by default.

Home > DemoDMSOnlineWestUS2 > Migration Wizard > Map to target databases

Migration Wizard X

Migrate from DStoSQLDB

1	Select source	✓
2	Select target	✓
3	Map to target databases	>
4	Configure migration settings	>
5	Summary	>

Map to target databases X

5 item(s) All

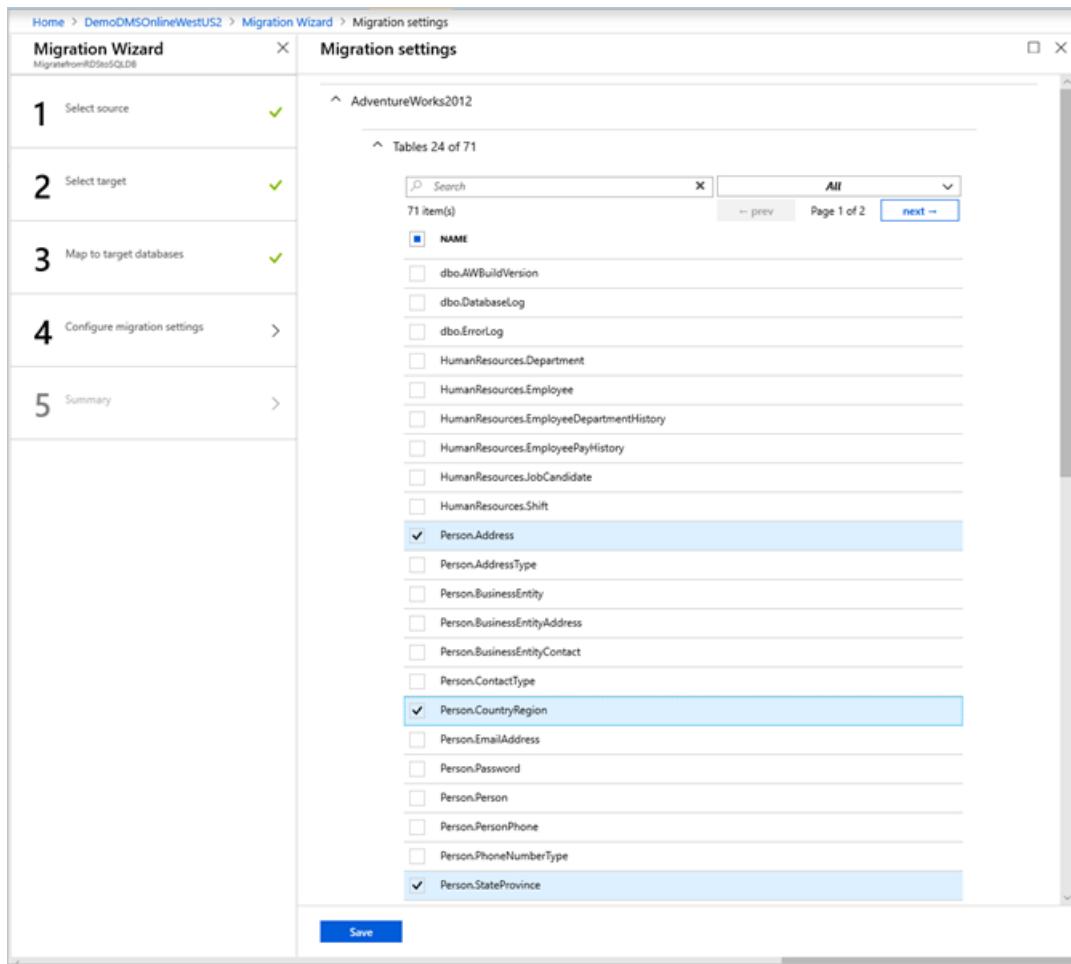
Search prev next Page 1 of 1

SOURCE DATABASE	SIZE	TARGET DATABASE
<input checked="" type="checkbox"/> AdventureWorks2012	232.06 MB	AdventureWorks2012
<input type="checkbox"/> arpaovicTestFk	17.00 MB	
<input type="checkbox"/> HRMinDowntime	147.31 MB	
<input type="checkbox"/> nesintest	9.00 MB	
<input type="checkbox"/> rdsadmin	144.00 MB	

Save

3. Select **Save**, on the **Select tables** screen, expand the table listing, and then review the list of affected fields.

The Azure Database Migration Service auto selects all the empty source tables that exist on the target Azure SQL Database instance. If you want to remigrate tables that already include data, you need to explicitly select the tables on this screen.



4. Select **Save**, after setting the following **Advanced online migration settings**.

SETTING	DESCRIPTION
Maximum number of tables to load in parallel	Specifies the number of tables that DMS executes in parallel during the migration. The default value is 5, but it can be set to an optimal value to meet specific migration needs based on any POC migrations.
When source table is truncated	Specifies whether DMS truncates the target table during migration. This setting can be helpful if one or more tables are truncated as part of the migration process.
Configure settings for large objects (LOB) data	Specifies whether DMS migrates unlimited LOB data or limits the LOB data migrated to a specific size. When there's a limit on the LOB data migrated, any LOB data beyond that limit is truncated. For production migrations, it's recommended to select Allow unlimited LOB size to prevent data loss. When specifying to allow unlimited LOB size, select the Migrate LOB data in a single block when the LOB size is less than (KB) specified check box to improve performance.

Home > DemoDMSOnlineWestUS2 > Migration Wizard > Migration settings

Migration Wizard

Migrate from RDG to SQL DB

Migration settings

1 Select source ✓

2 Select target ✓

3 Map to target databases ✓

4 Configure migration settings >

5 Summary >

AdventureWorks2012

Tables 3 of 71

Advanced online migration settings

Maximum number of tables to load in parallel: 5

When source table is truncated: Do not TRUNCATE target table

Configure settings for large objects (LOB) data:

Allow unlimited LOB size

Limit LOB size

Block size (KB): 64

Migrate LOB data in a single block when LOB size is less than (KB) below: 0

Save

The screenshot shows the 'Migration Wizard' interface with the 'Migration settings' step selected. The left sidebar lists five steps: 1. Select source (done), 2. Select target (done), 3. Map to target databases (done), 4. Configure migration settings (in progress), and 5. Summary. Step 4 is expanded, showing configuration for the 'AdventureWorks2012' database. It includes settings for parallel loading (5 tables), truncating target tables, configuring large object (LOB) data (allowing unlimited size or limiting to 64 KB), and migrating LOB data in single blocks if smaller than 0 KB. A 'Save' button is at the bottom.

5. Select **Save**, on the **Migration summary** screen, in the **Activity name** text box, specify a name for the migration activity, and then review the summary to ensure that the source and target details match what you previously specified.

Migration Wizard		X	Migration summary	□ X
MigrateToSQLDB				
1	Select source	✓	Please consider upgrading your Azure SQL Database to Premium tier (for example P11 or P15 for singleton databases) if DTU purchase model is used or Business critical tier (for example 8 or 16 vCORE) in the case of vCORE model. You can revert to a lower tier once migration is complete. Upgrade Your Database Service Tiers	
2	Select target	✓		
3	Map to target databases	✓	Migration project name MigrateToSQLDB	
4	Configure migration settings	✓	Activity name <input type="text" value="Mymigration"/> ✓ Source server name <source server name> Source server version SQL Server 2012 11.0.7462.6 Target server name <target server name> Target server version Azure SQL Database 12.0.2000.8 Database(s) to migrate 1 of 6 Type of activity Online data migration	
5	Summary	>		
Run migration				

Run the migration

- Select **Run migration**.

The migration activity window appears, and the **Status** of the activity is **Initializing**.

Home > <service instance> > Mymigration

Mymigration

Refresh Stop migration Delete activity

Source server	Source version	Source databases			
Target server	Target version	Type of activity			
Activity status		Duration			
10.105.129.164	SQL Server 2012	1			
dmazuresqldbserver.database.windows.net	Azure SQL Database	Online			
Running		---			
Database Name	Status	Migration Details	Duration	Estimated Application Downtime	Finish Date
AdventureWorks2012	Initializing	Initializing the migration pipeline	---	---	---

Monitor the migration

1. On the migration activity screen, select **Refresh** to update the display until the **Status** of the migration shows as **Running**.
2. Click on a specific database to get to the migration status for **Full data load** and **Incremental data sync** operations.

Home > <service instance> > MigrateToSQLDB > MyMigration1 > AdventureWorks2012

AdventureWorks2012

⟳ Refresh ▶ Start Cutover

Source database name	Full load completed	Incremental updates	Pending changes
AdventureWorks2012	46	0	0
Target database name	Full load queued	Incremental inserts	Applied changes
AdventureWorksAzure	0	0	0
Database status	Full load loading	Incremental deletes	Tables in error state ⓘ
Running	1	0	0
Migration details	Full load failed		
Full data load in progress	0		

Full load **Incremental data sync**

47 item(s) ← prev Page 1 of 5 next →

TABLE NAME	STATUS	COMPLETED	ROWS	VOLUME (MB)	DURATION
HumanResources.Department	Completed	7/23/2018 6:33:55 PM	16	1	00:00:03
HumanResources.Employee	Completed	7/23/2018 6:33:56 PM	290	1	00:00:04
HumanResources.EmployeeDe...	Completed	7/23/2018 6:33:56 PM	296	1	00:00:04
HumanResources.EmployeePa...	Completed	7/23/2018 6:34:00 PM	316	1	00:00:05
HumanResources.JobCandidate	Completed	7/23/2018 6:34:01 PM	13	1	00:00:05
HumanResources.Shift	Completed	7/23/2018 6:34:01 PM	3	1	00:00:05
Person.Address	Completed	7/23/2018 6:34:09 PM	19614	13.9	00:00:12
Person.AddressType	Completed	7/23/2018 6:34:05 PM	6	1	00:00:06
Person.BusinessEntity	Completed	7/23/2018 6:34:07 PM	20777	8.4	00:00:07
Person.BusinessEntityAddress	Completed	7/23/2018 6:34:07 PM	19614	9.9	00:00:06

Perform migration cutover

After the initial Full load is completed, the databases are marked **Ready to cutover**.

- When you're ready to complete the database migration, select **Start Cutover**.

Home > <service instance> > MigrateToSQLDB > MyMigration1 > AdventureWorks2012 > Complete cutover

AdventureWorks2012

⟳ Refresh ▶ Start Cutover

Source database name	Full load completed	Incremental updates	Pending changes
AdventureWorks2012	47	3	0
Target database name	Full load queued	Incremental inserts	Applied changes
AdventureWorksAzure	0	0	3
Database status	Full load loading	Incremental deletes	Tables in error state ⓘ
Running	0	0	0
Migration details	Full load failed		
Ready to cutover	0		

Full load **Incremental data sync**

47 item(s) ← prev Page 4 of 5 next →

TABLE NAME	STATUS	COMPLETED	ROWS	VOLUME (MB)	DURATION
Sales.PersonCreditCard	Completed	7/23/2018 6:34:39 PM	19118	7.4	00:00:06
Sales.SalesOrderDetail	Completed	7/23/2018 6:48:12 PM	121317	86.5	00:13:35
Sales.SalesOrderHeader	Completed	7/23/2018 6:34:53 PM	31465	37.7	00:00:15
Sales.SalesOrderHeaderSalesR...	Completed	7/23/2018 6:34:50 PM	27647	10.8	00:00:11
Sales.SalesPerson	Completed	7/23/2018 6:34:50 PM	17	1	00:00:11
Sales.SalesPersonQuotaHistory	Completed	7/23/2018 6:34:51 PM	163	1	00:00:10
Sales.SalesReason	Completed	7/23/2018 6:34:54 PM	10	1	00:00:04
Sales.SalesTaxRate	Completed	7/23/2018 6:34:55 PM	29	1	00:00:04
Sales.SalesTerritory	Completed	7/23/2018 6:35:39 PM	10	1	00:00:47
Sales.SalesTerritoryHistory	Completed	7/23/2018 6:34:59 PM	17	1	00:00:05

When you are ready to do the migration cutover, perform the following steps to complete the database migration. Please note that the database is ready for cutover only after the full data load is completed.

- Stop all the incoming transactions coming to the source database.
- Wait until all the pending transactions have been applied to the target database. At that time the pending changes counter will set to 0.

Pending changes 0

Confirm Apply

3. Reconnect your applications to the new Azure target database.

- Make sure to stop all the incoming transactions to the source database; wait until the **Pending changes** counter shows **0**.

3. Select **Confirm**, and the select **Apply**.

4. When the database migration status shows **Completed**, connect your applications to the new target Azure SQL Database.

The screenshot shows the 'Complete cutover' page for the 'AdventureWorks2012' service instance. It displays migration statistics and a step-by-step guide for performing a database migration.

Migration Statistics:

Source database name	Full load completed	Incremental updates	Pending changes
AdventureWorks2012	47	3	0
Target database name	Full load queued	Incremental inserts	Applied changes
AdventureWorksAzure	0	0	3
Database status	Full load loading	Incremental deletes	Tables in error state ⓘ
Complete	0	0	0
Migration details	Full load failed		
All changes applied	0		

Migration Details Table:

TABLE NAME	STATUS	COMPLETED	ROWS	VOLUME (MB)	DURATION
Sales.PersonCreditCard	Completed	7/23/2018 6:34:39 PM	19118	7.4	00:00:06
Sales.SalesOrderDetail	Completed	7/23/2018 6:48:12 PM	121317	86.5	00:13:35
Sales.SalesOrderHeader	Completed	7/23/2018 6:34:53 PM	31465	37.7	00:00:15
Sales.SalesOrderHeaderSalesR...	Completed	7/23/2018 6:34:50 PM	27647	10.8	00:00:11
Sales.SalesPerson	Completed	7/23/2018 6:34:50 PM	17	1	00:00:11
Sales.SalesPersonQuotaHistory	Completed	7/23/2018 6:34:51 PM	163	1	00:00:10
Sales.SalesReason	Completed	7/23/2018 6:34:54 PM	10	1	00:00:04
Sales.SalesTaxRate	Completed	7/23/2018 6:34:55 PM	29	1	00:00:04
Sales.SalesTerritory	Completed	7/23/2018 6:35:39 PM	10	1	00:00:47
Sales.SalesTerritoryHistory	Completed	7/23/2018 6:34:59 PM	17	1	00:00:05

Migration Guide:

1. Stop all the incoming transactions coming to the source database.
2. Wait until all the pending transactions have been applied to the target database. At that time the pending changes counter will set to 0.
3. Reconnect your applications to the new Azure target database.

Buttons:

- Pending changes: 0
- Confirm
- Apply

Next steps

- For information about known issues and limitations when performing online migrations to Azure SQL Database, see the article [Known issues and workarounds with Azure SQL Database online migrations](#).
- For information about the Azure Database Migration Service, see the article [What is the Azure Database Migration Service?](#).
- For information about Azure SQL Database, see the article [What is the Azure SQL Database service?](#).
- For information about Azure SQL Database managed instances, see the page [Azure SQL Database Managed Instance](#).

Tutorial: Migrate MySQL to Azure Database for MySQL online using DMS

1/8/2020 • 8 minutes to read • [Edit Online](#)

You can use Azure Database Migration Service to migrate the databases from an on-premises MySQL instance to [Azure Database for MySQL](#) with minimal downtime. In other words, migration can be achieved with minimum downtime to the application. In this tutorial, you migrate the **Employees** sample database from an on-premises instance of MySQL 5.7 to Azure Database for MySQL by using an online migration activity in Azure Database Migration Service.

In this tutorial, you learn how to:

- Migrate the sample schema using mysqldump utility.
- Create an instance of Azure Database Migration Service.
- Create a migration project by using Azure Database Migration Service.
- Run the migration.
- Monitor the migration.

NOTE

Using Azure Database Migration Service to perform an online migration requires creating an instance based on the Premium pricing tier.

IMPORTANT

For an optimal migration experience, Microsoft recommends creating an instance of Azure Database Migration Service in the same Azure region as the target database. Moving data across regions or geographies can slow down the migration process and introduce errors.

Prerequisites

To complete this tutorial, you need to:

- Download and install [MySQL community edition](#) 5.6 or 5.7. The on-premises MySQL version must match with Azure Database for MySQL version. For example, MySQL 5.6 can only migrate to Azure Database for MySQL 5.6 and not upgraded to 5.7.
- [Create an instance in Azure Database for MySQL](#). Refer to the article [Use MySQL Workbench to connect and query data](#) for details about how to connect and create a database using the Azure portal.
- Create a Microsoft Azure Virtual Network for Azure Database Migration Service by using Azure Resource Manager deployment model, which provides site-to-site connectivity to your on-premises source servers by using either [ExpressRoute](#) or [VPN](#). For more information about creating a virtual network, see the [Virtual Network Documentation](#), and especially the quickstart articles with step-by-step details.

NOTE

During virtual networkNet setup, if you use ExpressRoute with network peering to Microsoft, add the following service [endpoints](#) to the subnet in which the service will be provisioned:

- Target database endpoint (for example, SQL endpoint, Cosmos DB endpoint, and so on)
- Storage endpoint
- Service bus endpoint

This configuration is necessary because Azure Database Migration Service lacks internet connectivity.

- Ensure that your virtual network Network Security Group rules don't block the following inbound communication ports to Azure Database Migration Service: 443, 53, 9354, 445, 12000. For more detail on virtual network NSG traffic filtering, see the article [Filter network traffic with network security groups](#).
- Configure your [Windows Firewall for database engine access](#).
- Open your Windows firewall to allow Azure Database Migration Service to access the source MySQL Server, which by default is TCP port 3306.
- When using a firewall appliance in front of your source database(s), you may need to add firewall rules to allow Azure Database Migration Service to access the source database(s) for migration.
- Create a server-level [firewall rule](#) for Azure Database for MySQL to allow Azure Database Migration Service access to the target databases. Provide the subnet range of the virtual network used for Azure Database Migration Service.
- The source MySQL must be on supported MySQL community edition. To determine the version of MySQL instance, in the MySQL utility or MySQL Workbench, run the following command:

```
SELECT @@version;
```

- Azure Database for MySQL supports only InnoDB tables. To convert MyISAM tables to InnoDB, see the article [Converting Tables from MyISAM to InnoDB](#)
- Enable binary logging in the my.ini (Windows) or my.cnf (Unix) file in source database by using the following configuration:
 - **server_id** = 1 or greater (relevant only for MySQL 5.6)
 - **log-bin** = <path> (relevant only for MySQL 5.6) For example: log-bin = E:\MySQL_logs\BinLog
 - **binlog_format** = row
 - **Expire_logs_days** = 5 (it's recommended to not use zero; relevant only for MySQL 5.6)
 - **Binlog_row_image** = full (relevant only for MySQL 5.6)
 - **log_slave_updates** = 1
- The user must have the ReplicationAdmin role with the following privileges:
 - **REPLICATION CLIENT** - Required for Change Processing tasks only. In other words, Full Load only tasks don't require this privilege.
 - **REPLICATION REPLICA** - Required for Change Processing tasks only. In other words, Full Load only tasks don't require this privilege.
 - **SUPER** - Only required in versions earlier than MySQL 5.6.6.

Migrate the sample schema

To complete all the database objects like table schemas, indexes and stored procedures, we need to extract schema

from the source database and apply to the database. To extract schema, you can use mysqldump with the `--no-data` parameter.

Assuming you have MySQL **Employees** sample database in the on-premises system, the command to do schema migration using mysqldump is:

```
mysqldump -h [servername] -u [username] -p[password] --databases [db name] --no-data > [schema file path]
```

For example:

```
mysqldump -h 10.10.123.123 -u root -p --databases employees --no-data > d:\employees.sql
```

To import schema to Azure Database for MySQL target, run the following command:

```
mysql.exe -h [servername] -u [username] -p[password] [database]< [schema file path]
```

For example:

```
mysql.exe -h shausample.mysql.database.azure.com -u dms@shausample -p employees < d:\employees.sql
```

If you have foreign keys in your schema, the initial load and continuous sync of the migration will fail. Execute the following script in MySQL Workbench to extract the drop foreign key script and add foreign key script.

```
SET group_concat_max_len = 8192;
SELECT SchemaName, GROUP_CONCAT(DropQuery SEPARATOR ';\n') as DropQuery, GROUP_CONCAT(AddQuery SEPARATOR
';\n') as AddQuery
FROM
(SELECT
KCU.REFERENCED_TABLE_SCHEMA as SchemaName,
KCU.TABLE_NAME,
KCU.COLUMN_NAME,
CONCAT('ALTER TABLE ', KCU.TABLE_NAME, ' DROP FOREIGN KEY ', KCU.CONSTRAINT_NAME) AS DropQuery,
CONCAT('ALTER TABLE ', KCU.TABLE_NAME, ' ADD CONSTRAINT ', KCU.CONSTRAINT_NAME, ' FOREIGN KEY (`',
KCU.COLUMN_NAME, `)` REFERENCES `', KCU.REFERENCED_TABLE_NAME, '`(`', KCU.REFERENCED_COLUMN_NAME, `)` ON
UPDATE ', RC.UPDATE_RULE, ' ON DELETE ', RC.DELETE_RULE) AS AddQuery
FROM INFORMATION_SCHEMA.KEY_COLUMN_USAGE KCU, information_schema.REFERENTIAL_CONSTRAINTS RC
WHERE
KCU.CONSTRAINT_NAME = RC.CONSTRAINT_NAME
AND KCU.REFERENCED_TABLE_SCHEMA = RC.UNIQUE_CONSTRAINT_SCHEMA
AND KCU.REFERENCED_TABLE_SCHEMA = 'SchemaName') Queries
GROUP BY SchemaName;
```

Run the drop foreign key (which is the second column) in the query result to drop foreign key.

IMPORTANT

If importing data using a backup, remove the CREATE DEFINER commands manually or by using the `--skip-definer` command when performing a mysqldump. DEFINER requires super privileges to create and is restricted in Azure Database for MySQL.

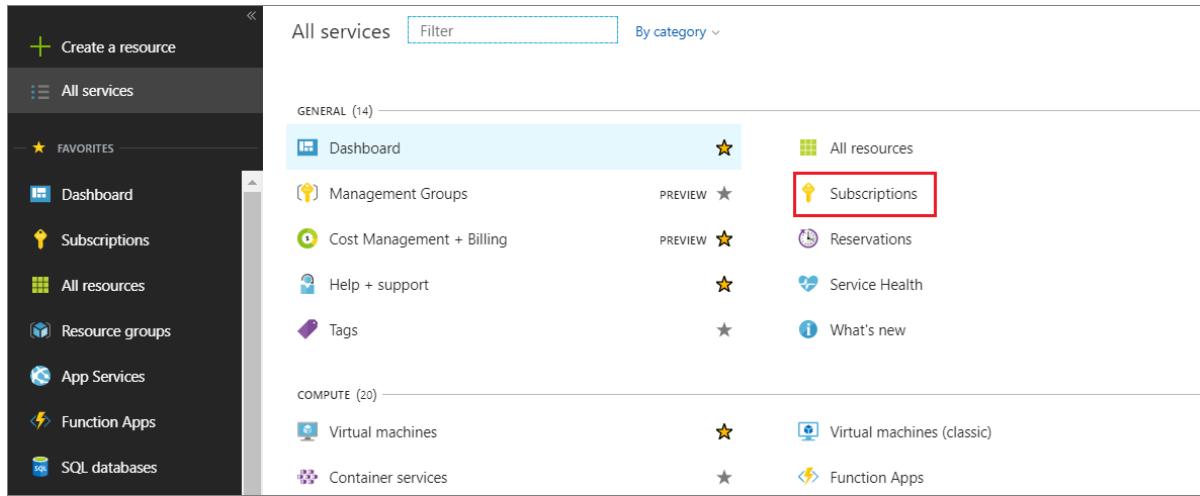
If you have a trigger in the data (insert or update trigger), it will enforce data integrity in the target ahead of the replicated data from the source. The recommendation is to disable triggers in all the tables at the target during migration, and then enable the triggers after migration is done.

To disable triggers in the target database, use the following command:

```
SELECT Concat('DROP TRIGGER ', Trigger_Name, ';') FROM information_schema.TRIGGERS WHERE TRIGGER_SCHEMA = 'your_schema';
```

Register the Microsoft.DataMigration resource provider

1. Sign in to the Azure portal, select **All services**, and then select **Subscriptions**.



The screenshot shows the Azure portal's 'All services' blade. On the left is a dark sidebar with various service icons and names: 'Create a resource', 'All services', 'Favorites' (Dashboard, Subscriptions, All resources, Resource groups, App Services, Function Apps, SQL databases), and 'GENERAL (14)' and 'COMPUTE (20)' sections. The 'Subscriptions' item is highlighted with a red box. In the main area, there are several service cards: 'Dashboard' (selected), 'Management Groups', 'Cost Management + Billing', 'Help + support', 'Tags', 'All resources', 'Subscriptions' (highlighted), 'Reservations', 'Service Health', 'Virtual machines', 'Container services', 'Virtual machines (classic)', and 'Function Apps'. Each card includes a star icon indicating it's a preview.

2. Select the subscription in which you want to create the instance of Azure Database Migration Service, and then select **Resource providers**.

The screenshot shows the Azure portal's Subscriptions page. On the left, there's a navigation bar with various service icons. The main area displays a table with columns for Subscription ID and Status. A search bar at the top allows filtering. On the right, a sidebar lists several options: Overview, Access control (IAM), Diagnose and solve problems, COST MANAGEMENT + BILLING, Partner information, SETTINGS, Programmatic deployment, Resource groups, Resources, Usage + quotas, Policies, Management certificates, My permissions, and Resource providers. The 'Resource providers' link is specifically highlighted with a red box.

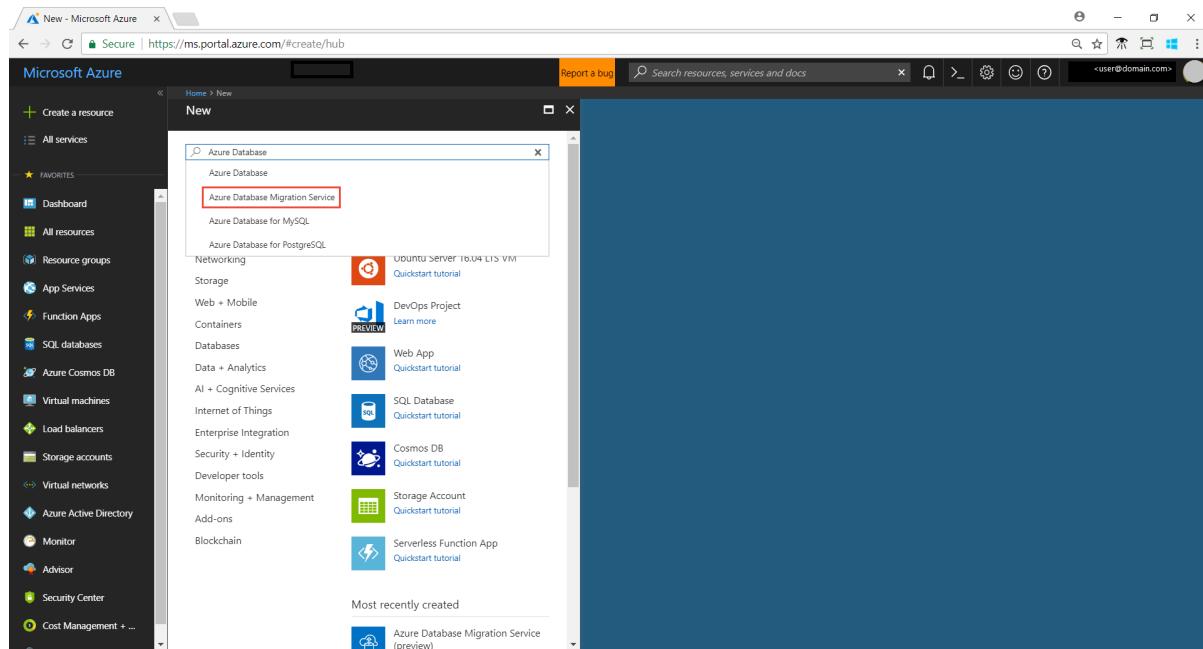
3. Search for migration, and then to the right of **Microsoft.DataMigration**, select **Register**.

This screenshot shows the 'Resource providers' page within the Azure portal. It features a search bar at the top and a table below it. The table has columns for PROVIDER, STATUS, and a Register button. One row in the table is highlighted with a yellow background, showing 'Microsoft.DataMigration' under PROVIDER and 'NotRegistered' under STATUS. The 'Register' button is also highlighted with a red box.

Create a DMS instance

1. In the Azure portal, select + **Create a resource**, search for Azure Database Migration Service, and then

select **Azure Database Migration Service** from the drop-down list.



2. On the **Azure Database Migration Service** screen, select **Create**.

The Azure Database Migration Service (DMS) is designed to streamline the process of migrating on-premises databases to Azure. DMS will simplify the migration of existing on-premises SQL Server and Oracle databases to Azure SQL Database, Azure SQL Managed Instance or Microsoft SQL Server in an Azure Virtual Machine. [Learn more](#)

Before using DMS we recommend you complete the following three steps:

1. Open the [Azure Database Migration Guide](#) for step by step guidance through the migration process
2. Assess your SQL Server on-premises database(s) for feature parity and potential compatibility issues by using [Data Migration Assistant \(DMA\)](#). Alternatively, if you are migrating from Oracle, use the [SQL Server Migration Assistant \(SSMA\)](#)
3. Based on your database needs, create a target database using one of the database services: Azure SQL Database, Azure SQL Managed Instance or SQL Server in an Azure Virtual Machine.

Once your assessments are complete, fixes are applied and schema is deployed, proceed with creating a migration service by clicking **Create** below to migrate the data from your source database to the target.

[Save for later](#)

PUBLISHER Microsoft

USEFUL LINKS [Documentation](#) [Privacy Statement](#)

Create

3. On the **Create Migration Service** screen, specify a name for the service, the subscription, and a new or

existing resource group.

4. Select an existing virtual network or create a new one.

The virtual network provides Azure Database Migration Service with access to the source SQL Server and the target Azure SQL Database instance.

For more information about how to create a virtual network in the Azure portal, see the article [Create a virtual network using the Azure portal](#).

5. Select a pricing tier.

For more information on costs and pricing tiers, see the [pricing page](#).

Create Migration Service

Service Name i
DMSTest ✓

* Subscription
DMSInternalPreview2

* Select a resource group i
 Create new Use existing

<resource group>

Virtual network i
 Create new Use existing

<virtual network>

* Location i
Central US

Pricing tier General Purpose: 4 vCores >

Estimated monthly cost i 0.00 USD

Pin to dashboard

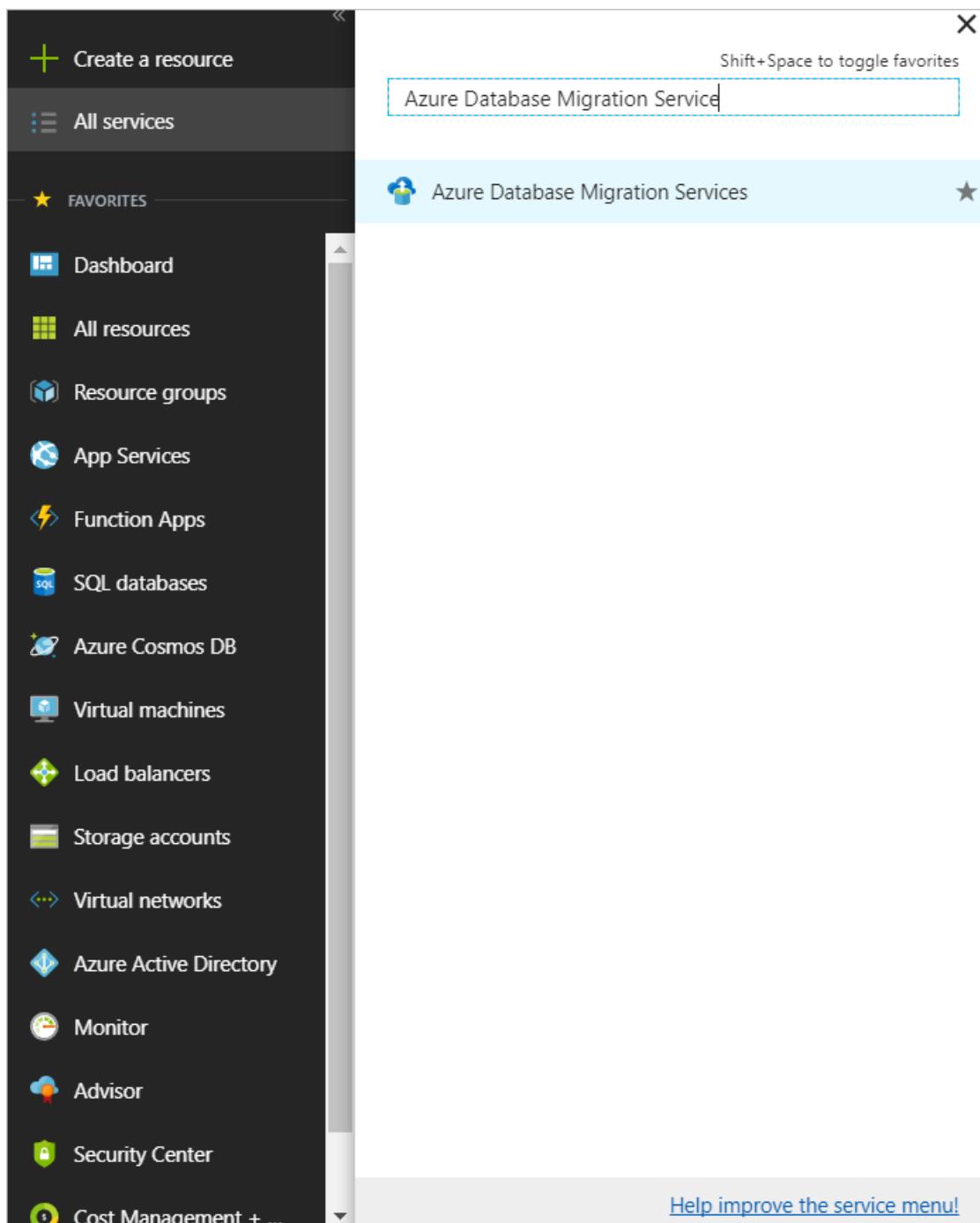
Create [Automation options](#)

6. Select **Create** to create the service.

Create a migration project

After the service is created, locate it within the Azure portal, open it, and then create a new migration project.

1. In the Azure portal, select **All services**, search for Azure Database Migration Service, and then select **Azure Database Migration Services**.



2. On the **Azure Database Migration Services** screen, search for the name of Azure Database Migration Service instance that you created, and then select the instance.

3. Select + New Migration Project.

4. On the **New migration project** screen, specify a name for the project, in the **Source server type** text box, select **MySQL**, in the **Target server type** text box, select **AzureDbForMySQL**.

5. In the **Choose type of activity** section, select **Online data migration**

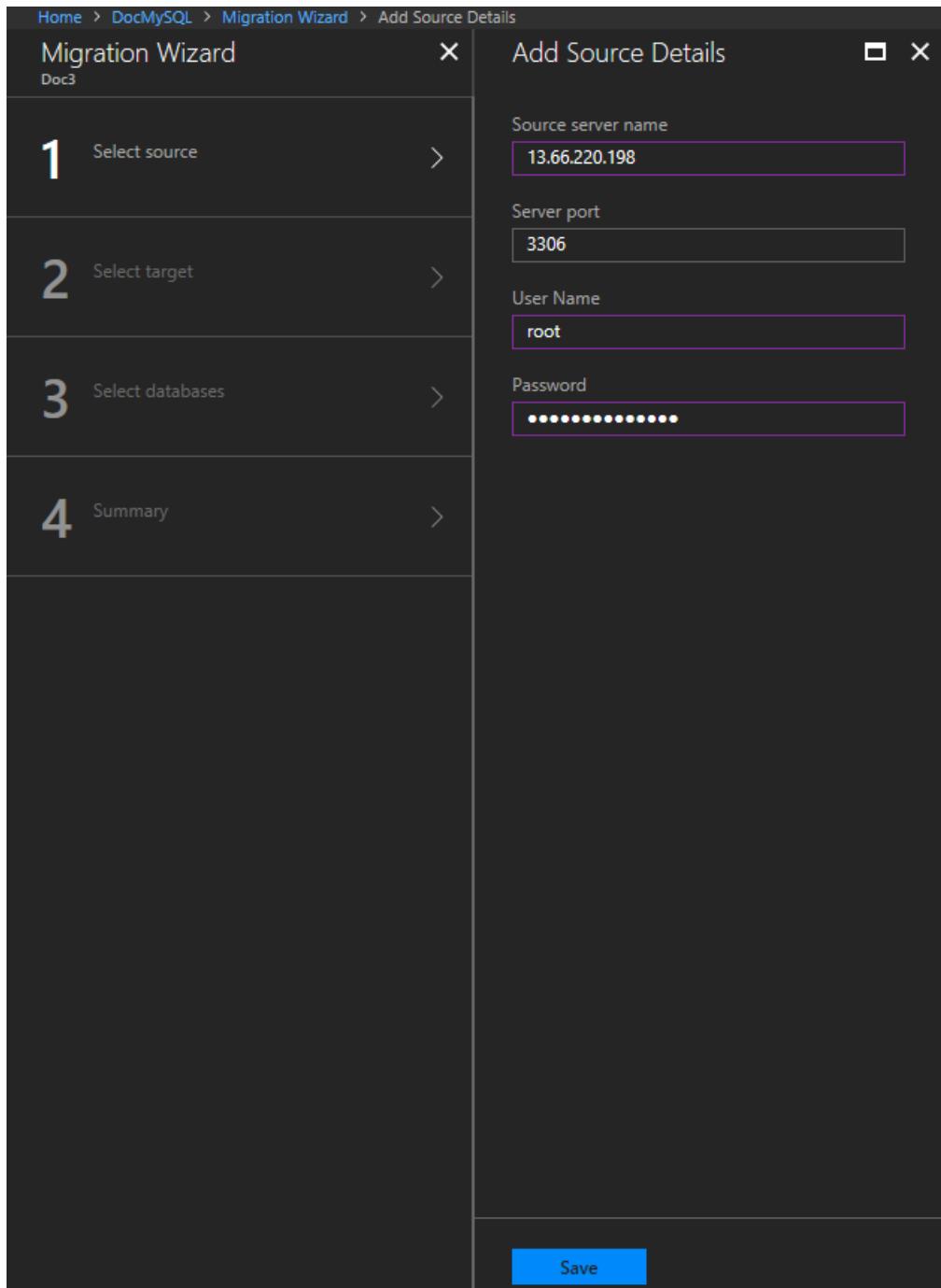
NOTE

Alternately, you can choose **Create project only** to create the migration project now and execute the migration later.

6. Select **Save**, note the requirements to successfully use DMS to migrate data, and then select **Create and run activity**.

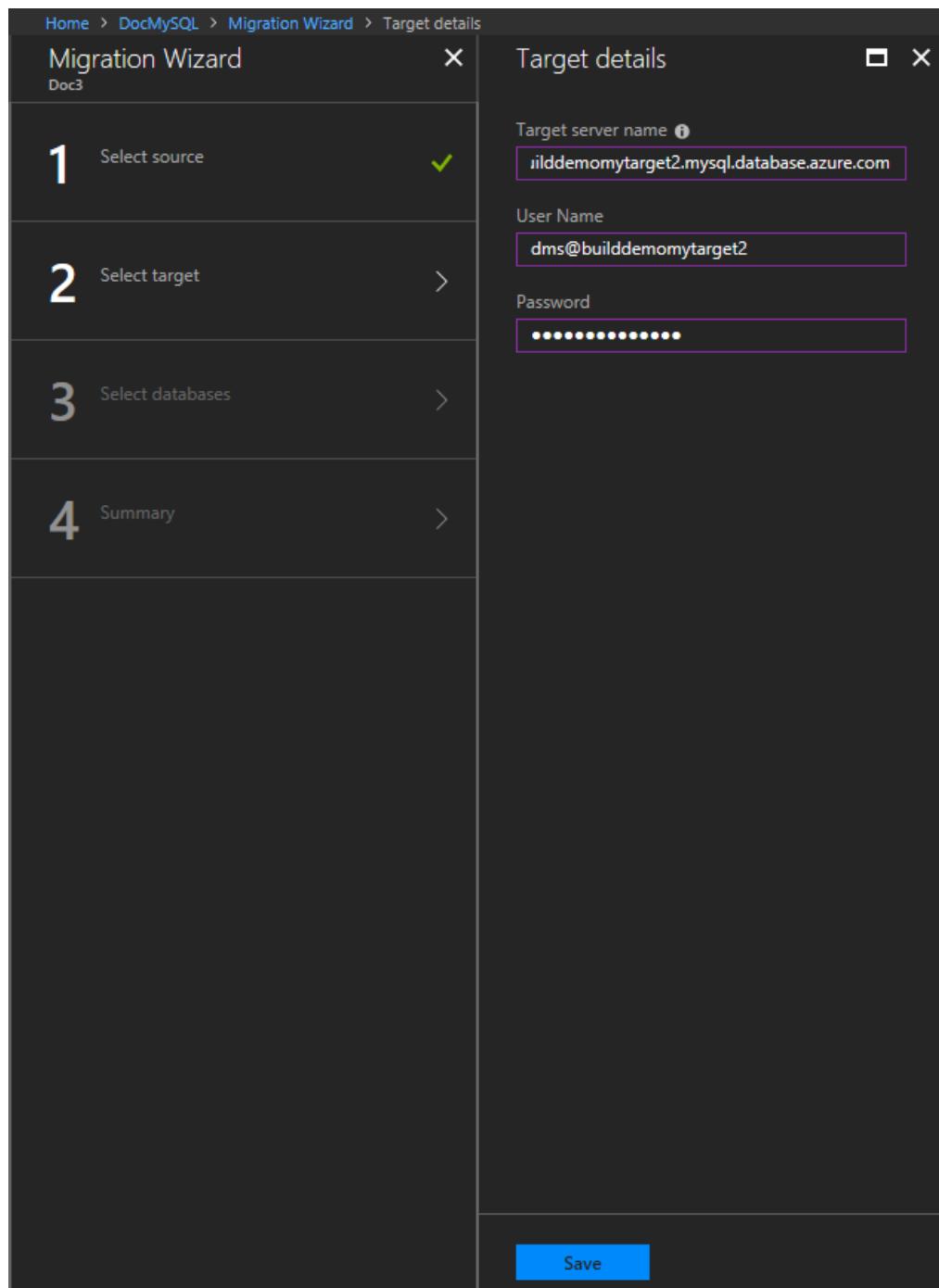
Specify source details

1. On the **Add Source Details** screen, specify the connection details for the source MySQL instance.



Specify target details

1. Select **Save**, and then on the **Target details** screen, specify the connection details for the target Azure Database for MySQL server, which is the pre-provisioned instance of Azure Database for MySQL to which the **Employees** schema was deployed by using mysqldump.



2. Select **Save**, and then on the **Map to target databases** screen, map the source and the target database for migration.

If the target database contains the same database name as the source database, Azure Database Migration Service selects the target database by default.

databases

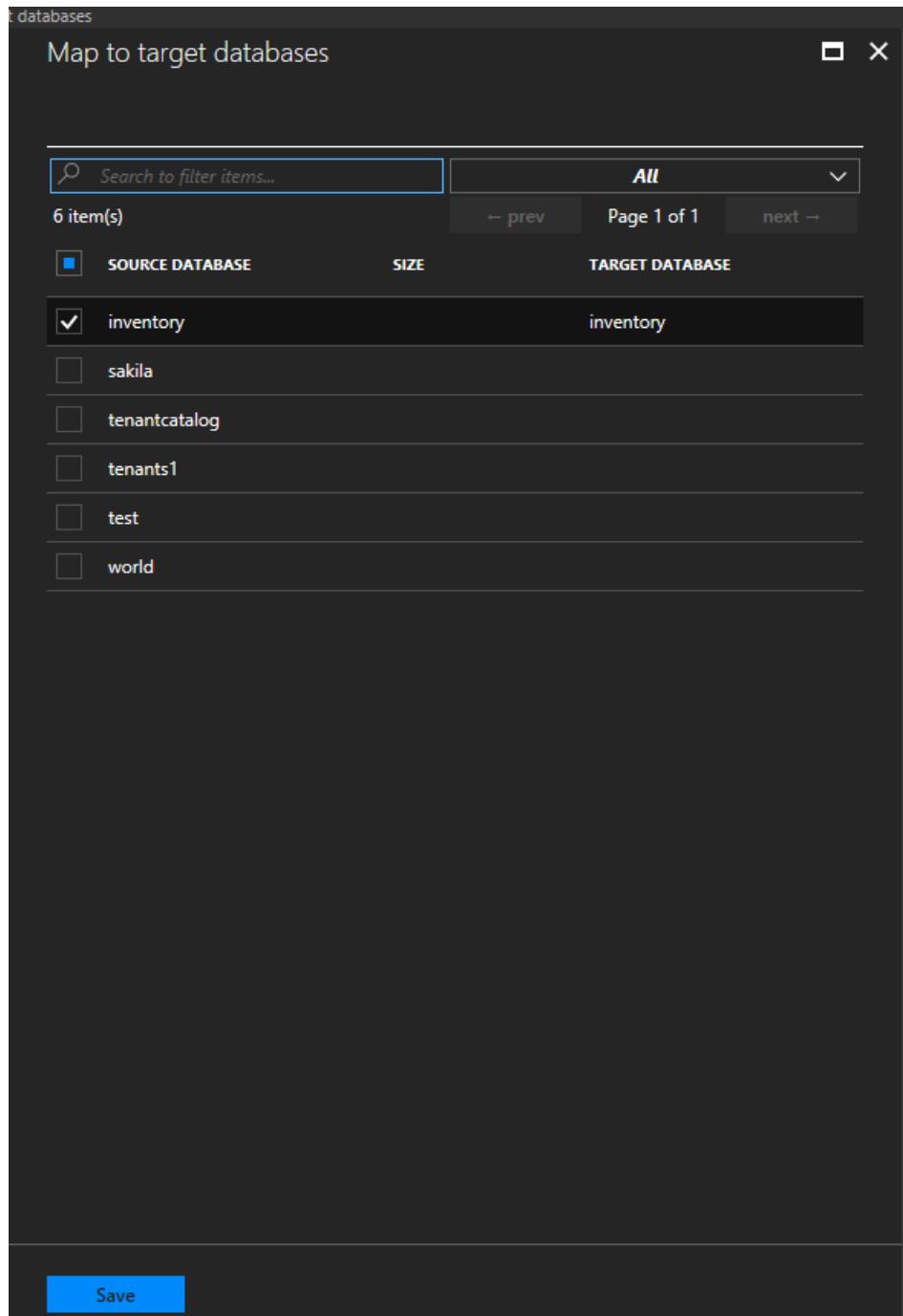
Map to target databases

Search to filter items... All

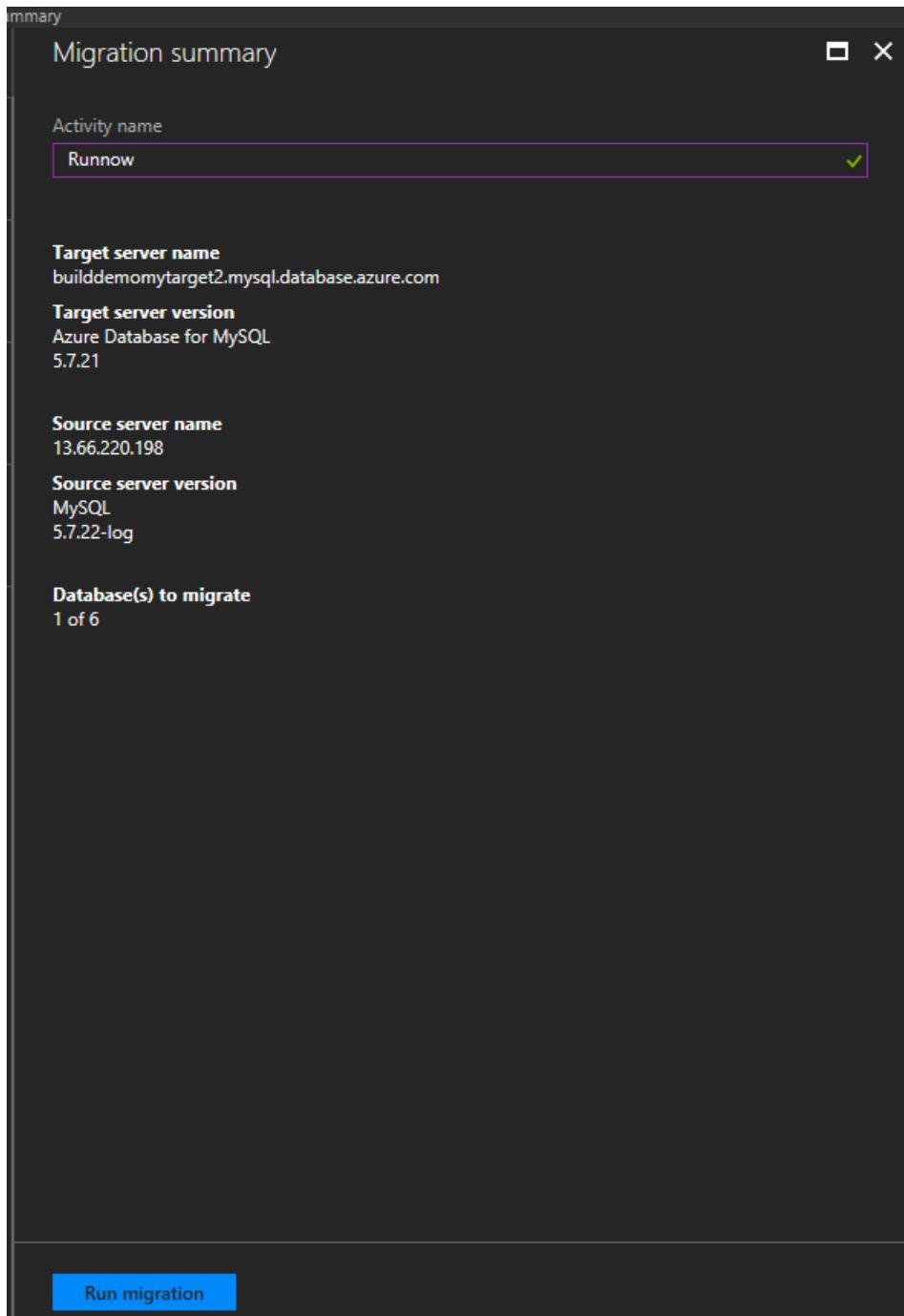
6 item(s) ← prev Page 1 of 1 next →

SOURCE DATABASE	SIZE	TARGET DATABASE
<input checked="" type="checkbox"/> inventory		inventory
<input type="checkbox"/> sakila		
<input type="checkbox"/> tenantcatalog		
<input type="checkbox"/> tenants1		
<input type="checkbox"/> test		
<input type="checkbox"/> world		

Save



3. Select **Save**, on the **Migration summary** screen, in the **Activity name** text box, specify a name for the migration activity, and then review the summary to ensure that the source and target details match what you previously specified.



Run the migration

- Select **Run migration**.

The migration activity window appears, and the **Status** of the activity is **initializing**.

Monitor the migration

1. On the migration activity screen, select **Refresh** to update the display until the **Status** of the migration shows as **Complete**.

Run Now 1					
13.66.220.198		MySQL	1		
Target server		Target version	Type of activity		
builddemomytarget2.mysql.database.azure.com		Azure Database for MySQL	Online		
Activity status			Duration		
Succeeded			00:04:28		
Database Name	Status	Migration Details	Duration	Estimated Application Downtime	Finish Date
inventory	Complete	All changes applied	00:04:08	---	7/3/2018 8:55:37 AM

2. Under **Database Name**, select specific database to get to the migration status for **Full data load** and **Incremental data sync** operations.

Full data load will show the initial load migration status while Incremental data sync will show change data capture (CDC) status.

Inventory			
Source database name		Full load completed	Incremental updates
inventory	2	0	0
Target database name	Full load queued	Incremental inserts	Applied changes
inventory	0	11	11
Database status	Full load loading	Incremental deletes	Tables in error state
Complete	0	0	0
Migration details	Full load failed		
All changes applied	0		
Full load Incremental data sync			
2 item(s)		← prev	Page 1 of 1
Table Name	Status	Completed	Rows
inventory.catalog	Completed	7/26/2018 2:12:38 PM	9
inventory.orders	Completed	7/26/2018 2:12:41 PM	218

Inventory							
Source database name		Full load completed	Incremental updates	Pending changes			
inventory	2	0	0	0			
Target database name	Full load queued	Incremental inserts	Applied changes				
inventory	0	11	11				
Database status	Full load loading	Incremental deletes	Tables in error state				
Complete	0	0	0				
Migration details	Full load failed						
All changes applied	0						
Full load Incremental data sync							
2 item(s)		← prev	Page 1 of 1				
Table Name	Status	Insert	Update	Delete	Total Applied	Data Errors	Last Modified
inventory.catalog	11				0	0	7/26/2018 2:12:36 ...
inventory.orders	11				11	0	7/26/2018 2:13:36 ...

Perform migration cutover

After the initial Full load is completed, the databases are marked **Ready to cutover**.

- When you're ready to complete the database migration, select **Start Cutover**.

The screenshot shows the 'Complete cutover' interface for the 'inventory' database. On the left, there's a summary table with columns for Source database name, Target database name, and Database status. The 'Pending changes' counter is highlighted with a red box and shows 0. On the right, a sidebar provides steps for migration completion:

- Stop all the incoming transactions coming to the source database.
- Wait until all the pending transactions have been applied to the target database. At that time the pending changes counter will set to 0.
- Reconnect your applications to the new Azure target database.

At the bottom right, there are 'Confirm' and 'Apply' buttons, with 'Apply' also highlighted by a red box.

- Make sure to stop all the incoming transactions to the source database; wait until the **Pending changes** counter shows **0**.
- Select **Confirm**, and then select **Apply**.
- When the database migration status shows **Completed**, connect your applications to the new target Azure SQL Database.

Next steps

- For information about known issues and limitations when performing online migrations to Azure Database for MySQL, see the article [Known issues and workarounds with Azure Database for MySQL online migrations](#).
- For information about Azure Database Migration Service, see the article [What is Azure Database Migration Service?](#).
- For information about Azure Database for MySQL, see the article [What is Azure Database for MySQL?](#).

Tutorial: Migrate RDS MySQL to Azure Database for MySQL online using DMS

1/8/2020 • 8 minutes to read • [Edit Online](#)

You can use Azure Database Migration Service to migrate databases from an RDS MySQL instance to [Azure Database for MySQL](#) while the source database remains online during migration. In other words, migration can be achieved with minimal downtime to the application. In this tutorial, you migrate the **Employees** sample database from an instance of RDS MySQL to Azure Database for MySQL by using the online migration activity in Azure Database Migration Service.

In this tutorial, you learn how to:

- Migrate the sample schema by using the mysqldump and mysql utilities.
- Create an instance of Azure Database Migration Service.
- Create a migration project by using Azure Database Migration Service.
- Run the migration.
- Monitor the migration.

NOTE

Using Azure Database Migration Service to perform an online migration requires creating an instance based on the Premium pricing tier. For more information, see the [Azure Database Migration Service pricing](#) page.

IMPORTANT

For an optimal migration experience, Microsoft recommends creating an instance of the Azure Database Migration Service in the same Azure region as the target database. Moving data across regions or geographies can slow down the migration process and introduce errors.

TIP

When you migrate databases to Azure by using Azure Database Migration Service, you can do an *offline* or an *online* migration. With an offline migration, application downtime starts when the migration starts. With an online migration, downtime is limited to the time to cut over at the end of migration. We suggest that you test an offline migration to determine whether the downtime is acceptable; if not, do an online migration.

This article describes how to perform an online migration from an instance of RDS MySQL to Azure Database for MySQL.

Prerequisites

To complete this tutorial, you need to:

- Ensure that the source MySQL server is running a supported MySQL community edition. To determine the version of your MySQL instance, in the mysql utility or MySQL Workbench, run the command:

```
SELECT @@version;
```

For more information, see the article [Supported Azure Database for MySQL versions](#).

- Download and install the [MySQL Employees sample database](#).
- Create an instance of [Azure Database for MySQL](#).
- Create a Microsoft Azure Virtual Network for Azure Database Migration Service by using the Azure Resource Manager deployment model, which provides site-to-site connectivity to your on-premises source servers by using either [ExpressRoute](#) or [VPN](#). For more information about creating a virtual network, see the [Virtual Network Documentation](#), and especially the quickstart articles with step-by-step details.
- Ensure that your virtual network Network Security Group rules don't block the following inbound communication ports to Azure Database Migration Service: 443, 53, 9354, 445, and 12000. For more detail on virtual network NSG traffic filtering, see the article [Filter network traffic with network security groups](#).
- Configure your [Windows Firewall](#) (or your Linux firewall) to allow for database engine access. For MySQL server, allow port 3306 for connectivity.

NOTE

Azure Database for MySQL only supports InnoDB tables. To convert MyISAM tables to InnoDB, please see the article [Converting Tables from MyISAM to InnoDB](#).

Set up AWS RDS MySQL for replication

1. To create a new parameter group, follow the instructions provided by AWS in the article [MySQL Database Log Files](#), in the **Binary Logging Format** section.
2. Create a new parameter group with the following configuration:
 - binlog_format = row
 - binlog_checksum = NONE
3. Save the new parameter group.
4. Associate the new parameter group with the RDS MySQL instance. A reboot might be required.

Migrate the schema

1. Extract the schema from the source database and apply to the target database to complete migration of all database objects such as table schemas, indexes, and stored procedures.

The easiest way to migrate only the schema is to use mysqldump with the --no-data parameter. The command to migrate the schema is:

```
mysqldump -h [servername] -u [username] -p[password] --databases [db name] --no-data > [schema file path]
```

For example, to dump a schema file for the **Employees** database, use the following command:

```
mysqldump -h 10.10.123.123 -u root -p --databases employees --no-data > d:\employees.sql
```

2. Import the schema to target service, which is Azure Database for MySQL. To restore the schema dump file, run the following command:

```
mysql.exe -h [servername] -u [username] -p[password] [database]< [schema file path]
```

For example, to import the schema for the **Employees** database:

```
mysql.exe -h shausample.mysql.database.azure.com -u dms@shausample -p employees < d:\employees.sql
```

3. If you have foreign keys in your schema, the initial load and continuous sync of the migration will fail. To extract the drop foreign key script and add foreign key script at the destination (Azure Database for MySQL), run the following script in MySQL Workbench:

```
SET group_concat_max_len = 8192;
SELECT SchemaName, GROUP_CONCAT(DropQuery SEPARATOR ';\n') as DropQuery, GROUP_CONCAT(AddQuery
SEPARATOR ';\n') as AddQuery
FROM
(SELECT
KCU.REFERENCED_TABLE_SCHEMA as SchemaName,
KCU.TABLE_NAME,
KCU.COLUMN_NAME,
CONCAT('ALTER TABLE ', KCU.TABLE_NAME, ' DROP FOREIGN KEY ', KCU.CONSTRAINT_NAME) AS
DropQuery,
CONCAT('ALTER TABLE ', KCU.TABLE_NAME, ' ADD CONSTRAINT ', KCU.CONSTRAINT_NAME, ' FOREIGN KEY (`',
KCU.COLUMN_NAME, '` REFERENCES `', KCU.REFERENCED_TABLE_NAME, '`(`', KCU.REFERENCED_COLUMN_NAME, ``)
ON UPDATE ', RC.UPDATE_RULE, ' ON DELETE ', RC.DELETE_RULE) AS AddQuery
FROM INFORMATION_SCHEMA.KEY_COLUMN_USAGE KCU, information_schema.REFERENTIAL_CONSTRAINTS
RC
WHERE
KCU.CONSTRAINT_NAME = RC.CONSTRAINT_NAME
AND KCU.REFERENCED_TABLE_SCHEMA = RC.UNIQUE_CONSTRAINT_SCHEMA
AND KCU.REFERENCED_TABLE_SCHEMA = 'SchemaName') Queries
GROUP BY SchemaName;
```

4. Run the drop foreign key (which is the second column) in the query result to drop the foreign key.
5. If you have triggers (insert or update trigger) in the data, it will enforce data integrity in the target before replicating data from the source. The recommendation is to disable triggers in all the tables *at the target* during migration, and then enable the triggers after migration is complete.

To disable triggers in target database:

```
select concat ('alter table ', event_object_table, ' disable trigger ', trigger_name)
from information_schema.triggers;
```

6. If there are instances of the ENUM data type in any tables, we recommend temporarily updating to the 'character varying' datatype in the target table. When data replication is complete, then revert the data type to ENUM.

Register the Microsoft.DataMigration resource provider

1. Sign in to the Azure portal, select **All services**, and then select **Subscriptions**.

All services Filter By category ▾

GENERAL (14) —

- Dashboard PREVIEW ★
- Management Groups PREVIEW ★
- Subscriptions
- Cost Management + Billing PREVIEW ★
- Reservations
- Help + support ★
- Service Health
- Tags ★
- What's new

COMPUTE (20) —

- Virtual machines ★
- Virtual machines (classic)
- Container services ★
- Function Apps

2. Select the subscription in which you want to create the instance of Azure Database Migration Service, and then select **Resource providers**.

Home > Subscriptions > <subscription>

Subscriptions Microsoft

Add

My role: Owner Status: 3 selected

Apply

Search to filter items...

SUBSCRIPTI... ↑↓ SUBSCRIPTION ID ↑↓

<subscription> <subscription ID>

Overview

Access control (IAM)

Diagnose and solve problems

COST MANAGEMENT + BILLING

Partner information

SETTINGS

Programmatic deployment

Resource groups

Resources

Usage + quotas

Policies

Management certificates

My permissions

Resource providers

Properties

Resource locks

SUPPORT + TROUBLESHOOTING

New support request

3. Search for migration, and then to the right of **Microsoft.DataMigration**, select **Register**.

The screenshot shows the Azure portal interface. On the left, the navigation menu includes 'Create a resource', 'All services', 'Dashboard', 'Resource groups', 'App Services', 'Function Apps', 'SQL databases', 'Azure Cosmos DB', 'Virtual machines', 'Load balancers', 'Storage accounts', 'Virtual networks', 'Azure Active Directory', 'Monitor', and 'Advisor'. The 'Resource providers' section on the right shows a table with one item: 'Microsoft.DataMigration' under 'PROVIDER', 'NotRegistered' under 'STATUS', and a red-bordered 'Register' button.

Create an instance of Azure Database Migration Service

1. In the Azure portal, select **+ Create a resource**, search for Azure Database Migration Service, and then select **Azure Database Migration Service** from the drop-down list.

The screenshot shows the 'New' blade in the Azure portal. The search bar at the top contains the text 'Azure Database Migration Service', which is highlighted with a red box. Below the search bar, there is a list of service categories: Networking, Storage, Web + Mobile, Containers, Databases, Data + Analytics, AI + Cognitive Services, Internet of Things, Enterprise Integration, Security + Identity, Developer tools, Monitoring + Management, Add-ons, and Blockchain. Under the 'Databases' category, 'Azure Database Migration Service' is listed as a PREVIEW service. A 'Quickstart tutorial' link is provided for this service.

2. On the **Azure Database Migration Service** screen, select **Create**.

The screenshot shows the Azure Database Migration Service page. On the left, a sidebar lists various Azure services like Subscriptions, Resource groups, App Services, etc. The main content area is titled 'Azure Database Migration Service' and contains an introduction about the service's purpose and migration steps. A 'Create' button at the bottom is highlighted with a red box.

3. On the **Create Migration Service** screen, specify a name for the service, the subscription, and a new or existing resource group.
4. Select the location in which you want to create the instance of Azure Database Migration Service.
5. Select an existing virtual network or create a new one.

The virtual network provides Azure Database Migration Service with access to the source MySQL instance and the target Azure Database for MySQL instance.

For more information about how to create a virtual network in the Azure portal, see the article [Create a virtual network using the Azure portal](#).

6. Select a pricing tier; for this online migration, be sure to select the Premium: 4vCores pricing tier.

Home > New > Azure Database Migration Service

Create Migration Service

Service Name ✓
DMSTest

* Subscription
<subscription ID>

* Select a resource group ✓
ADFHive

Create new

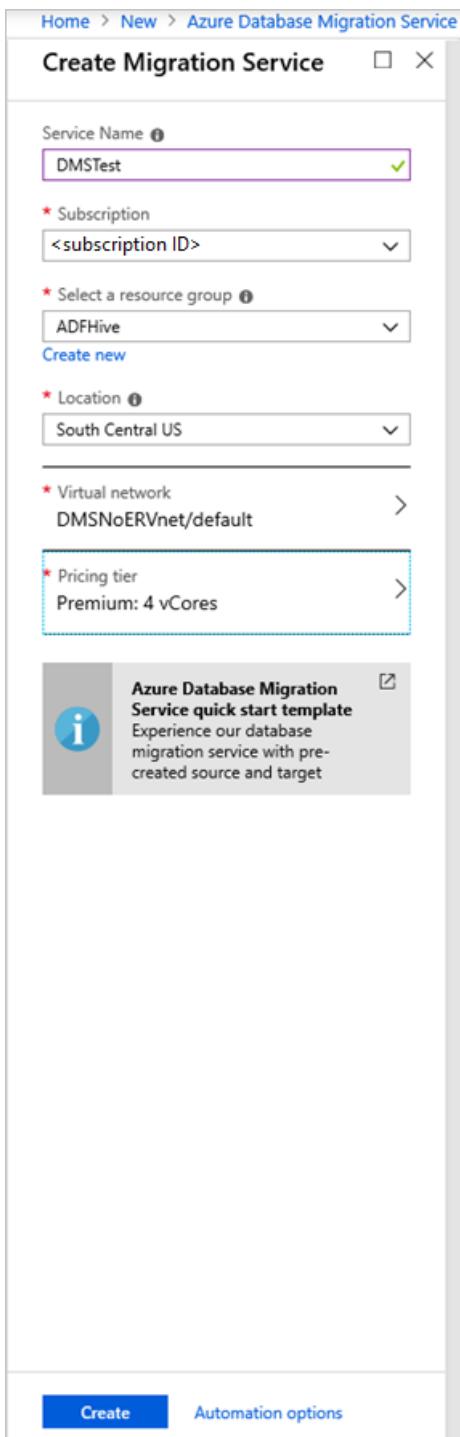
* Location ✓
South Central US

* Virtual network >
DMSNoERVnet/default

* Pricing tier >
Premium: 4 vCores

i Azure Database Migration Service quick start template
Experience our database migration service with pre-created source and target

Create [Automation options](#)

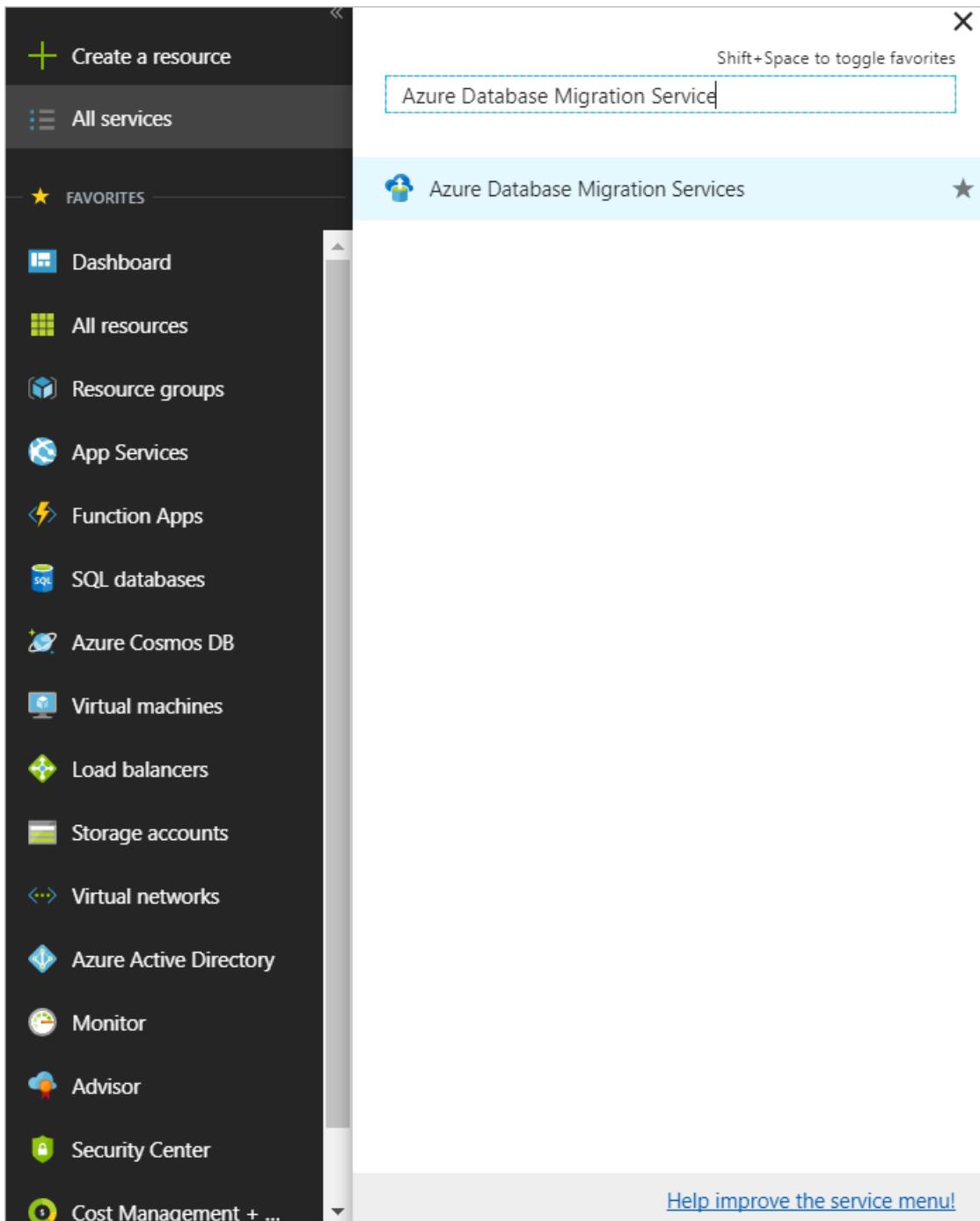


7. Select **Create** to create the service.

Create a migration project

After the service is created, locate it within the Azure portal, open it, and then create a new migration project.

1. In the Azure portal, select **All services**, search for Azure Database Migration Service, and then select **Azure Database Migration Services**.



2. On the **Azure Database Migration Services** screen, search for the name of the Azure Database Migration Service instance that you created, and then select the instance.

3. Select + **New Migration Project**.

4. On the **New migration project** screen, specify a name for the project, in the **Source server type** text box, select **MySQL**, and then in the **Target server type** text box, select **AzureDbForMySQL**.

5. In the **Choose type of activity** section, select **Online data migration**.

IMPORTANT

Be sure to select **Online data migration**; offline migrations are not supported for this scenario.

New migration project

Type of activity

Choose type of activity: Online data migration

Use this option to migrate databases that must be accessible and continuously updated during migration.

To continuously replicate data changes from your source to your target, please implement the steps below on your source server. These steps can be implemented anytime between the moment you create a project and the moment you start a migration activity. After your migration is complete, you will reverse those preparation steps.

- MySQL server source must match with the version that Azure Database for MySQL supports.
- Azure Database for MySQL supports - MySQL community edition, InnoDB engine and migration across source and target with same versions.
- Enable binary logging in my.ini (Windows) or my.cnf (Unix).
- User must have ReplicationAdmin role.

To successfully use Database Migration Service (DMS) to migrate data, you need to:

- Create the target Azure Database for MySQL.
- Deploy schema, indexes and routines to target database:
 - Using MySQL Workbench OR
 - Using mysqldump --no-data

Install MySQL Workbench

Migrating from MySQL server to Azure Database for MySQL using Azure Database Migration Service (DMS) is offered as preview program.

Please review the [Supplemental Terms of Use for Microsoft Azure Previews](#)

Create and run activity **Save**

NOTE

Alternately, you can choose **Create project only** to create the migration project now and execute the migration later.

6. Select **Save**.

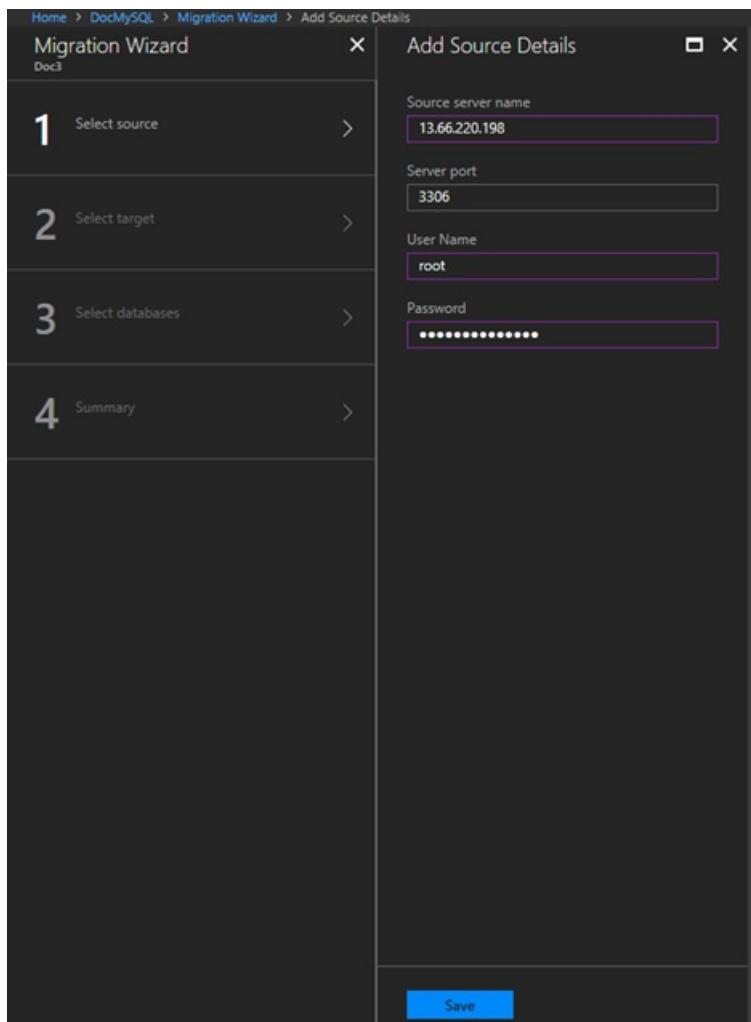
7. Select **Create and run activity** to create the project and run the migration activity.

NOTE

Please make a note of the pre-requisites needed to set up online migration in the project creation blade.

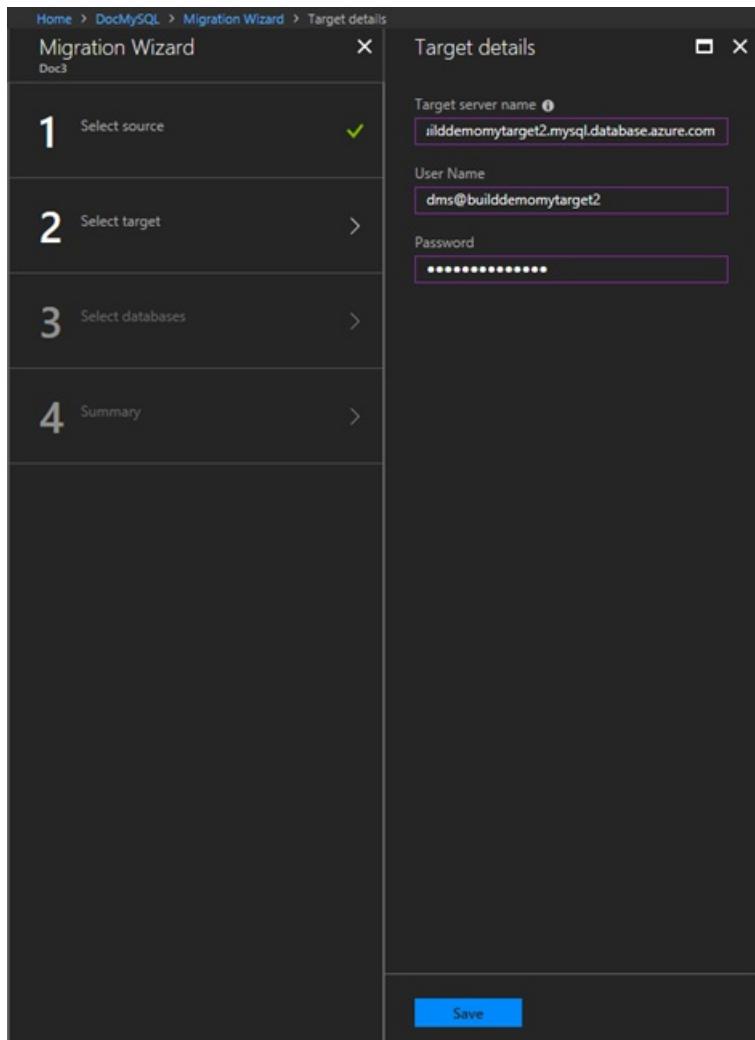
Specify source details

- On the **Migration source detail** screen, specify the connection details for the source MySQL instance.



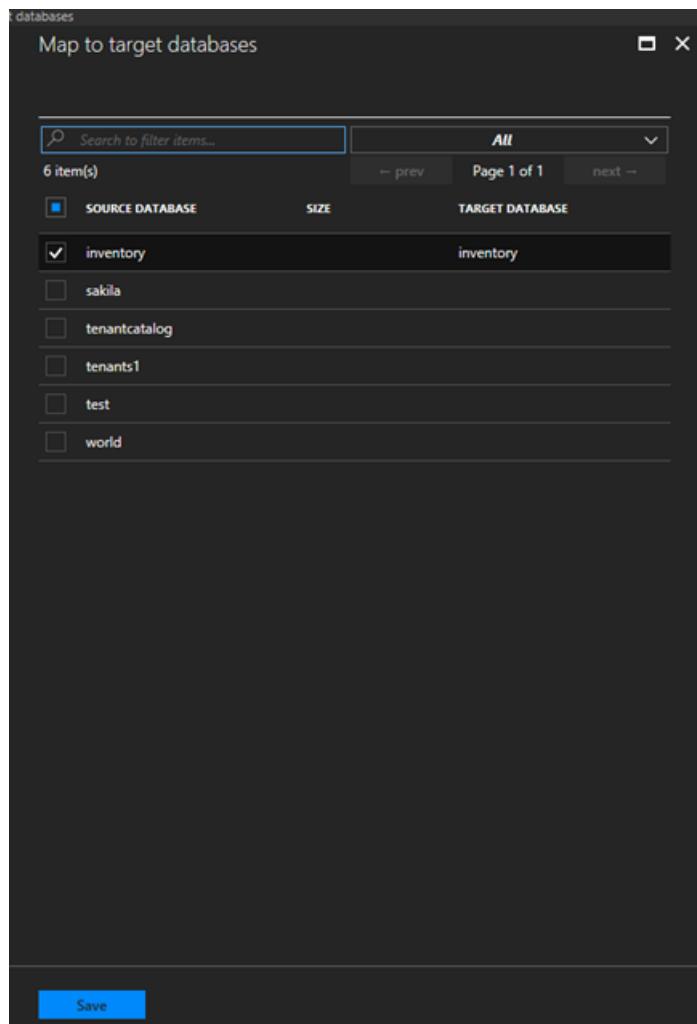
Specify target details

1. Select **Save**, and then on the **Target details** screen, specify the connection details for the target Azure Database for MySQL server, which is pre-provisioned and has the **Employees** schema deployed using MySQLDump.

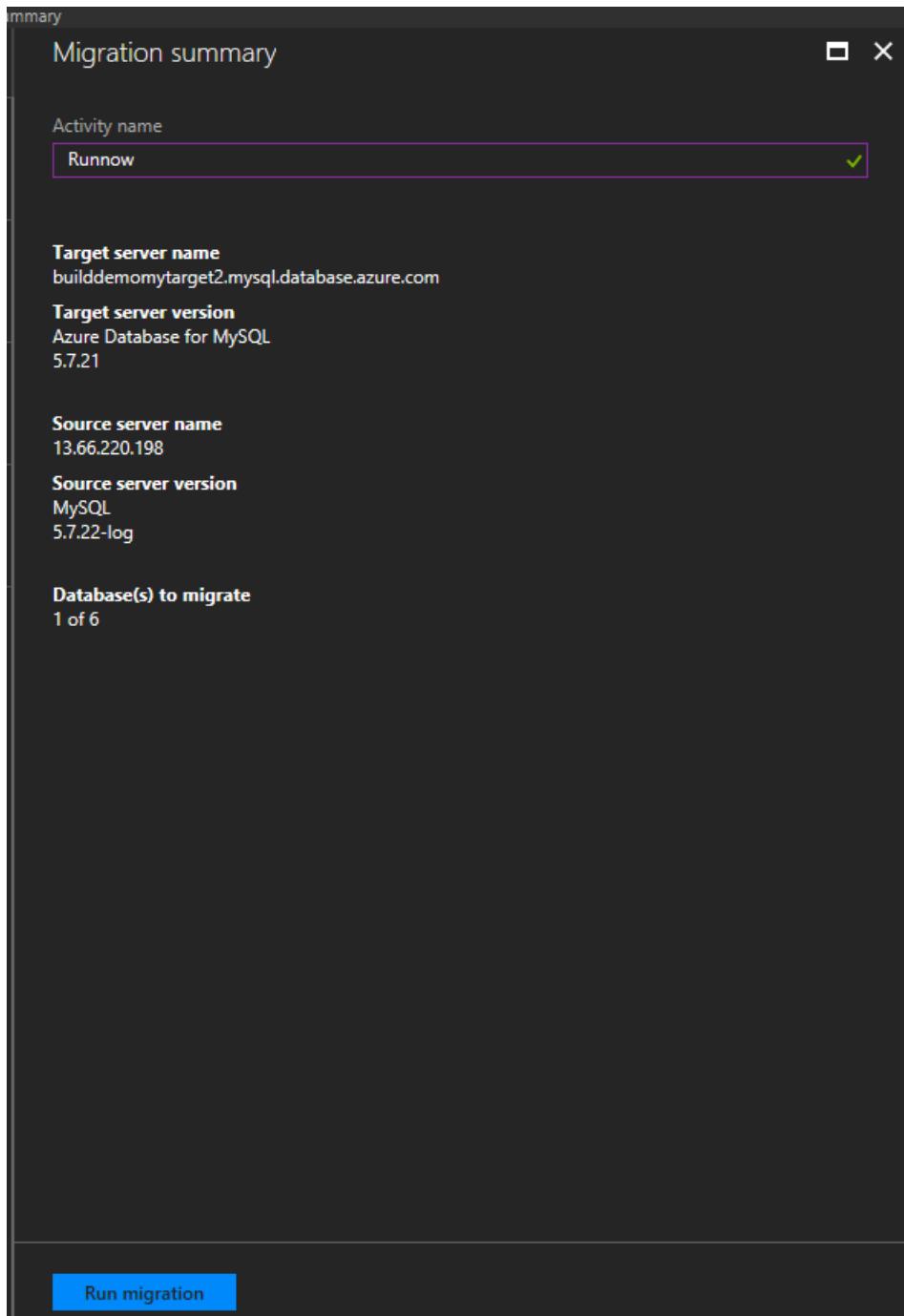


2. Select **Save**, and then on the **Map to target databases** screen, map the source and the target database for migration.

If the target database contains the same database name as the source database, Azure Database Migration Service selects the target database by default.



3. Select **Save**, on the **Migration summary** screen, in the **Activity name** text box, specify a name for the migration activity, and then review the summary to ensure that the source and target details match what you previously specified.



Run the migration

- Select **Run migration**.

The migration activity window appears, and the **Status** of the activity is **Initializing**.

Monitor the migration

1. On the migration activity screen, select **Refresh** to update the display until the **Status** of the migration shows as **Running**.

Source server	Source version	Source databases			
13.66.220.198	MySQL	1			
Target server	Target version	Type of activity			
builddemomytarget2.mysql.database.azure.com	Azure Database for MySQL	Online			
Activity status	Duration				
Succeeded	00:04:28				
Database Name	Status	Migration Details	Duration	Estimated Application Downtime	Finish Date
inventory	Complete	All changes applied	00:04:08	---	7/3/2018 8:55:37 AM

2. Under **DATABASE NAME**, select a specific database to get to the migration status for **Full data load** and **Incremental data sync** operations.

Full data load shows the initial load migration status, while **Incremental data sync** shows change data capture (CDC) status.

Source database name	Full load completed	
inventory	2	
Target database name	Full load queued	
Inventory	0	
Database status	Full load loading	
Complete	0	
Migration details	Full load failed	
All changes applied	0	
Full load	Incremental data sync	
2 item(s)		
Table Name	Status	Completed
inventory.catalog	Completed	7/26/2018 2:12:38 PM
inventory.orders	Completed	7/26/2018 2:12:41 PM

Source database name	Full load completed	Incremental updates	Pending changes
inventory	2	0	0
Target database name	Full load queued	Incremental inserts	Applied changes
Inventory	0	11	11
Database status	Full load loading	Incremental deletes	Tables in error state
Complete	0	0	0
Migration details	Full load failed		
All changes applied	0		
Full load	Incremental data sync		

Perform migration cutover

After the initial Full load is completed, the databases are marked **Ready to Cutover**.

- When you're ready to complete the database migration, select **Start Cutover**.

The screenshot shows the 'Complete cutover' section of the Azure Database Migration Service. It displays migration details for the 'inventory' database, including source and target database names, migration status (Full load completed, Full load queued, etc.), and pending changes (0). A sidebar on the right provides instructions for performing a migration cutover, such as stopping incoming transactions and reconnecting applications. The 'Pending changes' counter is highlighted with a red box.

2. Make sure to stop all the incoming transactions to the source database; wait until the **Pending changes** counter shows **0**.
3. Select **Confirm**, and then select **Apply**.
4. When the database migration status shows **Completed**, connect your applications to the new target Azure Database for MySQL database.

Your online migration of an on-premises instance of MySQL to Azure Database for MySQL is now complete.

Next steps

- For information about the Azure Database Migration Service, see the article [What is the Azure Database Migration Service?](#).
- For information about Azure Database for MySQL, see the article [What is Azure Database for MySQL?](#).
- For other questions, email the [Ask Azure Database Migrations](#) alias.

Tutorial: Migrate PostgreSQL to Azure DB for PostgreSQL online using DMS via the Azure portal

2/19/2020 • 8 minutes to read • [Edit Online](#)

You can use Azure Database Migration Service to migrate the databases from an on-premises PostgreSQL instance to [Azure Database for PostgreSQL](#) with minimal downtime to the application. In this tutorial, you migrate the **DVD Rental** sample database from an on-premises instance of PostgreSQL 9.6 to Azure Database for PostgreSQL by using the online migration activity in Azure Database Migration Service.

In this tutorial, you learn how to:

- Migrate the sample schema using the pg_dump utility.
- Create an instance of Azure Database Migration Service.
- Create a migration project in Azure Database Migration Service.
- Run the migration.
- Monitor the migration.
- Perform migration cutover.

NOTE

Using Azure Database Migration Service to perform an online migration requires creating an instance based on the Premium pricing tier.

IMPORTANT

For an optimal migration experience, Microsoft recommends creating an instance of Azure Database Migration Service in the same Azure region as the target database. Moving data across regions or geographies can slow down the migration process and introduce errors.

Prerequisites

To complete this tutorial, you need to:

- Download and install [PostgreSQL community edition](#) 9.4, 9.5, 9.6, or 10. The source PostgreSQL Server version must be 9.4, 9.5, 9.6, 10, or 11. For more information, see the article [Supported PostgreSQL Database Versions](#).

In addition, the on-premises PostgreSQL version must match the Azure Database for PostgreSQL version. For example, PostgreSQL 9.6 can only migrate to Azure Database for PostgreSQL 9.6, 10, or 11, but not to Azure Database for PostgreSQL 9.5.

- [Create an Azure Database for PostgreSQL server](#) or [Create an Azure Database for PostgreSQL - Hyperscale \(Citus\) server](#).
- Create a Microsoft Azure Virtual Network for Azure Database Migration Service by using the Azure Resource Manager deployment model, which provides site-to-site connectivity to your on-premises source servers by using either [ExpressRoute](#) or [VPN](#). For more information about creating a virtual network, see the [Virtual Network Documentation](#), and especially the quickstart articles with step-by-step details.

NOTE

During virtual network setup, if you use ExpressRoute with network peering to Microsoft, add the following service endpoints to the subnet in which the service will be provisioned:

- Target database endpoint (for example, SQL endpoint, Cosmos DB endpoint, and so on)
- Storage endpoint
- Service bus endpoint

This configuration is necessary because Azure Database Migration Service lacks internet connectivity.

- Ensure that the Network Security Group (NSG) rules for your virtual network don't block the following inbound communication ports to Azure Database Migration Service: 443, 53, 9354, 445, 12000. For more detail on virtual network NSG traffic filtering, see the article [Filter network traffic with network security groups](#).
- Configure your [Windows Firewall for database engine access](#).
- Open your Windows firewall to allow Azure Database Migration Service to access the source PostgreSQL Server, which by default is TCP port 5432.
- When using a firewall appliance in front of your source database(s), you may need to add firewall rules to allow the Azure Database Migration Service to access the source database(s) for migration.
- Create a server-level [firewall rule](#) for Azure Database for PostgreSQL to allow Azure Database Migration Service to access to the target databases. Provide the subnet range of the virtual network used for Azure Database Migration Service.
- Enable logical replication in the postgresql.config file, and set the following parameters:
 - wal_level = **logical**
 - max_replication_slots = [number of slots], recommend setting to **five slots**
 - max_wal_senders =[number of concurrent tasks] - The max_wal_senders parameter sets the number of concurrent tasks that can run, recommend setting to **10 tasks**

IMPORTANT

All tables in your existing database need a primary key to ensure that changes can be synced to the target database.

Migrate the sample schema

To complete all the database objects like table schemas, indexes and stored procedures, we need to extract schema from the source database and apply to the database.

1. Use pg_dump -s command to create a schema dump file for a database.

```
pg_dump -o -h hostname -U db_username -d db_name -s > your_schema.sql
```

For example, to create a schema dump file for the **dvdrental** database:

```
pg_dump -o -h localhost -U postgres -d dvdrental -s -0 -x > dvdrentalSchema.sql
```

For more information about using the pg_dump utility, see the examples in the [pg-dump](#) tutorial.

2. Create an empty database in your target environment, which is Azure Database for PostgreSQL.

For details on how to connect and create a database, see the article [Create an Azure Database for PostgreSQL server in the Azure portal](#) or [Create an Azure Database for PostgreSQL - Hyperscale \(Citus\) server in the Azure portal](#).

NOTE

An instance of Azure Database for PostgreSQL - Hyperscale (Citus) has only a single database: **citus**.

3. Import the schema into the target database you created by restoring the schema dump file.

```
psql -h hostname -U db_username -d db_name < your_schema.sql
```

For example:

```
psql -h mypgserver-20170401.postgres.database.azure.com -U postgres -d dvdrental citus < dvdrentalSchema.sql
```

4. To extract the drop foreign key script and add it at the destination (Azure Database for PostgreSQL), in PgAdmin or in psql, run the following script.

IMPORTANT

Foreign keys in your schema will cause the initial load and continuous sync of the migration to fail.

```
SELECT Queries.tablename
      ,concat('alter table ', Queries.tablename, ' ', STRING_AGG(concat('DROP CONSTRAINT ',
      Queries.foreignkey), ',')) as DropQuery
      ,concat('alter table ', Queries.tablename, ' ',
      STRING_AGG(concat('ADD CONSTRAINT ', Queries.foreignkey,
      'FOREIGN KEY (', column_name, ')', 'REFERENCES ', foreign_table_name, '(', foreign_column_name, ')'), ',')) as AddQuery
   FROM
  (SELECT
    tc.table_schema,
    tc.constraint_name as foreignkey,
    tc.table_name as tableName,
    kcu.column_name,
    ccu.table_schema AS foreign_table_schema,
    ccu.table_name AS foreign_table_name,
    ccu.column_name AS foreign_column_name
   FROM
    information_schema.table_constraints AS tc
   JOIN information_schema.key_column_usage AS kcu
     ON tc.constraint_name = kcu.constraint_name
    AND tc.table_schema = kcu.table_schema
   JOIN information_schema.constraint_column_usage AS ccu
     ON ccu.constraint_name = tc.constraint_name
    AND ccu.table_schema = tc.table_schema
  WHERE constraint_type = 'FOREIGN KEY') Queries
 GROUP BY Queries.tablename;
```

5. Run the drop foreign key (which is the second column) in the query result.
6. To disable triggers in target database, run the script below.

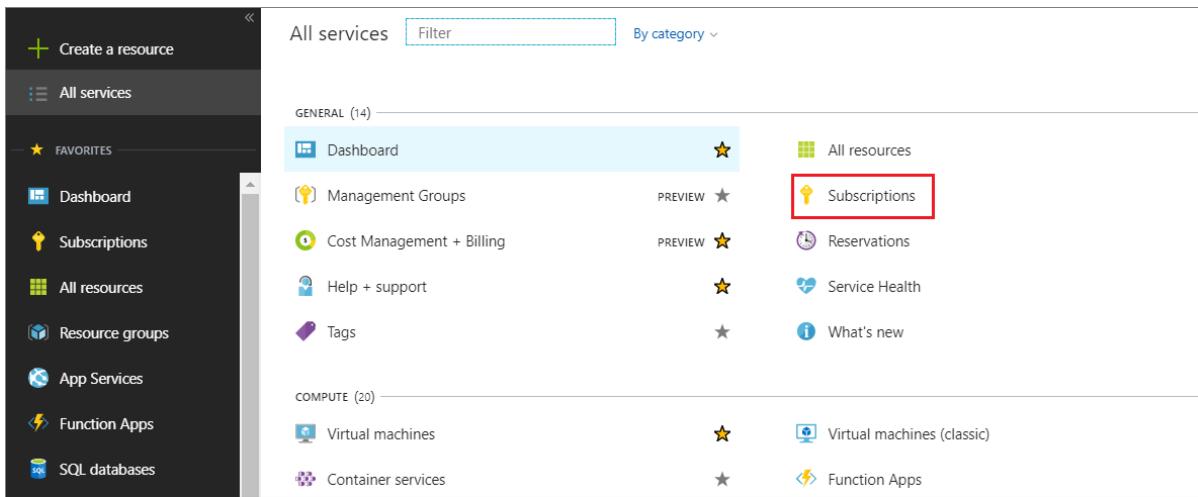
IMPORTANT

Triggers (insert or update) in the data enforce data integrity in the target ahead of the data being replicated from the source. As a result, it's recommended that you disable triggers in all the tables **at the target** during migration, and then re-enable the triggers after migration is complete.

```
select concat ('alter table ', event_object_table, ' disable trigger ', trigger_name)
from information_schema.triggers;
```

Register the Microsoft.DataMigration resource provider

1. Sign in to the Azure portal, select **All services**, and then select **Subscriptions**.



2. Select the subscription in which you want to create the instance of Azure Database Migration Service, and then select **Resource providers**.

The screenshot shows the Azure portal's Subscriptions page. On the left, there's a navigation bar with various service icons like App Services, SQL databases, and Azure Active Directory. The main area displays a list of subscriptions, with one entry selected: '<subscription> <subscription ID>'. The right sidebar contains sections for Overview, Access control (IAM), Diagnose and solve problems, Cost management + Billing, Partner information, Settings, and Support + Troubleshooting. The 'Resource providers' link under Settings is highlighted with a red box.

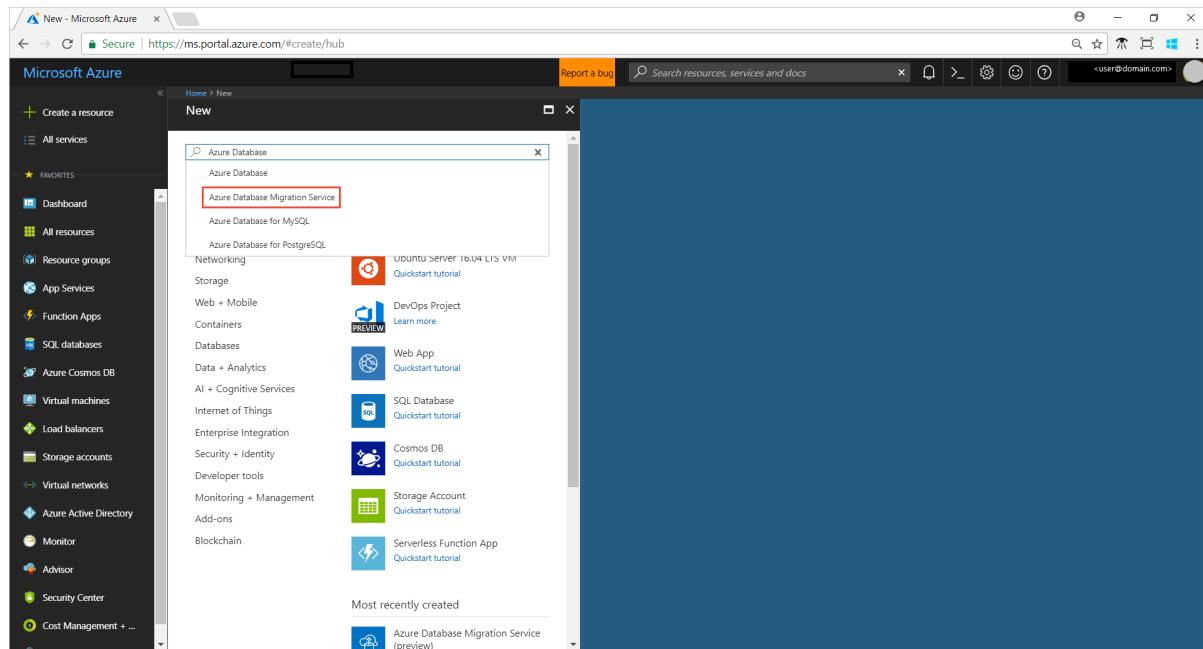
3. Search for migration, and then to the right of **Microsoft.DataMigration**, select **Register**.

This screenshot shows the 'Resource providers' page within the Azure portal. It lists a single provider named 'migration' with the status 'NotRegistered'. To the right of the provider name is a red-bordered 'Register' button. The rest of the interface is identical to the previous screenshot, showing the Subscriptions page with the 'Resource providers' link highlighted.

Create a DMS instance

1. In the Azure portal, select + **Create a resource**, search for Azure Database Migration Service, and then

select **Azure Database Migration Service** from the drop-down list.



2. On the **Azure Database Migration Service** screen, select **Create**.

The Azure Database Migration Service (DMS) is designed to streamline the process of migrating on-premises databases to Azure. DMS will simplify the migration of existing on-premises SQL Server and Oracle databases to Azure SQL Database, Azure SQL Managed Instance or Microsoft SQL Server in an Azure Virtual Machine. [Learn more](#)

Before using DMS we recommend you complete the following three steps:

1. Open the [Azure Database Migration Guide](#) for step by step guidance through the migration process
2. Assess your SQL Server on-premises database(s) for feature parity and potential compatibility issues by using [Data Migration Assistant \(DMA\)](#). Alternatively, if you are migrating from Oracle, use the [SQL Server Migration Assistant \(SSMA\)](#)
3. Based on your database needs, create a target database using one of the database services: Azure SQL Database, Azure SQL Managed Instance or SQL Server in an Azure Virtual Machine.

Once your assessments are complete, fixes are applied and schema is deployed, proceed with creating a migration service by clicking **Create** below to migrate the data from your source database to the target.

[Save for later](#)

PUBLISHER Microsoft

USEFUL LINKS [Documentation](#) [Privacy Statement](#)

Create

3. On the **Create Migration Service** screen, specify a name, the subscription, a new or existing resource

group, and the location for the service.

4. Select an existing virtual network or create a new one.

The virtual network provides Azure Database Migration Service with access to the source PostgreSQL server and the target Azure Database for PostgreSQL instance.

For more information about how to create a virtual network in the Azure portal, see the article [Create a virtual network using the Azure portal](#).

5. Select a pricing tier.

For more information on costs and pricing tiers, see the [pricing page](#).

Create Migration Service

Basics [Networking](#) [Tags](#) [Review + create](#)

Azure Database Migration Service is designed to streamline the process of migrating on-premises databases to Azure. [Learn more.](#)

Project details

Select the subscription to manage deployed resources and constants. Use resource groups as you would folders, to organize and manage all of your resources.

Subscription * ⓘ

Resource group * ⓘ [Create new](#)

Instance details

Migration service name * ⓘ

Location * ⓘ

Service mode * ⓘ [Azure](#) [Hybrid \(Preview\)](#)

Pricing tier *
Standard
1 vCores [Configure tier](#)

Tip: Use an Azure Database Migration Service quick start template with pre-created source and targets. [Learn more.](#)

[Review + create](#) [Next : Networking >>](#)

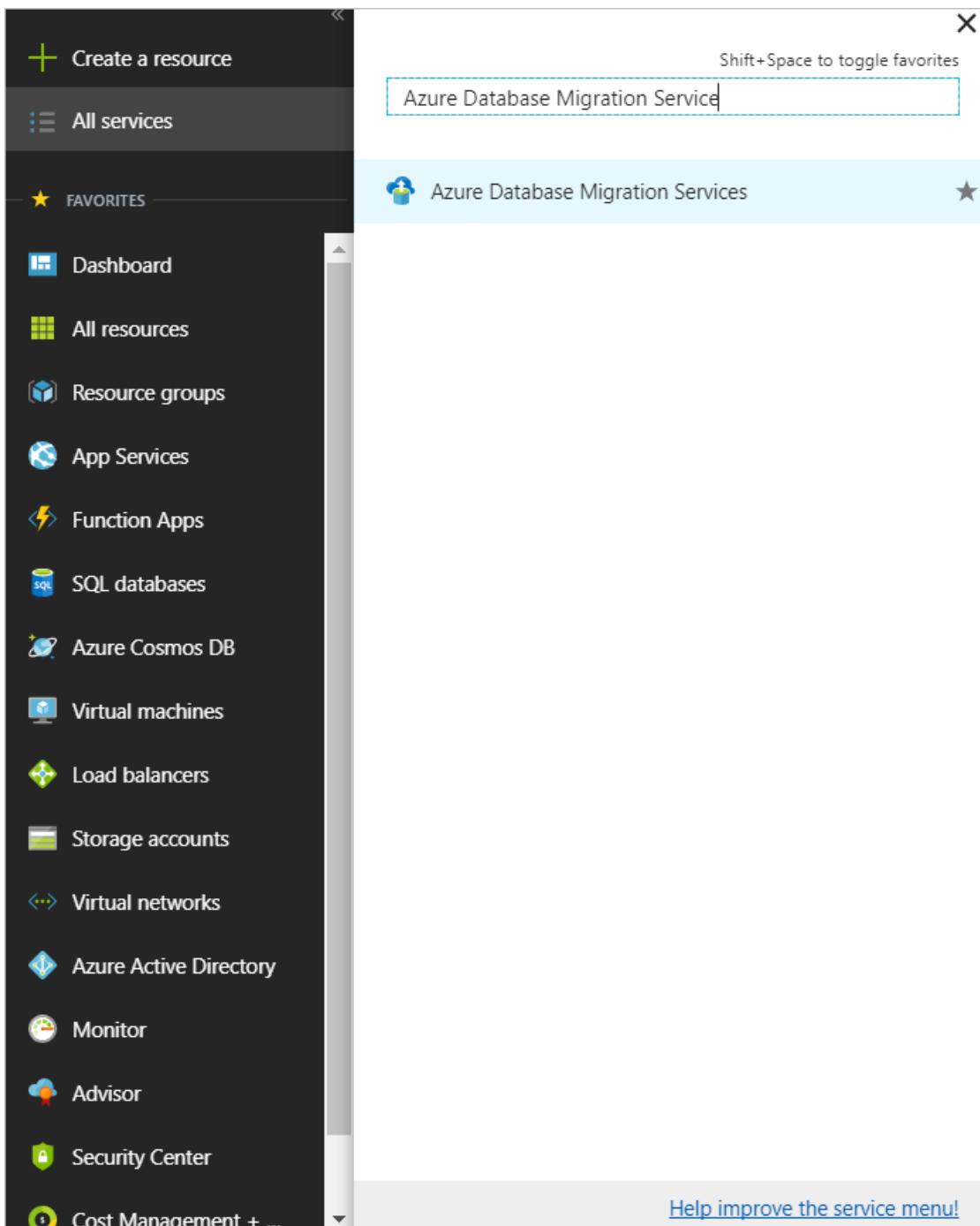
6. Select **Review + create** to create the service.

Service creation will complete within about 10 to 15 minutes.

Create a migration project

After the service is created, locate it within the Azure portal, open it, and then create a new migration project.

1. In the Azure portal, select **All services**, search for Azure Database Migration Service, and then select **Azure Database Migration Services**.



2. On the **Azure Database Migration Services** screen, search for the name of Azure Database Migration Service instance that you created, select the instance, and then select + **New Migration Project**.
3. On the **New migration project** screen, specify a name for the project, in the **Source server type** text box, select **PostgresSQL**, in the **Target server type** text box, select **Azure Database for PostgreSQL**.
4. In the **Choose type of activity** section, select **Online data migration**.

<p>New migration project</p> <p>Project name PGTest</p> <p>Source server type PostgreSQL</p> <p>Target server type Azure Database for PostgreSQL</p> <p>*Choose type of activity Online data migration</p> <p>To successfully use Database Migration Service (DMS) to migrate data, you need to:</p> <ol style="list-style-type: none"> 1. Migrate schema using pg_dump -o -h hostname -U db_username -d db_name -s > your_schema.sql 2. Remove foreign keys in schema at target Azure Database for PostgreSQL 3. Disable triggers at target Azure Database for PostgreSQL 4. Provision Database Migration Service and create a migration task <p>Please refer to this tutorial for more details.</p>	<p>Type of activity</p> <p>Choose type of activity Online data migration</p> <p>Use this option to migrate databases that must be accessible and continuously updated during migration.</p> <p>To continuously replicate data changes from your source to your target, please implement the steps below on your source server. These steps can be implemented anytime between the moment you create a project and the moment you start a migration activity. After your migration is complete, you will reverse those preparation steps.</p> <ul style="list-style-type: none"> • Azure Database for PostgreSQL supports community edition and the source PostgreSQL server must match the same major version that Azure Database for PostgreSQL supports. For example, upgrading from PostgreSQL 9.5.x to 9.6.x is not supported. • Enable logical replication in the postgresql.conf file.
Create and run activity	Save

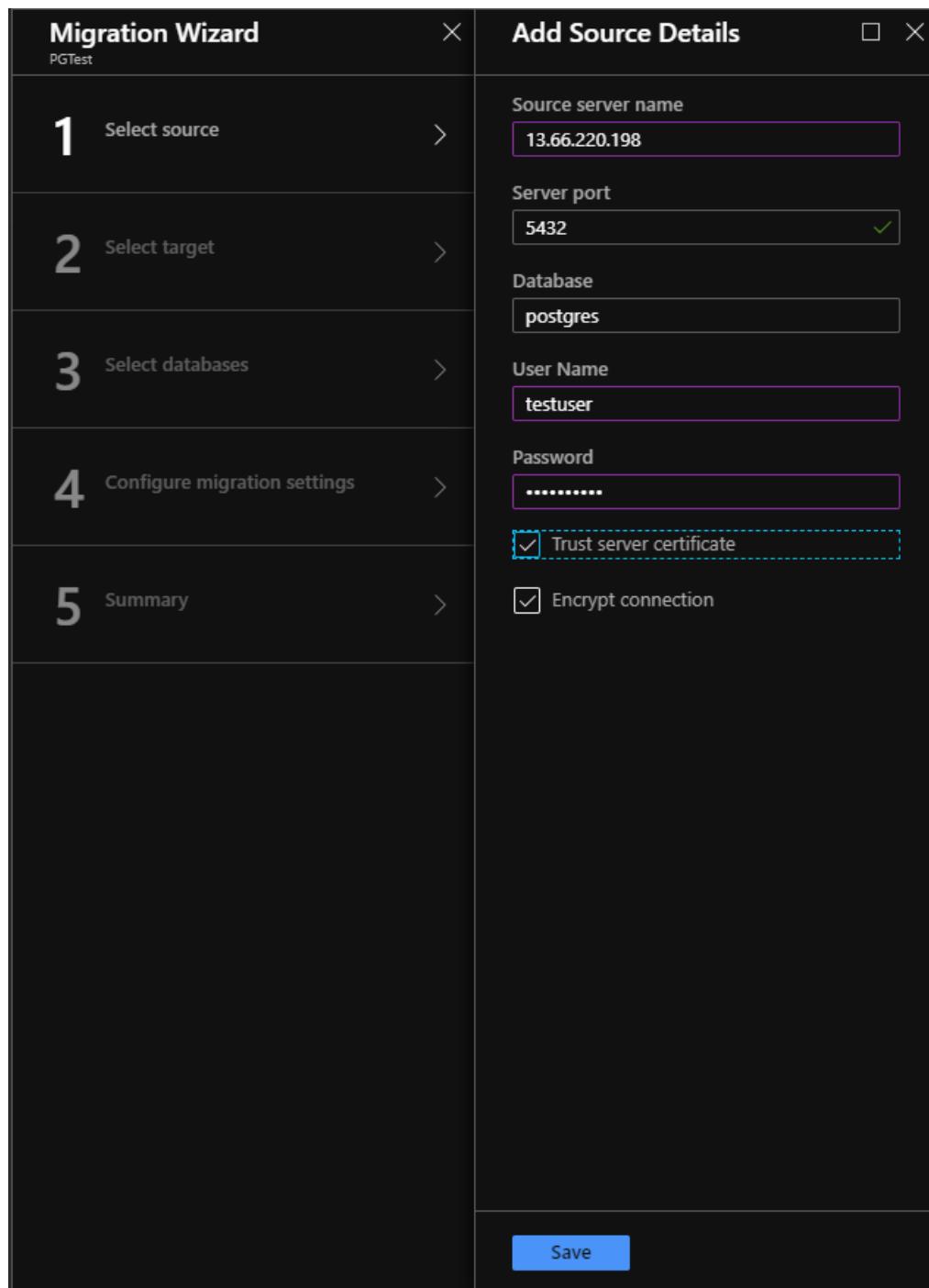
NOTE

Alternately, you can choose **Create project only** to create the migration project now and execute the migration later.

5. Select **Save**, note the requirements to successfully use Azure Database Migration Service to migrate data, and then select **Create and run activity**.

Specify source details

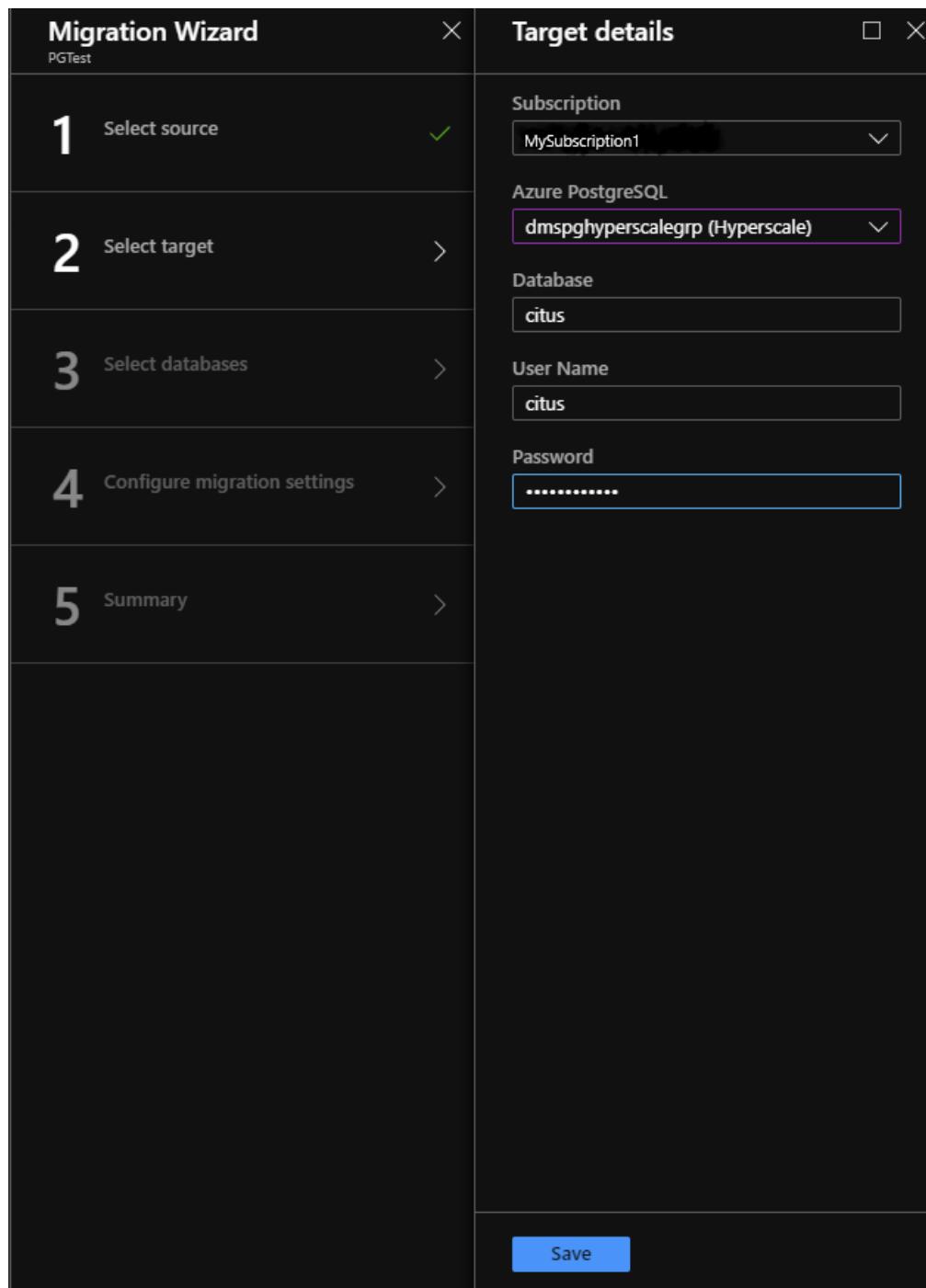
1. On the **Add Source Details** screen, specify the connection details for the source PostgreSQL instance.



2. Select **Save**.

Specify target details

1. On the **Target details** screen, specify the connection details for the target Hyperscale (Citus) server, which is the pre-provisioned instance of Hyperscale (Citus) to which the **DVD Rentals** schema was deployed by using pg_dump.

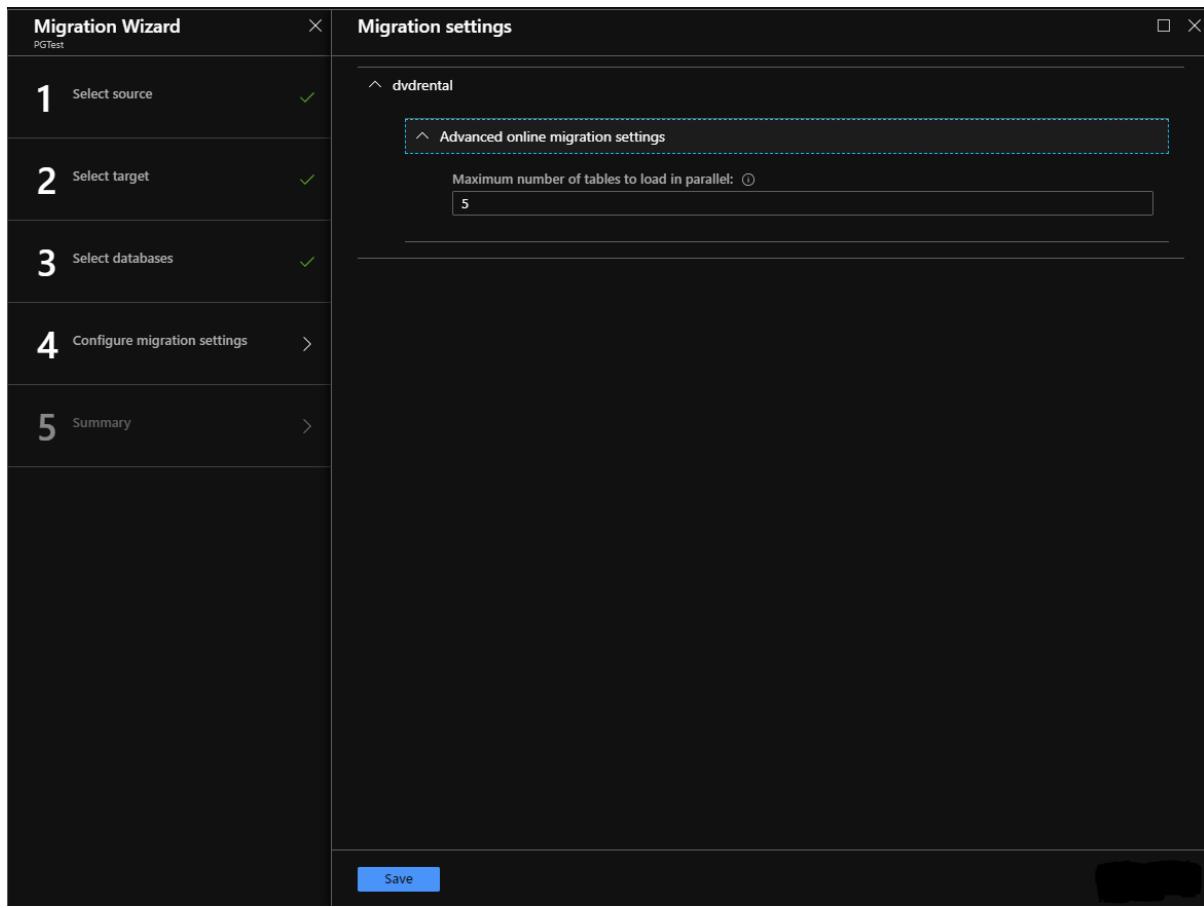


2. Select **Save**, and then on the **Map to target databases** screen, map the source and the target database for migration.

If the target database contains the same database name as the source database, Azure Database Migration Service selects the target database by default.

The screenshot shows the 'Migration Wizard' interface. On the left, a vertical navigation bar lists five steps: 1. Select source (green checkmark), 2. Select target (green checkmark), 3. Select databases (>), 4. Configure migration settings (>), and 5. Summary (>). The current step is 'Map to target databases'. The main area displays a table mapping source databases to target databases. The table has two columns: 'Source Database' and 'Target Database'. Under 'Source Database', there are two entries: 'dvdrental' (selected, indicated by a checked checkbox) and 'postgres' (unchecked). Under 'Target Database', there are two entries: 'citus' (selected, highlighted with a purple border) and 'postgres' (unchecked). A search bar at the top left is empty, and a status bar at the bottom indicates '2 item(s)'. Navigation buttons for 'prev', 'next', and 'Page 1 of 1' are also present. A blue 'Save' button is located at the bottom center.

3. Select **Save**, and then on the **Migration settings** screen, accept the default values.



4. Select **Save**, on the **Migration summary** screen, in the **Activity name** text box, specify a name for the migration activity, and then review the summary to ensure that the source and target details match what you previously specified.

Migration Wizard

PGTest

Migration summary

Activity name
Testmigration

Target server name
ezhyperscalepg-c.postgres.database.azure.com

Target server version
Azure Database for PostgreSQL
11.5

Source server name
13.66.220.198

Source server version
PostgreSQL
9.6.15

Database(s) to migrate
1 of 7

Run migration

Run the migration

- Select **Run migration**.

The migration activity window appears, and the **Status** of the activity should update to show as **Backup in Progress**.

Monitor the migration

- On the migration activity screen, select **Refresh** to update the display until the **Status** of the migration shows as **Complete**.

Database name	Status	Migration details	Duration	Estimated application downtime ⓘ	Finish Date
dvrental	Running	Ready to cutover	00:00:35	---	---

- When the migration is complete, under **Database Name**, select a specific database to get to the migration

status for **Full data load** and **Incremental data sync** operations.

NOTE

Full data load shows the initial load migration status, while **Incremental data sync** shows change data capture (CDC) status.

The screenshot shows the migration status for the dvrental database. The 'Full load completed' counter is highlighted with a red box and shows the value 19. Other counters include 'Pending changes' at 0, 'Applied changes' at 0, and 'Tables in error state' at 0.

The screenshot shows the migration status for the dvrental database. The 'Incremental updates' counter is highlighted with a red box and shows the value 28808. Other counters include 'Pending changes' at 0, 'Applied changes' at 43735, and 'Tables in error state' at 0.

Perform migration cutover

After the initial Full load is completed, the databases are marked **Ready to cutover**.

1. When you're ready to complete the database migration, select **Start Cutover**.
2. Wait until the **Pending changes** counter shows **0** to ensure that all incoming transactions to the source database are stopped, select the **Confirm** checkbox, and then select **Apply**.

The screenshot shows the 'Complete cutover' dialog box for the dvrental database. It contains a summary of migration status and a list of steps to perform the cutover. The 'Pending changes' counter is shown as 0. A checkbox labeled 'Confirm' is checked and highlighted with a red box. An 'Apply' button is also present.

3. When the database migration status shows **Completed**, connect your applications to the new target instance of Azure Database for PostgreSQL.

Next steps

- For information about known issues and limitations when performing online migrations to Azure Database for PostgreSQL, see the [Known issues and limitations](#) section.

PostgreSQL, see the article [Known issues and workarounds with Azure Database for PostgreSQL online migrations](#).

- For information about the Azure Database Migration Service, see the article [What is the Azure Database Migration Service?](#).
- For information about Azure Database for PostgreSQL, see the article [What is Azure Database for PostgreSQL?](#).

Tutorial: Migrate PostgreSQL to Azure DB for PostgreSQL online using DMS via the Azure CLI

2/19/2020 • 12 minutes to read • [Edit Online](#)

You can use Azure Database Migration Service to migrate the databases from an on-premises PostgreSQL instance to [Azure Database for PostgreSQL](#) with minimal downtime. In other words, migration can be achieved with minimal downtime to the application. In this tutorial, you migrate the **DVD Rental** sample database from an on-premises instance of PostgreSQL 9.6 to Azure Database for PostgreSQL by using the online migration activity in Azure Database Migration Service.

In this tutorial, you learn how to:

- Migrate the sample schema using pg_dump utility.
- Create an instance of the Azure Database Migration Service.
- Create a migration project by using the Azure Database Migration Service.
- Run the migration.
- Monitor the migration.

NOTE

Using Azure Database Migration Service to perform an online migration requires creating an instance based on the Premium pricing tier.

IMPORTANT

For an optimal migration experience, Microsoft recommends creating an instance of Azure Database Migration Service in the same Azure region as the target database. Moving data across regions or geographies can slow down the migration process and introduce errors.

Prerequisites

To complete this tutorial, you need to:

- Download and install [PostgreSQL community edition](#) 9.5, 9.6, or 10. The source PostgreSQL Server version must be 9.5.11, 9.6.7, 10, or later. For more information, see the article [Supported PostgreSQL Database Versions](#).

In addition, the on-premises PostgreSQL version must match the Azure Database for PostgreSQL version. For example, PostgreSQL 9.5.11.5 can only migrate to Azure Database for PostgreSQL 9.5.11 and not to version 9.6.7.

- [Create an instance in Azure Database for PostgreSQL](#) or [Create an Azure Database for PostgreSQL - Hyperscale \(Citus\) server](#).
- Create a Microsoft Azure Virtual Network for Azure Database Migration Service by using the Azure Resource Manager deployment model, which provides site-to-site connectivity to your on-premises source servers by using either [ExpressRoute](#) or [VPN](#). For more information about creating a virtual network, see the [Virtual Network Documentation](#), and especially the quickstart articles with step-by-step details.

NOTE

During virtual network setup, if you use ExpressRoute with network peering to Microsoft, add the following service endpoints to the subnet in which the service will be provisioned:

- Target database endpoint (for example, SQL endpoint, Cosmos DB endpoint, and so on)
- Storage endpoint
- Service bus endpoint

This configuration is necessary because Azure Database Migration Service lacks internet connectivity.

- Ensure that your virtual network Network Security Group (NSG) rules don't block the following inbound communication ports to Azure Database Migration Service: 443, 53, 9354, 445, 12000. For more detail on virtual network NSG traffic filtering, see the article [Filter network traffic with network security groups](#).
- Configure your [Windows Firewall for database engine access](#).
- Open your Windows firewall to allow Azure Database Migration Service to access the source PostgreSQL Server, which by default is TCP port 5432.
- When using a firewall appliance in front of your source database(s), you may need to add firewall rules to allow the Azure Database Migration Service to access the source database(s) for migration.
- Create a server-level [firewall rule](#) for Azure Database for PostgreSQL to allow Azure Database Migration Service to access to the target databases. Provide the subnet range of the virtual network used for Azure Database Migration Service.
- There are two methods for invoking the CLI:
 - In the upper-right corner of the Azure portal, select the Cloud Shell button:



- Install and run the CLI locally. CLI 2.0 is the command-line tool for managing Azure resources.

To download the CLI, follow the instructions in the article [Install Azure CLI 2.0](#). The article also lists the platforms that support CLI 2.0.

To set up Windows Subsystem for Linux (WSL), follow the instructions in the [Windows 10 Installation Guide](#)

- Enable logical replication in the postgresql.config file, and set the following parameters:
 - wal_level = **logical**
 - max_replication_slots = [number of slots], recommend setting to **five slots**
 - max_wal_senders =[number of concurrent tasks] - The max_wal_senders parameter sets the number of concurrent tasks that can run, recommend setting to **10 tasks**

Migrate the sample schema

To complete all the database objects like table schemas, indexes and stored procedures, we need to extract schema from the source database and apply to the database.

1. Use pg_dump -s command to create a schema dump file for a database.

```
pg_dump -o -h hostname -U db_username -d db_name -s > your_schema.sql
```

For example, to dump a schema file dvdrental database:

```
pg_dump -o -h localhost -U postgres -d dvdrental -s > dvdrentalSchema.sql
```

For more information about using the pg_dump utility, see the examples in the [pg-dump](#) tutorial.

2. Create an empty database in your target environment, which is Azure Database for PostgreSQL.

For details on how to connect and create a database, see the article [Create an Azure Database for PostgreSQL server in the Azure portal](#) or [Create an Azure Database for PostgreSQL - Hyperscale \(Citus\) server in the Azure portal](#).

3. Import the schema into the target database you created by restoring the schema dump file.

```
psql -h hostname -U db_username -d db_name < your_schema.sql
```

For example:

```
psql -h mypgserver-20170401.postgres.database.azure.com -U postgres -d dvdrental < dvdrentalSchema.sql
```

4. If you have foreign keys in your schema, the initial load and continuous sync of the migration will fail.

Execute the following script in PgAdmin or in psql to extract the drop foreign key script and add foreign key script at the destination (Azure Database for PostgreSQL).

```
SELECT Queries.tablename
      ,concat('alter table ', Queries.tablename, ' ', STRING_AGG(concat('DROP CONSTRAINT ',
      Queries.foreignkey), ',')) as DropQuery
      ,concat('alter table ', Queries.tablename, ' ',
      STRING_AGG(concat('ADD CONSTRAINT ', Queries.foreignkey, '
      FOREIGN KEY (', column_name, ')', 'REFERENCES ', foreign_table_name, '(', foreign_column_name, ')'
      ), ',')) as AddQuery
   FROM
  (SELECT
    tc.table_schema,
    tc.constraint_name as foreignkey,
    tc.table_name as tableName,
    kcu.column_name,
    ccu.table_schema AS foreign_table_schema,
    ccu.table_name AS foreign_table_name,
    ccu.column_name AS foreign_column_name
   FROM
    information_schema.table_constraints AS tc
   JOIN information_schema.key_column_usage AS kcu
     ON tc.constraint_name = kcu.constraint_name
    AND tc.table_schema = kcu.table_schema
   JOIN information_schema.constraint_column_usage AS ccu
     ON ccu.constraint_name = tc.constraint_name
    AND ccu.table_schema = tc.table_schema
  WHERE constraint_type = 'FOREIGN KEY') Queries
 GROUP BY Queries.tablename;
```

Run the drop foreign key (which is the second column) in the query result.

5. Triggers in the data (insert or update triggers) will enforce data integrity in the target ahead of the replicated data from the source. It's recommended that you disable triggers in all the tables **at the target** during migration and then re-enable the triggers after migration is complete.

To disable triggers in target database, use the following command:

```
select concat ('alter table ', event_object_table, ' disable trigger ', trigger_name)
from information_schema.triggers;
```

6. If there are ENUM data type in any tables, it's recommended that you temporarily update it to a 'character varying' datatype in the target table. After data replication is done, revert the datatype to ENUM.

Provisioning an instance of DMS using the CLI

1. Install the dms sync extension:

- Sign in to Azure by running the following command:

```
az login
```

- When prompted, open a web browser and enter a code to authenticate your device. Follow the instructions as listed.
- Add the dms extension:

- To list the available extensions, run the following command:

```
az extension list-available -otable
```

- To install the extension, run the following command:

```
az extension add -n dms-preview
```

- To verify you have dms extension installed correct, run the following command:

```
az extension list -otable
```

You should see the following output:

ExtensionType	Name
whl	dms

IMPORTANT

Make sure that your extension version is above 0.11.0.

- At any time, view all commands supported in DMS by running:

```
az dms -h
```

- If you have multiple Azure subscriptions, run the following command to set the subscription that you want to use to provision an instance of the DMS service.

```
az account set -s 97181df2-909d-420b-ab93-1bff15acb6b7
```

2. Provision an instance of DMS by running the following command:

```
az dms create -l [location] -n <newServiceName> -g <yourResourceGroupName> --sku-name Premium_4vCores --subnet/subscriptions/{vnet subscription id}/resourceGroups/{vnet resource group}/providers/Microsoft.Network/virtualNetworks/{vnet name}/subnets/{subnet name} -tags tagName1=tagValue1 tagWithNoValue
```

For example the following command will create a service in:

- Location: East US2
- Subscription: 97181df2-909d-420b-ab93-1bff15acb6b7
- Resource Group Name: PostgresDemo
- DMS Service Name: PostgresCLI

```
az dms create -l eastus2 -g PostgresDemo -n PostgresCLI --subnet /subscriptions/97181df2-909d-420b-ab93-1bff15acb6b7/resourceGroups/ERNetwork/providers/Microsoft.Network/virtualNetworks/AzureDMS-CORP-USC-VNET-5044/subnets/Subnet-1 --sku-name Premium_4vCores
```

It takes about 10-12 minutes to create the instance of the DMS service.

3. To identify the IP address of the DMS agent so that you can add it to the Postgres pg_hba.conf file, run the following command:

```
az network nic list -g <ResourceGroupName> --query '[].ipConfigurations | [].privateIpAddress'
```

For example:

```
az network nic list -g PostgresDemo --query '[].ipConfigurations | [].privateIpAddress'
```

You should get a result similar to the following address:

```
[  
    "172.16.136.18"  
]
```

4. Add the IP address of the DMS agent to the Postgres pg_hba.conf file.

- Take note of the DMS IP address after you finish provisioning in DMS.
- Add the IP address to pg_hba.conf file on the source, similar to the following entry:

```
host all all 172.16.136.18/10 md5  
host replication postgres 172.16.136.18/10 md5
```

5. Next, create a PostgreSQL migration project by running the following command:

```
az dms project create -l <location> -g <ResourceGroupName> --service-name <yourServiceName> --source-platform PostgreSQL --target-platform AzureDbforPostgreSQL -n <newProjectName>
```

For example, the following command creates a project using these parameters:

- Location: West Central US
- Resource Group Name: PostgresDemo

- Service Name: PostgresCLI
- Project name: PGMigration
- Source platform: PostgreSQL
- Target platform: AzureDbForPostgreSql

```
az dms project create -l westcentralus -n PGMigration -g PostgresDemo --service-name PostgresCLI
--source-platform PostgreSQL --target-platform AzureDbForPostgreSql
```

6. Create a PostgreSQL migration task using the following steps.

This step includes using the source IP, UserID and password, destination IP, UserID, password, and task type to establish connectivity.

- To see a full list of options, run the command:

```
az dms project task create -h
```

For both source and target connection, the input parameter is referring to a json file that has the object list.

The format of the connection JSON object for PostgreSQL connections.

```
{
    "userName: "user name", // if this is missing or null, you will be prompted
    "password": null, // if this is missing or null (highly recommended) you
will
be prompted
    "serverName": "server name",
    "databaseName": "database name", // if this is missing, it will default to the
'postgres'
    server
    "port": 5432 // if this is missing, it will default to 5432
}
```

- There's also a database option json file that lists the json objects. For PostgreSQL, the format of the database options JSON object is shown below:

```
[
{
    "name": "source database",
    "target_database_name": "target database",
},
...
]
```

- Create a json file with Notepad, copy the following commands and paste them into the file, and then save the file in C:\DMS\source.json.

```
{
    "userName": "postgres",
    "password": null,
    be prompted
    "serverName": "13.51.14.222",
    "databaseName": "dvdrental",
    "port": 5432
}
```

- Create another file named target.json and save as C:\DMS\target.json. Include the following commands:

```
{
    "userName": " dms@builddemotarget",
    "password": null,
    "serverName": " builddemotarget.postgres.database.azure.com",
    "databaseName": "inventory",
    "port": 5432
}
```

- Create a database options json file that lists inventory as the database to migrate:

```
[
{
    "name": "dvdrental",
    "target_database_name": "dvdrental",
}
]
```

- Run the following command, which takes in the source, destination, and the DB option json files.

```
az dms project task create -g PostgresDemo --project-name PGMigration --source-platform
postgresql --target-platform azuredbforpostgresql --source-connection-json c:\DMS\source.json --
database-options-json C:\DMS\option.json --service-name PostgresCLI --target-connection-json
c:\DMS\target.json -task-type OnlineMigration -n runnowtask
```

At this point, you've successfully submitted a migration task.

7. To show progress of the task, run the following command:

```
az dms project task show --service-name PostgresCLI --project-name PGMigration --resource-group
PostgresDemo --name Runnowtask
```

OR

```
az dms project task show --service-name PostgresCLI --project-name PGMigration --resource-group
PostgresDemo --name Runnowtask --expand output
```

8. You can also query for the migrationState from the expand output:

```
az dms project task show --service-name PostgresCLI --project-name PGMigration --resource-group
PostgresDemo --name Runnowtask --expand output --query 'properties.output[].migrationState | [0]'

"READY_TO_COMPLETE"
```

Understanding migration task status

In the output file, there are several parameters that indicate progress of migration. For example, see the output file below:

```
"output": [
    {
        "Database Level": [
            {
                "appliedChanges": 0, //Total incremental sync applied after full load
                "cdcDeleteCounter": 0 // Total delete operation applied after full load
                "cdcInsertCounter": 0, // Total insert operation applied after full load
                "cdcUpdateCounter": 0, // Total update operation applied after full load
                "databaseName": "inventory",
                "endedOn": null,
                "fullLoadCompletedTables": 2, //Number of tables completed full load
                "fullLoadErroredTables": 0, //Number of tables that contain migration error
                "fullLoadLoadingTables": 0, //Number of tables that are in loading status
                "fullLoadQueuedTables": 0, //Number of tables that are in queued status
                "id": "db|inventory",
                "incomingChanges": 0, //Number of changes after full load
                "initializationCompleted": true,
                "latency": 0,
                "migrationState": "READY_TO_COMPLETE", //Status of migration task. READY_TO_COMPLETE means the database is ready for cutover
                "resultType": "DatabaseLevelOutput",
                "startedOn": "2018-07-05T23:36:02.27839+00:00"
            },
            {
                "databaseCount": 1,
                "endedOn": null,
                "id": "dd27aa3a-ed71-4bff-ab34-77db4261101c",
                "resultType": "MigrationLevelOutput",
                "sourceServer": "138.91.123.10",
                "sourceVersion": "PostgreSQL",
                "startedOn": "2018-07-05T23:36:02.27839+00:00",
                "state": "PENDING",
                "targetServer": "builddemotarget.postgres.database.azure.com",
                "targetVersion": "Azure Database for PostgreSQL"
            },
            {
                "Table 1": [
                    {
                        "cdcDeleteCounter": 0,
                        "cdcInsertCounter": 0,
                        "cdcUpdateCounter": 0,
                        "dataErrorsCount": 0,
                        "databaseName": "inventory",
                        "fullLoadEndedOn": "2018-07-05T23:36:20.740701+00:00", //Full load completed time
                        "fullLoadEstFinishTime": "1970-01-01T00:00:00+00:00",
                        "fullLoadStartedOn": "2018-07-05T23:36:15.864552+00:00", //Full load started time
                        "fullLoadTotalRows": 10, //Number of rows loaded in full load
                        "fullLoadTotalVolumeBytes": 7056, //Volume in Bytes in full load
                        "id": "or|inventory|public|actor",
                        "lastModifiedTime": "2018-07-05T23:36:16.880174+00:00",
                        "resultType": "TableLevelOutput",
                        "state": "COMPLETED", //State of migration for this table
                        "tableName": "public.catalog", //Table name
                        "totalChangesApplied": 0 //Total sync changes that applied after full load
                    },
                    {
                        "Table 2": [
                            {
                                "cdcDeleteCounter": 0,
                                "cdcInsertCounter": 50,
                                "cdcUpdateCounter": 0,
                                "dataErrorsCount": 0,
                                "databaseName": "inventory",
                                "fullLoadEndedOn": "2018-07-05T23:36:23.963138+00:00",
                                "fullLoadEstFinishTime": "1970-01-01T00:00:00+00:00",
                                "fullLoadStartedOn": "2018-07-05T23:36:19.302013+00:00",
                                "fullLoadTotalRows": 112,
                                "fullLoadTotalVolumeBytes": 46592,
                            }
                        ]
                    }
                ]
            }
        ]
    }
]
```

```

        "id": "or|inventory|public|address",
        "lastModifiedTime": "2018-07-05T23:36:20.308646+00:00",
        "resultType": "TableLevelOutput",
        "state": "COMPLETED",
        "tableName": "public.orders",
        "totalChangesApplied": 0
    }
],
    DMS migration task state
"state": "Running", //Migration task state - Running means it is still listening to any changes that
might come in
    "taskType": null
},
"resourceGroup": "PostgresDemo",
"type": "Microsoft.DataMigration/services/projects/tasks"

```

Cutover migration task

The database is ready for cutover when full load is complete. Depending on how busy the source server is with new transactions is coming in, the DMS task might be still applying changes after the full load is complete.

To ensure all data is caught up, validate row counts between the source and target databases. For example, you can use the following command:

```

"migrationState": "READY_TO_COMPLETE", //Status of migration task. READY_TO_COMPLETE means database is ready
for cutover
"incomingChanges": 0, //continue to check for a period of 5-10 minutes to make sure no new incoming changes
that need to be applied to the target server
"fullLoadTotalRows": 10, //full load for table 1
"cdcDeleteCounter": 0, //delete, insert and update counter on incremental sync after full load
"cdcInsertCounter": 50,
"cdcUpdateCounter": 0,
"fullLoadTotalRows": 112, //full load for table 2

```

1. Perform the cutover database migration task by using the following command:

```
az dms project task cutover -h
```

For example:

```
az dms project task cutover --service-name PostgresCLI --project-name PGMigration --resource-group
PostgresDemo --name Runnowtask --object-name Inventory
```

2. To monitor the cutover progress, run the following command:

```
az dms project task show --service-name PostgresCLI --project-name PGMigration --resource-group
PostgresDemo --name Runnowtask
```

Service, project, task cleanup

If you need to cancel or delete any DMS task, project, or service, perform the cancellation in the following sequence:

- Cancel any running task
- Delete the task
- Delete the project
- Delete DMS service

1. To cancel a running task, use the following command:

```
az dms project task cancel --service-name PostgresCLI --project-name PGMigration --resource-group PostgresDemo --name Runnowtask
```

2. To delete a running task, use the following command:

```
az dms project task delete --service-name PostgresCLI --project-name PGMigration --resource-group PostgresDemo --name Runnowtask
```

3. To cancel a running project, use the following command:

```
az dms project task cancel -n runnowtask --project-name PGMigration -g PostgresDemo --service-name PostgresCLI
```

4. To delete a running project, use the following command:

```
az dms project task delete -n runnowtask --project-name PGMigration -g PostgresDemo --service-name PostgresCLI
```

5. To delete DMS service, use the following command:

```
az dms delete -g ProgresDemo -n PostgresCLI
```

Next steps

- For information about known issues and limitations when performing online migrations to Azure Database for PostgreSQL, see the article [Known issues and workarounds with Azure Database for PostgreSQL online migrations](#).
- For information about the Azure Database Migration Service, see the article [What is the Azure Database Migration Service?](#).
- For information about Azure Database for PostgreSQL, see the article [What is Azure Database for PostgreSQL?](#).

Tutorial: Migrate RDS PostgreSQL to Azure DB for PostgreSQL online using DMS

2/19/2020 • 9 minutes to read • [Edit Online](#)

You can use Azure Database Migration Service to migrate databases from an RDS PostgreSQL instance to [Azure Database for PostgreSQL](#) while the source database remains online during migration. In other words, migration can be achieved with minimal downtime to the application. In this tutorial, you migrate the **DVD Rental** sample database from an instance of RDS PostgreSQL 9.6 to Azure Database for PostgreSQL by using the online migration activity in Azure Database Migration Service.

In this tutorial, you learn how to:

- Migrate the sample schema by using the pg_dump utility.
- Create an instance of Azure Database Migration Service.
- Create a migration project by using Azure Database Migration Service.
- Run the migration.
- Monitor the migration.
- Perform migration cutover.

NOTE

Using the Azure Database Migration Service to perform an online migration requires creating an instance based on the Premium pricing tier. For more information, see the Azure Database Migration Service [pricing](#) page.

IMPORTANT

For an optimal migration experience, Microsoft recommends creating an instance of the Azure Database Migration Service in the same Azure region as the target database. Moving data across regions or geographies can slow down the migration process and introduce errors.

TIP

When you migrate databases to Azure by using Azure Database Migration Service, you can do an *offline* or an *online* migration. With an offline migration, application downtime starts when the migration starts. With an online migration, downtime is limited to the time to cut over at the end of migration. We suggest that you test an offline migration to determine whether the downtime is acceptable; if not, do an online migration.

This article describes how to perform an online migration from an on-premises instance of PostgreSQL to Azure Database for PostgreSQL.

Prerequisites

To complete this tutorial, you need to:

- Download and install [PostgreSQL community edition](#) 9.5, 9.6, or 10. The source PostgreSQL Server version must be 9.5.11, 9.6.7, 10, or later. For more information, see the article [Supported PostgreSQL Database Versions](#).

In addition, the RDS PostgreSQL version must match the Azure Database for PostgreSQL version. For example, RDS PostgreSQL 9.5.11.5 can only migrate to Azure Database for PostgreSQL 9.5.11 and not to version 9.6.7.

- Create an instance of [Azure Database for PostgreSQL](#) or [Azure Database for PostgreSQL - Hyperscale \(Citus\)](#). Refer to this [section](#) of the document for detail on how to connect to the PostgreSQL Server using pgAdmin.
- Create a Microsoft Azure Virtual Network for Azure Database Migration Service by using the Azure Resource Manager deployment model, which provides site-to-site connectivity to your on-premises source servers by using either [ExpressRoute](#) or [VPN](#). For more information about creating a virtual network, see the [Virtual Network Documentation](#), and especially the quickstart articles with step-by-step details.
- Ensure that your virtual network Network Security Group rules don't block the following inbound communication ports to Azure Database Migration Service: 443, 53, 9354, 445, and 12000. For more detail on virtual network NSG traffic filtering, see the article [Filter network traffic with network security groups](#).
- Configure your [Windows Firewall for database engine access](#).
- Open your Windows firewall to allow Azure Database Migration Service to access the source PostgreSQL server, which by default is TCP port 5432.
- When using a firewall appliance in front of your source database(s), you may need to add firewall rules to allow the Azure Database Migration Service to access the source database(s) for migration.
- Create a server-level [firewall rule](#) for the Azure Database for PostgreSQL server to allow Azure Database Migration Service access to the target databases. Provide the subnet range of the virtual network used for Azure Database Migration Service.

Set up AWS RDS PostgreSQL for replication

1. To create a new parameter group, follow the instructions provided by AWS in the article [Working with DB Parameter Groups](#).
2. Use the master user name to connect to the source from Azure Database Migration Service. If you use an account other than the master user account, the account must have the rds_superuser role and the rds_replication role. The rds_replication role grants permissions to manage logical slots and to stream data using logical slots.
3. Create a new parameter group with the following configuration:
 - a. Set the rds.logical_replication parameter in your DB parameter group to 1.
 - b. max_wal_senders =[number of concurrent tasks] - The max_wal_senders parameter sets the number of concurrent tasks that can run, recommend 10 tasks.
 - c. max_replication_slots – = [number of slots], recommend set to five slots.
4. Associate the parameter group you created to the RDS PostgreSQL instance.

Migrate the schema

1. Extract the schema from the source database and apply to the target database to complete migration of all database objects such as table schemas, indexes, and stored procedures.

The easiest way to migrate only the schema is to use pg_dump with the -s option. For more information, see the [examples](#) in the Postgres pg_dump tutorial.

```
pg_dump -o -h hostname -U db_username -d db_name -s > your_schema.sql
```

For example, to dump a schema file for the **dvdrental** database, use the following command:

```
pg_dump -o -h localhost -U postgres -d dvdrental -s > dvdrentalSchema.sql
```

2. Create an empty database in the target service, which is Azure Database for PostgreSQL. To connect and create a database, refer to one of the following articles:
 - [Create an Azure Database for PostgreSQL server by using the Azure portal](#)
 - [Create an Azure Database for PostgreSQL - Hyperscale \(Citus\) server using the Azure portal](#)
3. Import the schema to target service, which is Azure Database for PostgreSQL. To restore the schema dump file, run the following command:

```
psql -h hostname -U db_username -d db_name < your_schema.sql
```

For example:

```
psql -h mypgserver-20170401.postgres.database.azure.com -U postgres -d dvdrental < dvdrentalSchema.sql
```

4. If you have foreign keys in your schema, the initial load and continuous sync of the migration will fail. To extract the drop foreign key script and add foreign key script at the destination (Azure Database for PostgreSQL), run the following script in PgAdmin or in psql:

```
SELECT Queries.tablename
      ,concat('alter table ', Queries.tablename, ' ', STRING_AGG(concat('DROP CONSTRAINT ',
      Queries.foreignkey), ',')) as DropQuery
      ,concat('alter table ', Queries.tablename, ' ',
      STRING_AGG(concat('ADD CONSTRAINT ', Queries.foreignkey,
      'FOREIGN KEY (', column_name, ')', 'REFERENCES ', foreign_table_name, '(', foreign_column_name, ')'),
      ',')) as AddQuery
   FROM
  (SELECT
    tc.table_schema,
    tc.constraint_name as foreignkey,
    tc.table_name as tableName,
    kcu.column_name,
    ccu.table_schema AS foreign_table_schema,
    ccu.table_name AS foreign_table_name,
    ccu.column_name AS foreign_column_name
   FROM
    information_schema.table_constraints AS tc
   JOIN information_schema.key_column_usage AS kcu
     ON tc.constraint_name = kcu.constraint_name
     AND tc.table_schema = kcu.table_schema
   JOIN information_schema.constraint_column_usage AS ccu
     ON ccu.constraint_name = tc.constraint_name
     AND ccu.table_schema = tc.table_schema
  WHERE constraint_type = 'FOREIGN KEY') Queries
 GROUP BY Queries.tablename;
```

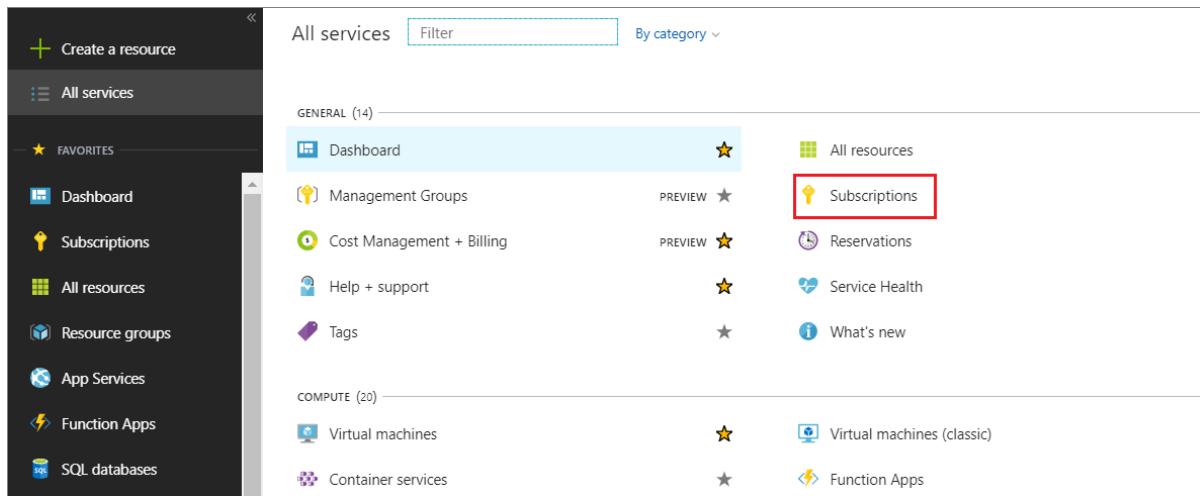
5. Run the drop foreign key (which is the second column) in the query result to drop the foreign key.
6. If you have triggers (insert or update trigger) in the data, it will enforce data integrity in the target before replicating data from the source. The recommendation is to disable triggers in all the tables *at the target* during migration, and then enable the triggers after migration is complete.

To disable triggers in target database:

```
SELECT Concat('DROP TRIGGER ', Trigger_Name, ';') FROM information_schema.TRIGGERS WHERE TRIGGER_SCHEMA = 'your_schema';
```

Register the Microsoft.DataMigration resource provider

1. Sign in to the Azure portal, select **All services**, and then select **Subscriptions**.



The screenshot shows the Azure portal's 'All services' blade. On the left is a navigation menu with items like 'Create a resource', 'All services', 'Favorites', 'Dashboard', 'Subscriptions', 'All resources', 'Resource groups', 'App Services', 'Function Apps', and 'SQL databases'. The 'Subscriptions' item is highlighted with a red box. The main area lists services under 'GENERAL' and 'COMPUTE' categories. In the 'GENERAL' section, 'Dashboard' is selected. In the 'COMPUTE' section, 'Virtual machines' and 'Container services' are listed.

2. Select the subscription in which you want to create the instance of Azure Database Migration Service, and then select **Resource providers**.

The screenshot shows the Azure portal interface. On the left, there's a sidebar with various service icons like App Services, SQL databases, and Storage accounts. The main area is titled 'Subscriptions' and shows a list of subscriptions. A search bar at the top right says 'Search (Ctrl+ /)'. On the right, a vertical navigation menu is open under the heading '<Subscription>'. The menu items include Overview, Access control (IAM), Diagnose and solve problems, COST MANAGEMENT + BILLING, Partner information, SETTINGS, Programmatic deployment, Resource groups, Resources, Usage + quotas, Policies, Management certificates, My permissions, and Resource providers. The 'Resource providers' item is highlighted with a red box.

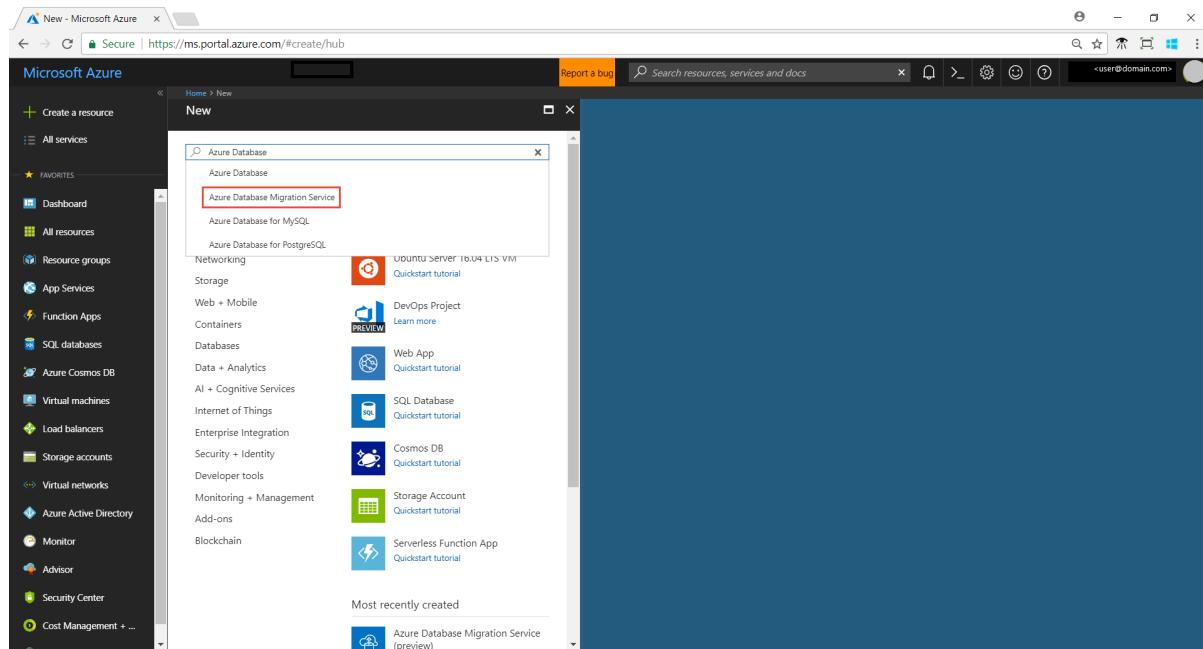
3. Search for migration, and then to the right of **Microsoft.DataMigration**, select **Register**.

This screenshot shows the 'Resource providers' page within the Azure portal. The left side has the same navigation and sidebar as the previous screenshot. The main area shows a table with one row for 'migration'. The 'PROVIDER' column contains 'Microsoft.DataMigration' (highlighted in yellow). The 'STATUS' column shows 'NotRegistered'. To the right of the status, there's a button labeled 'Register' with a red box around it.

Create an instance of Azure Database Migration Service

1. In the Azure portal, select + **Create a resource**, search for Azure Database Migration Service, and then

select **Azure Database Migration Service** from the drop-down list.



2. On the **Azure Database Migration Service** screen, select **Create**.

3. On the **Create Migration Service** screen, specify a name for the service, the subscription, and a new or

existing resource group.

4. Select the location in which you want to create the instance of Azure Database Migration Service.
5. Select an existing virtual network or create a new one.

The virtual network provides Azure Database Migration Service with access to the source PostgreSQL instance and the target Azure Database for PostgreSQL instance.

For more information about how to create a virtual network in the Azure portal, see the article [Create a virtual network using the Azure portal](#).

6. Select a pricing tier; for this online migration, be sure to select the Premium: 4vCores pricing tier.

Create Migration Service

[Basics](#) [Networking](#) [Tags](#) [Review + create](#)

Azure Database Migration Service is designed to streamline the process of migrating on-premises databases to Azure. [Learn more.](#)

Project details

Select the subscription to manage deployed resources and constants. Use resource groups as you would folders, to organize and manage all of your resources.

Subscription * ⓘ DMSInternalPreviewtest

Resource group * ⓘ Select existing... [Create new](#)

Instance details

Migration service name * ⓘ Enter service name

Location * ⓘ Brazil South

Service mode * ⓘ [Azure](#) [Hybrid \(Preview\)](#)

Pricing tier * Standard
1 vCores [Configure tier](#)

Tip: Use an Azure Database Migration Service quick start template with pre-created source and targets. [Learn more.](#)

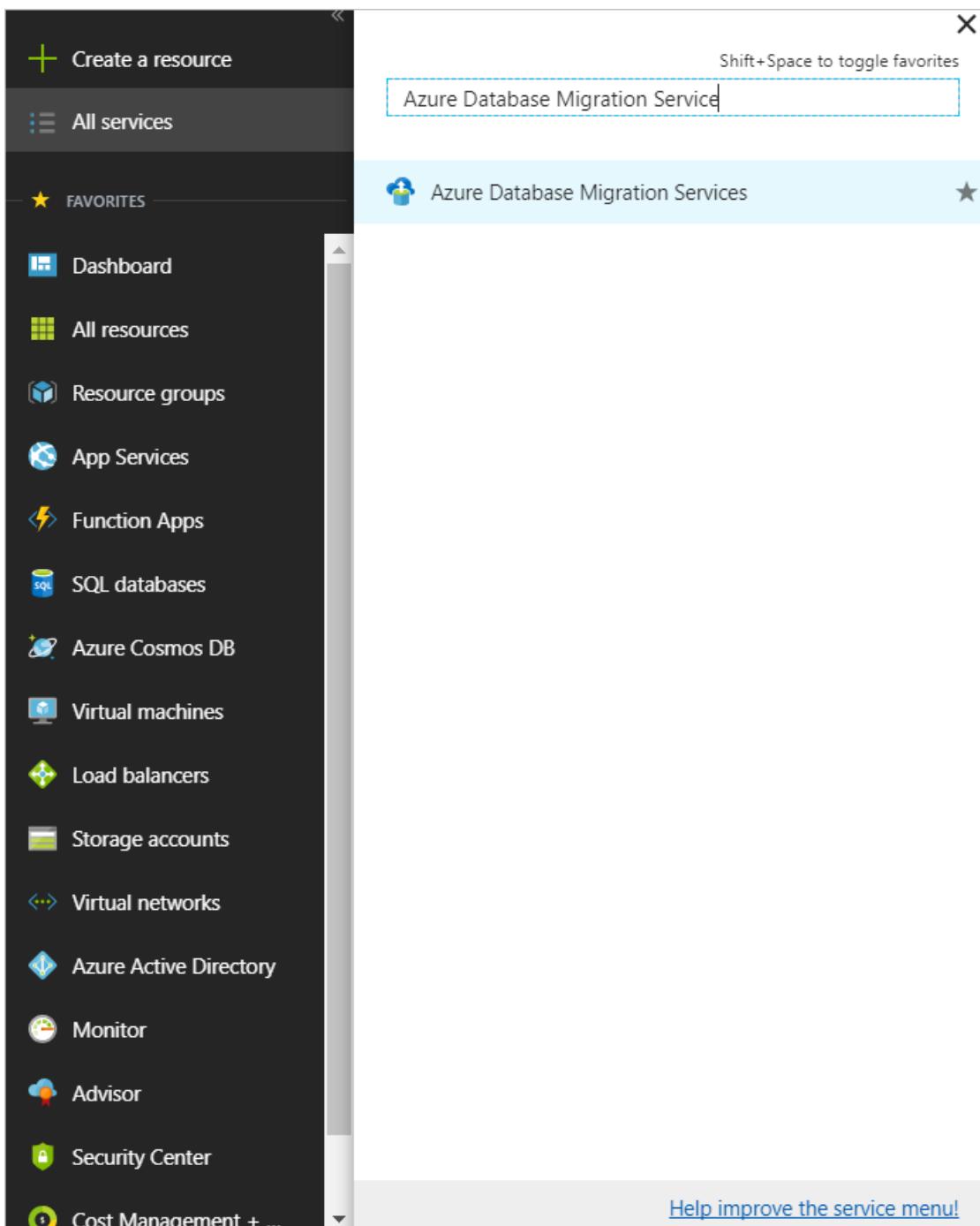
[Review + create](#) [Next : Networking >>](#)

7. Select **Create** to create the service.

Create a migration project

After the service is created, locate it within the Azure portal, open it, and then create a new migration project.

1. In the Azure portal, select **All services**, search for Azure Database Migration Service, and then select **Azure Database Migration Services**.



2. On the **Azure Database Migration Services** screen, search for the name of the Azure Database Migration Service instance that you created, select the instance, and then select + **New Migration Project**.
3. On the **New migration project** screen, specify a name for the project, in the **Source server type** text box, select **AWS RDS for PostgreSQL**, and then in the **Target server type** text box, select **Azure Database for PostgreSQL**.
4. In the **Choose type of activity** section, select **Online data migration**.

IMPORTANT

Be sure to select **Online data migration**; offline migrations are not supported for this scenario.

<p>Project name RDSPGtoAzPG</p> <p>* Source server type AWS RDS for PostgreSQL</p> <p>* Target server type Azure Database for PostgreSQL</p> <p>* Choose type of activity Online data migration ></p> <p>To successfully use Database Migration Service (DMS) to migrate data, you need to:</p> <ol style="list-style-type: none"> 1. Migrate schema using pg_dump -o -h hostname -U db_username -d db_name -> your_schema.sql 2. Remove foreign keys in schema at target Azure Database for PostgreSQL 3. Disable triggers at target Azure Database for PostgreSQL 4. Provision Database Migration Service and create a migration task <p>Please refer to this tutorial for more details.</p>	<p>Choose type of activity Online data migration</p> <p>Use this option to migrate databases that must be accessible and continuously updated during migration.</p> <p>To continuously replicate data changes from your source to your target, please implement the steps below on your source server. These steps can be implemented anytime between the moment you create a project and the moment you start a migration activity. After your migration is complete, you will reverse those preparation steps.</p> <ul style="list-style-type: none"> • Azure Database for PostgreSQL supports community edition and the source PostgreSQL server must match the same major version that Azure Database for PostgreSQL supports. For example, upgrading from PostgreSQL 9.5.x to 9.6.x is not supported. • Enable logical replication in the postgresql.conf file.
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

NOTE

Alternately, you can choose **Create project only** to create the migration project now and execute the migration later.

5. Select **Save**.

6. Select **Create and run activity** to create the project and run the migration activity.

NOTE

Please make a note of the pre-requisites needed to set up online migration in the project creation blade.

Specify source details

- On the **Add Source Details** screen, specify the connection details for the source PostgreSQL instance.

Migration Wizard		X	Add Source Details		□ X
RDSPGToAzPG					
1	Select source	>	Source server name <input type="text" value="Enter source server name"/>		
2	Select target	>	Server port <input type="text" value="5432"/> ✓		
3	Select databases	>	Database <input type="text" value="postgres"/>		
4	Configure migration settings	>	User Name <input type="text" value="Enter user name"/> Password <input type="text" value="Enter password"/>		
5	Summary	>	<input type="checkbox"/> Trust server certificate <input checked="" type="checkbox"/> Encrypt connection		
<input type="button" value="Save"/>					

Specify target details

1. Select **Save**, and then on the **Target details** screen, specify the connection details for the target Azure Database for PostgreSQL server, which is pre-provisioned and has the **DVD Rentals** schema deployed using pg_dump.

Migration Wizard		X	Target details		□ X
RDSPGToAzPG			Subscription <input type="text" value="DMSInternalPreviewtest"/>		
1	Select source	✓	Azure PostgreSQL <input type="text"/>		
2	Select target	>	Database <input type="text" value="postgres"/>		
3	Select databases	>	User Name <input type="text" value="Enter user name"/>		
4	Configure migration settings	>	Password <input type="text" value="Enter password"/>		
5	Summary	>			
<input type="button" value="Save"/>					

2. Select **Save**, and then on the **Map to target databases** screen, map the source and the target database for migration.

If the target database contains the same database name as the source database, Azure Database Migration Service selects the target database by default.

Source Database	Target Database
<input checked="" type="checkbox"/> citus	citus
<input type="checkbox"/> postgres	postgres
<input type="checkbox"/> rdsadmin	
<input type="checkbox"/> test	

Save

3. Select **Save**, on the **Migration summary** screen, in the **Activity name** text box, specify a name for the migration activity, and then review the summary to ensure that the source and target details match what you previously specified.

Activity name
runnow

Target server name
<server> postgres.database.azure.com

Target server version
Azure Database for PostgreSQL
10.5

Source server name
<server> us-east-2.rds.amazonaws.com

Source server version
Amazon RDS for PostgreSQL
10.3

Database(s) to migrate
1 of 2

Run the migration

- Select **Run migration**.

The migration activity window appears, and the **Status** of the activity is **Initializing**.

Monitor the migration

1. On the migration activity screen, select **Refresh** to update the display until the **Status** of the migration shows as **Running**.

runnow		
Refresh Stop migration Delete activity Download report		
Source server 138.91.123.10	Source version Amazon RDS for PostgreSQL 9.6	Source databases 1
Target server bulddemotarget.postgres.database.azure.com	Target version Azure Database for PostgreSQL 9.6	Type of activity Online
Activity status Running		Duration 00:01:01
DATABASE NAME	STATUS	MIGRATION DETAILS
inventory	Running	Ready to cutover
		DURATION
		ESTIMATED APPLICATION DOWNTIME ⓘ
		FINISH DATE

2. Under **DATABASE NAME**, select a specific database to get to the migration status for **Full data load** and **Incremental data sync** operations.

Full data load shows the initial load migration status, while **Incremental data sync** shows change data capture (CDC) status.

Home > StarWars1 > runnow2 > inventory			
inventory			
Refresh Start Cutover			
Source database name	Full load completed	Incremental updates	Pending changes
inventory	2	0	0
Target database name	Full load queued	Incremental inserts	Applied changes
Inventory	0	11	11
Database status	Full load loading	Incremental deletes	Tables in error state ⓘ
Complete	0	0	0
Migration details	Full load failed		
All changes applied	0		
Full load	Incremental data sync		
2 item(s)		prev	Page 1 of 1
			next →
TABLE NAME	STATUS	COMPLETED	ROWS
inventory.catalog	Completed	7/26/2018 2:12:38 PM	9
inventory.orders	Completed	7/26/2018 2:12:41 PM	218
			DURATION
			00:00:02

The screenshot shows the migration progress for the 'inventory' database. Key metrics include:

- Source database name:** inventory
- Target database name:** inventory
- Database status:** Complete
- Migration details:** All changes applied
- Pending changes:** 0
- Applied changes:** 11
- Tables in error state:** 0

The 'Incremental data sync' tab is selected. Below it, a table lists changes applied to two tables:

Table Name	Status	Insert	Update	Delete	Total Applied	Data Errors	Last Modified
inventory.catalog					0	0	7/26/2018 2:12:36 ...
inventory.orders		11			11	0	7/26/2018 2:13:36 ...

Perform migration cutover

After the initial Full load is completed, the databases are marked **Ready to Cutover**.

- When you're ready to complete the database migration, select **Start Cutover**.
- Wait until the **Pending changes** counter shows **0** to ensure that all incoming transactions to the source database are stopped, select the **Confirm** checkbox, and then select **Apply**.

The 'Complete cutover' dialog box contains the following information:

- Migration Statistics:**

Full load completed	Incremental updates	Pending changes
19	0	0
Full load queued	Incremental inserts	Applied changes
0	0	0
Full load loading	Incremental deletes	Tables in error state ⓘ
0	0	0
Full load failed		
0		
- Instructions:** When you are ready to do the migration cutover, perform the following steps to complete the database migration. Please note that the database is ready for cutover only after the full data load is completed.
- Step 1:** Stop all the incoming transactions coming to the source database.
- Step 2:** Wait until all the pending transactions have been applied to the target database. At that time the pending changes counter will set to 0.
- Step 3:** Reconnect your applications to the new Azure target database.
- Pending changes:** 0
- Buttons:** Confirm (checkbox checked) and Apply (button highlighted with a red box).

- When the database migration status shows **Completed**, connect your applications to the new target Azure Database for PostgreSQL database.

Your online migration of an on-premises instance of RDS PostgreSQL to Azure Database for PostgreSQL is now complete.

Next steps

- For information about the Azure Database Migration Service, see the article [What is the Azure Database Migration Service?](#).
- For information about Azure Database for PostgreSQL, see the article [What is Azure Database for PostgreSQL?](#).
- For other questions, email the [Ask Azure Database Migrations](#) alias.

Tutorial: Migrate MongoDB to Azure Cosmos DB's API for MongoDB offline using DMS

1/8/2020 • 7 minutes to read • [Edit Online](#)

You can use Azure Database Migration Service to perform an offline (one-time) migration of databases from an on-premises or cloud instance of MongoDB to Azure Cosmos DB's API for MongoDB.

In this tutorial, you learn how to:

- Create an instance of Azure Database Migration Service.
- Create a migration project by using Azure Database Migration Service.
- Run the migration.
- Monitor the migration.

In this tutorial, you migrate a dataset in MongoDB hosted in an Azure Virtual Machine to Azure Cosmos DB's API for MongoDB by using Azure Database Migration Service. If you don't have a MongoDB source set up already, see the article [Install and configure MongoDB on a Windows VM in Azure](#).

Prerequisites

To complete this tutorial, you need to:

- [Complete the pre-migration](#) steps such as estimating throughput, choosing a partition key, and the indexing policy.
- [Create an Azure Cosmos DB's API for MongoDB account](#).
- Create a Microsoft Azure Virtual Network for Azure Database Migration Service by using Azure Resource Manager deployment model, which provides site-to-site connectivity to your on-premises source servers by using either [ExpressRoute](#) or [VPN](#). For more information about creating a virtual network, see the [Virtual Network Documentation](#), and especially the quickstart articles with step-by-step details.

NOTE

During virtual network setup, if you use ExpressRoute with network peering to Microsoft, add the following service [endpoints](#) to the subnet in which the service will be provisioned:

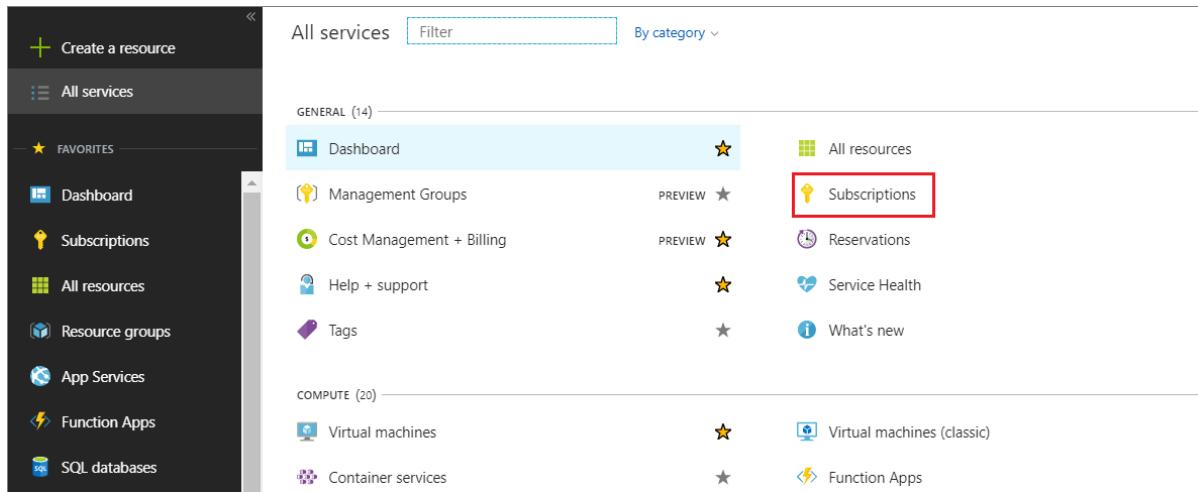
- Target database endpoint (for example, SQL endpoint, Cosmos DB endpoint, and so on)
- Storage endpoint
- Service bus endpoint

This configuration is necessary because Azure Database Migration Service lacks internet connectivity.

- Ensure that your virtual network Network Security Group (NSG) rules don't block the following communication ports: 53, 443, 445, 9354, and 10000-20000. For more detail on virtual network NSG traffic filtering, see the article [Filter network traffic with network security groups](#).
- Open your Windows firewall to allow Azure Database Migration Service to access the source MongoDB server, which by default is TCP port 27017.
- When using a firewall appliance in front of your source database(s), you may need to add firewall rules to allow Azure Database Migration Service to access the source database(s) for migration.

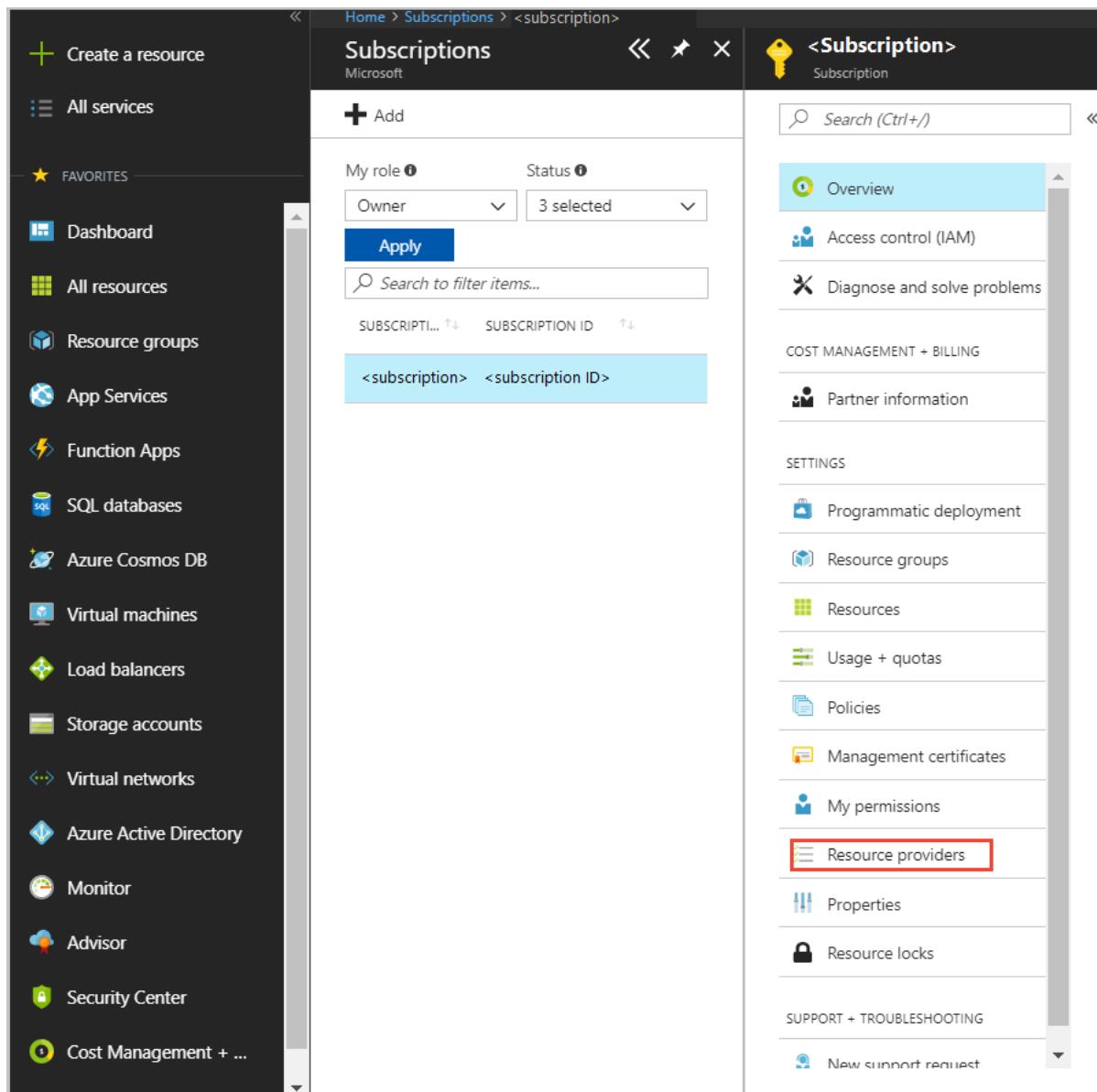
Register the Microsoft.DataMigration resource provider

1. Sign in to the Azure portal, select **All services**, and then select **Subscriptions**.



The screenshot shows the Azure portal's 'All services' blade. On the left is a navigation menu with items like 'Create a resource', 'Dashboard', 'Subscriptions', 'All resources', 'Resource groups', 'App Services', 'Function Apps', and 'SQL databases'. The 'Subscriptions' item is highlighted with a red box. The main area lists various services under 'GENERAL' and 'COMPUTE' categories, with 'Subscriptions' also listed under 'GENERAL'.

2. Select the subscription in which you want to create the instance of the Azure Database Migration Service, and then select **Resource providers**.



The screenshot shows the 'Subscriptions' blade for a specific subscription. The left sidebar has the same navigation menu as the previous screenshot. The main area shows a list of subscriptions with one selected, indicated by a blue highlight. On the right, there is a navigation menu with items like 'Overview', 'Access control (IAM)', 'Diagnose and solve problems', 'COST MANAGEMENT + BILLING', 'Partner information', 'SETTINGS', 'Programmatic deployment', 'Resource groups', 'Resources', 'Usage + quotas', 'Policies', 'Management certificates', 'My permissions', 'Resource providers' (which is highlighted with a red box), 'Properties', 'Resource locks', and 'SUPPORT + TROUBLESHOOTING'.

3. Search for migration, and then to the right of **Microsoft.DataMigration**, select **Register**.

The screenshot shows the Azure portal interface. On the left, there's a sidebar with various service icons like Dashboard, All resources, Resource groups, App Services, etc. The main area has a title bar "Home > Subscriptions > Subscription". Below it, there's a "Subscriptions" section with a "+ Add" button and a dropdown for "My role" set to "Owner". A search bar says "Search to filter items...". To the right, a large panel titled "<Subscription>" shows an "Overview" section with tabs for "Access control (IAM)", "Diagnose and solve problems", "COST MANAGEMENT + BILLING", and "Partner information". Under "SETTINGS", there are sections for "Programmatic deployment", "Resource groups", "Resources", "Usage + quotas", "Policies", "Management certificates", "My permissions", "Resource providers" (which is currently selected), and "Properties". On the far right, a "Resource providers" table lists one entry: "Microsoft.DataMigration" with status "NotRegistered" and a red-bordered "Register" button.

Create an instance

1. In the Azure portal, select **+ Create a resource**, search for Azure Database Migration Service, and then select **Azure Database Migration Service** from the drop-down list.

The screenshot shows the Azure portal "New" screen. The left sidebar is identical to the previous screenshot. The main search bar at the top contains the text "Azure Database". Below it, a dropdown menu is open, showing "Azure Database" and "Azure Database Migration Service" (which is highlighted with a red border). The main content area displays a grid of service tiles, including "Ubuntu Server 16.04 LTS VM", "DevOps Project", "Web App", "SQL Database", "Cosmos DB", "Storage Account", and "Serverless Function App". At the bottom, there's a "Most recently created" section with a single item: "Azure Database Migration Service (preview)".

2. On the **Azure Database Migration Service** screen, select **Create**.

The Azure Database Migration Service (DMS) is designed to streamline the process of migrating on-premises databases to Azure. DMS will simplify the migration of existing on-premises SQL Server and Oracle databases to Azure SQL Database, Azure SQL Managed Instance or Microsoft SQL Server in an Azure Virtual Machine. Learn [more](#)

Before using DMS we recommend you complete the following three steps:

1. Open the [Azure Database Migration Guide](#) for step by step guidance through the migration process
2. Assess your SQL Server on-premises database(s) for feature parity and potential compatibility issues by using [Data Migration Assistant \(DMA\)](#). Alternatively, if you are migrating from Oracle, use the [SQL Server Migration Assistant \(SSMA\)](#)
3. Based on your database needs, create a target database using one of the database services: Azure SQL Database, Azure SQL Managed Instance or SQL Server in an Azure Virtual Machine.

Once your assessments are complete, fixes are applied and schema is deployed, proceed with creating a migration service by clicking **Create** below to migrate the data from your source database to the target.

[Save for later](#)

PUBLISHER	Microsoft
USEFUL LINKS	Documentation Privacy Statement

Create

3. On the **Create Migration Service** screen, specify a name for the service, the subscription, and a new or existing resource group.
4. Select the location in which you want to create the instance of Azure Database Migration Service.
5. Select an existing virtual network or create a new one.

The virtual network provides Azure Database Migration Service with access to the source MongoDB instance and the target Azure Cosmos DB account.

For more information about how to create a virtual network in the Azure portal, see the article [Create a virtual network using the Azure portal](#).

6. Select a pricing tier.

For more information on costs and pricing tiers, see the [pricing page](#).

Create Migration Service

Service Name *
DMSTutorial

* Subscription
<subscription>

* Select a resource group *
<resource group>
[Create new](#)

* Location *
South Central US

* Virtual network
<virtual network>

* Pricing tier
Standard: 1 vCores

Azure Database Migration Service quick start template
Experience our database migration service with pre-created source and target

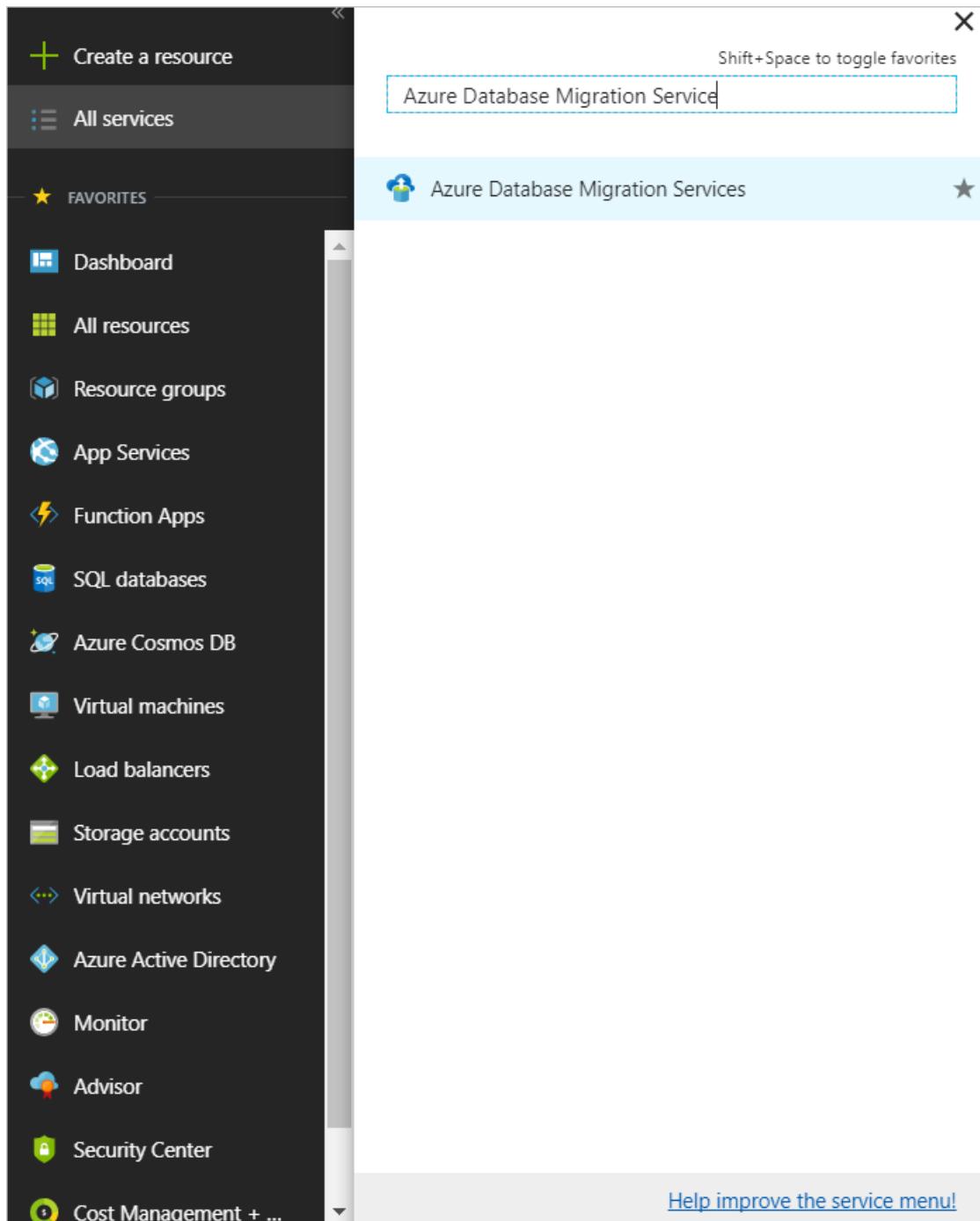
Create [Automation options](#)

7. Select **Create** to create the service.

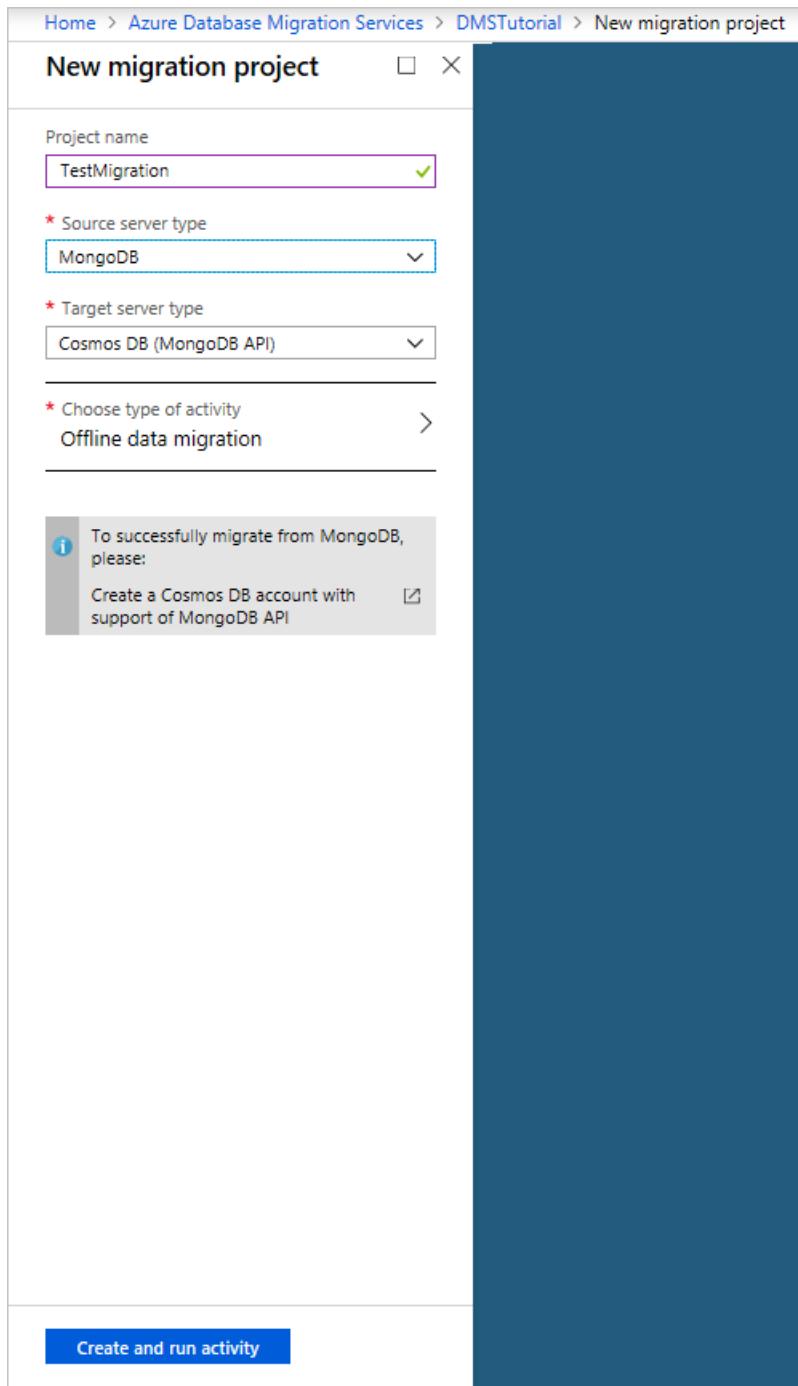
Create a migration project

After the service is created, locate it within the Azure portal, open it, and then create a new migration project.

1. In the Azure portal, select **All services**, search for Azure Database Migration Service, and then select **Azure Database Migration Services**.



2. On the **Azure Database Migration Services** screen, search for the name of Azure Database Migration Service instance that you created, and then select the instance.
3. Select + **New Migration Project**.
4. On the **New migration project** screen, specify a name for the project, in the **Source server type** text box, select **MongoDB**, in the **Target server type** text box, select **CosmosDB (MongoDB API)**, and then for **Choose type of activity**, select **Offline data migration**.



5. Select **Create and run activity** to create the project and run the migration activity.

Specify source details

1. On the **Source details** screen, specify the connection details for the source MongoDB server.

IMPORTANT

Azure Database Migration Service does not support Azure Cosmos DB as a source.

There are three modes to connect to a source:

- **Standard mode**, which accepts a fully qualified domain name or an IP address, Port number, and connection credentials.
- **Connection string mode**, which accepts a MongoDB Connection string as described in the article [Connection String URI Format](#).

- **Data from Azure storage**, which accepts a blob container SAS URL. Select **Blob contains BSON dumps** if the blob container has BSON dumps produced by the MongoDB [bsondump tool](#), and deselect it if the container contains JSON files.

If you select this option, be sure that the storage account connection string appears in the format:

```
https://blobnameurl/container?SASKEY
```

This blob container SAS connection string can be found in Azure Storage explorer. Creating the SAS for the concerned container will provide you the URL in above requested format.

Also, based on the type dump information in Azure Storage, keep the following detail in mind.

- For BSON dumps, the data within the blob container must be in bsondump format, such that data files are placed into folders named after the containing databases in the format *collection.bson*. Metadata files (if any) should be named using the format *collection.metadata.json*.
- For JSON dumps, the files in the blob container must be placed into folders named after the containing databases. Within each database folder, data files must be placed in a subfolder called "data" and named using the format *collection.json*. Metadata files (if any) must be placed in a subfolder called "metadata" and named using the same format, *collection.json*. The metadata files must be in the same format as produced by the MongoDB bsondump tool.

IMPORTANT

It is discouraged to use a self-signed certificate on the mongo server. However, if one is used, please connect to the server using **connection string mode** and ensure that your connection string has ""

```
&sslVerifyCertificate=false
```

You can also use the IP Address for situations in which DNS name resolution isn't possible.

Home > Azure Database Migration Services > DMSTutorial > Migration Wizard > Source details

Migration Wizard		Source details
TestMigration		X X
1	Select source	✓
2	Select target	>
3	Database setting	>
4	Collection setting	>
5	Migration summary	>

Mode
Standard mode

* Source server name ⓘ
Enter source server name
Please enter the name of your source server

Server port
27017

User Name ⓘ
Enter user name

Password
Enter password

Connection properties
 Require SSL

Save

2. Select **Save**.

Specify target details

1. On the **Migration target details** screen, specify the connection details for the target Azure Cosmos DB account, which is the pre-provisioned Azure Cosmos DB's API for MongoDB account to which you're migrating your MongoDB data.

Migration Wizard

TestMigration

Migration target details

Using the connection string mode is not recommended as it might reveal sensitive information

Mode: Select Cosmos DB target

Subscription: <subscription>

Select Cosmos DB name: wingtips-tutorial

* Connection string: mongodb://wingtips-tutorial:sJGOloXLCRjvryTXtdRA7efU9pMeKF3eDIXWh3Vwwdcjjz3Y8CuOtBfHCRo0f0HOQcbnEnY74sNQh0l24qolw==@wingtips-tutorial.documents.azure.com:10255/?ssl=true&replicaSet=globaldb

Save

2. Select **Save**.

Map to target databases

1. On the **Map to target databases** screen, map the source and the target database for migration.

If the target database contains the same database name as the source database, Azure Database Migration Service selects the target database by default.

If the string **Create** appears next to the database name, it indicates that Azure Database Migration Service didn't find the target database, and the service will create the database for you.

At this point in the migration, you can [provision throughput](#). In Cosmos DB, you can provision throughput either at the database-level or individually for each collection. Throughput is measured in [Request Units \(RUs\)](#). Learn more about [Azure Cosmos DB pricing](#).

The screenshot shows the 'Map to target databases' step of the Migration Wizard. The left sidebar lists five steps: 1. Select source (done), 2. Select target (done), 3. Database setting (in progress), 4. Collection setting (in progress), and 5. Migration summary (in progress). The main area displays a table for mapping collections. It shows one item: 'wingtips' from the 'SOURCE DATABASE' column is mapped to 'Create: wingtips' in the 'TARGET DATABASE' column. A note below the table says 'Blank for default or RU between 10000 to 100'. A 'CLEAN UP COLLECTIONS' checkbox is checked. At the bottom is a blue 'Save' button.

2. Select **Save**.
3. On the **Collection setting** screen, expand the collections listing, and then review the list of collections that will be migrated.

Azure Database Migration Service auto selects all the collections that exist on the source MongoDB instance that don't exist on the target Azure Cosmos DB account. If you want to remigrate collections that already include data, you need to explicitly select the collections on this blade.

You can specify the amount of RUs that you want the collections to use. Azure Database Migration Service suggests smart defaults based on the collection size.

NOTE

Perform the database migration and collection in parallel using multiple instances of Azure Database Migration Service, if necessary, to speed up the run.

You can also specify a shard key to take advantage of [partitioning in Azure Cosmos DB](#) for optimal scalability. Be sure to review the [best practices for selecting a shard/partition key](#).

Home > Azure Database Migration Services > DMSTutorial > TestMigration > Migration Wizard > Collection setting

Migration Wizard		X	Collection setting	X																																			
1	Select source	✓	Mongo collection settings for each database ^ wingtips 6 of 6	X																																			
2	Select target	✓	<input type="text"/> Search	← prev Page 1 of 1 next →																																			
3	Database setting	✓	6 item(s)																																				
4	Collection setting	>	<table border="1"> <thead> <tr> <th>NAME</th> <th>TARGET COLLECTION</th> <th>THROUGHPUT (RU/S)</th> <th>SHARD KEY</th> <th>UNIQUE</th> </tr> </thead> <tbody> <tr> <td>Configurations</td> <td>Create: Configurations</td> <td>RU: 1000 ✓</td> <td></td> <td></td> </tr> <tr> <td>Customers</td> <td>Create: Customers</td> <td>RU: 1000 ✓</td> <td>LastName ✓</td> <td></td> </tr> <tr> <td>CustomerTickets</td> <td>Create: CustomerTickets</td> <td>RU: 1000 ✓</td> <td>CustomerId ✓</td> <td></td> </tr> <tr> <td>EventSections</td> <td>Create: EventSections</td> <td>RU: 1000 ✓</td> <td>VenueId ✓</td> <td></td> </tr> <tr> <td>Sections</td> <td>Create: Sections</td> <td>RU: 1000 ✓</td> <td>SectionName ✓</td> <td></td> </tr> <tr> <td>Venues</td> <td>Create: Venues</td> <td>RU: 1000 ✓</td> <td>VenueName ✓</td> <td></td> </tr> </tbody> </table>	NAME	TARGET COLLECTION	THROUGHPUT (RU/S)	SHARD KEY	UNIQUE	Configurations	Create: Configurations	RU: 1000 ✓			Customers	Create: Customers	RU: 1000 ✓	LastName ✓		CustomerTickets	Create: CustomerTickets	RU: 1000 ✓	CustomerId ✓		EventSections	Create: EventSections	RU: 1000 ✓	VenueId ✓		Sections	Create: Sections	RU: 1000 ✓	SectionName ✓		Venues	Create: Venues	RU: 1000 ✓	VenueName ✓		
NAME	TARGET COLLECTION	THROUGHPUT (RU/S)	SHARD KEY	UNIQUE																																			
Configurations	Create: Configurations	RU: 1000 ✓																																					
Customers	Create: Customers	RU: 1000 ✓	LastName ✓																																				
CustomerTickets	Create: CustomerTickets	RU: 1000 ✓	CustomerId ✓																																				
EventSections	Create: EventSections	RU: 1000 ✓	VenueId ✓																																				
Sections	Create: Sections	RU: 1000 ✓	SectionName ✓																																				
Venues	Create: Venues	RU: 1000 ✓	VenueName ✓																																				
5	Migration summary	>																																					
Save																																							

4. Select **Save**.
5. On the **Migration summary** screen, in the **Activity name** text box, specify a name for the migration activity.

Home > Azure Database Migration Services > DMSTutorial > TestMigration > Migration Wizard > Migration summary

Migration Wizard		X	Migration summary	X
1	Select source	✓	Activity name <input type="text"/> TestMigration	X
2	Select target	✓	Target server name wingtips-tutorial.documents.azure.com	
3	Database setting	✓	Target server version 3.2.0	
4	Collection setting	✓	Source server name 40.74.232.123	
5	Migration summary	>	Source server version 3.2.21	
Database(s) to migrate 1 of 1 <input checked="" type="checkbox"/> Boost RU to during initial data copy				
Run migration				

Run the migration

- Select **Run migration**.

The migration activity window appears, and the **Status** of the activity is **Not started**.

The screenshot shows the Azure Database Migration Services interface. At the top, there's a breadcrumb navigation: Home > Azure Database Migration Services > DMSTutorial > TestMigration > TestMigration. Below the breadcrumb is a title bar for 'TestMigration' with 'TestMigration' repeated below it. On the right of the title bar are icons for Delete migration, Stop migration, Refresh, and Download logs. Under the title bar, there are two rows of metrics: 'Status' and 'Duration' (Throughput (bytes/s) and Throughput (documents/s)), and 'Total bytes' and 'Total documents'. A search bar labeled 'Search to filter items...' is located below these metrics. A table follows, with columns: NAME, STATUS, ERRORS, DOCUMENTS, BYTES C..., and DURATION. The single row in the table is for 'wingtips', showing 'Not started' in the STATUS column, '0' in the ERRORS column, and '0 Byte(s)' in the BYTES C... column.

NAME	STATUS	ERRORS	DOCUMENTS	BYTES C...	DURATION
wingtips	Not started	0	0	0 Byte(s)	---

Monitor the migration

- On the migration activity screen, select **Refresh** to update the display until the **Status** of the migration shows as **Completed**.

NOTE

You can select the Activity to get details of database- and collection-level migration metrics.

Home > Azure Database Migration Services > DMSTutorial > TestMigration > TestMigration

TestMigration

TestMigration

Delete migration Stop migration Refresh Download logs

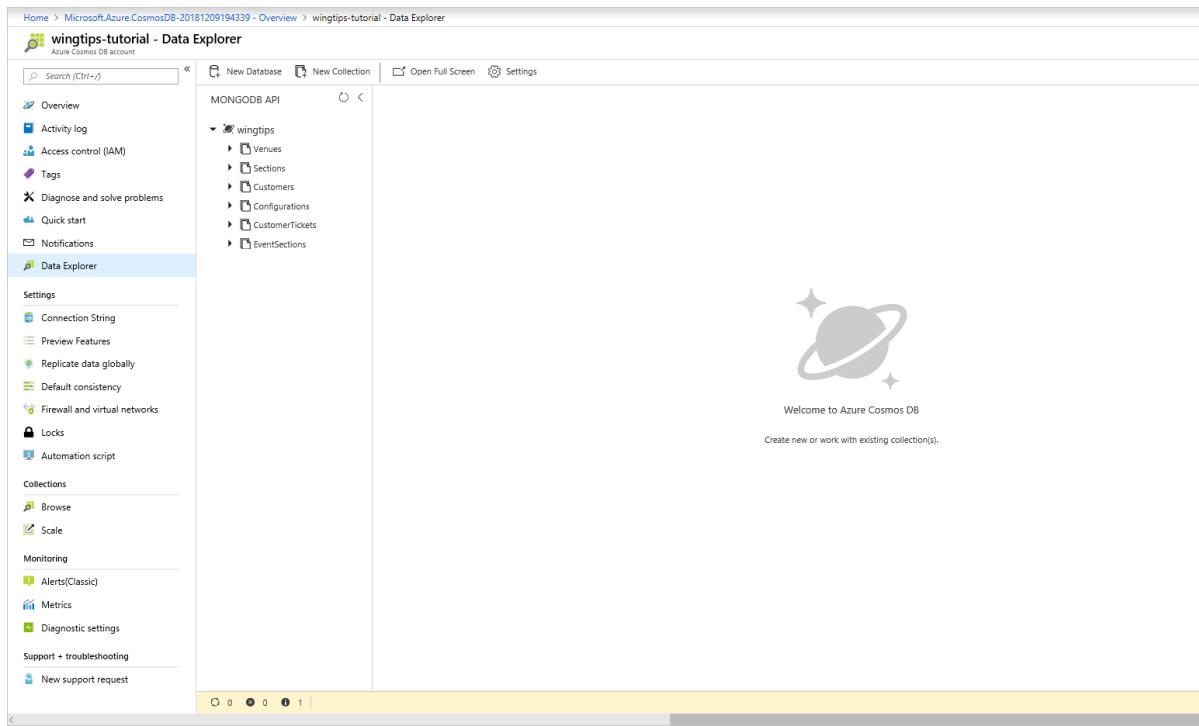
Status Complete	Duration 00:00:16
Throughput (bytes/s) Complete: 39.01 KB of 39.01 KB	Throughput (documents/s) Complete: 123 of 123
Total bytes 39.01 KB	Total documents 123

Search to filter items...

NAME	STATUS	ERRORS	DOCUM...	BYTES C...	DURATI...	...
wingtips	Complete		123/123	39.01 KB/3...	00:00:16	...

Verify data in Cosmos DB

- After the migration completes, you can check your Azure Cosmos DB account to verify that all the collections were migrated successfully.



Post-migration optimization

After you migrate the data stored in MongoDB database to Azure Cosmos DB's API for MongoDB, you can connect to Azure Cosmos DB and manage the data. You can also perform other post-migration optimization steps such as optimizing the indexing policy, update the default consistency level, or configure global distribution for your Azure Cosmos DB account. For more information, see the [Post-migration optimization](#) article.

Additional resources

- [Cosmos DB service information](#)

Next steps

- Review migration guidance for additional scenarios in the Microsoft [Database Migration Guide](#).

Tutorial: Migrate MongoDB to Azure Cosmos DB's API for MongoDB online using DMS

1/8/2020 • 9 minutes to read • [Edit Online](#)

You can use Azure Database Migration Service to perform an online (minimal downtime) migration of databases from an on-premises or cloud instance of MongoDB to Azure Cosmos DB's API for MongoDB.

In this tutorial, you learn how to:

- Create an instance of Azure Database Migration Service.
- Create a migration project by using Azure Database Migration Service.
- Run the migration.
- Monitor the migration.
- Complete the migration when you are ready.

In this tutorial, you migrate a dataset in MongoDB hosted in an Azure Virtual Machine to Azure Cosmos DB's API for MongoDB with minimal downtime by using Azure Database Migration Service. If you don't have a MongoDB source set up already, see the article [Install and configure MongoDB on a Windows VM in Azure](#).

NOTE

Using Azure Database Migration Service to perform an online migration requires creating an instance based on the Premium pricing tier.

IMPORTANT

For an optimal migration experience, Microsoft recommends creating an instance of Azure Database Migration Service in the same Azure region as the target database. Moving data across regions or geographies can slow down the migration process.

TIP

When you migrate databases to Azure by using Azure Database Migration Service, you can do an *offline* or an *online* migration. With an offline migration, application downtime starts when the migration starts. With an online migration, downtime is limited to the time to cut over at the end of migration. We suggest that you test an offline migration to determine whether the downtime is acceptable; if not, do an online migration.

This article describes an online migration from MongoDB to Azure Cosmos DB's API for MongoDB. For an offline migration, see [Migrate MongoDB to Azure Cosmos DB's API for MongoDB offline using DMS](#).

Prerequisites

To complete this tutorial, you need to:

- [Complete the pre-migration](#) steps such as estimating throughput, choosing a partition key, and the indexing policy.
- [Create an Azure Cosmos DB's API for MongoDB account](#).
- Create a Microsoft Azure Virtual Network for Azure Database Migration Service by using Azure Resource

Manager deployment model, which provides site-to-site connectivity to your on-premises source servers by using either [ExpressRoute](#) or [VPN](#).

NOTE

During virtual network setup, if you use ExpressRoute with network peering to Microsoft, add the following service endpoints to the subnet in which the service will be provisioned:

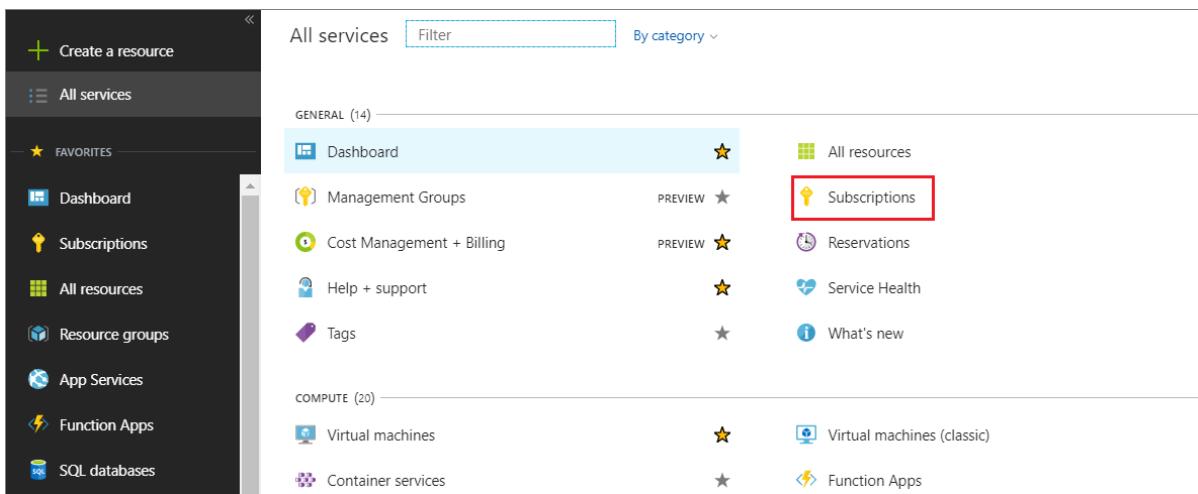
- Target database endpoint (for example, SQL endpoint, Cosmos DB endpoint, and so on)
- Storage endpoint
- Service bus endpoint

This configuration is necessary because Azure Database Migration Service lacks internet connectivity.

- Ensure that your virtual network Network Security Group (NSG) rules don't block the following communication ports: 53, 443, 445, 9354, and 10000-20000. For more detail on virtual network NSG traffic filtering, see the article [Filter network traffic with network security groups](#).
- Open your Windows firewall to allow Azure Database Migration Service to access the source MongoDB server, which by default is TCP port 27017.
- When using a firewall appliance in front of your source database(s), you may need to add firewall rules to allow Azure Database Migration Service to access the source database(s) for migration.

Register the Microsoft.DataMigration resource provider

1. Sign in to the Azure portal, select **All services**, and then select **Subscriptions**.



2. Select the subscription in which you want to create the instance of Azure Database Migration Service, and then select **Resource providers**.

The screenshot shows the Azure portal's Subscriptions page. On the left, there's a navigation bar with various service icons like App Services, SQL databases, and Storage accounts. The main area displays a table with columns for Subscription ID and Status. A search bar at the top says 'Search to filter items...'. On the right, a sidebar lists several options: Overview, Access control (IAM), Diagnose and solve problems, COST MANAGEMENT + BILLING, Partner information, SETTINGS, Programmatic deployment, Resource groups, Resources, Usage + quotas, Policies, Management certificates, My permissions, and Resource providers. The 'Resource providers' link is specifically highlighted with a red box.

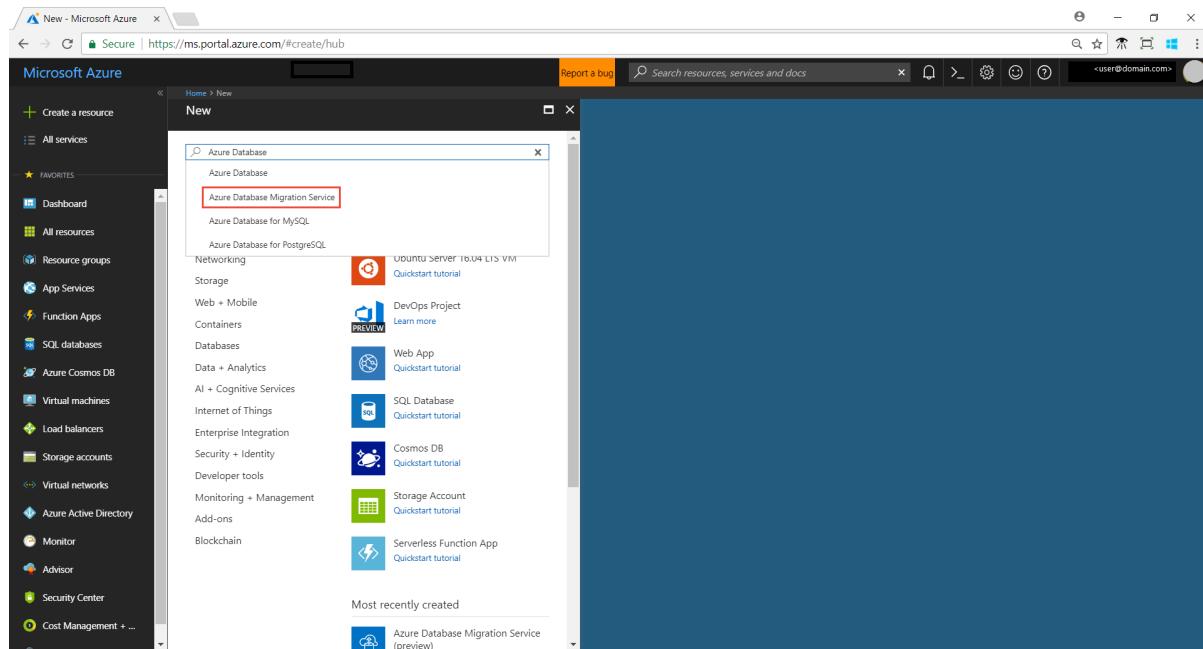
3. Search for migration, and then to the right of **Microsoft.DataMigration**, select **Register**.

This screenshot shows the 'Resource providers' page within the Azure portal. It features a search bar at the top and a table below it. The table has columns for PROVIDER, STATUS, and a Register button. One row in the table is highlighted with a yellow background and shows 'Microsoft.DataMigration' in the PROVIDER column, 'NotRegistered' in the STATUS column, and a red box around the 'Register' button.

Create an instance

1. In the Azure portal, select + **Create a resource**, search for Azure Database Migration Service, and then

select **Azure Database Migration Service** from the drop-down list.



2. On the **Azure Database Migration Service** screen, select **Create**.

The Azure Database Migration Service (DMS) is designed to streamline the process of migrating on-premises databases to Azure. DMS will simplify the migration of existing on-premises SQL Server and Oracle databases to Azure SQL Database, Azure SQL Managed Instance or Microsoft SQL Server in an Azure Virtual Machine. Learn [more](#)

Before using DMS we recommend you complete the following three steps:

1. Open the [Azure Database Migration Guide](#) for step by step guidance through the migration process
2. Assess your SQL Server on-premises database(s) for feature parity and potential compatibility issues by using [Data Migration Assistant \(DMA\)](#). Alternatively, if you are migrating from Oracle, use the [SQL Server Migration Assistant \(SSMA\)](#)
3. Based on your database needs, create a target database using one of the database services: Azure SQL Database, Azure SQL Managed Instance or SQL Server in an Azure Virtual Machine.

Once your assessments are complete, fixes are applied and schema is deployed, proceed with creating a migration service by clicking **Create** below to migrate the data from your source database to the target.

[Save for later](#)

PUBLISHER Microsoft

USEFUL LINKS [Documentation](#) [Privacy Statement](#)

Create

3. On the **Create Migration Service** screen, specify a name for the service, the subscription, and a new or

existing resource group.

4. Select the location in which you want to create the instance of Azure Database Migration Service.
5. Select an existing virtual network, or create a new one.

The virtual network provides Azure Database Migration Service with access to the source MongoDB instance and the target Azure Cosmos DB account.

For more information about how to create a virtual network in the Azure portal, see the article [Create a virtual network using the Azure portal](#).

6. Select a SKU from the Premium pricing tier.

NOTE

Online migrations are supported only when using the Premium tier. For more information on costs and pricing tiers, see the [pricing page](#).

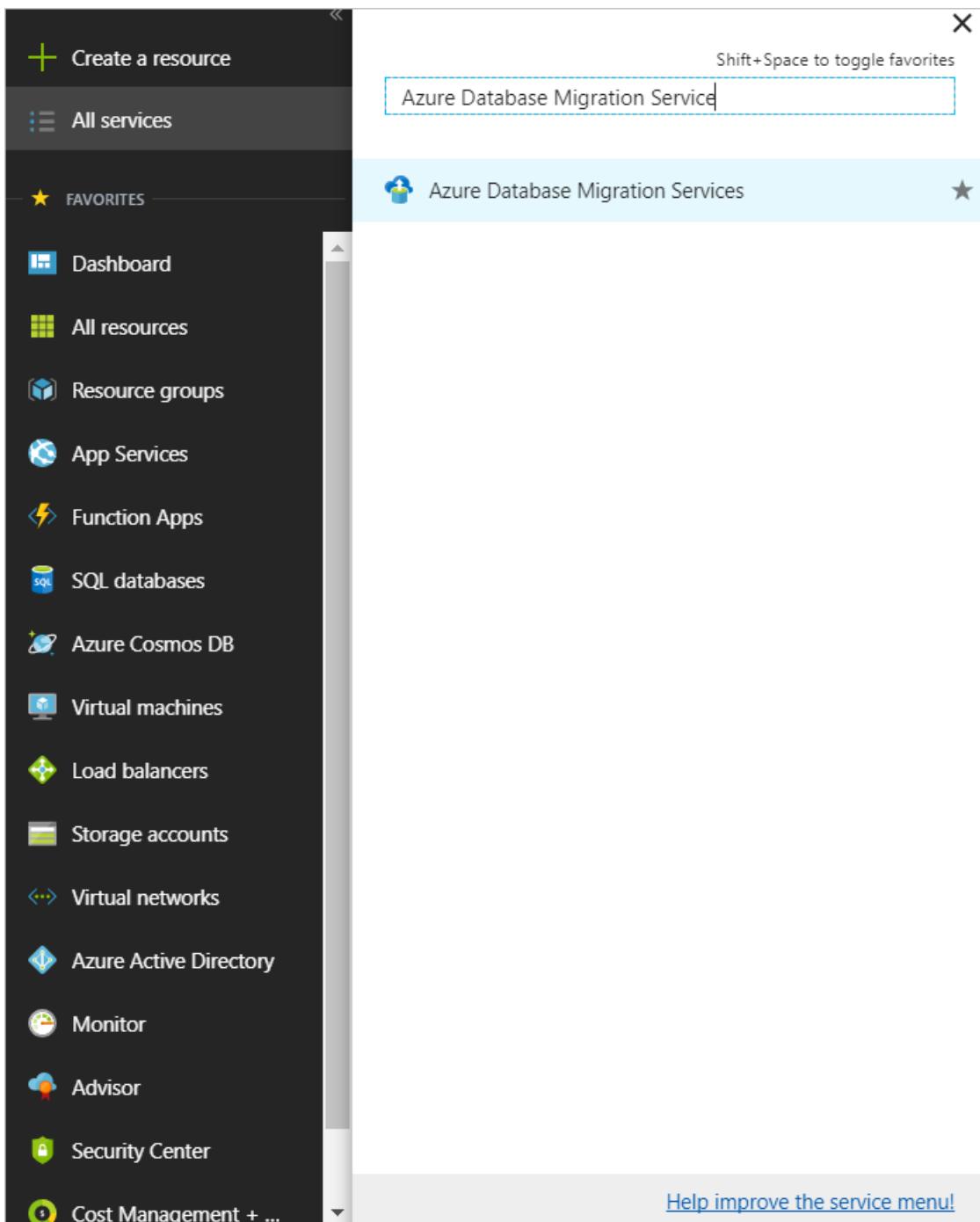
The screenshot shows the 'Create Migration Service' wizard in the Azure portal. The current step is 'Pricing tier'. On the left, there are fields for 'Service Name' (DMSOSSYNC), 'Subscription' (<subscription ID>), 'Resource Group' ((New) OSSYNC), 'Location' (West US 2), 'Virtual network' (<virtual network>), and 'Pricing tier' (Premium: 4 vCores). On the right, a note says 'The selected tier supports both offline and online migrations.' Below it, two options are shown: 'Standard' (For large data sizes, 1 vCore, 2 vCores, 4 vCores) and 'Premium' (For offline and online migrations with minimal downtime, 4 vCores). A 'vCores' slider is set to 4 vCores, with an estimated monthly cost of '\$0.00 USD/hour'. At the bottom right, a note says 'You can use the Database Migration Service Premium tier for free until February 28, 2019. Beginning March'.

7. Select **Create** to create the service.

Create a migration project

After the service is created, locate it within the Azure portal, open it, and then create a new migration project.

1. In the Azure portal, select **All services**, search for Azure Database Migration Service, and then select **Azure Database Migration Services**.



2. On the **Azure Database Migration Services** screen, search for the name of Azure Database Migration Service instance that you created, and then select the instance.

Alternately, you can discover Azure Database Migration service instance from the search pane in Azure portal.



3. Select + **New Migration Project**.

4. On the **New migration project** screen, specify a name for the project, in the **Source server type** text box, select **MongoDB**, in the **Target server type** text box, select **CosmosDB (MongoDB API)**, and then for **Choose type of activity**, select **Online data migration [preview]**.

New migration project X

Type of activity X

Project name <input type="text" value="MigratefromMongo"/> ✓ * Source server type <input type="text" value="MongoDB"/> * Target server type <input type="text" value="Cosmos DB (MongoDB API)"/> * Choose type of activity Online data migration [preview] >	Choose type of activity <input type="text" value="Online data migration [preview]"/> Use this option to migrate databases that must be accessible and continuously updated during migration. To continuously replicate MongoDB changes from your source to your target, please implement the steps below on your source. These steps can be implemented anytime between the moment you create a project and the moment you start a migration activity. After your migration is complete, you will reverse those preparation steps. <ul style="list-style-type: none"> • Prepare your Cosmos DB target <p>To successfully migrate from MongoDB, please: Create a Cosmos DB account with support of MongoDB API [x]</p>
Create and run activity	Save

5. Select **Save**, and then select **Create and run activity** to create the project and run the migration activity.

Specify source details

1. On the **Source details** screen, specify the connection details for the source MongoDB server.

IMPORTANT

Azure Database Migration Service does not support Azure Cosmos DB as a source.

There are three modes to connect to a source:

- **Standard mode**, which accepts a fully qualified domain name or an IP address, Port number, and connection credentials.
- **Connection string mode**, which accepts a MongoDB Connection string as described in the article [Connection String URI Format](#).
- **Data from Azure storage**, which accepts a blob container SAS URL. Select **Blob contains BSON dumps** if the blob container has BSON dumps produced by the MongoDB [bsondump tool](#), and de-select it if the container contains JSON files.

If you select this option, be sure that the storage account connection string appears in the format:

```
https://blobnameurl/container?SASKEY
```

Also, based on the type dump information in Azure Storage, keep the following detail in mind.

- For BSON dumps, the data within the blob container must be in bsondump format, such that data files are placed into folders named after the containing databases in the format `collection.bson`. Metadata files (if any) should be named using the format `collection.metadata.json`.
- For JSON dumps, the files in the blob container must be placed into folders named after the containing databases. Within each database folder, data files must be placed in a subfolder

called "data" and named using the format `collection.json`. Metadata files (if any) must be placed in a subfolder called "metadata" and named using the same format, `collection.json`. The metadata files must be in the same format as produced by the MongoDB `bsondump` tool.

IMPORTANT

It is discouraged to use a self-signed certificate on the mongo server. However, if one is used, please connect to the server using **connection string mode** and ensure that your connection string has ""

```
&sslVerifyCertificate=false
```

You can use the IP Address for situations in which DNS name resolution isn't possible.

Dashboard > Azure Database Migration Services > DMSOSSYNC > Migration Wizard > Source details

Migration Wizard		X	Source details		□ X
MigratefromMongo					
1	Select source	>	Mode Standard mode		
2	Select target	>	* Source server name ⓘ Enter source server name Please enter the name of your source server		
3	Database setting	>	Server port 27017		
4	Collection setting	>	User Name ⓘ Enter user name		
5	Migration summary	>	Password Enter password		
			Connection properties <input type="checkbox"/> Require SSL		
			Save		

2. Select **Save**.

NOTE

The Source server address should be the address of the primary if the source is a replica set, and the router if the source is a sharded MongoDB cluster. For a sharded MongoDB cluster, Azure Database Migration Service must be able to connect to the individual shards in the cluster, which may require opening the firewall on more machines.

Specify target details

1. On the **Migration target details** screen, specify the connection details for the target Azure Cosmos DB account, which is the pre-provisioned Azure Cosmos DB's API for MongoDB account to which you're

migrating your MongoDB data.

The screenshot shows the 'Migration Wizard' interface for migrating from MongoDB to Cosmos DB. The left sidebar lists five steps: 1. Select source (completed), 2. Select target (selected), 3. Database setting, 4. Collection setting, and 5. Migration summary. The 'Select target' step is highlighted with a green checkmark. The main panel, titled 'Migration target details', shows the following configuration:

- Mode:** Select Cosmos DB target
- Subscription:** <subscription ID>
- Select Cosmos DB name:** dmstestcosmos
- * Connection string:** <Connection string> (highlighted with a purple border)

A blue 'Save' button is located at the bottom of the panel.

2. Select **Save**.

Map to target databases

1. On the **Map to target databases** screen, map the source and the target database for migration.

If the target database contains the same database name as the source database, Azure Database Migration Service selects the target database by default.

If the string **Create** appears next to the database name, it indicates that Azure Database Migration Service didn't find the target database, and the service will create the database for you.

At this point in the migration, if you want share throughput on the database, specify a throughput RU. In Cosmos DB, you can provision throughput either at the database-level or individually for each collection. Throughput is measured in [Request Units \(RUs\)](#). Learn more about [Azure Cosmos DB pricing](#).

2. Select **Save**.
3. On the **Collection setting** screen, expand the collections listing, and then review the list of collections that will be migrated.

Azure Database Migration Service auto selects all the collections that exist on the source MongoDB instance that don't exist on the target Azure Cosmos DB account. If you want to remigrate collections that already include data, you need to explicitly select the collections on this screen.

You can specify the number of RUs that you want the collections to use. In most cases, a value between 500 (1000 minimum for sharded collections) and 4000 should suffice. Azure Database Migration Service suggests smart defaults based on the collection size.

NOTE

Perform the database migration and collection in parallel using multiple instances of Azure Database Migration Service, if necessary, to speed up the run.

You can also specify a shard key to take advantage of [partitioning in Azure Cosmos DB](#) for optimal scalability. Be sure to review the [best practices for selecting a shard/partition key](#). If you don't have a partition key, you can always use `_id` as the shard key for better throughput.

Migration Wizard > **Migration from MongoDB** > **Migration Wizard** > **Collection setting**

Collection setting

Mongo collection settings for each database						
^ LongRun1GB_Sharded 6 of 6						
<input type="text"/> Search 6 item(s)						
		THROUGHPUT (RU/S)	SHARD KEY	UNIQUE		
<input checked="" type="checkbox"/>	NAME	TARGET COLLECTION	RU: 1000	_id	<input checked="" type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	docs10KB	docs10KB	RU: 1000	_id	<input checked="" type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	docs10KB	docs10KB	RU: 1000	_id	<input checked="" type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	docs135B	docs135B	RU: 1000	_id	<input checked="" type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	docs1KB	docs1KB	RU: 1000	_id	<input checked="" type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	docs1MB	docs1MB	RU: 1000	_id	<input checked="" type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	docs35B	docs35B	RU: 1000	_id	<input checked="" type="checkbox"/>	<input type="checkbox"/>

Save

4. Select **Save**.
5. On the **Migration summary** screen, in the **Activity name** text box, specify a name for the migration activity.

Migration Wizard > **Migration from MongoDB** > **Migration Wizard** > **Migration summary**

Migration summary

Activity name	OnlineShardMigrate
Target server name	dmstestcosmos.documents.azure.com
Target server version	3.2.0
Source server name	172.16.223.101
Source server version	3.6.7
Database(s) to migrate	1 of 2 <input checked="" type="checkbox"/> Boost RU during initial data copy
Run migration	

Run the migration

- Select **Run migration**.

The migration activity window appears, and the **Status** of the activity is displayed.

Home > DMSOSSYNC > MigratefromMongo (DMSOSSYNC/MigratefromMongo) > OnlineShardMigrate

OnlineShardMigrate

MigratefromMongo

>Delete migration Stop migration Refresh Download logs

Status Replaying	Duration 00:02:59
Throughput (bytes/s) Replaying: 800.95 MB of 800.95 MB	Throughput (documents/s) Replaying: 1778655 of 1778655
Total bytes 800.95 MB	Total documents 1778655

▲

Search to filter items...

NAME	STATUS	ERRORS	DOCUM...	BYTES C...	DURATI...
LongRun1G...	Replaying		1778655/17...	800.95 MB/...	00:02:59
...					

Monitor the migration

- On the migration activity screen, select **Refresh** to update the display until the **Status** of the migration shows as **Replaying**.

NOTE

You can select the Activity to get details of database- and collection-level migration metrics.

Home > DMSOSSYNC > MigratefromMongo (DMSOSSYNC/MigratefromMongo) > OnlineShardMigrate > LongRun1GB_Sharded

dMigrate

LongRun1GB_Sharded

Delete migration Stop migration Refresh Download logs

(bytes/s) 00.95 MB of 800.95 MB	Duration 00:02:59
Throughput (documents/s) Replaying: 1778655 of 1778655	Throughput (bytes/s) Replaying: 800.95 MB of 800.95 MB
Total documents 1778655	Total bytes 800.95 MB

▲

Search to filter items...

NAME	STATUS	ERRORS	DOCUM...	BYTES C...	DURATI...
docs100KB	Replaying		7897/7897	19.72 MB/1...	00:00:09
docs10KB	Replaying		23077/23077	81.79 MB/8...	00:00:16
docs135B	Replaying		819683/81...	233.21 MB/...	00:02:49
docs1KB	Replaying		221049/22...	294.61 MB...	00:01:22
docs1MB	Replaying		7563/7563	18.79 MB/1...	00:00:09
docs35B	Replaying		699386/69...	152.84 MB/...	00:02:17

Verify data in Cosmos DB

1. Make changes to your source MongoDB database.
2. Connect to COSMOS DB to verify if the data is replicated from the source MongoDB server.

The screenshot shows the Azure Cosmos DB Data Explorer interface. On the left, the sidebar has 'Data Explorer' selected. The main area shows a tree view of collections: 'docs4KB', 'LongRun1GB_Sharded' (selected), 'docs100KB', 'docs1KB' (selected), 'docs35B', 'docs1MB', 'docs135B', 'docs10KB', and 'BulkNoShard3GB'. A query window titled 'Query 1' contains the command '1 { _id:266991 }'. The results pane shows one document with the following JSON structure:

```
{ "_id" : 266991, "b" : "WJ2S44JNCPV97NGQ6SJ534YLX5LY4SWTWAG6YEE3EX5X9T", "sizeByte" : 1058 }
```

Complete the migration

- After all documents from the source are available on the COSMOS DB target, select **Finish** from the migration activity's context menu to complete the migration.

This action will finish replaying all the pending changes and complete the migration.

The screenshot shows the DMSOSSYNC migration activity details for 'OnlineShardMigrate'. The status is 'Replaying'. The throughput is 'Replaying: 800.95 MB of 800.95 MB'. The total bytes are '800.95 MB' and the total documents are '1778655'. The duration is '00:02:59'. The throughput is 'Replaying: 1778655 of 1778655'. A context menu is open on the right side, listing 'Abort', 'Restart', 'Finish' (which is highlighted with a dashed border), and 'Finish immediately'.

Post-migration optimization

After you migrate the data stored in MongoDB database to Azure Cosmos DB's API for MongoDB, you can connect to Azure Cosmos DB and manage the data. You can also perform other post-migration optimization steps

such as optimizing the indexing policy, update the default consistency level, or configure global distribution for your Azure Cosmos DB account. For more information, see the [Post-migration optimization](#) article.

Additional resources

- [Cosmos DB service information](#)

Next steps

- Review migration guidance for additional scenarios in the Microsoft [Database Migration Guide](#).

Tutorial: Migrate Oracle to Azure Database for PostgreSQL online using DMS (Preview)

1/24/2020 • 13 minutes to read • [Edit Online](#)

You can use Azure Database Migration Service to migrate the databases from Oracle databases hosted on-premises or on virtual machines to [Azure Database for PostgreSQL](#) with minimal downtime. In other words, you can complete the migration with minimal downtime to the application. In this tutorial, you migrate the **HR** sample database from an on-premises or virtual machine instance of Oracle 11g to Azure Database for PostgreSQL by using the online migration activity in Azure Database Migration Service.

In this tutorial, you learn how to:

- Assess the migration effort using the ora2pg tool.
- Migrate the sample schema using the ora2pg tool.
- Create an instance of Azure Database Migration Service.
- Create a migration project by using Azure Database Migration Service.
- Run the migration.
- Monitor the migration.

NOTE

Using Azure Database Migration Service to perform an online migration requires creating an instance based on the Premium pricing tier.

IMPORTANT

For an optimal migration experience, Microsoft recommends creating an instance of Azure Database Migration Service in the same Azure region as the target database. Moving data across regions or geographies can slow down the migration process and introduce errors.

TIP

When you migrate databases to Azure by using Azure Database Migration Service, you can do an *offline* or an *online* migration. With an offline migration, application downtime starts when the migration starts. With an online migration, downtime is limited to the time to cut over at the end of migration. We suggest that you test an offline migration to determine whether the downtime is acceptable; if not, do an online migration.

This article describes how to perform an online migration from Oracle to Azure Database for PostgreSQL.

Prerequisites

To complete this tutorial, you need to:

- Download and install [Oracle 11g Release 2 \(Standard Edition, Standard Edition One, or Enterprise Edition\)](#).
- Download the sample **HR** database from [here](#).
- Download and install [ora2pg](#) on either Windows or Linux.

- Create an instance in Azure Database for PostgreSQL.
- Connect to the instance and create a database using the instruction in this [document](#).
- Create a Microsoft Azure Virtual Network for Azure Database Migration Service by using the Azure Resource Manager deployment model, which provides site-to-site connectivity to your on-premises source servers by using either [ExpressRoute](#) or [VPN](#). For more information about creating a virtual network, see the [Virtual Network Documentation](#), and especially the quickstart articles with step-by-step details.

NOTE

During virtual network setup, if you use ExpressRoute with network peering to Microsoft, add the following service [endpoints](#) to the subnet in which the service will be provisioned:

- Target database endpoint (for example, SQL endpoint, Cosmos DB endpoint, and so on)
- Storage endpoint
- Service bus endpoint

This configuration is necessary because Azure Database Migration Service lacks internet connectivity.

- Ensure that your virtual network Network Security Group (NSG) rules don't block the following inbound communication ports to Azure Database Migration Service: 443, 53, 9354, 445, 12000. For more detail on virtual network NSG traffic filtering, see the article [Filter network traffic with network security groups](#).
- Configure your [Windows Firewall for database engine access](#).
- Open your Windows firewall to allow Azure Database Migration Service to access the source Oracle server, which by default is TCP port 1521.
- When using a firewall appliance in front of your source database(s), you may need to add firewall rules to allow Azure Database Migration Service to access the source database(s) for migration.
- Create a server-level [firewall rule](#) for Azure Database for PostgreSQL to allow Azure Database Migration Service access to the target databases. Provide the subnet range of the virtual network used for Azure Database Migration Service.
- Enable access to the source Oracle databases.

NOTE

The DBA role is required for a user to connect to the Oracle source.

- Archive Redo Logs is required for incremental sync in Azure Database Migration Service to capture data change. Follow these steps to configure the Oracle source:
 - Sign in using SYSDBA privilege by running the following command:

```
sqlplus (user)/(password) as sysdba
```

- Shut down the database instance by running the following command.

```
SHUTDOWN IMMEDIATE;
```

Wait for the confirmation `'ORACLE instance shut down'`.

- Start the new instance and mount (but don't open) the database to enable or disable archiving

but running the following command:

```
STARTUP MOUNT;
```

The database must be mounted; wait for confirmation 'Oracle instance started'.

- o Change the database archiving mode by running the following command:

```
ALTER DATABASE ARCHIVELOG;
```

- o Open the database for normal operations by running the following command:

```
ALTER DATABASE OPEN;
```

You may need to restart for the ARC file to show up.

- o To verify, run the following command:

```
SELECT log_mode FROM v$database;
```

You should receive a response '`ARCHIVELOG`'. If the response is '`NOARCHIVELOG`', then the requirement isn't met.

- o Enable supplemental logging for replication using one of the following options.

- o **Option 1.** Change the database level supplemental logging to cover all the tables with PK and unique index. The detection query will return '`IMPLICIT`'.

```
ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (PRIMARY KEY, UNIQUE) COLUMNS;
```

Change the table level supplemental logging. Run only for tables that have data manipulation and don't have PKs or unique indexes.

```
ALTER TABLE [TABLENAME] ADD SUPPLEMENTAL LOG DATA (ALL) COLUMNS;
```

- o **Option 2.** Change the database level supplemental logging to cover all the tables, and the detection query returns '`YES`'.

```
ALTER DATABASE ADD SUPPLEMENTAL LOG DATA;
```

Change the table level supplemental logging. Follow the logic below to run only one statement for every table.

If the table has a primary key:

```
ALTER TABLE xxx ADD SUPPLEMENTAL LOG DATA (PRIMARY KEY) COLUMNS;
```

If the table has a unique index:

```
ALTER TABLE xxx ADD SUPPLEMENTAL LOG GROUP (first unique index columns) ALWAYS;
```

Otherwise, run the following command:

```
ALTER TABLE xxx ADD SUPPLEMENTAL LOG DATA (ALL) COLUMNS;
```

To verify, run the following command:

```
SELECT supplemental_log_data_min FROM v$database;
```

You should receive a response `'YES'`.

Assess the effort for an Oracle to Azure Database for PostgreSQL migration

We recommend using ora2pg to assess the effort required to migrate from Oracle to Azure Database for PostgreSQL. Use the `ora2pg -t SHOW_REPORT` directive to create a report listing all the Oracle objects, the estimated migration cost (in developer days), and certain database objects that may require special attention as part of the conversion.

Most customers will spend a considerable amount time reviewing the assessment report and considering the automatic and manual conversion effort.

To configure and run ora2pg to create an assessment report, see the **Premigration: Assessment** section of the [Oracle to Azure Database for PostgreSQL Cookbook](#). A sample ora2pg assessment report is available for reference [here](#).

Export the Oracle schema

We recommend that you use ora2pg to convert the Oracle schema and other Oracle objects (types, procedures, functions, etc.) to a schema that is compatible with Azure Database for PostgreSQL. ora2pg includes many directives to help you pre-define certain data types. For example, you can use the `DATA_TYPE` directive to replace all `NUMBER(*,0)` with `bignum` rather than `NUMERIC(38)`.

You can run ora2pg to export each of the database objects in .sql files. You can then review the .sql files before importing them to Azure Database for PostgreSQL using psql or you can execute the .SQL script in PgAdmin.

```
psql -f [FILENAME] -h [AzurePostgreConnection] -p 5432 -U [AzurePostgreUser] -d database
```

For example:

```
psql -f %namespace%\schema\sequences\sequence.sql -h server1-server.postgres.database.azure.com -p 5432 -U username@server1-server -d database
```

To configure and run ora2pg for schema conversion, see the **Migration: Schema and data** section of the [Oracle to Azure Database for PostgreSQL Cookbook](#).

Set up the schema in Azure Database for PostgreSQL

You can choose to convert Oracle table schemas, stored procedures, packages, and other database objects to make them Postgres compatible by using ora2pg before starting a migration pipeline in Azure Database Migration Service. See the links below for how to work with ora2pg:

- [Install ora2pg on Windows](#)

- Oracle to Azure PostgreSQL Migration Cookbook

Azure Database Migration Service can also create the PostgreSQL table schema. The service accesses the table schema in the connected Oracle source and creates a compatible table schema in Azure Database for PostgreSQL. Be sure to validate and check the schema format in Azure Database for PostgreSQL after Azure Database Migration Service finishes creating the schema and moving the data.

IMPORTANT

Azure Database Migration Service only creates the table schema; other database objects such as stored procedures, packages, indexes, etc., are not created.

Also be sure to drop the foreign key in the target database for the full load to run. Refer to the **Migrate the sample schema** section of the article [here](#) for a script that you can use to drop the foreign key. Use Azure Database Migration Service to run for full load and sync.

When the PostgreSQL table schema already exists

If you create a PostgreSQL schema using tools such as ora2pg before starting the data movement with Azure Database Migration Service, map the source tables to the target tables in Azure Database Migration Service.

1. When you create a new Oracle to Azure Database for PostgreSQL migration project, you're prompted to select target database and target schema in Select schemas step. Fill in the target database and target schema.

SOURCE SCHEMA	TARGET DATABASE	TARGET SCHEMA
HR	hr	public
LHTA_LT1		Hr
Lpu		
LPU		

2. The **Migration settings** screen presents a list of tables in the Oracle source. Azure Database Migration Service tries to match tables in the source and the target tables based on table name. If multiple matching target tables with different casing exist, you can select which target table to map to.

NAME	TARGET	Notes
HR.LOCATIONS	public.locations	Target table is not empty... ⓘ
HR.EMPLOYEES	public.employees	Target table is not empty... ⓘ
HR.JOB_HISTORY	public.job_history	Target table is not empty... ⓘ
HR.JOB	public.jobs	Target table is not empty... ⓘ
HR.DEPARTMENTS	public.departments	Target table is not empty... ⓘ
HR.REGION	public.regions	Target table is not empty... ⓘ
HR.COUNTRY	public.countries	Target table is not empty... ⓘ

NOTE

If you need to map source table names to tables with different names, email dmsfeedback@microsoft.com and we can provide a script to automate the process.

When the PostgreSQL table schema doesn't exist

If the target PostgreSQL database doesn't contain any table schema information, Azure Database Migration Service converts the source schema and recreates it in the target database. Remember, Azure Database Migration Service creates only the table schema, not other database objects such as stored procedures, packages, and indexes. To have Azure Database Migration Service create the schema for you, ensure that your target environment includes a schema with no existing tables. If Azure Database Migration Service discovers any table, the service assumes that the schema was created by an external tool such as ora2pg.

IMPORTANT

Azure Database Migration Service requires that all tables be created the same way, by using either Azure Database Migration Service or a tool such as ora2pg, but not both.

To get started:

1. Create a schema in the target database based on your application requirements. By default, PostgreSQL table schema and columns names are lower cased. Oracle table schema and columns, on the other hand, are by default in all capital case.
2. In Select schemas step, specify the target database and the target schema.
3. Based on the schema you create in Azure Database for PostgreSQL, Azure Database Migration Service uses the following transformation rules:

If the schema name in the Oracle source and matches that in Azure Database for PostgreSQL, then Azure Database Migration Service *creates the table schema using the same case as in the target*.

For example:

SOURCE ORACLE SCHEMA	TARGET POSTGRESQL DATABASE.SCHEMA	DMS CREATED SCHEMA.TABLE.COLUMN
HR	targetHR.public	public.countries.country_id
HR	targetHR.trgthr	trgthr.countries.country_id
HR	targetHR.TARGETHR	"TARGETHR"."COUNTRIES"."COUNTR Y_ID"
HR	targetHR.HR	"HR"."COUNTRIES"."COUNTRY_ID"
HR	targetHR.Hr	*Unable to map mixed cases

*To create mixed case schema and table names in target PostgreSQL, contact dmsfeedback@microsoft.com. We can provide a script to set up mixed case table schema in the target PostgreSQL database.

Register the Microsoft.DataMigration resource provider

1. Sign in to the Azure portal, select **All services**, and then select **Subscriptions**.

All services Filter By category ▾

GENERAL (14) —

- Dashboard
- Management Groups PREVIEW
- Cost Management + Billing PREVIEW
- Help + support
- Tags
- All resources
- Subscriptions
- Reservations
- Service Health
- What's new

COMPUTE (20) —

- Virtual machines
- Virtual machines (classic)
- Container services
- Function Apps

2. Select the subscription in which you want to create the instance of the Azure Database Migration Service, and then select **Resource providers**.

Home > Subscriptions > <subscription>

Subscriptions Microsoft

Add

My role: Owner Status: 3 selected

Apply

Search to filter items...

SUBSCRIPTI... ↑↓ SUBSCRIPTION ID ↑↓

<subscription> <subscription ID>

Overview

Access control (IAM)

Diagnose and solve problems

COST MANAGEMENT + BILLING

Partner information

SETTINGS

Programmatic deployment

Resource groups

Resources

Usage + quotas

Policies

Management certificates

My permissions

Resource providers

Properties

Resource locks

SUPPORT + TROUBLESHOOTING

New support request

3. Search for migration, and then to the right of **Microsoft.DataMigration**, select **Register**.

The screenshot shows the Azure portal interface. On the left, the navigation menu includes 'Create a resource', 'All services', 'Dashboard', 'Resource groups', 'App Services', 'Function Apps', 'SQL databases', 'Azure Cosmos DB', 'Virtual machines', 'Load balancers', 'Storage accounts', 'Virtual networks', 'Azure Active Directory', 'Monitor', and 'Advisor'. The main content area is titled '<Subscription>' and shows a list of registered providers. One provider, 'Microsoft.DataMigration', is listed with the status 'NotRegistered'. A red box highlights the 'Register' button next to it.

Create a DMS instance

1. In the Azure portal, select **+ Create a resource**, search for Azure Database Migration Service, and then select **Azure Database Migration Service** from the drop-down list.

The screenshot shows the Azure portal's 'New' blade. The search bar at the top contains the text 'Azure Database'. Below the search bar, the results list shows 'Azure Database Migration Service' highlighted with a red box. Other results include 'Azure Database for MySQL' and 'Azure Database for PostgreSQL'. The main pane displays various service categories like Networking, Storage, Web + Mobile, etc., with their respective icons and quickstart tutorials.

2. On the **Azure Database Migration Service** screen, select **Create**.

Create a resource

Home

Dashboard

All services

FAVORITES

Resource groups

All resources

Recent

Virtual machines

App Services

SQL databases

Virtual machines (classic)

Cloud services (classic)

Subscriptions

SQL servers

Azure Active Directory

Monitor

Security Center

Cost Management + Billing

Help + support

Advisor

Azure Database Migration Service

Microsoft

Azure Database Migration Service
Microsoft

Create Saved

The Azure Database Migration Service (DMS) is designed to streamline the process of migrating on-premise databases to Azure. Get started with [step-by-step guidance](#).

Common scenarios:

- SQL Server → [Azure SQL Database](#)
- SQL Server → [Azure SQL Database Managed Instance](#)
- MongoDB → [Azure Cosmos DB](#)
- MySQL → [Azure Database for MySQL](#)
- PostgreSQL → [Azure Database for PostgreSQL](#)
- DB2 → [Azure SQL Database](#)
- Oracle → [Azure SQL Database](#) (requires preview, sign up [here](#))
- Oracle → [Azure SQL Database Managed Instance](#) (requires preview, sign up [here](#))
- Oracle → [Azure Database for PostgreSQL](#) (requires [ora2pg](#))

Don't see your migration scenario here? Find it in the [Azure Database Migration Guide](#)

Once you've completed the step by step guidance for your migration scenario, proceed with creating a migration service by clicking the **Create** button below.

Additional resources:

Use these tools to assess your database(s) for feature parity and potential compatibility issues.

- SQL Server on-premise database(s): [Data Migration Assistant \(DMA\)](#)
- Migrating from Oracle: [SQL Server Migration Assistant \(SSMA\)](#)

[Useful Links](#)
[Documentation](#)
[Privacy Statement](#)

3. On the **Create Migration Service** screen, specify a name for the service, the subscription, and a new or existing resource group.
4. Select an existing virtual network or create a new one.

The virtual network provides Azure Database Migration Service with access to the source Oracle and the target Azure Database for PostgreSQL instance.

For more information about how to create a virtual network in the Azure portal, see the article [Create a virtual network using the Azure portal](#).

5. Select a pricing tier.

For more information on costs and pricing tiers, see the [pricing page](#).

Create Migration Service

Service Name * ✓

* Subscription

* Select a resource group * ✓

[Create new](#)

* Location *

* Virtual network >

* Pricing tier >

Azure Database Migration Service quick start template ↗
Experience our database migration service with pre-created source and target

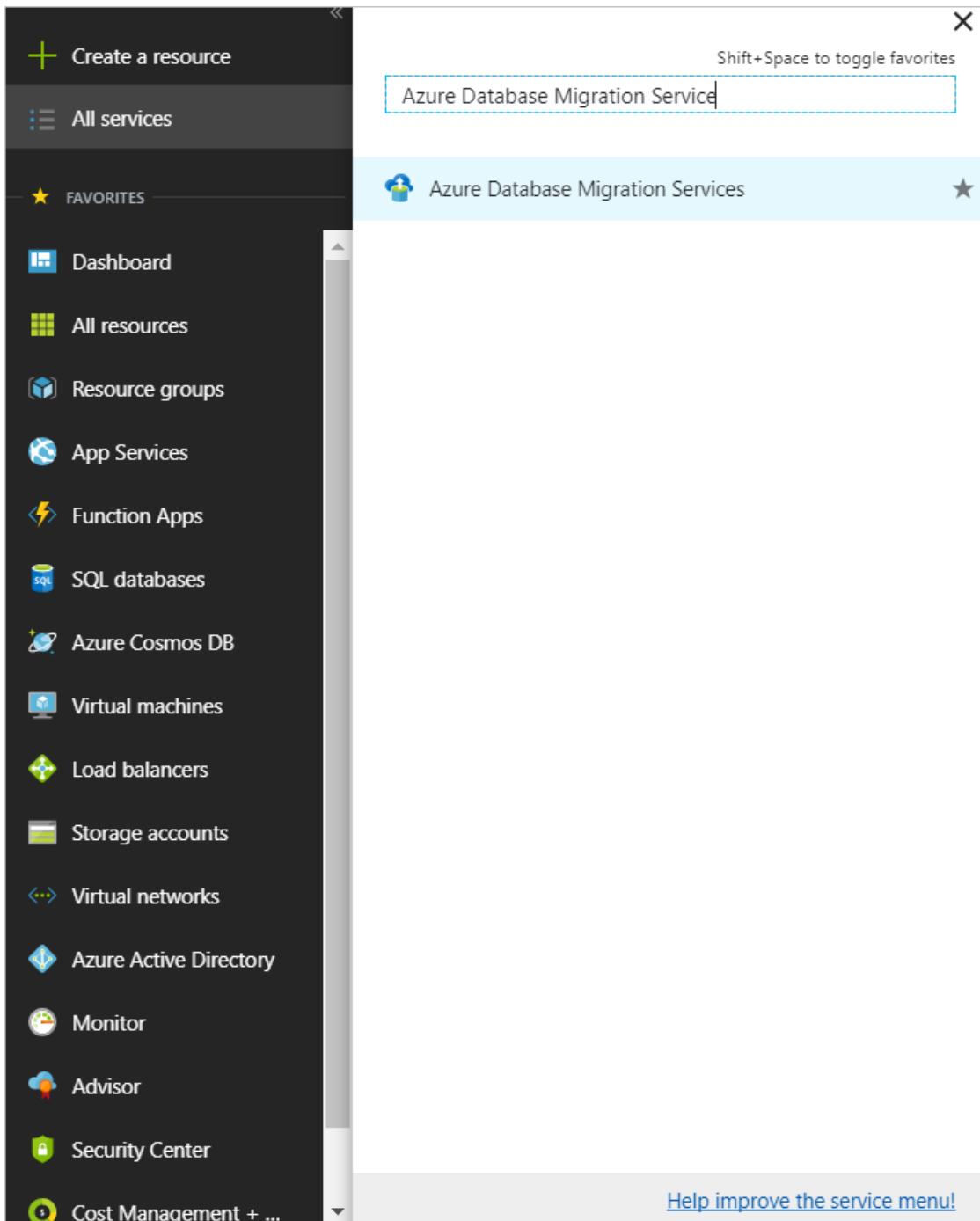
Create [Automation options](#)

6. Select **Create** to create the service.

Create a migration project

After the service is created, locate it within the Azure portal, open it, and then create a new migration project.

1. In the Azure portal, select **All services**, search for Azure Database Migration Service, and then select **Azure Database Migration Services**.



2. On the **Azure Database Migration Services** screen, search for the name of the Azure Database Migration Service instance that you created, and then select the instance.

3. Select **+ New Migration Project**.
4. On the **New migration project** screen, specify a name for the project, in the **Source server type** text box, select **Oracle**, in the **Target server type** text box, select **Azure Database for PostgreSQL**.
5. In the **Choose type of activity** section, select **Online data migration**.

Project name
ShauTest

*** Source server type**
Oracle

*** Target server type**
Azure Database for PostgreSQL

*** Choose type of activity**
Online data migration [preview]

Type of activity

Choose type of activity
Online data migration [preview]

Use this option to migrate databases that must be accessible and continuously updated during migration.

Click here for the overview of Oracle to Azure Database for PostgreSQL migration process which includes Oracle database objects assessment, conversion, data movement and post-migration steps.

To continuously replicate data changes from your source to your target, please implement the steps below on your source server. These steps can be implemented anytime between the moment you create a project and the moment you start a migration activity. After your migration is complete, you will reverse those preparation steps.

- Enable archive redo logs to capture data changes for incremental sync.
- Enable supplemental logging needed for replication.
- Oracle Version 10g or 11g is supported in public preview of Azure Database Migration Service (DMS). For customers with Oracle version 12c, please click [here](#).

NOTE

Alternately, you can choose **Create project only** to create the migration project now and execute the migration later.

6. Select **Save**, note the requirements to successfully use Azure Database Migration Service to perform an

online migration, and then select **Create and run activity**.

Specify source details

- On the **Add Source Details** screen, specify the connection details for the source Oracle instance.

The screenshot shows the 'Add Source Details' dialog box. On the left, there is a vertical list of steps: 1. Select source, 2. Install OCI driver, 3. Select target, 4. Select schemas, and 5. Summary. The 'Source server name' field is populated with '<server>.redmond.corp.microsoft.co'. The 'Server port' is set to '1521'. The 'Oracle SID' is 'STD'. The 'User Name' is 'system'. The 'Password' field contains several redacted dots. At the bottom of the dialog is a blue 'Save' button.

Upload Oracle OCI driver

- Select **Save**, and then on the **Install OCI driver** screen, sign into your Oracle account and download the driver **instantclient-basiclite-windows.x64-12.2.0.1.0.zip** (37,128,586 Byte(s)) (SHA1 Checksum: 865082268) from [here](#).
- Download the driver to a shared folder.

Make sure the folder is shared with the username that you specified with minimum Read-only access. Azure Database Migration Service accesses and reads from the share to upload the OCI driver to Azure by impersonating the username you specify.

The username you specify must be a Windows user account.

The screenshot shows a 'Driver install detail' page with a sidebar on the left listing five steps:

- 1 Select source (status: ✓)
- 2 Install OCI driver
- 3 Select target
- 4 Select schemas
- 5 Summary

The main panel displays the following information:

Step 1:
Sign into your Oracle account and download the following driver.
Driver
instantclient-basiclite-windows.x64-12.2.0.1.0.zip (37,128,586 Byte(s))
(SHA1 Checksum: 865082268)
[Oracle driver download](#)

Step 2:
Upload the driver to Azure.

* OCI driver path.

* User Name

Password

Save

Specify target details

1. Select **Save**, and then on the **Target details** screen, specify the connection details for the target Azure Database for PostgreSQL server, which is the pre-provisioned instance of Azure Database for PostgreSQL to which the **HR** schema was deployed.

Home > shuhuan_BDSCUS_Bash3 > ShauOracle (shuhuan_BDSCUS_Bash3/ShauOracle) > ShauOracle >

ShauOracle		X	Migration target details		□ X
1	Select source	✓	Target server name ⓘ <code><server> postgres.database.azure.com</code>		
2	Install OCI driver	✓	Default Database <code>postgres</code>		
3	Select target	✓	User Name <code><username></code>		
4	Select schemas	>	Password *****		
5	Summary	>			
Save					

2. Select **Save**, and then on the **Map to target databases** screen, map the source and the target database for migration.

If the target database contains the same database name as the source database, Azure Database Migration Service selects the target database by default.

Home > shuhuan_BDSCUS_Bash3 > ShauOracle (shuhuan_BDSCUS_Bash3/ShauOracle) > ShauOracle > Map to target databases

ShauOracle		X	Map to target databases		X														
1	Select source	✓	<input type="text" value="Search"/> 6 item(s) ← prev Page 1 of 1 next → <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="background-color: #e0f2ff;">SOURCE SCHEMA</th> <th style="background-color: #e0f2ff;">TARGET DATABASE</th> </tr> </thead> <tbody> <tr> <td><input type="checkbox"/> DISCOVERY</td> <td></td> </tr> <tr> <td><input checked="" type="checkbox"/> HR</td> <td>HR</td> </tr> <tr> <td><input type="checkbox"/> TPCC1000W</td> <td></td> </tr> <tr> <td><input type="checkbox"/> TPCC100W</td> <td></td> </tr> <tr> <td><input type="checkbox"/> TPCC10W</td> <td></td> </tr> <tr> <td><input type="checkbox"/> TPCC1W</td> <td></td> </tr> </tbody> </table>			SOURCE SCHEMA	TARGET DATABASE	<input type="checkbox"/> DISCOVERY		<input checked="" type="checkbox"/> HR	HR	<input type="checkbox"/> TPCC1000W		<input type="checkbox"/> TPCC100W		<input type="checkbox"/> TPCC10W		<input type="checkbox"/> TPCC1W	
SOURCE SCHEMA	TARGET DATABASE																		
<input type="checkbox"/> DISCOVERY																			
<input checked="" type="checkbox"/> HR	HR																		
<input type="checkbox"/> TPCC1000W																			
<input type="checkbox"/> TPCC100W																			
<input type="checkbox"/> TPCC10W																			
<input type="checkbox"/> TPCC1W																			
2	Install OCI driver	✓																	
3	Select target	✓																	
4	Select schemas	>																	
5	Summary	>																	
<input type="button" value="Save"/>																			

3. Select **Save**, on the **Migration summary** screen, in the **Activity name** text box, specify a name for the migration activity, and then review the summary to ensure that the source and target details match what you previously specified.

Home > shuhuan_BDSCUS_Bash3 > ShauOracle (shuhuan_BDSCUS_Bash3/ShauOracle) > ShauOracle > Migration summary

ShauOracle		X	Migration summary		X
1	Select source	✓	Activity name <input type="text" value="runnow"/> ✓		
2	Install OCI driver	✓	Source server name <server> redmond.corp.microsoft.com/STD Source server version Oracle 12.2.0.1.0		
3	Select target	✓	Target server name <server> postgres.database.azure.com Target server version Azure Database for PostgreSQL 10.7		
4	Select schemas	✓	Schema to migrate 1 of 6		
5	Summary	>	Type of activity Online data migration [preview]		
Run migration					

Run the migration

- Select **Run migration**.

The migration activity window appears, and the **Status** of the activity is **initializing**.

Monitor the migration

- On the migration activity screen, select **Refresh** to update the display until the **Status** of the migration shows as **Running**.

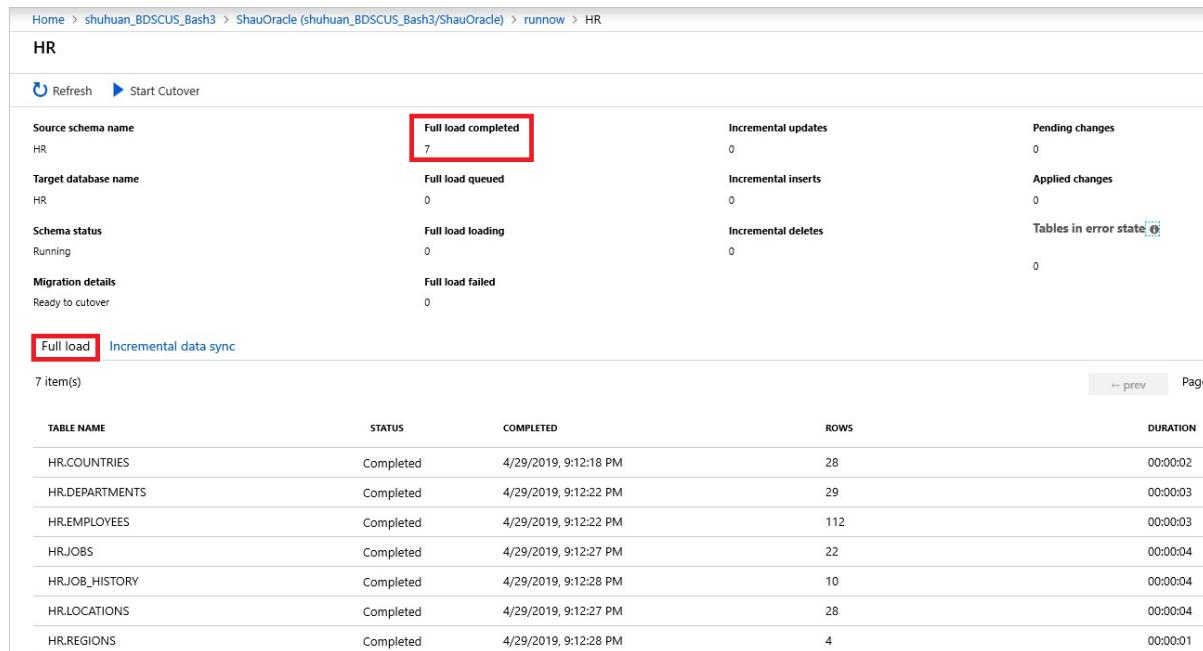
Home > shuhuan_BDSCUS_Bash3 > ShauOracle (shuhuan_BDSCUS_Bash3/ShauOracle) > runnow

runnow					
<input type="button" value="⟳ Refresh"/>	<input type="button" value="⟳ Retry"/>	<input type="button" value="Stop migration"/>	<input type="button" value="Delete activity"/>	<input type="button" value="Download report"/>	
Source server	Source version	Source schemas			
//AAALAB03-12cR2.redmond.corp.microsoft.com:1521/STD	Oracle 12.2	1			
Target server	Target version	Type of activity			
ora2pgsqlshau.postgres.database.azure.com	Azure Database for PostgreSQL 10.7	Online			
Activity status	Duration				
Running	00:07:13				
SCHEMA NAME	STATUS	MIGRATION DETAILS	DURATION	ESTIMATED APPLICATION DOWNTIME ⓘ	FINISH DATE
HR	Running	Ready to cutover	00:07:13	00:00:18	---

- Under **Database Name**, select a specific database to get to the migration status for **Full data load** and **Incremental data sync** operations.

Full data load will show the initial load migration status while Incremental data sync will show change data

capture (CDC) status.



Home > shuhuan_BDSCUS_Bash3 > ShauOracle (shuhuan_BDSCUS_Bash3/ShauOracle) > runnow > HR

HR

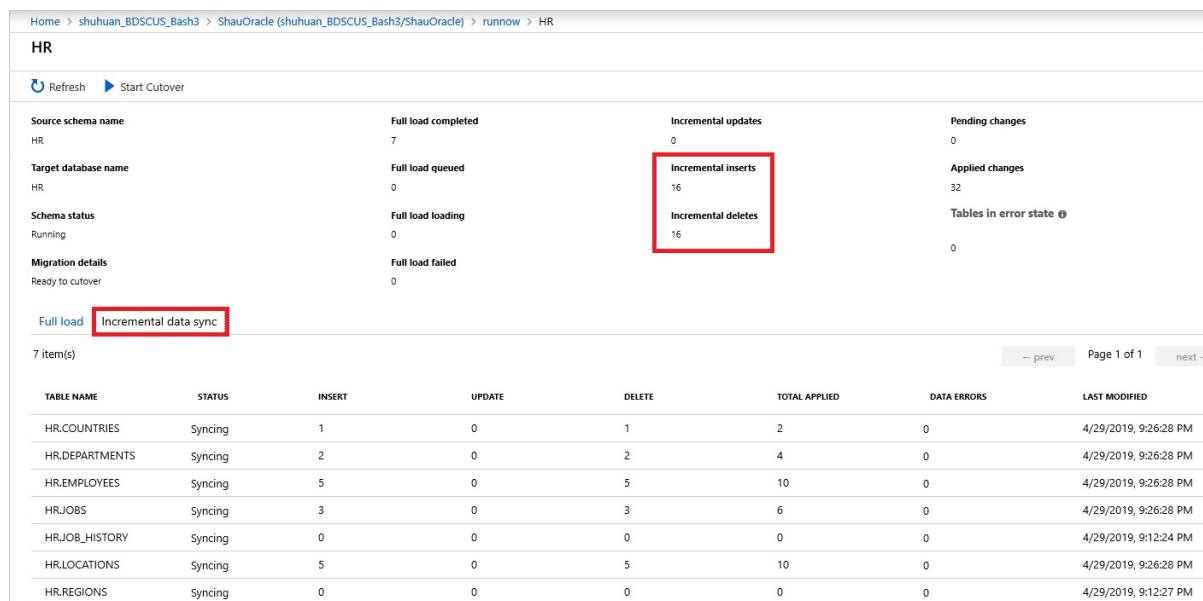
Refresh ▶ Start Cutover

Source schema name	Full load completed 7	Incremental updates 0	Pending changes 0
Target database name	Full load queued 0	Incremental inserts 0	Applied changes 0
Schema status	Full load loading Running	Incremental deletes 0	Tables in error state ⓘ 0
Migration details	Full load failed Ready to cutover	0	

Full load Incremental data sync

7 item(s)

TABLE NAME	STATUS	COMPLETED	ROWS	DURATION
HR.COUNTRIES	Completed	4/29/2019, 9:12:18 PM	28	00:00:02
HR.DEPARTMENTS	Completed	4/29/2019, 9:12:22 PM	29	00:00:03
HR.EMPLOYEES	Completed	4/29/2019, 9:12:22 PM	112	00:00:03
HR.JOBS	Completed	4/29/2019, 9:12:27 PM	22	00:00:04
HRJOB_HISTORY	Completed	4/29/2019, 9:12:28 PM	10	00:00:04
HR.LOCATIONS	Completed	4/29/2019, 9:12:27 PM	28	00:00:04
HR.REGION	Completed	4/29/2019, 9:12:28 PM	4	00:00:01



Home > shuhuan_BDSCUS_Bash3 > ShauOracle (shuhuan_BDSCUS_Bash3/ShauOracle) > runnow > HR

HR

Refresh ▶ Start Cutover

Source schema name	Full load completed 7	Incremental updates 0	Pending changes 0
Target database name	Full load queued 0	Incremental inserts 16	Applied changes 32
Schema status	Full load loading Running	Incremental deletes 16	Tables in error state ⓘ 0
Migration details	Full load failed Ready to cutover	0	

Full load Incremental data sync

7 item(s)

TABLE NAME	STATUS	INSERT	UPDATE	DELETE	TOTAL APPLIED	DATA ERRORS	LAST MODIFIED
HR.COUNTRIES	Syncing	1	0	1	2	0	4/29/2019, 9:26:28 PM
HR.DEPARTMENTS	Syncing	2	0	2	4	0	4/29/2019, 9:26:28 PM
HR.EMPLOYEES	Syncing	5	0	5	10	0	4/29/2019, 9:26:28 PM
HR.JOBS	Syncing	3	0	3	6	0	4/29/2019, 9:26:28 PM
HRJOB_HISTORY	Syncing	0	0	0	0	0	4/29/2019, 9:12:24 PM
HR.LOCATIONS	Syncing	5	0	5	10	0	4/29/2019, 9:26:28 PM
HR.REGION	Syncing	0	0	0	0	0	4/29/2019, 9:12:27 PM

Perform migration cutover

After the initial Full load is completed, the databases are marked **Ready to cutover**.

- When you're ready to complete the database migration, select **Start Cutover**.
- Make sure to stop all the incoming transactions to the source database; wait until the **Pending changes** counter shows **0**.

Complete cutover

When you are ready to do the migration cutover, perform the following steps to complete the database migration. Please note that the database is ready for cutover only after the full data load is completed.

1. Stop all the incoming transactions coming to the source database.
2. Wait until all the pending transactions have been applied to the target database. At that time the pending changes counter will set to 0:

Pending changes 0

Confirm

Apply

3. By default database objects in PostgreSQL are case insensitive and are stored as lowercase. If you like to change the schema, table and column from UPPER CASE to lower case, [follow this script in this document](#).
4. Reconnect your applications to the new Azure target database.

3. Select **Confirm**, and then select **Apply**.
4. When the database migration status shows **Completed**, connect your applications to the new target Azure Database for PostgreSQL instance.

NOTE

Since PostgreSQL by default has schema.table.column in lower case, you can revert from upper case to lower case by using the script in the **Set up the schema in Azure Database for PostgreSQL** section earlier in this article.

Next steps

- For information about known issues and limitations when performing online migrations to Azure Database for PostgreSQL, see the article [Known issues and workarounds with Azure Database for PostgreSQL online migrations](#).
- For information about the Azure Database Migration Service, see the article [What is Azure Database Migration Service?](#).
- For information about Azure Database for PostgreSQL, see the article [What is Azure Database for PostgreSQL?](#).

Migrate SQL Server on-premises to Azure SQL Database using Azure PowerShell

2/26/2020 • 7 minutes to read • [Edit Online](#)

In this article, you migrate the **Adventureworks2012** database restored to an on-premises instance of SQL Server 2016 or above to an Azure SQL Database by using Microsoft Azure PowerShell. You can migrate databases from an on-premises SQL Server instance to Azure SQL Database by using the `Az.DataMigration` module in Microsoft Azure PowerShell.

In this article, you learn how to:

- Create a resource group.
- Create an instance of the Azure Database Migration Service.
- Create a migration project in an Azure Database Migration Service instance.
- Run the migration.

Prerequisites

To complete these steps, you need:

- [SQL Server 2016 or above](#) (any edition)
- To enable the TCP/IP protocol, which is disabled by default with SQL Server Express installation. Enable the TCP/IP protocol by following the article [Enable or Disable a Server Network Protocol](#).
- To configure your [Windows Firewall for database engine access](#).
- An Azure SQL Database instance. You can create an Azure SQL Database instance by following the detail in the article [Create an Azure SQL database in the Azure portal](#).
- [Data Migration Assistant v3.3 or later](#).
- To have created a Microsoft Azure Virtual Network by using the Azure Resource Manager deployment model, which provides the Azure Database Migration Service with site-to-site connectivity to your on-premises source servers by using either [ExpressRoute](#) or [VPN](#).
- To have completed assessment of your on-premises database and schema migration using Data Migration Assistant as described in the article [Performing a SQL Server migration assessment](#)
- To download and install the `Az.DataMigration` module from the PowerShell Gallery by using [Install-Module PowerShell cmdlet](#); be sure to open the powershell command window using run as an Administrator.
- To ensure that the credentials used to connect to source SQL Server instance has the [CONTROL SERVER](#) permission.
- To ensure that the credentials used to connect to target Azure SQL DB instance has the [CONTROL DATABASE](#) permission on the target Azure SQL Database databases.
- An Azure subscription. If you don't have one, create a [free](#) account before you begin.

Log in to your Microsoft Azure subscription

Use the directions in the article [Log in with Azure PowerShell](#) to sign in to your Azure subscription by using PowerShell.

Create a resource group

An Azure resource group is a logical container into which Azure resources are deployed and managed. Create a

resource group before you can create a virtual machine.

Create a resource group by using the [New-AzResourceGroup](#) command.

The following example creates a resource group named *myResourceGroup* in the *EastUS* region.

```
New-AzResourceGroup -ResourceGroupName myResourceGroup -Location EastUS
```

Create an instance of Azure Database Migration Service

You can create new instance of Azure Database Migration Service by using the [New-AzDataMigrationService](#) cmdlet. This cmdlet expects the following required parameters:

- *Azure Resource Group name*. You can use [New-AzResourceGroup](#) command to create Azure Resource group as previously shown and provide its name as a parameter.
- *Service name*. String that corresponds to the desired unique service name for Azure Database Migration Service
- *Location*. Specifies the location of the service. Specify an Azure data center location, such as West US or Southeast Asia
- *Sku*. This parameter corresponds to DMS Sku name. The currently supported Sku name is *GeneralPurpose_4vCores*.
- *Virtual Subnet Identifier*. You can use cmdlet [New-AzVirtualNetworkSubnetConfig](#) to create a subnet.

The following example creates a service named *MyDMS* in the resource group *MyDMSResourceGroup* located in the *East US* region using a virtual network named *MyVNET* and subnet called *MySubnet*.

```
$vNet = Get-AzVirtualNetwork -ResourceGroupName MyDMSResourceGroup -Name MyVNET  
  
$vSubNet = Get-AzVirtualNetworkSubnetConfig -VirtualNetwork $vNet -Name MySubnet  
  
$service = New-AzDms -ResourceGroupName myResourceGroup `  
    -ServiceName MyDMS `  
    -Location EastUS `  
    -Sku Basic_2vCores `  
    -VirtualSubnetId $vSubNet.Id`
```

Create a migration project

After creating an Azure Database Migration Service instance, create a migration project. An Azure Database Migration Service project requires connection information for both the source and target instances, as well as a list of databases that you want to migrate as part of the project.

Create a Database Connection Info object for the source and target connections

You can create a Database Connection Info object by using the [New-AzDmsConnInfo](#) cmdlet. This cmdlet expects the following parameters:

- *ServerType*. The type of database connection requested, for example, SQL, Oracle, or MySQL. Use SQL for SQL Server and Azure SQL.
- *DataSource*. The name or IP of a SQL Server instance or Azure SQL database.
- *AuthType*. The authentication type for connection, which can be either SqlAuthentication or WindowsAuthentication.
- *TrustServerCertificate* parameter sets a value that indicates whether the channel is encrypted while bypassing walking the certificate chain to validate trust. Value can be true or false.

The following example creates Connection Info object for source SQL Server called MySourceSQLServer using sql authentication:

```
$sourceConnInfo = New-AzDmsConnInfo -ServerType SQL  
-DataSource MySourceSQLServer  
-AuthType SqlAuthentication  
-TrustServerCertificate:$true
```

The next example shows creation of Connection Info for an Azure SQL database server called SQLAzureTarget using sql authentication:

```
$targetConnInfo = New-AzDmsConnInfo -ServerType SQL  
-DataSource "sqlazurereadtarget.database.windows.net"  
-AuthType SqlAuthentication  
-TrustServerCertificate:$false
```

Provide databases for the migration project

Create a list of `AzDataMigrationDatabaseInfo` objects that specifies databases as part of the Azure Database Migration project that can be provided as parameter for creation of the project. The Cmdlet `New-AzDataMigrationDatabaseInfo` can be used to create `AzDataMigrationDatabaseInfo`.

The following example creates `AzDataMigrationDatabaseInfo` project for the **AdventureWorks2016** database and adds it to the list to be provided as parameter for project creation.

```
$dbInfo1 = New-AzDataMigrationDatabaseInfo -SourceDatabaseName AdventureWorks2016  
$dbList = @($dbInfo1)
```

Create a project object

Finally you can create Azure Database Migration project called *MyDMSProject* located in *East US* using `New-AzDataMigrationProject` and adding the previously created source and target connections and the list of databases to migrate.

```
$project = New-AzDataMigrationProject -ResourceGroupName myResourceGroup  
-ServiceName $service.Name  
-ProjectName MyDMSProject  
-Location EastUS  
-SourceType SQL  
-TargetType SQLDB  
-SourceConnection $sourceConnInfo  
-TargetConnection $targetConnInfo  
-DatabaseInfo $dbList
```

Create and start a migration task

Finally, create and start Azure Database Migration task. Azure Database Migration task requires connection credential information for both source and target and list of database tables to be migrated in addition to the information already provided with the project created as a prerequisite.

Create credential parameters for source and target

Connection security credentials can be created as a `PSCredential` object.

The following example shows the creation of `PSCredential` objects for both source and target connections providing passwords as string variables `$sourcePassword` and `$targetPassword`.

```
$secpasswd = ConvertTo-SecureString -String $sourcePassword -AsPlainText -Force
$sourceCred = New-Object System.Management.Automation.PSCredential ($sourceUserName, $secpasswd)
$secpasswd = ConvertTo-SecureString -String $targetPassword -AsPlainText -Force
$targetCred = New-Object System.Management.Automation.PSCredential ($targetUserName, $secpasswd)
```

Create a table map and select source and target parameters for migration

Another parameter needed for migration is mapping of tables from source to target to be migrated. Create dictionary of tables that provides a mapping between source and target tables for migration. The following example illustrates mapping between source and target tables Human Resources schema for the AdventureWorks 2016 database.

```
$tableMap = New-Object 'System.Collections.Generic.Dictionary[string,string]'
$tableMap.Add("HumanResources.Department", "HumanResources.Department")
$tableMap.Add("HumanResources.Employee", "HumanResources.Employee")
$tableMap.Add("HumanResources.EmployeeDepartmentHistory", "HumanResources.EmployeeDepartmentHistory")
$tableMap.Add("HumanResources.EmployeePayHistory", "HumanResources.EmployeePayHistory")
$tableMap.Add("HumanResources.JobCandidate", "HumanResources.JobCandidate")
$tableMap.Add("HumanResources.Shift", "HumanResources.Shift")
```

The next step is to select the source and target databases and provide table mapping to migrate as a parameter by using the `New-AzDmsSelectedDB` cmdlet, as shown in the following example:

```
$selectedDbs = New-AzDmsSelectedDB -MigrateSqlServerSqlDb -Name AdventureWorks2016 ` 
    -TargetDatabaseName AdventureWorks2016 ` 
    -TableMap $tableMap
```

Create the migration task and start it

Use the `New-AzDataMigrationTask` cmdlet to create and start a migration task. This cmdlet expects the following parameters:

- *TaskType*. Type of migration task to create for SQL Server to Azure SQL Database migration type `MigrateSqlServerSqlDb` is expected.
- *Resource Group Name*. Name of Azure resource group in which to create the task.
- *ServiceName*. Azure Database Migration Service instance in which to create the task.
- *ProjectName*. Name of Azure Database Migration Service project in which to create the task.
- *TaskName*. Name of task to be created.
- *SourceConnection*. `AzDmsConnInfo` object representing source SQL Server connection.
- *TargetConnection*. `AzDmsConnInfo` object representing target Azure SQL Database connection.
- *SourceCred*. `PSCredential` object for connecting to source server.
- *TargetCred*. `PSCredential` object for connecting to target server.
- *SelectedDatabase*. `AzDataMigrationSelectedDB` object representing the source and target database mapping.
- *SchemaValidation*. (optional, switch parameter) Following the migration, performs a comparison of the schema information between source and target.
- *DataIntegrityValidation*. (optional, switch parameter) Following the migration, performs a checksum-based data integrity validation between source and target.
- *QueryAnalysisValidation*. (optional, switch parameter) Following the migration, performs a quick and intelligent query analysis by retrieving queries from the source database and executes them in the target.

The following example creates and starts a migration task named myDMSTask:

```
$migTask = New-AzDataMigrationTask -TaskType MigrateSqlServerSqlDb `  
-ResourceGroupName myResourceGroup `  
-ServiceName $service.Name `  
-ProjectName $project.Name `  
-TaskName myDMSTask `  
-SourceConnection $sourceConnInfo `  
-SourceCred $sourceCred `  
-TargetConnection $targetConnInfo `  
-TargetCred $targetCred `  
-SelectedDatabase $selectedDbs `
```

The following example creates and starts the same migration task as above but also performs all three validations:

```
$migTask = New-AzDataMigrationTask -TaskType MigrateSqlServerSqlDb `  
-ResourceGroupName myResourceGroup `  
-ServiceName $service.Name `  
-ProjectName $project.Name `  
-TaskName myDMSTask `  
-SourceConnection $sourceConnInfo `  
-SourceCred $sourceCred `  
-TargetConnection $targetConnInfo `  
-TargetCred $targetCred `  
-SelectedDatabase $selectedDbs `  
-SchemaValidation `  
-DataIntegrityValidation `  
-QueryAnalysisValidation `
```

Monitor the migration

You can monitor the migration task running by querying the state property of the task as shown in the following example:

```
if (($mytask.ProjectTask.Properties.State -eq "Running") -or ($mytask.ProjectTask.Properties.State -eq  
"Queued"))  
{  
    write-host "migration task running"  
}
```

Deleting the DMS instance

After the migration is complete, you can delete the Azure DMS instance:

```
Remove-AzDms -ResourceGroupName myResourceGroup -ServiceName MyDMS
```

Next step

- Review the migration guidance in the Microsoft [Database Migration Guide](#).

Migrate SQL Server to SQL Database managed instance with PowerShell & Azure Database Migration Service

2/26/2020 • 11 minutes to read • [Edit Online](#)

In this article, you migrate the **Adventureworks2016** database restored to an on-premises instance of SQL Server 2005 or above to an Azure SQL Database managed instance by using Microsoft Azure PowerShell. You can migrate databases from an on-premises SQL Server instance to an Azure SQL Database managed instance by using the `Az.DataMigration` module in Microsoft Azure PowerShell.

In this article, you learn how to:

- Create a resource group.
- Create an instance of Azure Database Migration Service.
- Create a migration project in an instance of Azure Database Migration Service.
- Run the migration.

TIP

When you migrate databases to Azure by using Azure Database Migration Service, you can do an *offline* or an *online* migration. With an offline migration, application downtime starts when the migration starts. With an online migration, downtime is limited to the time to cut over at the end of migration. We suggest that you test an offline migration to determine whether the downtime is acceptable; if not, do an online migration.

This article includes detail on how to perform both online and offline migrations.

Prerequisites

To complete these steps, you need:

- [SQL Server 2016 or above](#) (any edition).
- A local copy of the **AdventureWorks2016** database, which is available for download [here](#).
- To enable the TCP/IP protocol, which is disabled by default with SQL Server Express installation. Enable the TCP/IP protocol by following the article [Enable or Disable a Server Network Protocol](#).
- To configure your [Windows Firewall for database engine access](#).
- An Azure subscription. If you don't have one, [create a free account](#) before you begin.
- An Azure SQL Database managed instance. You can create an Azure SQL Database managed instance by following the detail in the article [Create an Azure SQL Database managed instance](#).
- To download and install [Data Migration Assistant](#) v3.3 or later.
- A Microsoft Azure Virtual Network created using the Azure Resource Manager deployment model, which provides the Azure Database Migration Service with site-to-site connectivity to your on-premises source servers by using either [ExpressRoute](#) or [VPN](#).
- A completed assessment of your on-premises database and schema migration using Data Migration

Assistant, as described in the article [Performing a SQL Server migration assessment](#).

- To download and install the `Az.DataMigration` module (version 0.7.2 or later) from the PowerShell Gallery by using [Install-Module PowerShell cmdlet](#).
- To ensure that the credentials used to connect to source SQL Server instance have the [CONTROL SERVER](#) permission.
- To ensure that the credentials used to connect to target Azure SQL Database managed instance has the [CONTROL DATABASE](#) permission on the target Azure SQL Database managed instance databases.

IMPORTANT

For online migrations, you must already have set up your Azure Active Directory credentials. For more information, see the article [Use the portal to create an Azure AD application and service principal that can access resources](#).

Sign in to your Microsoft Azure subscription

Sign in to your Azure subscription by using PowerShell. For more information, see the article [Sign in with Azure PowerShell](#).

Create a resource group

An Azure resource group is a logical container in which Azure resources are deployed and managed.

Create a resource group by using the `New-AzResourceGroup` command.

The following example creates a resource group named *myResourceGroup* in the *East US* region.

```
New-AzResourceGroup -ResourceGroupName myResourceGroup -Location EastUS
```

Create an instance of Azure Database Migration Service

You can create new instance of Azure Database Migration Service by using the `New-AzDataMigrationService` cmdlet. This cmdlet expects the following required parameters:

- *Azure Resource Group name*. You can use `New-AzResourceGroup` command to create an Azure Resource group as previously shown and provide its name as a parameter.
- *Service name*. String that corresponds to the desired unique service name for Azure Database Migration Service.
- *Location*. Specifies the location of the service. Specify an Azure data center location, such as West US or Southeast Asia.
- *Sku*. This parameter corresponds to DMS Sku name. Currently supported Sku names are *Basic_1vCore*, *Basic_2vCores*, *GeneralPurpose_4vCores*.
- *Virtual Subnet Identifier*. You can use the cmdlet `New-AzVirtualNetworkSubnetConfig` to create a subnet.

The following example creates a service named *MyDMS* in the resource group *MyDMSResourceGroup* located in the *East US* region using a virtual network named *MyVNET* and a subnet named *MySubnet*.

IMPORTANT

The code snippet below is for an offline migration, which does not require an instance of Azure Database Migration Service based on a Premium SKU. For an online migration, the value of the *-Sku* parameter must include a Premium SKU.

```
$vNet = Get-AzVirtualNetwork -ResourceGroupName MyDMSResourceGroup -Name MyVNET

$vSubNet = Get-AzVirtualNetworkSubnetConfig -VirtualNetwork $vNet -Name MySubnet

$service = New-AzDms -ResourceGroupName myResourceGroup `

-ServiceName MyDMS `

-Location EastUS `

-Sku Basic_2vCores `

-VirtualSubnetId $vSubNet.Id`
```

Create a migration project

After creating an Azure Database Migration Service instance, create a migration project. An Azure Database Migration Service project requires connection information for both the source and target instances, as well as a list of databases that you want to migrate as part of the project.

Create a Database Connection Info object for the source and target connections

You can create a Database Connection Info object by using the `New-AzDmsConnInfo` cmdlet, which expects the following parameters:

- *ServerType*. The type of database connection requested, for example, SQL, Oracle, or MySQL. Use SQL for SQL Server and Azure SQL.
- *DataSource*. The name or IP of a SQL Server instance or Azure SQL Database instance.
- *AuthType*. The authentication type for connection, which can be either SqlAuthentication or WindowsAuthentication.
- *TrustServerCertificate*. This parameter sets a value that indicates whether the channel is encrypted while bypassing walking the certificate chain to validate trust. The value can be `$true` or `$false`.

The following example creates a Connection Info object for a source SQL Server called *MySourceSQLServer* using sql authentication:

```
$sourceConnInfo = New-AzDmsConnInfo -ServerType SQL `

-DataSource MySourceSQLServer `

-AuthType SqlAuthentication `

-TrustServerCertificate:$true
```

The next example shows creation of Connection Info for an Azure SQL Database managed instance server named 'targetmanagedinstance.database.windows.net' using sql authentication:

```
$targetConnInfo = New-AzDmsConnInfo -ServerType SQL `

-DataSource "targetmanagedinstance.database.windows.net" `

-AuthType SqlAuthentication `

-TrustServerCertificate:$false
```

Provide databases for the migration project

Create a list of `AzDataMigrationDatabaseInfo` objects that specifies databases as part of the Azure Database Migration Service project, which can be provided as parameter for creation of the project. You can use the cmdlet `New-AzDataMigrationDatabaseInfo` to create `AzDataMigrationDatabaseInfo`.

The following example creates the `AzDataMigrationDatabaseInfo` project for the **AdventureWorks2016** database and adds it to the list to be provided as parameter for project creation.

```
$dbInfo1 = New-AzDataMigrationDatabaseInfo -SourceDatabaseName AdventureWorks  
$dbList = @($dbInfo1)
```

Create a project object

Finally, you can create an Azure Database Migration Service project called *MyDMSProject* located in *East US* using `New-AzDataMigrationProject` and add the previously created source and target connections and the list of databases to migrate.

```
$project = New-AzDataMigrationProject -ResourceGroupName myResourceGroup `  
-ServiceName $service.Name `  
-ProjectName MyDMSProject `  
-Location EastUS `  
-SourceType SQL `  
-TargetType SQLMI `  
-SourceConnection $sourceConnInfo `  
-TargetConnection $targetConnInfo `  
-DatabaseInfo $dbList
```

Create and start a migration task

Next, create and start an Azure Database Migration Service task. This task requires connection credential information for both the source and target, as well as the list of database tables to be migrated and the information already provided with the project created as a prerequisite.

Create credential parameters for source and target

Create connection security credentials as a [PSCredential](#) object.

The following example shows the creation of *PSCredential* objects for both the source and target connections, providing passwords as string variables `$sourcePassword` and `$targetPassword`.

```
$secpasswd = ConvertTo-SecureString -String $sourcePassword -AsPlainText -Force  
$sourceCred = New-Object System.Management.Automation.PSCredential ($sourceUserName, $secpasswd)  
$secpasswd = ConvertTo-SecureString -String $targetPassword -AsPlainText -Force  
$targetCred = New-Object System.Management.Automation.PSCredential ($targetUserName, $secpasswd)
```

Create a backup FileShare object

Now create a FileShare object representing the local SMB network share to which Azure Database Migration Service can take the source database backups using the `New-AzDmsFileShare` cmdlet.

```
$backupPassword = ConvertTo-SecureString -String $password -AsPlainText -Force  
$backupCred = New-Object System.Management.Automation.PSCredential ($backupUserName, $backupPassword)  
  
$backupFilePath="\\10.0.0.76\SharedBackup"  
$backupFileShare = New-AzDmsFileShare -Path $backupFilePath -Credential $backupCred
```

Create selected database object

The next step is to select the source and target databases by using the `New-AzDmsSelectedDB` cmdlet.

The following example is for migrating a single database from SQL Server to an Azure SQL Database managed instance:

```
$selectedDbs = @()
$selectedDbs += New-AzDmsSelectedDB -MigrateSqlServerSqlDbMi ` 
-Name AdventureWorks2016 ` 
-TargetDatabaseName AdventureWorks2016 ` 
-BackupFileShare $backupFileShare `
```

If an entire SQL Server instance needs a lift-and-shift into an Azure SQL Database managed instance, then a loop to take all databases from the source is provided below. In the following example, for \$Server, \$SourceUserName, and \$SourcePassword, provide your source SQL Server details.

```
$Query = "(select name as Database_Name from master.sys.databases where Database_id>4)";
$Databases= (Invoke-Sqlcmd -ServerInstance "$Server" -Username $SourceUserName
-Password $SourcePassword -database master -Query $Query)
$selectedDbs=@()
foreach($DataBase in $Databases.Database_Name)
{
    $SourceDB=$DataBase
    $TargetDB=$DataBase

$selectedDbs += New-AzureRmDmsSelectedDB -MigrateSqlServerSqlDbMi ` 
-Name $SourceDB ` 
-TargetDatabaseName $TargetDB ` 
-BackupFileShare $backupFileShare
}
```

SAS URI for Azure Storage Container

Create variable containing the SAS URI that provides the Azure Database Migration Service with access to the storage account container to which the service uploads the backup files.

```
$blobSasUri="https://mystorage.blob.core.windows.net/test?st=2018-07-13T18%3A10%3A33Z&se=2019-07-14T18%3A10%3A00Z&sp=rw&sv=2018-03-28&sr=c&sig=qK1SA512EVtest3xYjvUg139tYSDrasbftY%3D"
```

Additional configuration requirements

There are a few additional requirements you need to address, but they differ depending on whether you're performing an offline or online migration.

Offline migrations

For offline migrations only, perform the following additional configuration tasks.

- **Select logins.** Create a list of logins to be migrated as shown in the following example:

```
$selectedLogins = @("user1", "user2")
```

IMPORTANT

Currently, Azure Database Migration Service only supports migrating SQL logins.

- **Select agent jobs.** Create list of agent jobs to be migrated as shown in the following example:

```
$selectedAgentJobs = @("agentJob1", "agentJob2")
```

IMPORTANT

Currently, Azure Database Migration Service only supports jobs with T-SQL subsystem job steps.

Online migrations

For online migrations only, perform the following additional configuration tasks.

- **Configure Azure Active Directory App**

```
$pwd = "Azure App Key"  
$appId = "Azure App Id"  
$AppPasswd = ConvertTo-SecureString $pwd -AsPlainText -Force  
$app = New-AzDmsAdApp -ApplicationId $appId -AppKey $AppPasswd
```

- **Configure Storage Resource**

```
$storageResourceId = "Storage Resource Id"
```

Create and start the migration task

Use the `New-AzDataMigrationTask` cmdlet to create and start a migration task.

Specify parameters

Common Parameters

Regardless of whether you're performing an offline or online migration, the `New-AzDataMigrationTask` cmdlet expects the following parameters:

- *TaskType*. Type of migration task to create for SQL Server to Azure SQL Database Managed Instance migration type *MigrateSqlServerSqlDbMi* is expected.
- *Resource Group Name*. Name of Azure resource group in which to create the task.
- *ServiceName*. Azure Database Migration Service instance in which to create the task.
- *ProjectName*. Name of Azure Database Migration Service project in which to create the task.
- *TaskName*. Name of task to be created.
- *SourceConnection*. `AzDmsConnInfo` object representing source SQL Server connection.
- *TargetConnection*. `AzDmsConnInfo` object representing target Azure SQL Database Managed Instance connection.
- *SourceCred*. `PSCredential` object for connecting to source server.
- *TargetCred*. `PSCredential` object for connecting to target server.
- *SelectedDatabase*. `AzDataMigrationSelectedDB` object representing the source and target database mapping.
- *BackupFileShare*. `FileShare` object representing the local network share that the Azure Database Migration Service can take the source database backups to.
- *BackupBlobSasUri*. The SAS URI that provides the Azure Database Migration Service with access to the storage account container to which the service uploads the backup files. Learn how to get the SAS URI for blob container.
- *SelectedLogins*. List of selected logins to migrate.
- *SelectedAgentJobs*. List of selected agent jobs to migrate.

Additional parameters

The `New-AzDataMigrationTask` cmdlet also expects parameters that are unique to the type of migration, offline or online, that you are performing.

- **Offline migrations.** For offline migrations, the `New-AzDataMigrationTask` cmdlet also expects the following parameters:

- *SelectedLogins*. List of selected logins to migrate.
- *SelectedAgentJobs*. List of selected agent jobs to migrate.
- **Online migrations.** For online migrations, the `New-AzDataMigrationTask` cmdlet also expects the following parameters.
 - *AzureActiveDirectoryApp*. Active Directory Application.
 - *StorageResourceId*. Resource ID of Storage Account.

Create and start an offline migration task

The following example creates and starts an offline migration task named **myDMSTask**:

```
$migTask = New-AzDataMigrationTask -TaskType MigrateSqlServerSqlDbMi ` 
-ResourceGroupName myResourceGroup ` 
-ServiceName $service.Name ` 
-ProjectName $project.Name ` 
-TaskName myDMSTask ` 
-SourceConnection $sourceConnInfo ` 
-SourceCred $sourceCred ` 
-TargetConnection $targetConnInfo ` 
-TargetCred $targetCred ` 
-SelectedDatabase $selectedDbs ` 
-BackupFileShare $backupFileShare ` 
-BackupBlobSasUri $blobSasUri ` 
-SelectedLogins $selectedLogins ` 
-SelectedAgentJobs $selectedJobs `
```

Create and start an online migration task

The following example creates and starts an online migration task named **myDMSTask**:

```
$migTask = New-AzDataMigrationTask -TaskType MigrateSqlServerSqlDbMiSync ` 
-ResourceGroupName myResourceGroup ` 
-ServiceName $service.Name ` 
-ProjectName $project.Name ` 
-TaskName myDMSTask ` 
-SourceConnection $sourceConnInfo ` 
-SourceCred $sourceCred ` 
-TargetConnection $targetConnInfo ` 
-TargetCred $targetCred ` 
-SelectedDatabase $selectedDbs ` 
-BackupFileShare $backupFileShare ` 
-SelectedDatabase $selectedDbs ` 
-AzureActiveDirectoryApp $app ` 
-StorageResourceId $storageResourceId
```

Monitor the migration

To monitor the migration, perform the following tasks.

1. Consolidate all the migration details into a variable called `$CheckTask`.

To combine migration details such as properties, state, and database information associated with the migration, use the following code snippet:

```
$CheckTask= Get-AzDataMigrationTask -ResourceGroupName myResourceGroup ` 
    -ServiceName $service.Name ` 
    -ProjectName $project.Name ` 
    -Name myDMSTask ` 
    -ResultType DatabaseLevelOutput ` 
    -Expand 
Write-Host '$CheckTask.ProjectTask.Properties.Output'
```

2. Use the `$CheckTask` variable to get the current state of the migration task.

To use the `$checkTask` variable to get the current state of the migration task, you can monitor the migration task running by querying the state property of the task, as shown in the following example:

```
if (($CheckTask.ProjectTask.Properties.State -eq "Running") -or ($CheckTask.ProjectTask.Properties.State -eq "Queued"))
{
    Write-Host "migration task running"
}
else if($CheckTask.ProjectTask.Properties.State -eq "Succeeded")
{
    Write-Host "Migration task is completed Successfully"
}
else if($CheckTask.ProjectTask.Properties.State -eq "Failed" -or $CheckTask.ProjectTask.Properties.State -eq "FailedInputValidation" -or $CheckTask.ProjectTask.Properties.State -eq "Faulted")
{
    Write-Host "Migration Task Failed"
}
```

Performing the cutover (online migrations only)

With an online migration, a full backup and restore of databases is performed, and then work proceeds on restoring the Transaction Logs stored in the BackupFileShare.

When the database in an Azure SQL Database managed instance is updated with latest data and is in sync with the source database, you can perform a cutover.

The following example will complete the cutover\migration. Users invoke this command at their discretion.

```
$command = Invoke-AzDmsCommand - CommandType CompleteSqlMiSync ` 
    -ResourceGroupName myResourceGroup ` 
    -ServiceName $service.Name ` 
    -ProjectName $project.Name ` 
    -TaskName myDMSTask ` 
    -DatabaseName "Source DB Name"
```

Deleting the instance of Azure Database Migration Service

After the migration is complete, you can delete the Azure Database Migration Service instance:

```
Remove-AzDms -ResourceGroupName myResourceGroup -ServiceName MyDMS
```

Additional resources

For information about additional migrating scenarios (source/target pairs), see the Microsoft [Database Migration Guide](#).

Next steps

Find out more about Azure Database Migration Service in the article [What is the Azure Database Migration Service?](#).

Monitor migration activity using the Azure Database Migration Service

2/26/2020 • 3 minutes to read • [Edit Online](#)

In this article, you learn how to monitor the progress of a migration at both a database level and a table level.

Monitor at the database level

To monitor activity at the database level, view the database-level blade:

The screenshot shows the 'runnow1' database-level blade. It includes sections for Source server (13.66.220.198, MySQL), Target server (builddemomytarget2.mysql.database.azure.com, Azure Database for MySQL), Activity status (Succeeded), and a table of migration details. The table has columns: DATABASE NAME, STATUS, MIGRATION DETAILS, DURATION, ESTIMATED APPLICATION DOWNTIME, and FINISH DATE. One row is shown for 'inventory' with STATUS 'Complete', MIGRATION DETAILS 'All changes applied', and DURATION '---'. The entire table is highlighted with a red border.

DATABASE NAME	STATUS	MIGRATION DETAILS	DURATION	ESTIMATED APPLICATION DOWNTIME	FINISH DATE
inventory	Complete	All changes applied	---	---	---

NOTE

Selecting the database hyperlink will show you the list of tables and their migration progress.

The following table lists the fields on the database-level blade and describes the various status values associated with each.

FIELD NAME	FIELD SUBSTATUS	DESCRIPTION
Activity status	Running	Migration activity is running.
	Succeeded	Migration activity succeeded without issues.
	Faulted	Migration failed. Select the 'See error details' link under migration details for the complete error message.
Status	Initializing	DMS is setting up the migration pipeline.
	Running	DMS pipeline is running and performing migration.
	Complete	Migration completed.

FIELD NAME	FIELD SUBSTATUS	DESCRIPTION
	Failed	Migration failed. Click on migration details to see migration errors.
Migration details	Initiating the migration pipeline	DMS is setting up the migration pipeline.
	Full data load in progress	DMS is performing initial load.
	Ready for Cutover	After initial load is completed, DMS will mark database as ready for cutover. User should check if data has caught up on continuous sync.
	All changes applied	Initial load and continuous sync are complete. This status also occurs after the database is cutover successfully.
	See error details	Click on the link to show error details.
Duration	N/A	Total time from migration activity being initialized to migration completed or migration faulted.

Monitor at table level – Quick Summary

To monitor activity at the table level, view the table-level blade. The top portion of the blade shows the detailed number of rows migrated in full load and incremental updates.

The bottom portion of the blade lists the tables and shows a quick summary of migration progress.

The screenshot shows the 'inventory' table-level blade. At the top, there are two buttons: 'Refresh' (highlighted with a blue dashed box) and 'Start Cutover'. Below these are four sections: 'Source database name' (inventory), 'Target database name' (inventory), 'Database status' (Running), and 'Migration details' (Ready to cutover). Under 'Migration details', it says 'Full load failed'. At the bottom, there are tabs for 'Full load' (selected) and 'Incremental data sync'. A message indicates '2 item(s)' and shows 'Page 1 of 1'. A red box highlights the table below:

TABLE NAME	STATUS	COMPLETED	ROWS	DURATION
inventory.catalog	Completed	8/2/2018 3:44:08 PM	9	00:00:02
inventory.orders	Completed	8/2/2018 3:44:11 PM	306	00:00:02

The following table describes the fields shown in the table-level details.

FIELD NAME	DESCRIPTION
Full load completed	Number of tables completed full data load.
Full load queued	Number of tables being queued for full load.
Full load loading	Number of tables failed.
Incremental updates	Number of change data capture (CDC) updates in rows applied to target.
Incremental inserts	Number of CDC inserts in rows applied to target.
Incremental deletes	Number of CDC deletes in rows applied to target.
Pending changes	Number of CDC in rows that are still waiting to get applied to target.
Applied changes	Total of CDC updates, inserts, and deletes in rows applied to target.
Tables in error state	Number of tables that are in 'error' state during migration. Some examples that tables can go into error state are when there are duplicates identified in the target or data isn't compatible loading in the target table.

Monitor at table level – Detailed Summary

There are two tabs that show migration progress in Full load and Incremental data sync.

Full load	Incremental data sync															
2 item(s)																
← prev Page 1 of 1 next →																
<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>TABLE NAME</th> <th>STATUS</th> <th>COMPLETED</th> <th>ROWS</th> <th>DURATION</th> </tr> </thead> <tbody> <tr> <td>inventory.catalog</td> <td>Completed</td> <td>8/2/2018 3:44:08 PM</td> <td>9</td> <td>00:00:02</td> </tr> <tr> <td>inventory.orders</td> <td>Completed</td> <td>8/2/2018 3:44:11 PM</td> <td>306</td> <td>00:00:02</td> </tr> </tbody> </table>		TABLE NAME	STATUS	COMPLETED	ROWS	DURATION	inventory.catalog	Completed	8/2/2018 3:44:08 PM	9	00:00:02	inventory.orders	Completed	8/2/2018 3:44:11 PM	306	00:00:02
TABLE NAME	STATUS	COMPLETED	ROWS	DURATION												
inventory.catalog	Completed	8/2/2018 3:44:08 PM	9	00:00:02												
inventory.orders	Completed	8/2/2018 3:44:11 PM	306	00:00:02												

Full load	Incremental data sync																								
2 item(s)																									
← prev Page 1 of 1 next →																									
<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>TABLE NAME</th> <th>STATUS</th> <th>INSERT</th> <th>UPDATE</th> <th>DELETE</th> <th>TOTAL APPLIED</th> <th>DATA ERRORS</th> <th>LAST MODIFIED</th> </tr> </thead> <tbody> <tr> <td>inventory.catalog</td> <td></td> <td></td> <td></td> <td></td> <td>0</td> <td>---</td> <td></td> </tr> <tr> <td>inventory.orders</td> <td></td> <td>64</td> <td></td> <td>254</td> <td>318</td> <td>0</td> <td>8/6/2018 7:28:55 PM</td> </tr> </tbody> </table>		TABLE NAME	STATUS	INSERT	UPDATE	DELETE	TOTAL APPLIED	DATA ERRORS	LAST MODIFIED	inventory.catalog					0	---		inventory.orders		64		254	318	0	8/6/2018 7:28:55 PM
TABLE NAME	STATUS	INSERT	UPDATE	DELETE	TOTAL APPLIED	DATA ERRORS	LAST MODIFIED																		
inventory.catalog					0	---																			
inventory.orders		64		254	318	0	8/6/2018 7:28:55 PM																		

The following table describes the fields shown in table level migration progress.

FIELD NAME	DESCRIPTION
Status - Syncing	Continuous sync is running.
Insert	Number of CDC inserts in rows applied to target.
Update	Number of CDC updates in rows applied to target.

FIELD NAME	DESCRIPTION
Delete	Number of CDC deletes in rows applied to target.
Total Applied	Total of CDC updates, inserts, and deletes in rows applied to target.
Data Errors	Number of data errors happened in this table. Some examples of the errors are 511: <i>Cannot create a row of size %d which is greater than the allowable maximum row size of %d</i> , 8114: <i>Error converting data type %ls to %ls</i> . Customer should query from dms_apply_exceptions table in Azure target to see the error details.

NOTE

CDC values of Insert, Update and Delete and Total Applied may decrease when database is cutover or migration is restarted.

Next steps

- Review the migration guidance in the Microsoft [Database Migration Guide](#).

Redeploy SSIS packages to Azure SQL Database with Azure Database Migration Service

2/26/2020 • 3 minutes to read • [Edit Online](#)

If you use SQL Server Integration Services (SSIS) and want to migrate your SSIS projects/packages from the source SSISDB hosted by SQL Server to the destination SSISDB hosted by Azure SQL Database, you can redeploy them using the Integration Services Deployment Wizard. You can launch the wizard from within SQL Server Management Studio (SSMS).

If the version of SSIS you use is earlier than 2012, before redeploying your SSIS projects/packages into the project deployment model, you first need to convert them by using the Integration Services Project Conversion Wizard, which can also be launched from SSMS. For more information, see the article [Converting projects to the project deployment model](#).

NOTE

The Azure Database Migration Service (DMS) currently does not support the migration of a source SSISDB to an Azure SQL Database server, but you can redeploy your SSIS projects/packages using the following process.

In this article, you learn how to:

- Assess source SSIS projects/packages.
- Migrate SSIS projects/packages to Azure.

Prerequisites

To complete these steps, you need:

- SSMS version 17.2 or later.
- An instance of your target database server to host SSISDB. If you don't already have one, create an Azure SQL Database server (without a database) using the Azure portal by navigating to the SQL Server (logical server only) [form](#).
- SSIS must be provisioned in Azure Data Factory (ADF) containing Azure-SSIS Integration Runtime (IR) with the destination SSISDB hosted by the instance of Azure SQL Database server (as described in the article [Provision the Azure-SSIS Integration Runtime in Azure Data Factory](#)).

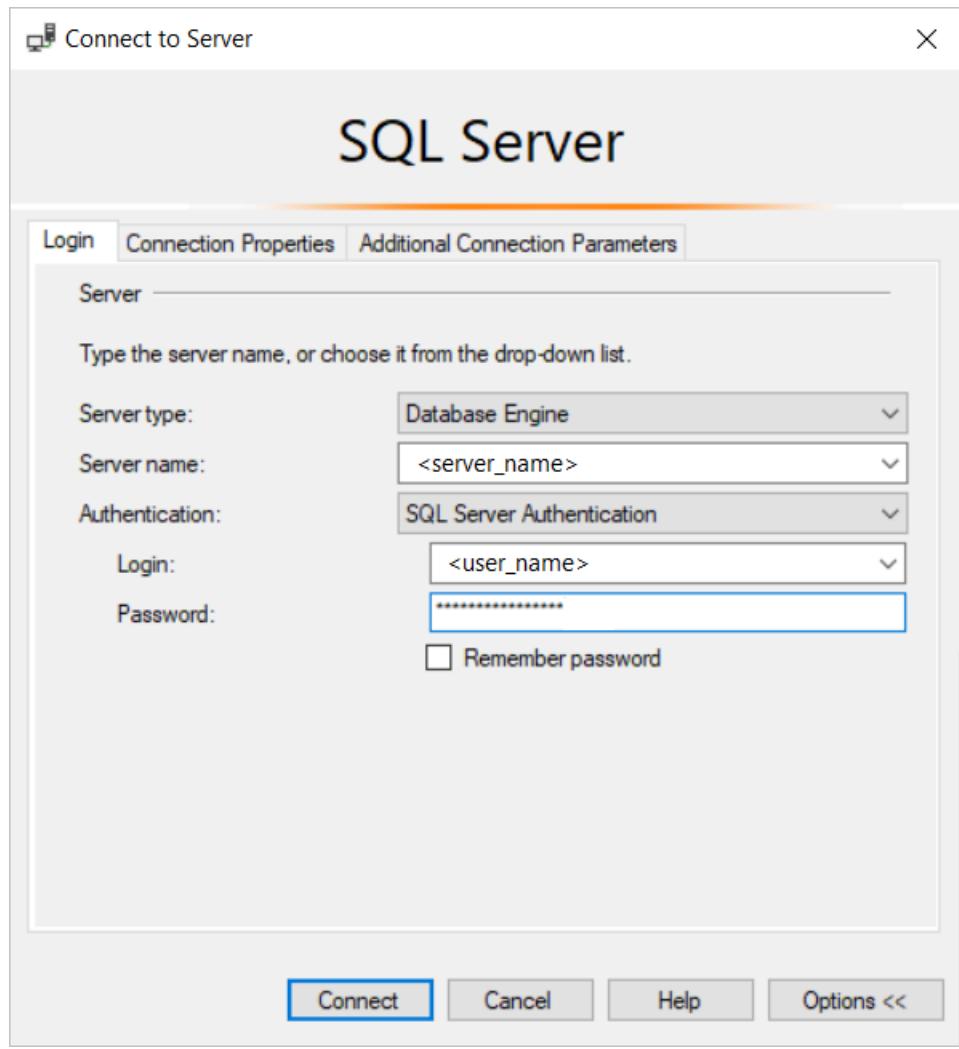
Assess source SSIS projects/packages

While assessment of source SSISDB is not yet integrated into the Database Migration Assistant (DMA) or the Azure Database Migration Service (DMS), your SSIS projects/packages will be assessed/validated as they are redeployed to the destination SSISDB hosted on an Azure SQL Database server.

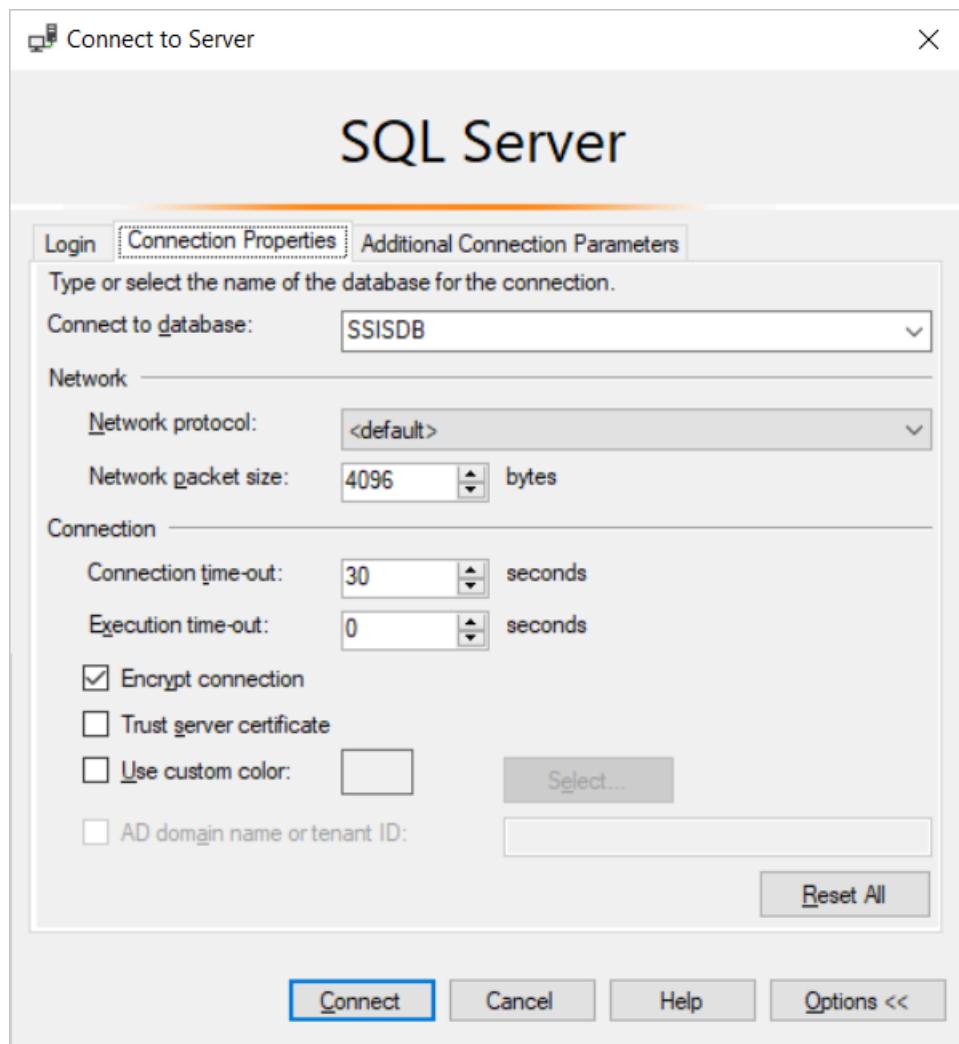
Migrate SSIS projects/packages

To migrate SSIS projects/packages to Azure SQL Database server, perform the following steps.

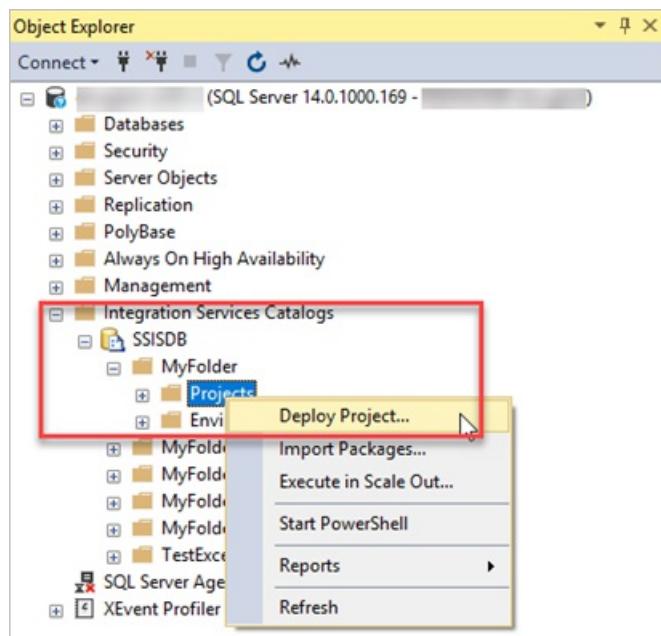
1. Open SSMS, and then select **Options** to display the **Connect to Server** dialog box.
2. On the **Login** tab, specify the information necessary to connect to the Azure SQL Database server that will host the destination SSISDB.



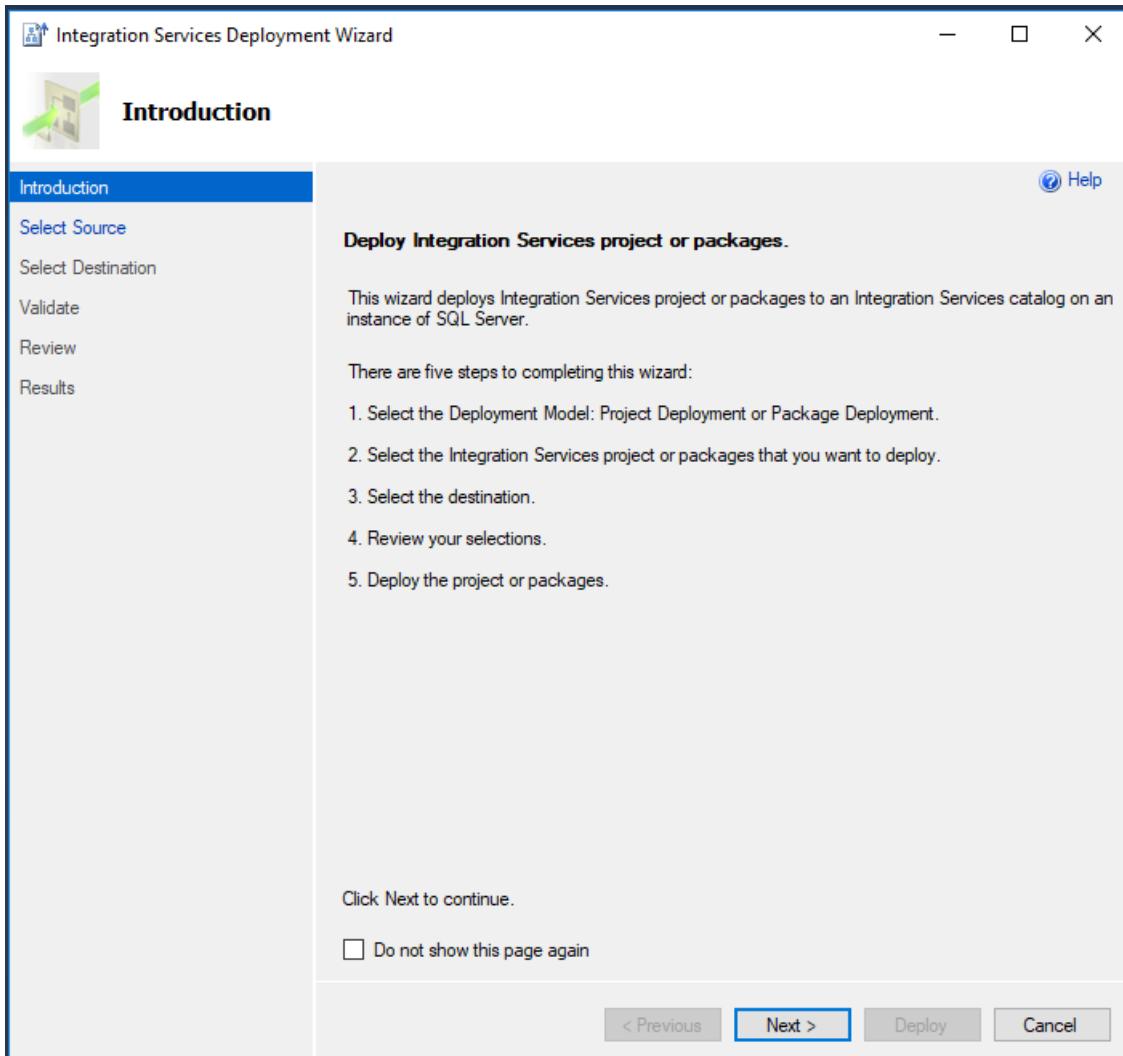
3. On the **Connection Properties** tab, in the **Connect to database** text box, select or enter **SSISDB**, and then select **Connect**.



4. In the SSMS Object Explorer, expand the **Integration Services Catalogs** node, expand **SSISDB**, and if there are no existing folders, then right-click **SSISDB** and create a new folder.
5. Under **SSISDB**, expand any folder, right-click **Projects**, and then select **Deploy Project**.



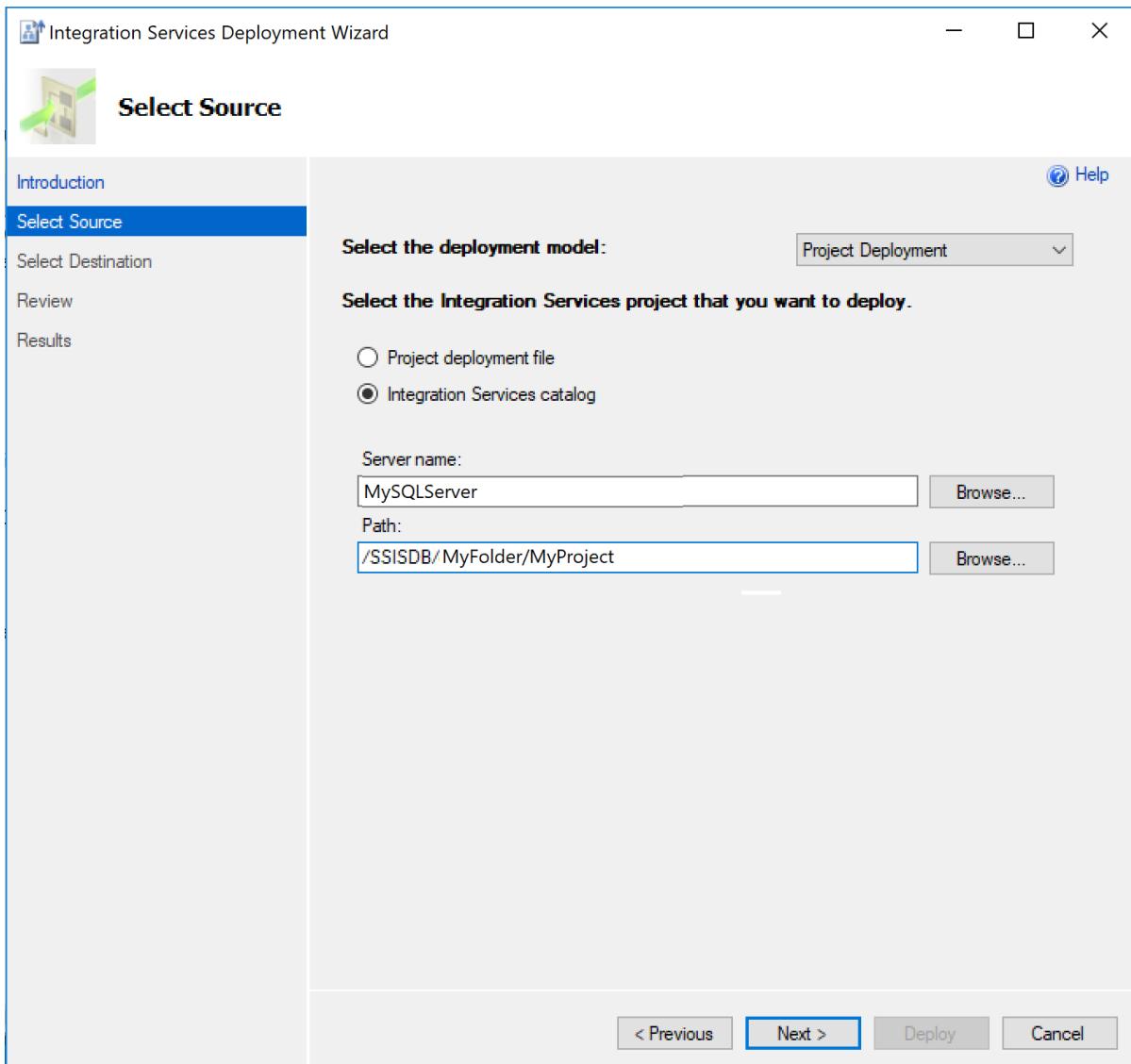
6. In the Integration Services Deployment Wizard, on the **Introduction** page, review the information, and then select **Next**.



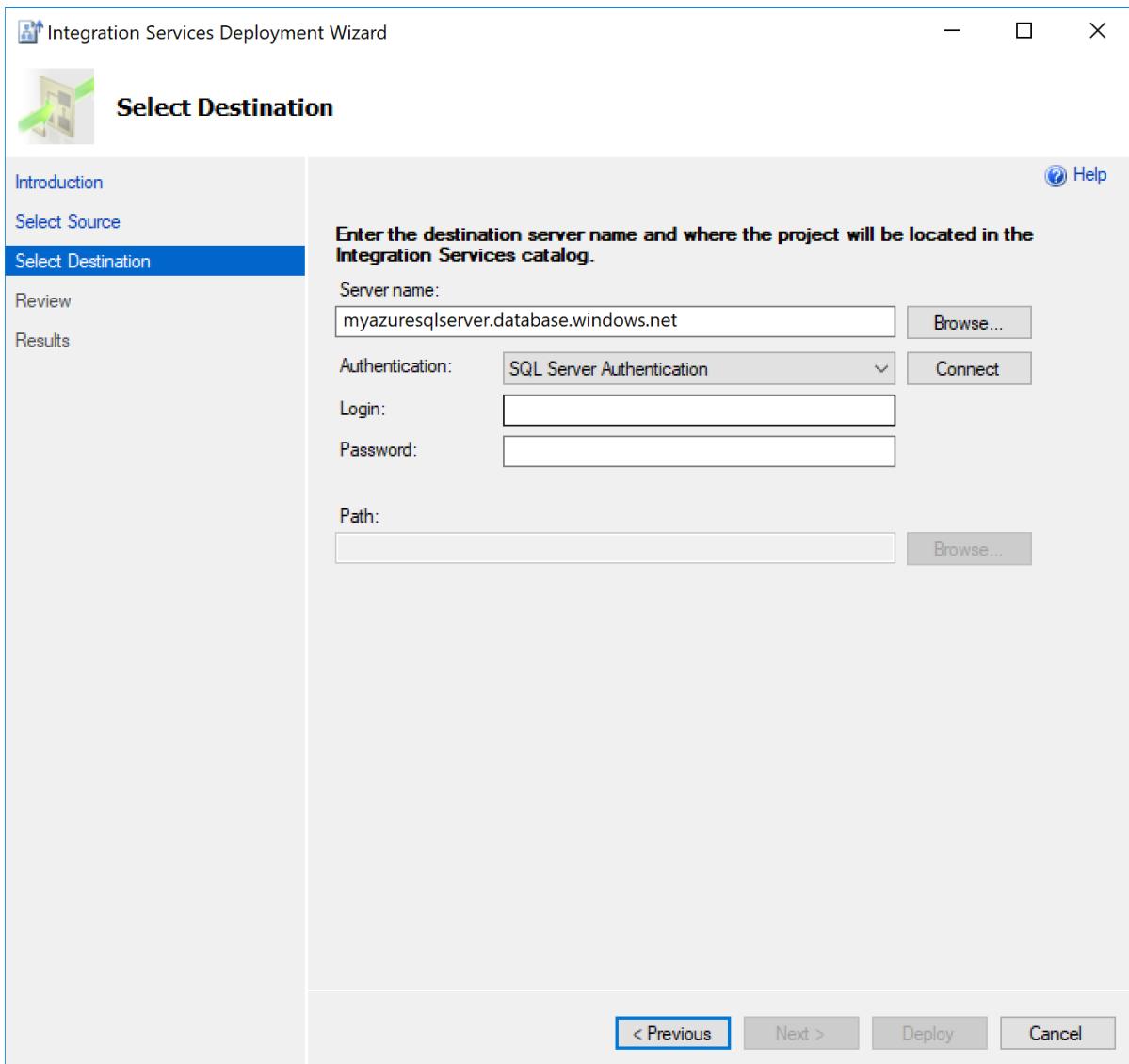
7. On the **Select Source** page, specify the existing SSIS project that you want to deploy.

If SSMS is also connected to the SQL Server hosting the source SSISDB, select **Integration Services catalog**, and then enter the server name and project path in your catalog to deploy your project directly.

Alternately, select **Project deployment file**, and then specify the path to an existing project deployment file (.ispac) to deploy your project.



8. Select **Next**.
9. On the **Select Destination** page, specify the destination for your project.
 - a. In the Server name text box, enter the fully qualified Azure SQL Database server name (<server_name>.database.windows.net).
 - b. Provide the authentication information, and then select **Connect**.

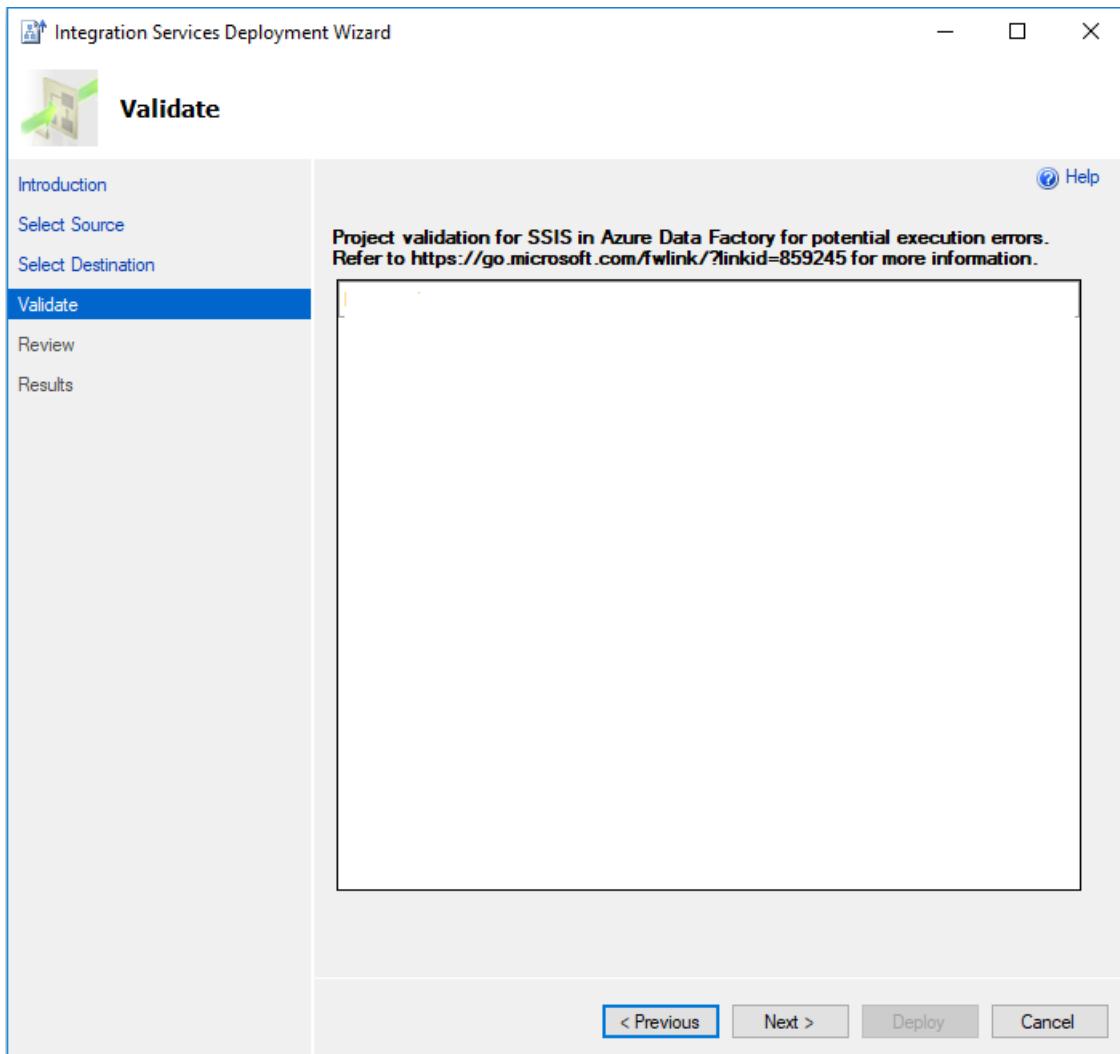


c. Select **Browse** to specify the destination folder in SSISDB, and then select **Next**.

NOTE

The **Next** button is enabled only after you've selected **Connect**.

10. On the **Validate** page, view any errors/warnings, and then if necessary, modify your packages accordingly.



11. Select **Next**.

12. On the **Review** page, review your deployment settings.

NOTE

You can change your settings by selecting **Previous** or by selecting any of the step links in the left pane.

13. Select **Deploy** to start the deployment process.

14. After the deployment process is completed, you can view the Results page, which displays the success or failure of each deployment action. a. If any action failed, in the **Result** column, select **Failed** to display an explanation of the error. b. Optionally, select **Save Report** to save the results to an XML file.

15. Select **Close** to exit the Integration Services Deployment Wizard.

If the deployment of your project succeeds without failure, you can select any packages it contains to run on your Azure-SSIS IR.

Next steps

- Review the migration guidance in the Microsoft [Database Migration Guide](#).

Migrate SQL Server Integration Services packages to an Azure SQL Database managed instance

2/26/2020 • 5 minutes to read • [Edit Online](#)

If you use SQL Server Integration Services (SSIS) and want to migrate your SSIS projects/packages from the source SSISDB hosted by SQL Server to the destination SSISDB hosted by an Azure SQL Database managed instance, you can use Azure Database Migration Service.

If the version of SSIS you use is earlier than 2012 or you use non-SSISDB package store types, before migrating your SSIS projects/packages, you need to convert them by using the Integration Services Project Conversion Wizard, which can also be launched from SSMS. For more information, see the article [Converting projects to the project deployment model](#).

NOTE

Azure Database Migration Service (DMS) currently does not support Azure SQL Database as a target migration destination. To redeploy SSIS projects/packages to Azure SQL Database, see the article [Redeploy SQL Server Integration Services packages to Azure SQL Database](#).

In this article, you learn how to:

- Assess source SSIS projects/packages.
- Migrate SSIS projects/packages to Azure.

Prerequisites

To complete these steps, you need:

- To create a Microsoft Azure Virtual Network for the Azure Database Migration Service by using the Azure Resource Manager deployment model, which provides site-to-site connectivity to your on-premises source servers by using either [ExpressRoute](#) or [VPN](#). For more information, see the article [Network topologies for Azure SQL Database managed instance migrations using Azure Database Migration Service](#). For more information about creating a virtual network, see the [Virtual Network Documentation](#), and especially the quickstart articles with step-by-step details.
- To ensure that your virtual network Network Security Group rules don't block the following inbound communication ports to Azure Database Migration Service: 443, 53, 9354, 445, 12000. For more detail on virtual network NSG traffic filtering, see the article [Filter network traffic with network security groups](#).
- To configure your [Windows Firewall for source database engine access](#).
- To open your Windows Firewall to allow the Azure Database Migration Service to access the source SQL Server, which by default is TCP port 1433.
- If you're running multiple named SQL Server instances using dynamic ports, you may wish to enable the SQL Browser Service and allow access to UDP port 1434 through your firewalls so that the Azure Database Migration Service can connect to a named instance on your source server.
- If you're using a firewall appliance in front of your source databases, you may need to add firewall rules to allow the Azure Database Migration Service to access the source database(s) for migration, as well as files via SMB port 445.
- An Azure SQL Database managed instance to host SSISDB. If you need to create one, follow the detail in the article [Create an Azure SQL Database Managed Instance](#).

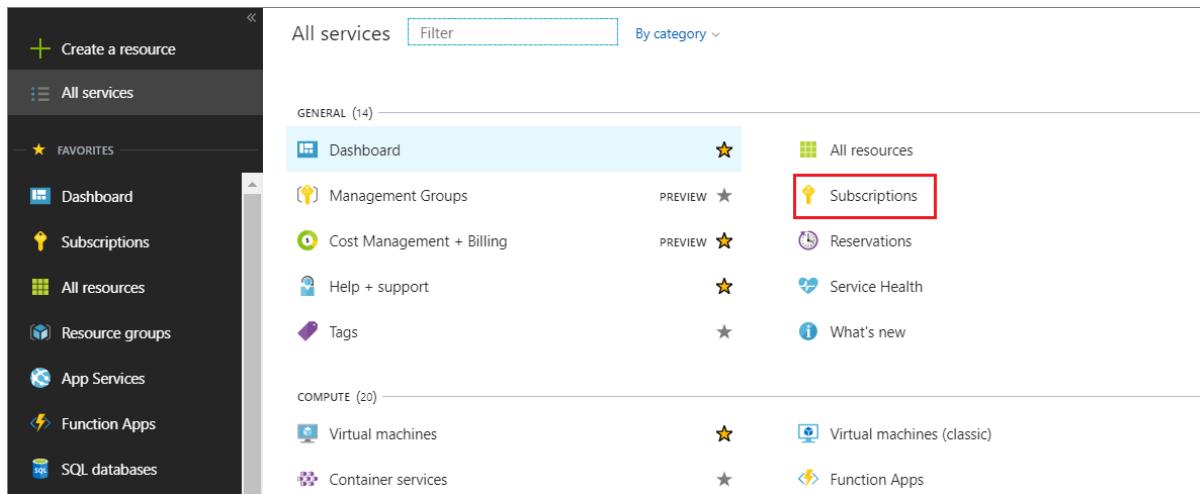
- To ensure that the logins used to connect the source SQL Server and target managed instance are members of the sysadmin server role.
- To verify that SSIS is provisioned in Azure Data Factory (ADF) containing Azure-SSIS Integration Runtime (IR) with the destination SSISDB hosted by an Azure SQL Database managed instance (as described in the article [Create the Azure-SSIS integration runtime in Azure Data Factory](#)).

Assess source SSIS projects/packages

While assessment of source SSISDB isn't yet integrated into the Database Migration Assistant (DMA), your SSIS projects/packages will be assessed/validated as they're redeployed to the destination SSISDB hosted on an Azure SQL Database managed instance.

Register the Microsoft.DataMigration resource provider

1. Sign in to the Azure portal, select **All services**, and then select **Subscriptions**.



2. Select the subscription in which you want to create the instance of Azure Database Migration Service, and then select **Resource providers**.

The screenshot shows the Azure portal's Subscriptions page. On the left, there's a navigation bar with various service icons. The main area displays a table with columns for Subscription ID and Status. A search bar at the top says 'Search to filter items...'. On the right, a sidebar lists several options: Overview, Access control (IAM), Diagnose and solve problems, COST MANAGEMENT + BILLING, Partner information, SETTINGS, Programmatic deployment, Resource groups, Resources, Usage + quotas, Policies, Management certificates, My permissions, and Resource providers. The 'Resource providers' link is highlighted with a red box.

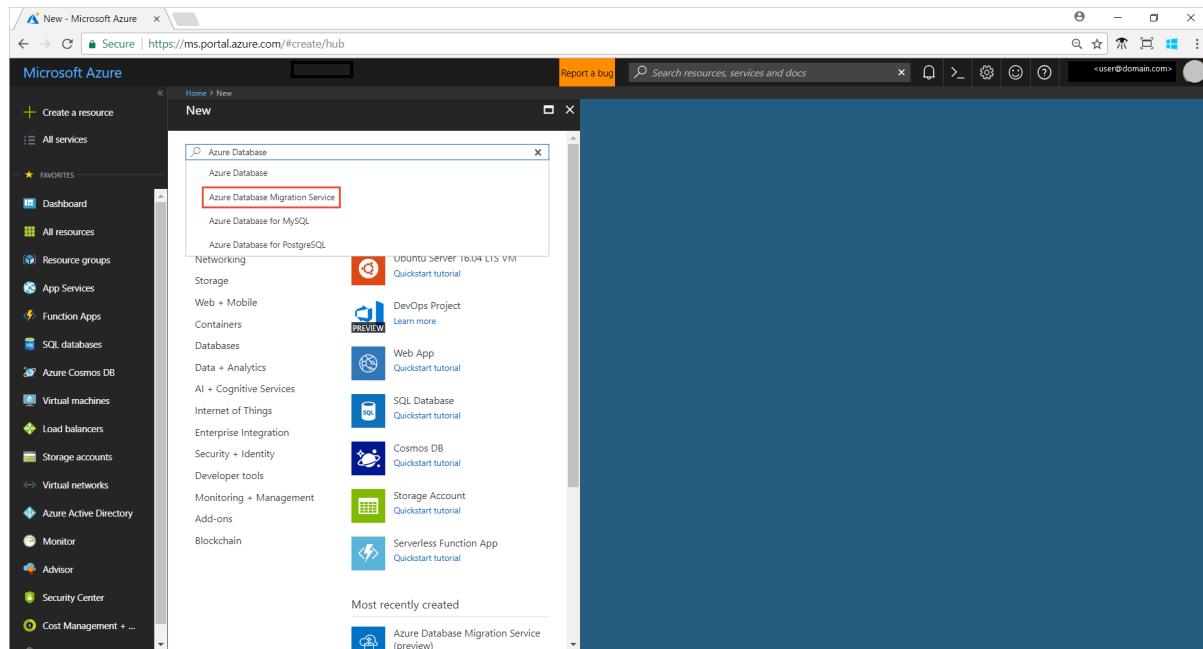
3. Search for migration, and then to the right of **Microsoft.DataMigration**, select **Register**.

This screenshot shows the 'Resource providers' page within the Azure portal. It features a search bar at the top and a table below it. The table has columns for PROVIDER, STATUS, and a Register button. One row in the table is highlighted with a yellow background and shows 'Microsoft.DataMigration' under PROVIDER, 'NotRegistered' under STATUS, and a red box around the 'Register' button.

Create an Azure Database Migration Service instance

1. In the Azure portal, select + **Create a resource**, search for **Azure Database Migration Service**, and then

select **Azure Database Migration Service** from the drop-down list.



2. On the **Azure Database Migration Service** screen, select **Create**.

The Azure Database Migration Service (DMS) is designed to streamline the process of migrating on-premises databases to Azure. DMS will simplify the migration of existing on-premises SQL Server and Oracle databases to Azure SQL Database, Azure SQL Managed Instance or Microsoft SQL Server in an Azure Virtual Machine. [Learn more](#)

Before using DMS we recommend you complete the following three steps:

1. Open the [Azure Database Migration Guide](#) for step by step guidance through the migration process
2. Assess your SQL Server on-premises database(s) for feature parity and potential compatibility issues by using [Data Migration Assistant \(DMA\)](#). Alternatively, if you are migrating from Oracle, use the [SQL Server Migration Assistant \(SSMA\)](#)
3. Based on your database needs, create a target database using one of the database services: Azure SQL Database, Azure SQL Managed Instance or SQL Server in an Azure Virtual Machine.

Once your assessments are complete, fixes are applied and schema is deployed, proceed with creating a migration service by clicking **Create** below to migrate the data from your source database to the target.

[Save for later](#)

PUBLISHER Microsoft

USEFUL LINKS [Documentation](#) [Privacy Statement](#)

Create

3. On the **Create Migration Service** screen, specify a name for the service, the subscription, and a new or

existing resource group.

4. Select the location in which you want to create the instance of DMS.
5. Select an existing virtual network or create one.

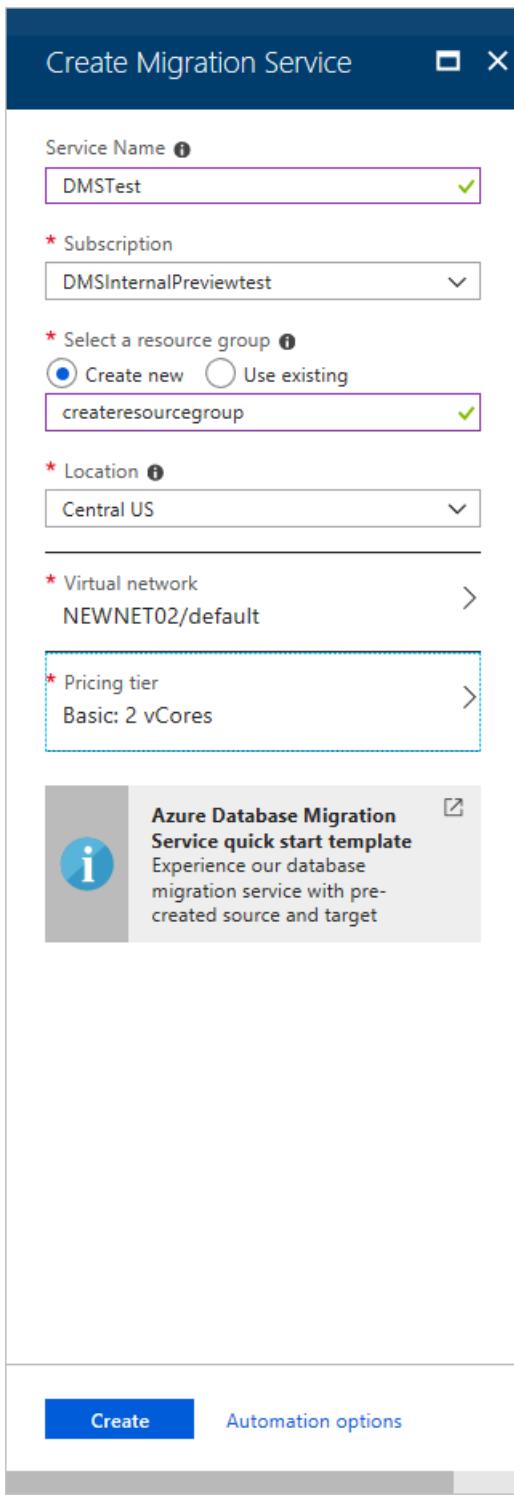
The virtual network provides Azure Database Migration Service with access to the source SQL Server and target Azure SQL Database managed instance.

For more information on how to create a virtual network in Azure portal, see the article [Create a virtual network using the Azure portal](#).

For additional detail, see the article [Network topologies for Azure SQL DB managed instance migrations using the Azure Database Migration Service](#).

6. Select a pricing tier.

For more information on costs and pricing tiers, see the [pricing page](#).

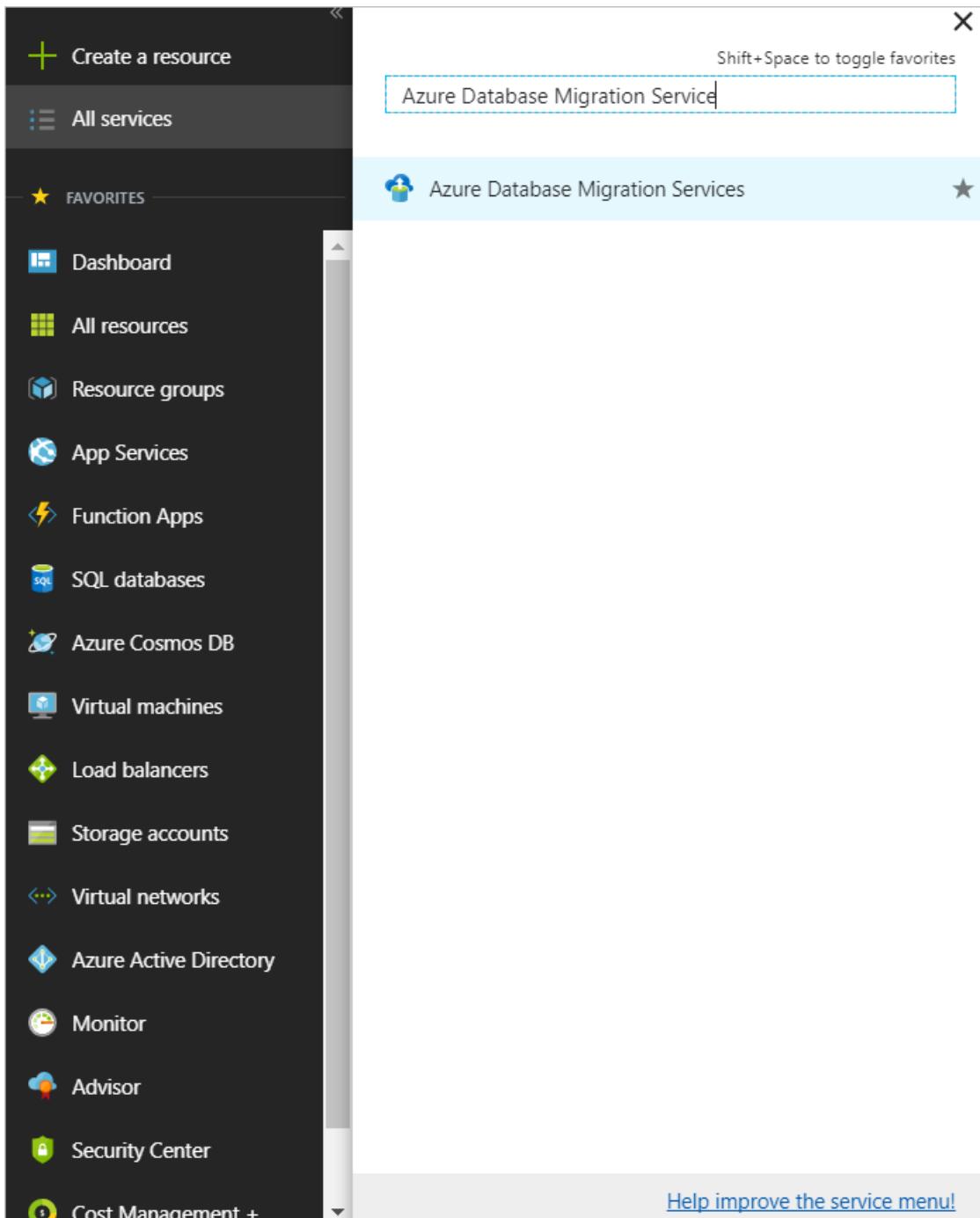


7. Select **Create** to create the service.

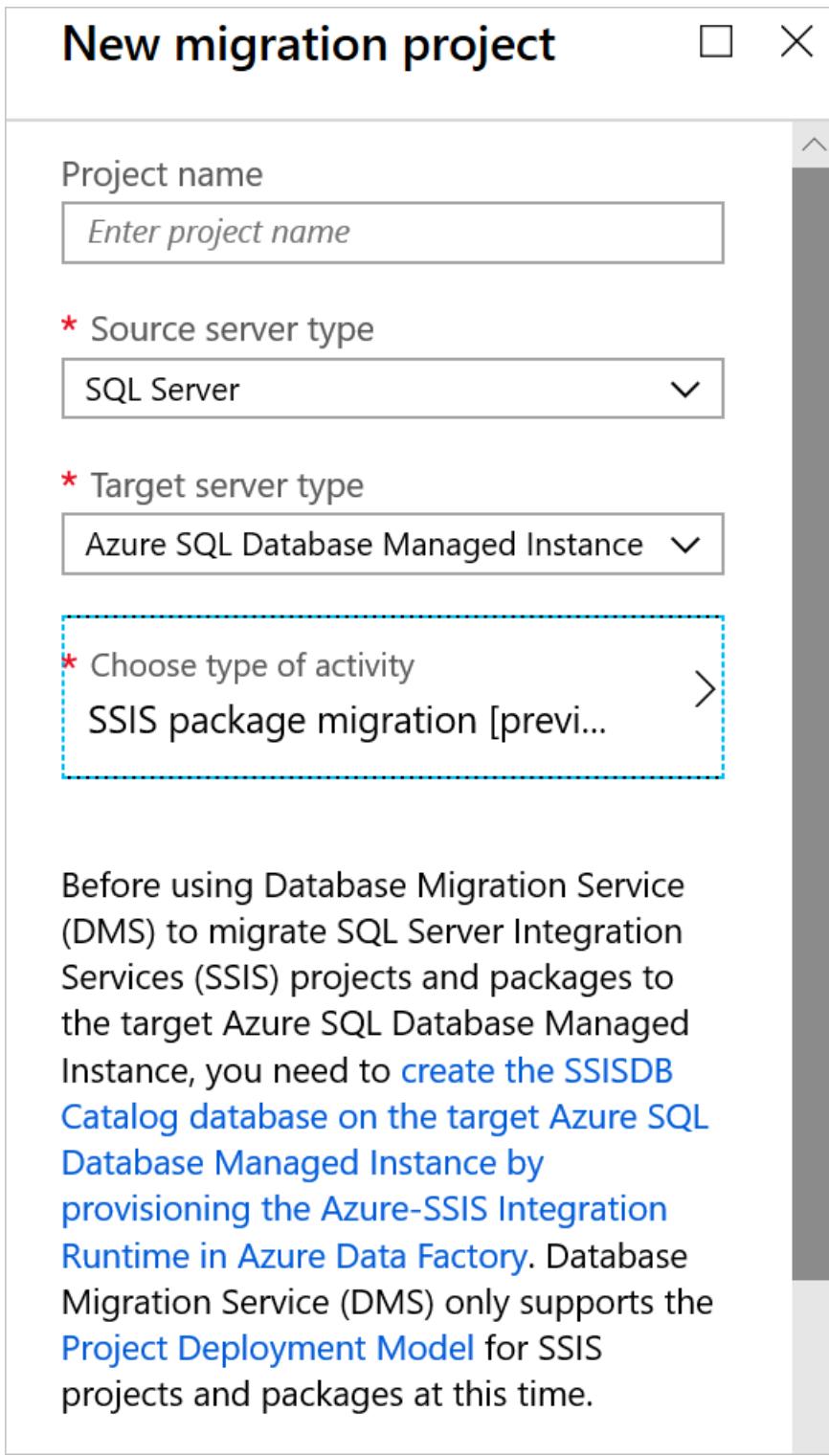
Create a migration project

After an instance of the service is created, locate it within the Azure portal, open it, and then create a new migration project.

1. In the Azure portal, select **All services**, search for Azure Database Migration Service, and then select **Azure Database Migration Services**.



2. On the **Azure Database Migration Service** screen, search for the name of the instance that you created, and then select the instance.
3. Select + **New Migration Project**.
4. On the **New migration project** screen, specify a name for the project, in the **Source server type** text box, select **SQL Server**, in the **Target server type** text box, select **Azure SQL Database Managed Instance**, and then for **Choose type of activity**, select **SSIS package migration**.



5. Select **Create** to create the project.

Specify source details

1. On the **Migration source detail** screen, specify the connection details for the source SQL Server.
2. If you haven't installed a trusted certificate on your server, select the **Trust server certificate** check box.

When a trusted certificate isn't installed, SQL Server generates a self-signed certificate when the instance is started. This certificate is used to encrypt the credentials for client connections.

Caution

SSL connections that are encrypted using a self-signed certificate does not provide strong security. They are susceptible to man-in-the-middle attacks. You should not rely on SSL using self-signed certificates in a production environment or on servers that are connected to the internet.

Migration Wizard

MigrateToMI

1 Select source >

2 Select target >

3 Summary >

Migration source detail

* Source SQL Server instance name [?](#)
servername.domainname.com

Authentication type [?](#)
Windows Authentication

* User Name [?](#)
Enter domain\user name

Password
Enter password

Connection properties

Encrypt connection

Trust server certificate

Save

3. Select **Save**.

Specify target details

1. On the **Migration target details** screen, specify the connection details for the target.

Migration Wizard		X	Migration target details		□ X
MigrateToMI					
1	Select source	✓			
2	Select target	>	* Target server name <small>i</small> <input type="text" value="MyAzureSQLDBMI.database.windows.net"/>		
3	Summary	>	Authentication type <small>i</small> SQL Authentication		
			* User Name <small>i</small> <input type="text" value="Enter user name"/>		
			Password <input type="text" value="Enter password"/>		
					Save

2. Select **Save**.

Review the migration summary

1. On the **Migration summary** screen, in the **Activity name** text box, specify a name for the migration activity.
2. For the **SSIS project(s) and environment(s) overwrite option**, specify whether to overwrite or ignore existing SSIS projects and environments.

Migration Wizard		X	Migration summary		□ X
MigrateToMI					
1	Select source	✓	Activity name <input type="text" value="Create Activity"/>		
2	Select target	✓	Source server name <source server> Source server version SQL Server 2016 13.0.4550.1		
3	Summary	>	Target server name <target server> Target server version Azure SQL Database Managed Instance 12.0.2000.8		
			Type of activity SSIS package migration [preview] * SSIS project(s) and environment(s) overwrite option: <input checked="" type="radio"/> Ignore <input type="radio"/> Overwrite Existing SSIS project(s) and environment(s) will be ignored.		
			Run migration		

3. Review and verify the details associated with the migration project.

Run the migration

- Select **Run migration**.

Next steps

- Review the migration guidance in the Microsoft [Database Migration Guide](#).

Status of migration scenarios supported by Azure Database Migration Service

12/18/2019 • 2 minutes to read • [Edit Online](#)

Azure Database Migration Service is designed to support different migration scenarios (source/target pairs) for both offline (one-time) and online (continuous sync) migrations. The scenario coverage provided by Azure Database Migration Service is being extended over time. New scenarios are being added on a regular basis. This article identifies migration scenarios currently supported by Azure Database Migration Service and the status (private preview, public preview, or general availability) for each scenario.

Offline versus online migrations

With Azure Database Migration Service, you can do an offline or an online migration. With *offline* migrations, application downtime begins at the same time that the migration starts. To limit downtime to the time required to cut over to the new environment when the migration completes, use an *online* migration. It's recommended to test an offline migration to determine whether the downtime is acceptable; if not, do an online migration.

Migration scenario status

The status of migration scenarios supported by Azure Database Migration Service varies with time. Generally, scenarios are first released in **private preview**. Participating in private preview requires customers to submit a nomination via the [DMS Preview site](#). After private preview, the scenario status changes to **public preview**. Azure Database Migration Service users can try out migration scenarios in public preview directly from the user interface. No sign-up is required. However, migration scenarios in public preview may not be available in all regions and may undergo additional changes before final release. After public preview, the scenario status changes to **generally availability**. General availability (GA) is the final release status, and the functionality is complete and accessible to all users.

Migration scenario support

The following tables show which migration scenarios are supported when using Azure Database Migration Service.

NOTE

If a scenario listed as supported below does not appear within the user interface, please contact the [Ask Azure Database Migrations](#) alias for additional information.

IMPORTANT

To view all scenarios currently supported by Azure Database Migration Service in Private Preview, see the [DMS Preview site](#).

Offline (one-time) migration support

The following table shows Azure Database Migration Service support for offline migrations.

TARGET	SOURCE	SUPPORT	STATUS
Azure SQL DB	SQL Server	✓	GA
	RDS SQL		
	Oracle		
Azure SQL DB MI	SQL Server	✓	GA
	RDS SQL		
	Oracle		
Azure SQL VM	SQL Server	✓	GA
	Oracle		
Azure Cosmos DB	MongoDB	✓	GA
Azure DB for MySQL	MySQL		
	RDS MySQL		
Azure DB for PostgreSQL	PostgreSQL		
	RDS PostgreSQL		

Online (continuous sync) migration support

The following table shows Azure Database Migration Service support for online migrations.

TARGET	SOURCE	SUPPORT	STATUS
Azure SQL DB	SQL Server	✓	GA
	RDS SQL	✓	GA
	Oracle		
Azure SQL DB MI	SQL Server	✓	GA
	RDS SQL	✓	GA
	Oracle		Private preview
Azure SQL VM	SQL Server		
	Oracle		
Azure Cosmos DB	MongoDB	✓	GA
Azure DB for MySQL	MySQL	✓	GA

TARGET	SOURCE	SUPPORT	STATUS
	RDS MySQL	✓	GA
Azure DB for PostgreSQL	PostgreSQL	✓	GA
	RDS PostgreSQL	✓	GA
	Oracle	✓	Public preview

Next steps

For an overview of Azure Database Migration Service and regional availability, see the article [What is the Azure Database Migration Service](#).

Overview of prerequisites for using the Azure Database Migration Service

2/26/2020 • 5 minutes to read • [Edit Online](#)

There are several prerequisites required to ensure Azure Database Migration Service runs smoothly when performing database migrations. Some of the prerequisites apply across all scenarios (source-target pairs) supported by the service, while other prerequisites are unique to a specific scenario.

Prerequisites associated with using the Azure Database Migration Service are listed in the following sections.

Prerequisites common across migration scenarios

Azure Database Migration Service prerequisites that are common across all supported migration scenarios include the need to:

- Create a Microsoft Azure Virtual Network for Azure Database Migration Service by using the Azure Resource Manager deployment model, which provides site-to-site connectivity to your on-premises source servers by using either [ExpressRoute](#) or [VPN](#).
- Ensure that your virtual network Network Security Group (NSG) rules don't block the following communication ports 443, 53, 9354, 445, 12000. For more detail on virtual network NSG traffic filtering, see the article [Filter network traffic with network security groups](#).
- When using a firewall appliance in front of your source database(s), you may need to add firewall rules to allow Azure Database Migration Service to access the source database(s) for migration.
- Configure your [Windows Firewall for database engine access](#).
- Enable the TCP/IP protocol, which is disabled by default during SQL Server Express installation, by following the instructions in the article [Enable or Disable a Server Network Protocol](#).

IMPORTANT

Creating an instance of Azure Database Migration Service requires access to virtual network settings that are normally not within the same resource group. As a result, the user creating an instance of DMS requires permission at subscription level. To create the required roles, which you can assign as needed, run the following script:

```

$readerActions = `
"Microsoft.Network/networkInterfaces/ipConfigurations/read", `
"Microsoft.DataMigration/*/read", `
"Microsoft.Resources/subscriptions/resourceGroups/read"

$writerActions = `
"Microsoft.DataMigration/services/*/write", `
"Microsoft.DataMigration/services/*/delete", `
"Microsoft.DataMigration/services/*/action", `
"Microsoft.Network/virtualNetworks/subnets/join/action", `
"Microsoft.Network/virtualNetworks/write", `
"Microsoft.Network/virtualNetworks/read", `
"Microsoft.Resources/deployments/validate/action", `
"Microsoft.Resources/deployments/*/read", `
"Microsoft.Resources/deployments/*/write"

$writerActions += $readerActions

# TODO: replace with actual subscription IDs
$subScopes = ,"/subscriptions/00000000-0000-0000-0000-000000000000","/subscriptions/11111111-1111-
1111-1111-111111111111"

function New-DmsReaderRole() {
    $aRole = [Microsoft.Azure.Commands.Resources.Models.Authorization.PSRoleDefinition]::new()
    $aRole.Name = "Azure Database Migration Reader"
    $aRole.Description = "Lets you perform read only actions on DMS service/project/tasks."
    $aRole.IsCustom = $true
    $aRole.Actions = $readerActions
    $aRole.NotActions = @()

    $aRole.AssignableScopes = $subScopes
    #Create the role
    New-AzRoleDefinition -Role $aRole
}

function New-DmsContributorRole() {
    $aRole = [Microsoft.Azure.Commands.Resources.Models.Authorization.PSRoleDefinition]::new()
    $aRole.Name = "Azure Database Migration Contributor"
    $aRole.Description = "Lets you perform CRUD actions on DMS service/project/tasks."
    $aRole.IsCustom = $true
    $aRole.Actions = $writerActions
    $aRole.NotActions = @()

    $aRole.AssignableScopes = $subScopes
    #Create the role
    New-AzRoleDefinition -Role $aRole
}

function Update-DmsReaderRole() {
    $aRole = Get-AzRoleDefinition "Azure Database Migration Reader"
    $aRole.Actions = $readerActions
    $aRole.NotActions = @()
    Set-AzRoleDefinition -Role $aRole
}

function Update-DmsConributorRole() {
    $aRole = Get-AzRoleDefinition "Azure Database Migration Contributor"
    $aRole.Actions = $writerActions
    $aRole.NotActions = @()
    Set-AzRoleDefinition -Role $aRole
}

# Invoke above functions
New-DmsReaderRole
New-DmsContributorRole
Update-DmsReaderRole
Update-DmsConributorRole

```

Prerequisites for migrating SQL Server to Azure SQL Database

In addition to Azure Database Migration Service prerequisites that are common to all migration scenarios, there are also prerequisites that apply specifically to one scenario or another.

When using the Azure Database Migration Service to perform SQL Server to Azure SQL Database migrations, in addition to the prerequisites that are common to all migration scenarios, be sure to address the following additional prerequisites:

- Create an instance of Azure SQL Database instance, which you do by following the detail in the article [Create an Azure SQL database in the Azure portal](#).
- Download and install the [Data Migration Assistant](#) v3.3 or later.
- Open your Windows Firewall to allow the Azure Database Migration Service to access the source SQL Server, which by default is TCP port 1433.
- If you are running multiple named SQL Server instances using dynamic ports, you may wish to enable the SQL Browser Service and allow access to UDP port 1434 through your firewalls so that the Azure Database Migration Service can connect to a named instance on your source server.
- Create a server-level [firewall rule](#) for the Azure SQL Database server to allow the Azure Database Migration Service access to the target databases. Provide the subnet range of the virtual network used for the Azure Database Migration Service.
- Ensure that the credentials used to connect to source SQL Server instance have [CONTROL SERVER](#) permissions.
- Ensure that the credentials used to connect to target Azure SQL Database instance have [CONTROL DATABASE](#) permission on the target Azure SQL databases.

NOTE

For a complete listing of the prerequisites required to use the Azure Database Migration Service to perform migrations from SQL Server to Azure SQL Database, see the tutorial [Migrate SQL Server to Azure SQL Database](#).

Prerequisites for migrating SQL Server to an Azure SQL Database managed instance

- Create an Azure SQL Database managed instance by following the detail in the article [Create an Azure SQL Database Managed Instance in the Azure portal](#).
- Open your firewalls to allow SMB traffic on port 445 for the Azure Database Migration Service IP address or subnet range.
- Open your Windows Firewall to allow the Azure Database Migration Service to access the source SQL Server, which by default is TCP port 1433.
- If you are running multiple named SQL Server instances using dynamic ports, you may wish to enable the SQL Browser Service and allow access to UDP port 1434 through your firewalls so that the Azure Database Migration Service can connect to a named instance on your source server.
- Ensure that the logins used to connect the source SQL Server and target Managed Instance are members of the sysadmin server role.
- Create a network share that the Azure Database Migration Service can use to back up the source database.
- Ensure that the service account running the source SQL Server instance has write privileges on the network share that you created and that the computer account for the source server has read/write access to the same share.

- Make a note of a Windows user (and password) that has full control privilege on the network share that you previously created. The Azure Database Migration Service impersonates the user credential to upload the backup files to Azure Storage container for restore operation.
- Create a blob container and retrieve its SAS URI by using the steps in the article [Manage Azure Blob Storage resources with Storage Explorer](#). Be sure to select all permissions (Read, Write, Delete, List) on the policy window while creating the SAS URI.

NOTE

For a complete listing of the prerequisites required to use the Azure Database Migration Service to perform migrations from SQL Server to Azure SQL Database Managed Instance, see the tutorial [Migrate SQL Server to Azure SQL Database Managed Instance](#).

Next steps

For an overview of the Azure Database Migration Service and regional availability, see the article [What is the Azure Database Migration Service](#).

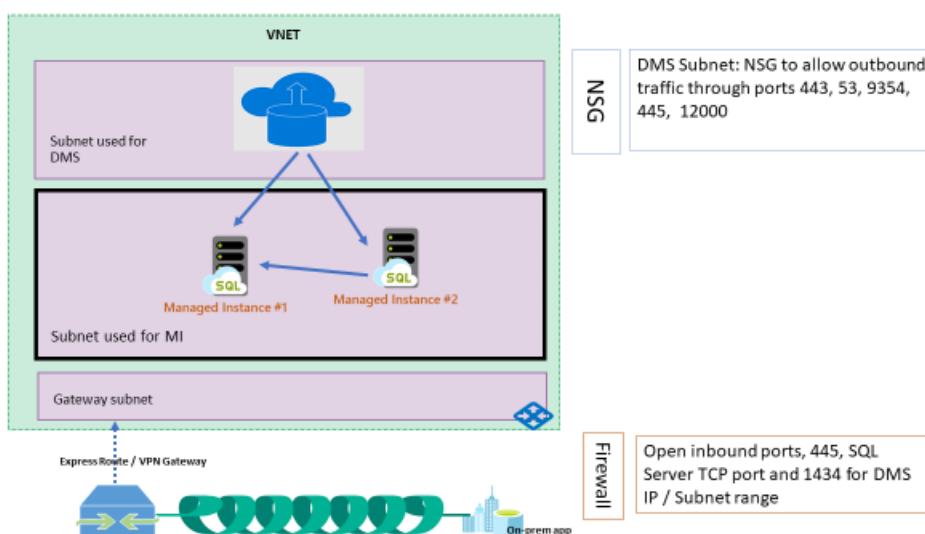
Network topologies for Azure SQL DB Managed Instance migrations using Azure Database Migration Service

1/8/2020 • 3 minutes to read • [Edit Online](#)

This article discusses various network topologies that Azure Database Migration Service can work with to provide a comprehensive migration experience from on-premises SQL Servers to Azure SQL Database Managed Instance.

Azure SQL Database Managed Instance configured for Hybrid workloads

Use this topology if your Azure SQL Database Managed Instance is connected to your on-premises network. This approach provides the most simplified network routing and yields maximum data throughput during the migration.



Requirements

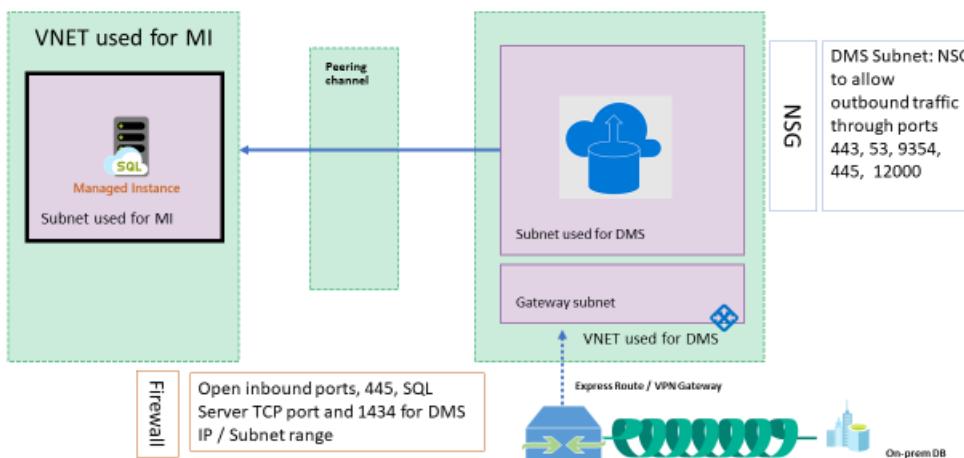
- In this scenario, the Azure SQL Database managed instance and the Azure Database Migration Service instance are created in the same Microsoft Azure Virtual Network, but they use different subnets.
- The virtual network used in this scenario is also connected to the on-premises network by using either [ExpressRoute](#) or [VPN](#).

Azure SQL Database Managed Instance isolated from the on-premises network

Use this network topology if your environment requires one or more of the following scenarios:

- The Azure SQL Database managed instance is isolated from on-premises connectivity, but your Azure Database Migration Service instance is connected to the on-premises network.
- If Role Based Access Control (RBAC) policies are in place and you need to limit the users to accessing the same subscription that is hosting the Azure SQL Database managed instance.
- The virtual networks used for the Azure SQL Database Managed Instance and Azure Database Migration

Service are in different subscriptions.

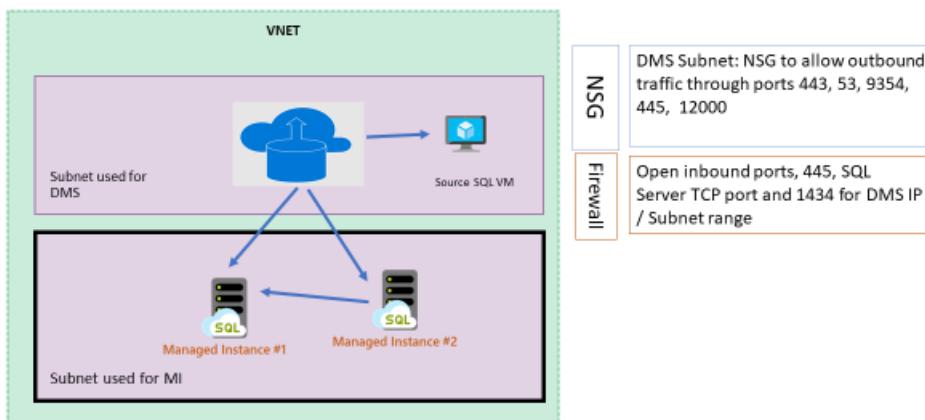


Requirements

- The virtual network that Azure Database Migration Service uses for this scenario must also be connected to the on-premises network by using either (<https://docs.microsoft.com/azure/expressroute/expressroute-introduction>) or **VPN**.
- Set up **VNet network peering** between the virtual network used for Azure SQL Database managed instance and Azure Database Migration Service.

Cloud-to-cloud migrations: Shared virtual network

Use this topology if the source SQL Server is hosted in an Azure VM and shares the same virtual network with Azure SQL Database managed instance and Azure Database Migration Service.



Requirements

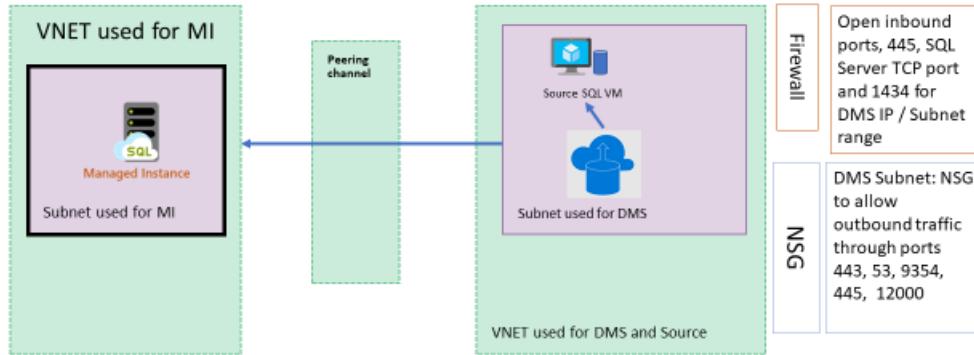
- No additional requirements.

Cloud to cloud migrations: Isolated virtual network

Use this network topology if your environment requires one or more of the following scenarios:

- The Azure SQL Database managed instance is provisioned in an isolated virtual network.

- If Role Based Access Control (RBAC) policies are in place and you need to limit the users to accessing the same subscription that is hosting the Azure SQL Database managed instance.
- The virtual networks used for Azure SQL Database Managed Instance and Azure Database Migration Service are in different subscriptions.



Requirements

- Set up [VNet network peering](#) between the virtual network used for Azure SQL Database managed instance and Azure Database Migration Service.

Inbound security rules

NAME	PORT	PROTOCOL	SOURCE	DESTINATION	ACTION
DMS_subnet	Any	Any	DMS SUBNET	Any	Allow

Outbound security rules

NAME	PORT	PROTOCOL	SOURCE	DESTINATION	ACTION	REASON FOR RULE
management	443,9354	TCP	Any	Any	Allow	Management plane communication through Service Bus and Azure blob storage. (If Microsoft peering is enabled, you may not need this rule.)

Name	Port	Protocol	Source	Destination	Action	Reason for Rule
Diagnostics	12000	TCP	Any	Any	Allow	DMS uses this rule to collect diagnostic information for troubleshooting purposes.
SQL Source server	1433 (or TCP IP port that SQL Server is listening to)	TCP	Any	On-premises address space	Allow	SQL Server source connectivity from DMS (If you have site-to-site connectivity, you may not need this rule.)
SQL Server named instance	1434	UDP	Any	On-premises address space	Allow	SQL Server named instance source connectivity from DMS (If you have site-to-site connectivity, you may not need this rule.)
SMB share	445	TCP	Any	On-premises address space	Allow	SMB network share for DMS to store database backup files for migrations to Azure SQL Database MI and SQL Servers on Azure VM (If you have site-to-site connectivity, you may not need this rule).
DMS_subnet	Any	Any	Any	DMS_Subnet	Allow	

See also

- [Migrate SQL Server to Azure SQL Database Managed Instance](#)
- [Overview of prerequisites for using Azure Database Migration Service](#)
- [Create a virtual network using the Azure portal](#)

Next steps

- For an overview of Azure Database Migration Service, see the article [What is Azure Database Migration Service?](#).
- For current information about regional availability of Azure Database Migration Service, see the [Products available by region](#) page.

Custom roles for SQL Server to SQL Database managed instance online migrations

12/18/2019 • 3 minutes to read • [Edit Online](#)

Azure Database Migration Service uses an APP ID to interact with Azure Services. The APP ID requires either the Contributor role at the Subscription level (which many Corporate security departments won't allow) or creation of custom roles that grant the specific permissions that Azure database Migrations Service requires. Since there's a limit of 2,000 custom roles in Azure Active Directory, you may want to combine all permissions required specifically by the APP ID into one or two custom roles, and then grant the APP ID the custom role on specific objects or resource groups (vs. at the subscription level). If the number of custom roles isn't a concern, you can split the custom roles by resource type, to create three custom roles in total as described below.

The AssignableScopes section of the role definition json string allows you to control where the permissions appear in the **Add Role Assignment** UI in the portal. You'll likely want to define the role at the resource group or even resource level to avoid cluttering the UI with extra roles. Note that this doesn't perform the actual role assignment.

Minimum number of roles

We currently recommend creating a minimum of two custom roles for the APP ID, one at the resource level and the other at the subscription level.

NOTE

The last custom role requirement may eventually be removed, as new SQL Database managed instance code is deployed to Azure.

Custom Role for the APP ID. This role is required for Azure Database Migration Service migration at the resource or resource group level (for more information about the APP ID, see the article [Use the portal to create an Azure AD application and service principal that can access resources](#)).

```
{
  "Name": "DMS Role - App ID",
  "IsCustom": true,
  "Description": "DMS App ID access to complete MI migrations",
  "Actions": [
    "Microsoft.Storage/storageAccounts/read",
    "Microsoft.Storage/storageAccounts/listKeys/action",
    "Microsoft.Storage/storageAccounts/blobServices/read",
    "Microsoft.Storage/storageAccounts/blobServices/write",
    "Microsoft.Sql/managedInstances/read",
    "Microsoft.Sql/managedInstances/write",
    "Microsoft.Sql/managedInstances/databases/read",
    "Microsoft.Sql/managedInstances/databases/write",
    "Microsoft.Sql/managedInstances/databases/delete",
    "Microsoft.Sql/managedInstances/metrics/read",
    "Microsoft.DataMigration/locations/*",
    "Microsoft.DataMigration/services/*"
  ],
  "NotActions": [
  ],
  "AssignableScopes": [
    "/subscriptions/<subscription_id>/ResourceGroups/<StorageAccount_rg_name>",
    "/subscriptions/<subscription_id>/ResourceGroups/<ManagedInstance_rg_name>",
    "/subscriptions/<subscription_id>/ResourceGroups/<DMS_rg_name>",
  ]
}
}
```

Custom role for the APP ID - subscription. This role is required for Azure Database Migration Service migration at *subscription* level.

```
{
  "Name": "DMS Role - App ID - Sub",
  "IsCustom": true,
  "Description": "DMS App ID access at subscription level to complete MI migrations",
  "Actions": [
    "Microsoft.Sql/locations/managedDatabaseRestoreAzureAsyncOperation/*"
  ],
  "NotActions": [
  ],
  "AssignableScopes": [
    "/subscriptions/<subscription_id>"
  ]
}
```

The json above must be stored in three text files, and you can use either the AzureRM, AZ PowerShell cmdlets, or Azure CLI to create the roles using either **New-AzureRmRoleDefinition (AzureRM)** or **New-AzRoleDefinition (AZ)**.

For more information, see the article [Custom roles for Azure resources](#).

After you create these custom roles, you must add role assignments to users and APP ID(s) to the appropriate resources or resource groups:

- The “DMS Role - App ID” role must be granted to the APP ID that will be used for the migrations, and also at the Storage Account, Azure Database Migration Service instance, and SQL Database managed instance resource levels.
- The “DMS Role - App ID - Sub” role must be granted to the APP ID at the subscription level (granting at the resource or resource group will fail). This requirement is temporary until a code update is deployed.

Expanded number of roles

If the number of custom roles in your Azure Active Directory isn't a concern, we recommend you create a total of three roles. You'll still need the "DMS Role - App ID – Sub" role, but the "DMS Role - App ID" role above is split by resource type into two different roles.

Custom role for the APP ID for SQL Database managed instance

```
{  
  "Name": "DMS Role - App ID - SQL MI",  
  "IsCustom": true,  
  "Description": "DMS App ID access to complete MI migrations",  
  "Actions": [  
    "Microsoft.Sql/managedInstances/read",  
    "Microsoft.Sql/managedInstances/write",  
    "Microsoft.Sql/managedInstances/databases/read",  
    "Microsoft.Sql/managedInstances/databases/write",  
    "Microsoft.Sql/managedInstances/databases/delete",  
    "Microsoft.Sql/managedInstances/metrics/read"  
,  
  ],  
  "NotActions": [  
  ],  
  "AssignableScopes": [  
    "/subscriptions/<subscription_id>/resourceGroups/<ManagedInstance_rg_name>"  
  ]  
}
```

Custom role for the APP ID for Storage

```
{  
  "Name": "DMS Role - App ID - Storage",  
  "IsCustom": true,  
  "Description": "DMS App ID storage access to complete MI migrations",  
  "Actions": [  
    "Microsoft.Storage/storageAccounts/read",  
    "Microsoft.Storage/storageAccounts/listKeys/action",  
    "Microsoft.Storage/storageaccounts/blobservices/read",  
    "Microsoft.Storage/storageaccounts/blobservices/write"  
,  
  ],  
  "NotActions": [  
  ],  
  "AssignableScopes": [  
    "/subscriptions/<subscription_id>/resourceGroups/<StorageAccount_rg_name>"  
  ]  
}
```

Role assignment

To assign a role to users/APP ID, open the Azure portal, perform the following steps:

1. Navigate to the resource group or resource (except for the role that needs to be granted on the subscription), go to **Access Control**, and then scroll to find the custom roles you just created.
2. Select the appropriate role, select the APP ID, and then save the changes.

Your APP ID(s) now appears listed on the **Role assignments** tab.

Next steps

- Review the migration guidance for your scenario in the Microsoft [Database Migration Guide](#).

Troubleshoot DMS errors when connecting to source databases

2/26/2020 • 7 minutes to read • [Edit Online](#)

The following article provides detail about how to address potential issues you might encounter when connecting the Azure Database Migration Service (DMS) to your source database. Each section below relates to a specific type of source database, listing the error you might encounter together with detail and links to information about how to troubleshoot the connectivity.

SQL Server

Potential issues associated with connecting to a source SQL Server database and how to address them are provided in the following table.

ERROR	CAUSE AND TROUBLESHOOTING DETAIL
SQL connection failed. A network-related or instance-specific error occurred while establishing a connection to SQL Server. The server was not found or was not accessible. Verify that the instance name is correct, and that SQL Server is configured to allow remote connections.	This error occurs if the service can't locate the source server. To address the issue, see the article Error connecting to source SQL Server when using dynamic port or named instance .
Error 53 - SQL connection failed. (Also, for error codes 1, 2, 5, 53, 233, 258, 1225, 11001)	This error occurs if the service can't connect to the source server. To address the issue, refer to the following resources, and then try again. Interactive user guide to troubleshoot the connectivity issue Prerequisites for migrating SQL Server to Azure SQL Database Prerequisites for migrating SQL Server to an Azure SQL Database managed instance
Error 18456 - Login failed.	This error occurs if the service can't connect to the source database using the provided T-SQL credentials. To address the issue, verify the entered credentials. You can also refer to MSSQLSERVER_18456 or to the troubleshooting documents listed in the note below this table, and then try again.
Malformed AccountName value '{0}' provided. Expected format for AccountName is DomainName\UserName	This error occurs if the user selects Windows authentication but provides the username in an invalid format. To address the issue, either provide username in the correct format for Windows authentication or select SQL Authentication .

AWS RDS MySQL

Potential issues associated with connecting to a source AWS RDS MySQL database and how to address them are provided in the following table.

ERROR	CAUSE AND TROUBLESHOOTING DETAIL
Error [2003][HY000] - connection failed. ERROR [HY000] [MySQL][ODBC x.x(w) driver] Can't connect to MySQL server on '{server}' (10060)	This error occurs if the MySQL ODBC driver can't connect to the source server. To address the issue, refer to the troubleshooting documents listed in the note below this table, and then try again.
Error [2005][HY000] - connection failed. ERROR [HY000] [MySQL][ODBC x.x(w) driver] Unknown MySQL server host '{server}'	This error occurs if the service can't find the source host on RDS. The issue could either be because the listed source does not exist or there is a problem with RDS infrastructure. To address the issue, refer to the troubleshooting documents listed in the note below this table, and then try again.
Error [1045][HY000] - connection failed. ERROR [HY000] [MySQL][ODBC x.x(w) driver] Access denied for user '{user}'@'{server}' (using password: YES)	This error occurs if MySQL ODBC driver cannot connect to the source server due to invalid credentials. Verify the credentials you have entered. If the issue continues, verify that source computer has the correct credentials. You may need to reset the password in the console. If you still encounter the issue, refer to the troubleshooting documents listed in the note below this table, and then try again.
Error [9002][HY000] - connection failed. ERROR [HY000] [MySQL][ODBC x.x(w) driver] The connection string may not be right. Visit portal for references.	This error occurs if the connection is failing due to an issue with the connection string. Verify the connection string provided is valid. To address the issue, refer to the troubleshooting documents listed in the note below this table, and then try again.
Error in binary logging. Variable binlog_format has value '{value}'. Please change it to 'row'.	This error occurs if there is an error in binary logging; the variable binlog_format has the wrong value. To address the issue, change the binlog_format in parameter group to 'ROW', and then reboot the instance. For more information, see to Binary Logging Options and Variables or AWS RDS MySQL Database Log Files documentation .

NOTE

For more information about troubleshooting issues related to connecting to a source AWS RDS MySQL database, see the following resources:

- [Troubleshooting for Amazon RDS Connectivity issues](#)
- [How do I resolve problems connecting to my Amazon RDS database instance?](#)

AWS RDS PostgreSQL

Potential issues associated with connecting to a source AWS RDS PostgreSQL database and how to address them are provided in the following table.

ERROR	CAUSE AND TROUBLESHOOTING DETAIL
Error [101][08001] - connection failed. ERROR [08001] timeout expired.	This error occurs if the Postgres driver can't connect to the source server. To address the issue, refer to the troubleshooting documents listed in the note below this table, and then try again.

ERROR	CAUSE AND TROUBLESHOOTING DETAIL
Error: Parameter wal_level has value '{value}'. Please change it to 'logical' to allow replication.	<p>This error occurs if the parameter wal_level has the wrong value. To address the issue, change the rds.logical_replication in parameter group to 1, and then reboot the instance. For more information, see to Pre-requisites for migrating to Azure PostgreSQL using DMS or PostgreSQL on Amazon RDS.</p>

NOTE

For more information about troubleshooting issues related to connecting to a source AWS RDS PostgreSQL database, see the following resources:

- [Troubleshooting for Amazon RDS Connectivity issues](#)
- [How do I resolve problems connecting to my Amazon RDS database instance?](#)

AWS RDS SQL Server

Potential issues associated with connecting to a source AWS RDS SQL Server database and how to address them are provided in the following table.

ERROR	CAUSE AND TROUBLESHOOTING DETAIL
Error 53 - SQL connection failed. A network-related or instance-specific error occurred while establishing a connection to SQL Server. The server wasn't found or wasn't accessible. Verify that the instance name is correct, and that SQL Server is configured to allow remote connections. (provider: Named Pipes Provider, error: 40 - Could not open a connection to SQL Server)	<p>This error occurs if the service can't connect to the source server. To address the issue, refer to the troubleshooting documents listed in the note below this table, and then try again.</p>
Error 18456 - Login failed. Login failed for user '{user}'	<p>This error occurs if the service can't connect to the source database with the T-SQL credentials provided. To address the issue, verify the entered credentials. You can also refer to MSSQLSERVER_18456 or to the troubleshooting documents listed in the note below this table, and try again.</p>
Error 87 - Connection string is not valid. A network-related or instance-specific error occurred while establishing a connection to SQL Server. The server was not found or was not accessible. Verify that the instance name is correct, and that SQL Server is configured to allow remote connections. (provider: SQL Network Interfaces, error: 25 - Connection string is not valid)	<p>This error occurs if the service can't connect to the source server because of an invalid connection string. To address the issue, verify the connection string provided. If the issue persists, refer to the troubleshooting documents listed in the note below this table, and then try again.</p>
Error - Server certificate not trusted. A connection was successfully established with the server, but then an error occurred during the login process. (provider: SSL Provider, error: 0 - The certificate chain was issued by an authority that is not trusted.)	<p>This error occurs if the certificate used isn't trusted. To address the issue, you need to find a certificate that can be trusted, and then enable it on the server. Alternatively, you can select the Trust Certificate option while connecting. Take this action only if you're familiar with the certificate used and you trust it. SSL connections that are encrypted using a self-signed certificate don't provide strong security -- they're susceptible to man-in-the-middle attacks. Do not rely on SSL using self-signed certificates in a production environment or on servers that are connected to the internet.</p> <p>For more information, see to Using SSL with a Microsoft SQL Server DB Instance or Tutorial: Migrate RDS SQL Server to Azure using DMS.</p>

ERROR	CAUSE AND TROUBLESHOOTING DETAIL
Error 300 - User does not have required permissions. VIEW SERVER STATE permission was denied on object '{server}', database '{database}'	This error occurs if user doesn't have permission to perform the migration. To address the issue, refer to GRANT Server Permissions - Transact-SQL or Tutorial: Migrate RDS SQL Server to Azure using DMS for more details.

NOTE

For more information about troubleshooting issues related to connecting to a source AWS RDS SQL Server, see the following resources:

- [Solving Connectivity errors to SQL Server](#)
- [How do I resolve problems connecting to my Amazon RDS database instance?](#)

Known issues

- [Known issues/migration limitations with online migrations to Azure SQL Database](#)
- [Known issues/migration limitations with online migrations to Azure Database for MySQL](#)
- [Known issues/migration limitations with online migrations to Azure Database for PostgreSQL](#)

Next steps

- View the article [Azure Database Migration Service PowerShell](#).
- View the article [How to configure server parameters in Azure Database for MySQL by using the Azure portal](#).
- View the article [Overview of prerequisites for using Azure Database Migration Service](#).
- See the [FAQ about using Azure Database Migration Service](#).

Troubleshoot common Azure Database Migration Service issues and errors

2/26/2020 • 7 minutes to read • [Edit Online](#)

This article describes some common issues and errors that Azure Database Migration Service users can come across. The article also includes information about how to resolve these issues and errors.

Migration activity in queued state

When you create new activities in an Azure Database Migration Service project, the activities remain in a queued state.

CAUSE	RESOLUTION
This issue happens when the Azure Database Migration Service instance has reached maximum capacity for ongoing tasks that concurrently run. Any new activity is queued until the capacity becomes available.	Validate the Data Migration Service instance has running activities across projects. You can continue to create new activities that automatically get added to the queue for execution. As soon as any of the existing running activities complete, the next queued activity starts running and the status changes to running state automatically. You don't need to take any additional action to start migration of queued activity.

Max number of databases selected for migration

The following error occurs when creating an activity for a database migration project for moving to Azure SQL Database or an Azure SQL Database managed instance:

- **Error:** Migration settings validation error", "errorDetail":"More than max number '4' objects of 'Databases' has been selected for migration."

CAUSE	RESOLUTION
This error displays when you've selected more than four databases for a single migration activity. At present, each migration activity is limited to four databases.	Select four or fewer databases per migration activity. If you need to migrate more than four databases in parallel, provision another instance of Azure Database Migration Service. Currently, each subscription supports up to two Azure Database Migration Service instances.

Errors for MySQL migration to Azure MySQL with recovery failures

When you migrate from MySQL to Azure Database for MySQL using Azure Database Migration Service, the migration activity fails with the following error:

- **Error:** Database migration error - Task 'TaskID' was suspended due to [n] successive recovery failures.

CAUSE	RESOLUTION
This error may occur when the user doing the migration is missing ReplicationAdmin role and/or privileges of REPLICATION CLIENT, REPLICATION REPLICA, and SUPER (versions earlier than MySQL 5.6.6).	<p>Make sure the pre-requisite privileges for the user account are configured accurately on the Azure Database for MySQL instance. For example, the following steps can be followed to create a user named 'migrateuser' with required privileges:</p> <ol style="list-style-type: none"> 1. CREATE USER migrateuser@'%' IDENTIFIED BY 'secret'; 2. Grant all privileges on db_name.* to 'migrateuser'@'%' identified by 'secret'; // repeat this step to grant access on more databases 3. Grant replication slave on . to 'migrateuser'@'%' identified by 'secret'; 4. Grant replication client on . to 'migrateuser'@'%' identified by 'secret'; 5. Flush privileges;

Error when attempting to stop Azure Database Migration Service

You receive following error when stopping the Azure Database Migration Service instance:

- **Error:** Service failed to Stop. Error: {`'error':{'code':'InvalidRequest','message':'One or more activities are currently running. To stop the service, wait until the activities have completed or stop those activities manually and try again.'}}`

CAUSE	RESOLUTION
This error displays when the service instance you're attempting to stop includes activities that are still running or present in migration projects.	<p>Ensure that there are no activities running in the instance of Azure Database Migration Service you're trying to stop. You may also delete the activities or projects before attempting to stop the service. The following steps illustrate how to remove projects to clean up the migration service instance by deleting all running tasks:</p> <ol style="list-style-type: none"> 1. Install-Module -Name AzureRM.DataMigration 2. Login-AzureRmAccount 3. Select-AzureRmSubscription -SubscriptionName "<code><subName></code>" 4. Remove-AzureRmDataMigrationProject -Name <code>< projectName ></code> -ResourceGroupName <code><rgName></code> -ServiceName <code>< serviceName ></code> -DeleteRunningTask

Error when attempting to start Azure Database Migration Service

You receive following error when starting the Azure Database Migration Service instance:

- **Error:** Service fails to Start. Error: {`'errorDetail':'The service failed to start, please contact Microsoft support'`}

CAUSE	RESOLUTION
This error displays when the previous instance failed internally. This error occurs rarely, and the engineering team is aware of it.	Delete the instance of the service that you cannot start, and then provision new one to replace it.

Error restoring database while migrating SQL to Azure SQL DB managed instance

When you perform an online migration from SQL Server to an Azure SQL Database managed instance, the cutover fails with following error:

- **Error:** Restore Operation failed for operation Id 'operationId'. Code 'AuthorizationFailed', Message 'The client 'clientId' with object id 'objectId' does not have authorization to perform action 'Microsoft.Sql/locations/managedDatabaseRestoreAzureAsyncOperation/read' over scope '/subscriptions/subscriptionId'.'

CAUSE	RESOLUTION
This error indicates the application principal being used for online migration from SQL Server to an Azure SQL Database managed instance doesn't have contribute permission on the subscription. Certain API calls with Managed Instance at present require this permission on subscription for the restore operation.	<p>Use the <code>Get-AzureADServicePrincipal</code> PowerShell cmdlet with <code>-ObjectId</code> available from the error message to list the display name of the application ID being used.</p> <p>Validate the permissions to this application and ensure it has the contributor role at the subscription level.</p> <p>The Azure Database Migration Service Engineering Team is working to restrict the required access from current contribute role on subscription. If you have a business requirement that doesn't allow use of contribute role, contact Azure support for additional help.</p>

Error when deleting NIC associated with Azure Database Migration Service

When you try to delete a Network Interface Card associated with Azure Database Migration Service, the deletion attempt fails with this error:

- **Error:** Cannot delete the NIC associated to Azure Database Migration Service due to the DMS service utilizing the NIC

CAUSE	RESOLUTION
This issue happens when the Azure Database Migration Service instance may still be present and consuming the NIC.	<p>To delete this NIC, delete the DMS service instance that automatically deletes the NIC used by the service.</p> <p>Important: Make sure the Azure Database Migration Service instance being deleted has no running activities.</p> <p>After all the projects and activities associated to the Azure Database Migration Service instance are deleted, you can delete the service instance. The NIC used by the service instance is automatically cleaned as part of service deletion.</p>

Connection error when using ExpressRoute

When you try to connect to source in the Azure Database Migration service project wizard, the connection fails after prolonged timeout if source is using ExpressRoute for connectivity.

Cause	Resolution
<p>When using ExpressRoute, Azure Database Migration Service requires provisioning three service endpoints on the Virtual Network subnet associated with the service:</p> <ul style="list-style-type: none"> -- Service Bus endpoint -- Storage endpoint -- Target database endpoint (e.g. SQL endpoint, Cosmos DB endpoint) 	<p>Enable the required service endpoints for ExpressRoute connectivity between source and Azure Database Migration Service.</p>

Lock wait timeout error when migrating a MySQL database to Azure DB for MySQL

When you migrate a MySQL database to an Azure Database for MySQL instance via Azure Database Migration Service, the migration fails with following lock wait timeout error:

- **Error:** Database migration error - Failed to load file - Failed to start load process for file 'n' RetCode: SQL_ERROR SqlState: HY000 NativeError: 1205 Message: [MySQL][ODBC Driver][mysqld] Lock wait timeout exceeded; try restarting transaction

Cause	Resolution
<p>This error occurs when migration fails because of the lock wait timeout during migration.</p>	<p>Consider increasing the value of server parameter 'innodb_lock_wait_timeout'. The highest allowed value is 1073741824.</p>

Error connecting to source SQL Server when using dynamic port or named instance

When you try to connect Azure Database Migration Service to SQL Server source that runs on either named instance or a dynamic port, the connection fails with this error:

- **Error:** -1 - SQL connection failed. A network-related or instance-specific error occurred while establishing a connection to SQL Server. The server was not found or was not accessible. Verify that the instance name is correct and that SQL Server is configured to allow remote connections. (provider: SQL Network Interfaces, error: 26 - Error Locating Server/Instance Specified)

Cause	Resolution
<p>This issue happens when the source SQL Server instance that Azure Database Migration Service tries to connect to either has a dynamic port or is using a named instance. The SQL Server Browser service listens to UDP port 1434 for incoming connections to a named instance or when using a dynamic port. The dynamic port may change each time SQL Server service restarts. You can check the dynamic port assigned to an instance via network configuration in SQL Server Configuration Manager.</p>	<p>Verify that Azure Database Migration Service can connect to the source SQL Server Browser service on UDP port 1434 and the SQL Server instance through the dynamically assigned TCP port as applicable.</p>

Additional known issues

- [Known issues/migration limitations with online migrations to Azure SQL Database](#)
- [Known issues/migration limitations with online migrations to Azure Database for MySQL](#)
- [Known issues/migration limitations with online migrations to Azure Database for PostgreSQL](#)

Next steps

- View the article [Azure Database Migration Service PowerShell](#).
- View the article [How to configure server parameters in Azure Database for MySQL by using the Azure portal](#).
- View the article [Overview of prerequisites for using Azure Database Migration Service](#).
- See the [FAQ about using Azure Database Migration Service](#).

Known issues/migration limitations with online migrations to Azure SQL Database

2/26/2020 • 3 minutes to read • [Edit Online](#)

Known issues and limitations associated with online migrations from SQL Server to Azure SQL Database are described below.

IMPORTANT

With online migrations of SQL Server to Azure SQL Database, migration of SQL_variant data types is not supported.

Migration of temporal tables not supported

Symptom

If your source database consists of one or more temporal tables, your database migration fails during the "Full data load" operation and you may see the following message:

```
{ "resourceId":"/subscriptions/<subscription id>/resourceGroups/migrateready/providers/Microsoft.DataMigration/services/<DMS Service name>",
  "errorType":"Database migration error", "errorEvents": "["Capture functionalities could not be set. RetCode: SQL_ERROR SqlState: 42000 NativeError: 13570 Message: [Microsoft][SQL Server Native Client 11.0][SQL Server]The use of replication is not supported with system-versioned temporal table '[Application. Cities]' Line: 1 Column: -1 "] }
```

The screenshot shows the Azure Database Migration Service portal. On the left, there's a navigation tree: Home > migrateready > Onlinemigration > test. Below it, a 'test' activity is listed with a 'Failed' status. The activity details show the source server (10.139.160.105) and target server (dmazuresqldbservice.database.windows.net). The migration status is 'Failed' with a duration of 00:00:41. A red banner at the top indicates 'Database migration errors'. On the right, an 'Error Detail' modal is open, showing the error type as 'Database migration error' and the error events as: 'Capture functionalities could not be set. RetCode: SQL_ERROR SqlState: 42000 NativeError: 13570 Message: [Microsoft][SQL Server Native Client 11.0][SQL Server]The use of replication is not supported with system-versioned temporal table '[Application. Cities]' Line: 1 Column: -1 '. There's also a link to the 'Feedback Forum for Azure Database Migration Service'.

Workaround

Use the following steps.

1. Find the temporal tables in your source schema using the query below.

```
select name,temporal_type,temporal_type_desc,* from sys.tables where temporal_type <>0
```

2. Exclude these tables from the **Configure migration settings** blade, on which you specify tables for migration.
3. Rerun the migration activity.

Resources

For more information, see the article [Temporal Tables](#).

Migration of tables includes one or more columns with the hierarchyid data type

Symptom

You may see a SQL Exception suggesting "ntext is incompatible with hierarchyid" during the "Full data load" operation:

The screenshot shows the Azure Database Migration Service portal. On the left, a list of migration activities is shown, with one activity named 'test' failing. A red banner at the top of the activity details page indicates 'Database migration errors'. The activity summary table shows the source server (10.105.129.164) and target server (dmazuresqldbserver.database.windows.net). The activity status is 'Failed' with a duration of 00:00:19. In the 'Migration Details' section, there is a link to 'See error details'. On the right, an 'Error Detail' pane is open, displaying the error message: "Truncation of a column occurred while fetching a value from array", followed by several SQL error messages related to the 'Production.Document' table and its 'Document' column, which is defined as a hierarchyid type.

Workaround

Use the following steps.

1. Find the user tables that include columns with the hierarchyid data type using the query below.

```
select object_name(object_id) 'Table name' from sys.columns where system_type_id = 240 and object_id in (select object_id from sys.objects where type='U')
```

2. Exclude these tables from the **Configure migration settings** blade, on which you specify tables for migration.
3. Rerun the migration activity.

Migration failures with various integrity violations with active triggers in the schema during "Full data load" or "Incremental data sync"

Workaround

Use the following steps.

1. Find the triggers that are currently active in the source database using the query below:

```
select * from sys.triggers where is_disabled =0
```

2. Disable the triggers on your source database using the steps provided in the article [DISABLE TRIGGER \(Transact-SQL\)](#).

3. Rerun the migration activity.

Support for LOB data types

Symptom

If the length of Large Object (LOB) column is bigger than 32 KB, data might get truncated at the target. You can check the length of LOB column using the query below:

```
SELECT max(DATALENTH(Column Name)) as LEN from TableName
```

Workaround

If you have an LOB column that is bigger than 32 KB, contact the engineering team at [Ask Azure Database Migrations](#).

Issues with timestamp columns

Symptom

Azure Database Migration Service doesn't migrate the source timestamp value; instead, Azure Database Migration Service generates a new timestamp value in the target table.

Workaround

If you need Azure Database Migration Service to migrate the exact timestamp value stored in the source table, contact the engineering team at [Ask Azure Database Migrations](#).

Data migration errors don't provide additional details on the Database detailed status blade

Symptom

When you come across migration failures in the Databases details status view, selecting the **Data migration errors** link on the top ribbon may not provide additional details specific to the migration failures.

The screenshot shows the Azure Database Migration Service interface. At the top, there's a navigation bar with 'Home > migrateready > Onlinemigration > test > WideWorldImporters'. Below this is a 'Database migration errors' section with a red warning icon. The main area displays migration status for 'WideWorldImporters' across three categories: Source database name, Target database name, and Database status. Under 'Migration details', it shows 'Full load failed' for Failed items. A table below lists migration progress for various tables like Application.Cities, Application.Countries, etc., all in 'Initializing' status. On the right, an 'Error Detail' pane is open, showing 'Database migration error' and a message about failed databases. It also includes a 'Feedback Forum' link and a 'Still having trouble?' section.

Workaround

To get to specific failure details, use the following steps.

1. Close the Database detailed status blade to display the Migration activity screen.

This screenshot shows the migration activity screen for the 'test' database. It includes sections for 'Source server' (10.139.160.105), 'Target server' (dmazuresql01.database.windows.net), and 'Activity status' (Failed). Below these are tables for 'Migration Details' and 'Downtime'. The 'Migration Details' table shows 'WideWorldImporters' failed with a duration of 00:00:41. The 'Downtime' table is empty. To the right, an 'Error Detail' pane is open, showing 'Database migration error' and a detailed error message about capture functionalities not being supported. It also includes a 'Feedback Forum' link and a 'Still having trouble?' section.

2. Select **See error details** to view specific error messages that help you to troubleshoot migration errors.

Geography datatype not supported in SQLDB online migration

Symptom

Migration fails with an error message containing the following text:

```
“** encountered a fatal error”, “errorEvents”:<Table>.<Column> is of type ‘GEOGRAPHY’, which is not supported by ‘Full Load’ under ‘Full LOB’ support mode.”
```

Workaround

While Azure Database Migration Service supports the Geography data type for offline migrations to Azure SQL Database, for online migrations, the Geography datatype is not supported. Try alternate methods to change the datatype at the source to a supported type before attempting to use Azure Database Migration Service for an online migration of this database.

Supported editions

Symptom

Migration fails with an error message containing the following text:

```
Migration settings validation error: The edition of the server [Business Intelligence Edition (64-bit)] does not match the supported edition(s) [Enterprise,Standard,Developer].
```

Workaround

Support for online migrations to Azure SQL Database using Azure Database Migration Service extends only to the Enterprise, Standard, and Developer editions. Be sure that you are using a supported edition before beginning the migration process.

Known issues/migration limitations with online migrations to Azure SQL Database managed instance

2/26/2020 • 2 minutes to read • [Edit Online](#)

Known issues and limitations that are associated with online migrations from SQL Server to Azure SQL Database managed instance are described below.

IMPORTANT

With online migrations of SQL Server to Azure SQL Database, migration of SQL_variant data types is not supported.

Backup requirements

- **Backups with checksum**

Azure Database Migration Service uses the backup and restore method to migrate your on-premises databases to SQL Database managed instance. Azure Database Migration Service only supports backups created using checksum.

[Enable or Disable Backup Checksums During Backup or Restore \(SQL Server\)](#)

NOTE

If you take the database backups with compression, the checksum is a default behavior unless explicitly disabled.

With offline migrations, if you choose **I will let Azure Database Migration Service...**, then Azure Database Migration Service will take the database backup with the checksum option enabled.

- **Backup media**

Make sure to take every backup on a separate backup media (backup files). Azure Database Migration Service doesn't support backups that are appended to a single backup file. Take full backup and log backups to separate backup files.

Data and log file layout

- **Number of log files**

Azure Database Migration Service doesn't support databases with multiple log files. If you have multiple log files, shrink and reorganize them into a single transaction log file. Because you can't remote to log files that aren't empty, you need to back up the log file first.

SQL Server features

- **FileStream/FileTables**

SQL Database managed instance currently doesn't support FileStream and FileTables. For workloads dependent on these features, we recommend that you opt for SQL Servers running on Azure VMs as your Azure target.

- **In-memory tables**

In-memory OLTP is available in the Premium and Business Critical tiers for SQL Database managed instance; the General Purpose tier doesn't support In-memory OLTP.

Migration resets

- **Deployments**

SQL Database managed instance is a PaaS service with automatic patching and version updates. During migration of your SQL Database managed instance, non-critical updates are held up to 36 hours.

Afterwards (and for critical updates), if the migration is disrupted, the process resets to a full restore state.

Migration cutover can only be called after the full backup is restored and catches up with all log backups. If your production migration cutovers are affected, contact the [Azure DMS Feedback alias](#).

Online migration issues & limitations to Azure DB for MySQL with Azure Database Migration Service

2/26/2020 • 5 minutes to read • [Edit Online](#)

Known issues and limitations associated with online migrations from MySQL to Azure Database for MySQL are described in the following sections.

Online migration configuration

- The source MySQL Server version must be version 5.6.35, 5.7.18 or later
- Azure Database for MySQL supports:
 - MySQL community edition
 - InnoDB engine
- Same version migration. Migrating MySQL 5.6 to Azure Database for MySQL 5.7 isn't supported.
- Enable binary logging in my.ini (Windows) or my.cnf (Unix)
 - Set Server_id to any number larger or equals to 1, for example, Server_id=1 (only for MySQL 5.6)
 - Set log-bin = <path> (only for MySQL 5.6)
 - Set binlog_format = row
 - Expire_logs_days = 5 (recommended - only for MySQL 5.6)
- User must have the ReplicationAdmin role.
- Collations defined for the source MySQL database are the same as the ones defined in target Azure Database for MySQL.
- Schema must match between source MySQL database and target database in Azure Database for MySQL.
- Schema in target Azure Database for MySQL must not have foreign keys. Use the following query to drop foreign keys:

```
SET group_concat_max_len = 8192;
SELECT SchemaName, GROUP_CONCAT(DropQuery SEPARATOR ';\n') as DropQuery, GROUP_CONCAT(AddQuery SEPARATOR
';\n') as AddQuery
FROM
(SELECT
KCU.REFERENCED_TABLE_SCHEMA as SchemaName, KCU.TABLE_NAME, KCU.COLUMN_NAME,
CONCAT('ALTER TABLE ', KCU.TABLE_NAME, ' DROP FOREIGN KEY ', KCU.CONSTRAINT_NAME) AS DropQuery,
CONCAT('ALTER TABLE ', KCU.TABLE_NAME, ' ADD CONSTRAINT ', KCU.CONSTRAINT_NAME, ' FOREIGN KEY (`',
KCU.COLUMN_NAME, `)` REFERENCES `', KCU.REFERENCED_TABLE_NAME, ``(`', KCU.REFERENCED_COLUMN_NAME, ``)
ON UPDATE ', RC.UPDATE_RULE, ' ON DELETE ', RC.DELETE_RULE) AS AddQuery
FROM INFORMATION_SCHEMA.KEY_COLUMN_USAGE KCU, information_schema.REFERENTIAL_CONSTRAINTS RC
WHERE
KCU.CONSTRAINT_NAME = RC.CONSTRAINT_NAME
AND KCU.REFERENCED_TABLE_SCHEMA = RC.UNIQUE_CONSTRAINT_SCHEMA
AND KCU.REFERENCED_TABLE_SCHEMA = ['schema_name']) Queries
GROUP BY SchemaName;
```

Run the drop foreign key (which is the second column) in the query result.

- Schema in target Azure Database for MySQL must not have any triggers. To drop triggers in target database:

```
SELECT Concat('DROP TRIGGER ', Trigger_Name, ';') FROM information_schema.TRIGGERS WHERE TRIGGER_SCHEMA = 'your_schema';
```

Datatype limitations

- **Limitation:** If there's a JSON datatype in the source MySQL database, migration will fail during continuous sync.

Workaround: Modify JSON datatype to medium text or longtext in source MySQL database.

- **Limitation:** If there's no primary key on tables, continuous sync will fail.

Workaround: Temporarily set a primary key for the table for migration to continue. You can remove the primary key after data migration is complete.

LOB limitations

Large Object (LOB) columns are columns that could grow large in size. For MySQL, Medium text, Longtext, Blob, Mediumblob, Longblob, etc., are some of the datatypes of LOB.

- **Limitation:** If LOB data types are used as primary keys, migration will fail.

Workaround: Replace primary key with other datatypes or columns that aren't LOB.

- **Limitation:** If the length of Large Object (LOB) column is bigger than 32 KB, data might be truncated at the target. You can check the length of LOB column using this query:

```
SELECT max(length(description)) as LEN from catalog;
```

Workaround: If you have LOB object that is bigger than 32 KB, contact engineering team at [Ask Azure Database Migrations](#).

Limitations when migrating online from AWS RDS MySQL

When you try to perform an online migration from AWS RDS MySQL to Azure Database for MySQL, you may come across the following errors.

- **Error:** Database '{0}' has foreign key(s) on target. Fix the target and start a new data migration activity.
Execute below script on target to list the foreign key(s)

Limitation: If you have foreign keys in your schema, the initial load and continuous sync of the migration will fail. **Workaround:** Execute the following script in MySQL workbench to extract the drop foreign key script and add foreign key script:

```
SET group_concat_max_len = 8192; SELECT SchemaName, GROUP_CONCAT(DropQuery SEPARATOR '\n') as DropQuery, GROUP_CONCAT(AddQuery SEPARATOR '\n') as AddQuery FROM (SELECT KCU.REFERENCED_TABLE_SCHEMA as SchemaName, KCU.TABLE_NAME, KCU.COLUMN_NAME, CONCAT('ALTER TABLE ', KCU.TABLE_NAME, ' DROP FOREIGN KEY ', KCU.CONSTRAINT_NAME) AS DropQuery, CONCAT('ALTER TABLE ', KCU.TABLE_NAME, ' ADD CONSTRAINT ', KCU.CONSTRAINT_NAME, ' FOREIGN KEY (`', KCU.COLUMN_NAME, '` REFERENCES `', KCU.REFERENCED_TABLE_NAME, '`(`', KCU.REFERENCED_COLUMN_NAME, `') ON UPDATE ', RC.UPDATE_RULE, ' ON DELETE ', RC.DELETE_RULE) AS AddQuery FROM INFORMATION_SCHEMA.KEY_COLUMN_USAGE KCU, information_schema.REFERENTIAL_CONSTRAINTS RC WHERE KCU.CONSTRAINT_NAME = RC.CONSTRAINT_NAME AND KCU.REFERENCED_TABLE_SCHEMA = RC.UNIQUE_CONSTRAINT_SCHEMA AND KCU.REFERENCED_TABLE_SCHEMA = 'SchemaName') Queries GROUP BY SchemaName;
```

- **Error:** Database '{0}' does not exist on server. Provided MySQL source server is case sensitive. Please check the database name.

Limitation: When migrating a MySQL database to Azure using Command Line Interface (CLI), users may hit this error. The service couldn't locate the database on the source server, which could be because you may have provided incorrect database name or the database doesn't exist on the listed server. Note database names are case-sensitive.

Workaround: Provide the exact database name, and then try again.

- **Error:** There are tables with the same name in the database '{database}'. Azure Database for MySQL does not support case sensitive tables.

Limitation: This error happens when you have two tables with the same name in the source database. Azure Database for MySQL doesn't support case-sensitive tables.

Workaround: Update the table names to be unique, and then try again.

- **Error:** The target database {database} is empty. Please migrate the schema.

Limitation: This error occurs when the target Azure Database for MySQL database doesn't have the required schema. Schema migration is required to enable migrating data to your target.

Workaround: [Migrate the schema](#) from your source database to the target database.

Other limitations

- A password string that has opening and closing curly brackets {} at the beginning and end of the password string isn't supported. This limitation applies to both connecting to source MySQL and target Azure Database for MySQL.
- The following DDLs aren't supported:
 - All partition DDLs
 - Drop table
 - Rename table
- Using the `alter table <table_name> add column <column_name>` statement to add columns to the beginning or to the middle of a table isn't supported. The `alter table <table_name> add column <column_name>` adds the column at the end of the table.
- Indexes created on only part of the column data aren't supported. The following statement is an example that creates an index using only part of the column data:

```
CREATE INDEX partial_name ON customer (name(10));
```

- In Azure Database Migration Service, the limit of databases to migrate in one single migration activity is four.
- **Error:** Row size too large (> 8126). Changing some columns to TEXT or BLOB may help. In current row format, BLOB prefix of 0 bytes is stored inline.

Limitation: This error happens when you're migrating to Azure Database for MySQL using the InnoDB storage engine and any table row size is too large (>8126 bytes).

Workaround: Update the schema of the table that has a row size greater than 8126 bytes. We don't recommend changing the strict mode because the data will be truncated. Changing the page_size isn't supported.

Known issues/migration limitations with online migrations from PostgreSQL to Azure DB for PostgreSQL

2/26/2020 • 4 minutes to read • [Edit Online](#)

Known issues and limitations associated with online migrations from PostgreSQL to Azure Database for PostgreSQL are described in the following sections.

Online migration configuration

- The source PostgreSQL server must be running version 9.4, 9.5, 9.6, 10, or 11. For more information, see the article [Supported PostgreSQL Database Versions](#).
- Only migrations to the same or a higher version are supported. For example, migrating PostgreSQL 9.5 to Azure Database for PostgreSQL 9.6 or 10 is supported, but migrating from PostgreSQL 11 to PostgreSQL 9.6 isn't supported.
- To enable logical replication in the **source PostgreSQL postgresql.conf** file, set the following parameters:
 - **wal_level** = logical
 - **max_replication_slots** = [at least max number of databases for migration]; if you want to migrate four databases, set the value to at least 4.
 - **max_wal_senders** = [number of databases running concurrently]; the recommended value is 10
- Add DMS agent IP to the source PostgreSQL pg_hba.conf
 1. Make a note of the DMS IP address after you finish provisioning an instance of Azure Database Migration Service.
 2. Add the IP address to the pg_hba.conf file as shown:

```
host all 172.16.136.18/10 md5
host replication postgres 172.16.136.18/10 md5
```

- The user must have the REPLICATION role on the server hosting the source database.
- The source and target database schemas must match.
- The schema in the target Azure Database for PostgreSQL-Single server must not have foreign keys. Use the following query to drop foreign keys:

```

SELECT Queries.tablename
    ,concat('alter table ', Queries.tablename, ' ', STRING_AGG(concat('DROP CONSTRAINT ',
    Queries.foreignkey), ',')) as DropQuery
    ,concat('alter table ', Queries.tablename, ' ',
        STRING_AGG(concat('ADD CONSTRAINT ', Queries.foreignkey,
    FOREIGN KEY (' , column_name, ')', 'REFERENCES ', foreign_table_name, '(', foreign_column_name, ')'),
    ',')) as AddQuery
    FROM
    (SELECT
    tc.table_schema,
    tc.constraint_name as foreignkey,
    tc.table_name as tableName,
    kcu.column_name,
    ccu.table_schema AS foreign_table_schema,
    ccu.table_name AS foreign_table_name,
    ccu.column_name AS foreign_column_name
    FROM
    information_schema.table_constraints AS tc
    JOIN information_schema.key_column_usage AS kcu
        ON tc.constraint_name = kcu.constraint_name
        AND tc.table_schema = kcu.table_schema
    JOIN information_schema.constraint_column_usage AS ccu
        ON ccu.constraint_name = tc.constraint_name
        AND ccu.table_schema = tc.table_schema
    WHERE constraint_type = 'FOREIGN KEY') Queries
    GROUP BY Queries.tablename;

```

Run the drop foreign key (which is the second column) in the query result.

- The schema in target Azure Database for PostgreSQL-Single server must not have any triggers. Use the following to disable triggers in target database:

```

SELECT Concat('DROP TRIGGER ', Trigger_Name, ';') FROM  information_schema.TRIGGERS WHERE
TRIGGER_SCHEMA = 'your_schema';

```

Datatype limitations

Limitation: If there's no primary key on tables, changes may not be synced to the target database.

Workaround: Temporarily set a primary key for the table for migration to continue. You can remove the primary key after data migration is complete.

Limitations when migrating online from AWS RDS PostgreSQL

When you try to perform an online migration from AWS RDS PostgreSQL to Azure Database for PostgreSQL, you may encounter the following errors.

- Error:** The Default value of column '{column}' in table '{table}' in database '{database}' is different on source and target servers. It's '{value on source}' on source and '{value on target}' on target.

Limitation: This error occurs when the default value on a column schema is different between the source and target databases. **Workaround:** Ensure that the schema on the target matches schema on the source. For detail on migrating schema, refer to the [Azure PostgreSQL online migration documentation](#).

- Error:** Target database '{database}' has '{number of tables}' tables where as source database '{database}' has '{number of tables}' tables. The number of tables on source and target databases should match.

Limitation: This error occurs when the number of tables is different between the source and target databases.

Workaround: Ensure that the schema on the target matches schema on the source. For detail on migrating schema, refer to the [Azure PostgreSQL online migration documentation](#).

- **Error:** The source database {database} is empty.

Limitation: This error occurs when the source database is empty. It is most likely because you have selected the wrong database as source.

Workaround: Double-check the source database you selected for migration, and then try again.

- **Error:** The target database {database} is empty. Please migrate the schema.

Limitation: This error occurs when there's no schema on the target database. Make sure schema on the target matches schema on the source. **Workaround:** Ensure that the schema on the target matches schema on the source. For detail on migrating schema, refer to the [Azure PostgreSQL online migration documentation](#).

Other limitations

- The database name can't include a semi-colon (;).
- A captured table must have a Primary Key. If a table doesn't have a primary key, the result of DELETE and UPDATE record operations will be unpredictable.
- Updating a Primary Key segment is ignored. In such cases, applying such an update will be identified by the target as an update that didn't update any rows and will result in a record written to the exceptions table.
- Migration of multiple tables with the same name but a different case (e.g. table1, TABLE1, and Table1) may cause unpredictable behavior and is therefore not supported.
- Change processing of [CREATE | ALTER | DROP | TRUNCATE] table DDLs isn't supported.
- In Azure Database Migration Service, a single migration activity can only accommodate up to four databases.

Known issues/migration limitations with online migrations from Oracle to Azure DB for PostgreSQL-Single server

2/26/2020 • 2 minutes to read • [Edit Online](#)

Known issues and limitations associated with online migrations from Oracle to Azure Database for PostgreSQL-Single server are described in the following sections.

Oracle versions supported as a source database

Azure Database Migration Service supports connecting to:

- Oracle version 10g, 11g, and 12c.
- Oracle Enterprise, Standard, Express, and Personal Edition.

Azure Database Migration Service doesn't support connecting to multi-tenant container databases (CDBs).

PostgreSQL versions supported as a target database

Azure Database Migration Service supports migrations to Azure Database for PostgreSQL-Single server version 9.5, 9.6, 10 and 11. See the article [Supported PostgreSQL database versions](#) for current information on version support in Azure Database for PostgreSQL-Single server.

Datatype limitations

The following datatypes **won't** be migrated:

- BFILE
- ROWID
- REF
- UROWID
- ANYDATA
- SDO_GEOmetry
- Nested tables
- User-defined data types
- Notes
- Virtual columns
- Materialized views based on ROWID column

Also, empty BLOB/CLOB columns are mapped to NULL on the target.

LOB limitations

- When Limited-size LOB mode is enabled, empty LOBs on the Oracle source are replicated as NULL values.
- Long object names (over 30 bytes) aren't supported.
- Data in LONG and LONG RAW column can't exceed 64k. Any data beyond 64k will be truncated.
- In Oracle 12 only, any changes to LOB columns aren't supported (migrated).

- UPDATES to XMLTYPE and LOB columns aren't supported (migrated).

Known issues and limitations

- Customers must use SYSDBA to connect to Oracle.
- Data changes resulting from partition/sub-partition operations (ADD, DROP, EXCHANGE, and TRUNCATE) won't be migrated and may cause the following errors:
 - For ADD operations, updates and deletes on the added data may return a "0 rows affected" warning.
 - For DROP and TRUNCATE operations, new inserts may result in "duplicates" errors.
 - For EXCHANGE operations, both a "0 rows affected" warning and "duplicates" errors may occur.
- Tables whose names contain apostrophes can't be replicated.

Known issues/migration limitations with using hybrid mode

2/26/2020 • 3 minutes to read • [Edit Online](#)

Known issues and limitations associated with using Azure Database Migration Service in hybrid mode are described in the following sections.

Installer fails to authenticate

After uploading the certificate to your AdApp, there is a delay of up to a couple of minutes before it can authenticate with Azure. The installer will attempt to retry with some delay, but it's possible for the propagation delay to be longer than the retry, and you'll see a **FailedToGetAccessTokenException** message. If the certificate was uploaded to the correct AdApp and the correct AppId was provided in dmsSettings.json, try running the install command again.

Service "offline" after successful installation

If the service shows as offline after the installation process completes successfully, try using the following steps.

1. In the Azure portal, in your instance of Azure Database Migration Service, navigate to the **Hybrid** settings tab, and then verify that the worker is registered by checking the grid of registered workers.

The status of this worker should be **Online**, but it may show as **Offline** if there's a problem.

2. On the worker computer, check the status of the service by running the following PowerShell command:

```
Get-Service Scenario*
```

This command gives you the status of the Windows service running the worker. There should only be a single result. If the worker is stopped, you can attempt to restart it by using the following PowerShell command:

```
Start-Service Scenario*
```

You can also check the service in the Windows Services UI.

3. If the Windows service cycles between Running and Stopped, then the worker encountered problems starting up. Check the Azure Database Migration Service hybrid worker logs to determine the problem.
 - Installation process logs are stored in the "logs" folder within the folder from which the installer executable was run.
 - Azure Database Migration Service hybrid worker logs are stored in the **WorkerLogs** folder, in the folder in which worker is installed. The default location for the hybrid worker log files is **C:\Program Files\DatabaseMigrationServiceHybrid\WorkerLogs**.

Using your own signed certificate

The certificate generated by the action GenerateCert is a self-signed certificate, which may not be acceptable based on your internal security policies. Instead of using this certificate, you can provide your own certificate and provide

the thumbprint in dmsSettings.json. This certificate will need to be uploaded to your AdApp and installed on the computer on which you're installing the Azure Database Migration Service hybrid worker. Then, install this certificate with the private key into the Local Machine certificate store.

Running the worker service as a low-privilege account

By default, the Azure Database Migration Service hybrid worker service runs as the Local System account. You can change the account used for this service as long as the account you use has network permissions. To change the service 'run as' account, use the following process.

1. Stop the service, either through Windows Services or by using the Stop-Service command in PowerShell.
2. Update the service to use a different logon account.
3. In certmgr for Local Computer certificates, give private key permissions to the new account for the **DMS Hybrid App Key** and **DMS Scenario Engine Key Pair** certificates.
 - a. Open certmgr to view the following keys:
 - DMS Hybrid App Key
 - DMS Hybrid Worker Setup Key
 - DMS Scenario Engine Key Pair
 - b. Right-click the **DMS Hybrid App Key** entry, point to **All Tasks**, and then select **Manage Private Keys**.
 - c. On the **Security** tab, select **Add**, and then enter the name of the account.
 - d. Use the same steps to grant private key permission for the new account to the **DMS Scenario Engine Key Pair** certificate.

Unregistering the worker manually

If you no longer have access to the worker computer, you can unregister the worker and reuse your Azure Database Migration Service instance by performing the following steps:

1. In the Azure portal, go to your Azure Database Migration Service instance, and then navigate to the **Hybrid** settings page.

Your worker entry appears in the list, with the status showing as **Offline**.
2. To the far right of the worker entry listing, select the ellipses, and then select **Unregister**.

Addressing issues for specific migration scenarios

The sections below describe scenario-specific issues related to using Azure Database Migration Service hybrid mode to perform an online migration.

Online migrations to Azure SQL Database managed instance

High CPU usage

Issue: For online migrations to SQL Database managed instance, the computer running the hybrid worker will encounter high CPU usage if there are too many backups or if the backups are too large.

Mitigation: To mitigate this issue, use compressed backups, split the migration so that it uses multiple shares, or scale up the computer running the hybrid worker.

Services and tools available for data migration scenarios

2/26/2020 • 4 minutes to read • [Edit Online](#)

This article provides a matrix of the Microsoft and third-party services and tools available to assist you with various database and data migration scenarios and specialty tasks.

The following tables identify the services and tools that you can use to plan successfully for data migration and to complete its various phases.

NOTE

In the following tables, items marked with an asterisk (*) represent third-party tools.

Business justification phase

SOURCE	TARGET	DISCOVER / INVENTORY	TARGET AND SKU RECOMMENDATION	TCO/ROI AND BUSINESS CASE
SQL Server	Azure SQL DB	MAP Toolkit Azure Migrate Cloudamize*	DMA Cloud Atlas* Cloudamize*	TCO Calculator
SQL Server	Azure SQL DB MI	MAP Toolkit Azure Migrate Cloudamize*	DMA Cloud Atlas* Cloudamize*	TCO Calculator
SQL Server	Azure SQL VM	MAP Toolkit Azure Migrate Cloudamize*	Cloud Atlas* Cloudamize*	TCO Calculator
SQL Server	SQL DW	MAP Toolkit Azure Migrate Cloudamize*		TCO Calculator
RDS SQL	Azure SQL DB, MI, VM		DMA	TCO Calculator
Oracle	Azure SQL DB, MI, VM	MAP Toolkit Azure Migrate	SSMA MigVisor*	
Oracle	SQL DW	MAP Toolkit Azure Migrate	SSMA	
Oracle	Azure DB for PostgreSQL - Single server	MAP Toolkit Azure Migrate		
MongoDB	Cosmos DB	Cloudamize*	Cloudamize*	

SOURCE	TARGET	DISCOVER / INVENTORY	TARGET AND SKU RECOMMENDATION	TCO/ROI AND BUSINESS CASE
Cassandra	Cosmos DB			
MySQL	Azure SQL DB, MI, VM	Azure Migrate	SSMA Cloud Atlas*	TCO Calculator
MySQL	Azure DB for MySQL	Azure Migrate		TCO Calculator
RDS MySQL	Azure DB for MySQL			TCO Calculator
PostgreSQL	Azure DB for PostgreSQL - Single server	Azure Migrate		TCO Calculator
RDS PostgreSQL	Azure DB for PostgreSQL - Single server			TCO Calculator
DB2	Azure SQL DB, MI, VM	Azure Migrate	SSMA	
Access	Azure SQL DB, MI, VM	Azure Migrate	SSMA	
Sybase - SAP ASE	Azure SQL DB, MI, VM	Azure Migrate	SSMA	
Sybase - SAP IQ	Azure SQL DB, MI, VM			

Pre-migration phase

SOURCE	TARGET	APP DATA ACCESS LAYER ASSESSMENT	DATABASE ASSESSMENT	PERFORMANCE ASSESSMENT
SQL Server	Azure SQL DB	DAMT / DMA	DMA Cloud Atlas* Cloudamize*	DEA Cloudamize*
SQL Server	Azure SQL DB MI	DAMT / DMA	DMA Cloud Atlas* Cloudamize*	DEA Cloudamize*
SQL Server	Azure SQL VM	DAMT / DMA	DMA Cloud Atlas* Cloudamize*	DEA Cloudamize*
SQL Server	SQL DW	DAMT		
RDS SQL	Azure SQL DB, MI, VM	DAMT / DMA	DMA	DEA

SOURCE	TARGET	APP DATA ACCESS LAYER ASSESSMENT	DATABASE ASSESSMENT	PERFORMANCE ASSESSMENT
Oracle	Azure SQL DB, MI, VM	DAMT / SSMA	SSMA	
Oracle	SQL DW	DAMT / SSMA	SSMA	
Oracle	Azure DB for PostgreSQL - Single server		Ora2Pg*	
MongoDB	Cosmos DB		Cloudamize*	Cloudamize*
Cassandra	Cosmos DB			
MySQL	Azure SQL DB, MI, VM	DAMT / SSMA	SSMA Cloud Atlas*	
MySQL	Azure DB for MySQL			
RDS MySQL	Azure DB for MySQL			
PostgreSQL	Azure DB for PostgreSQL - Single server			
RDS PostgreSQL	Azure DB for PostgreSQL - Single server			
DB2	Azure SQL DB, MI, VM	DAMT / SSMA	SSMA	
Access	Azure SQL DB, MI, VM		SSMA	
Sybase - SAP ASE	Azure SQL DB, MI, VM	DAMT / SSMA	SSMA	
Sybase - SAP IQ	Azure SQL DB, MI, VM			

Migration phase

SOURCE	TARGET	SCHEMA	DATA (OFFLINE)	DATA (ONLINE)
SQL Server	Azure SQL DB	DMA Cloudamize*	DMS Cloudamize* Attunity* Striim*	DMS Cloudamize* Attunity* Striim*

SOURCE	TARGET	SCHEMA	DATA (OFFLINE)	DATA (ONLINE)
SQL Server	Azure SQL DB MI	DMS Cloudamize*	DMS Cloudamize*	DMS Cloudamize* Attunity* Striim*
SQL Server	Azure SQL VM	DMA DMS Cloudamize*	DMA DMS Cloudamize*	DMS Cloudamize* Attunity* Striim*
SQL Server	SQL DW			
RDS SQL	Azure SQL DB, MI, VM	DMA	DMA DMS	DMS Attunity* Striim*
Oracle	Azure SQL DB, MI, VM	SSMA SharePlex*	SSMA SharePlex*	DMS SharePlex* Attunity* Striim*
Oracle	SQL DW	SSMA	SSMA	DMS SharePlex* Attunity* Striim*
Oracle	Azure DB for PostgreSQL - Single server			DMS
MongoDB	Cosmos DB	DMS Cloudamize* Imanis Data*	DMS Cloudamize* Imanis Data*	DMS Cloudamize* Imanis Data* Striim*
Cassandra	Cosmos DB	Imanis Data*	Imanis Data*	Imanis Data*
MySQL	Azure SQL DB, MI, VM	SSMA	SSMA	Attunity* Striim*
MySQL	Azure DB for MySQL	MySQL dump*		DMS Attunity* Striim*
RDS MySQL	Azure DB for MySQL	MySQL dump*		DMS Attunity* Striim*
PostgreSQL	Azure DB for PostgreSQL - Single server	PG dump*		DMS Attunity* Striim*
RDS PostgreSQL	Azure DB for PostgreSQL - Single server	PG dump*		DMS Attunity* Striim*

SOURCE	TARGET	SCHEMA	DATA (OFFLINE)	DATA (ONLINE)
DB2	Azure SQL DB, MI, VM	SSMA	SSMA	Attunity* Striim*
Access	Azure SQL DB, MI, VM	SSMA	SSMA	SSMA
Sybase - SAP ASE	Azure SQL DB, MI, VM	SSMA	SSMA	Attunity* Striim*
Sybase - SAP IQ	Azure SQL DB, MI, VM	Ispirer*	Ispirer*	

Post-migration phase

SOURCE	TARGET	OPTIMIZE
SQL Server	Azure SQL DB	Cloud Atlas* Cloudamize*
SQL Server	Azure SQL DB MI	Cloud Atlas* Cloudamize*
SQL Server	Azure SQL VM	Cloud Atlas* Cloudamize*
SQL Server	SQL DW	
RDS SQL	Azure SQL DB, MI, VM	
Oracle	Azure SQL DB, MI, VM	
Oracle	SQL DW	
Oracle	Azure DB for PostgreSQL - Single server	
MongoDB	Cosmos DB	Cloudamize*
Cassandra	Cosmos DB	
MySQL	Azure SQL DB, MI, VM	
MySQL	Azure DB for MySQL	
RDS MySQL	Azure DB for MySQL	
PostgreSQL	Azure DB for PostgreSQL - Single server	

SOURCE	TARGET	OPTIMIZE
RDS PostgreSQL	Azure DB for PostgreSQL - Single server	
DB2	Azure SQL DB, MI, VM	
Access	Azure SQL DB, MI, VM	
Sybase - SAP ASE	Azure SQL DB, MI, VM	
Sybase - SAP IQ	Azure SQL DB, MI, VM	

Next steps

For an overview of the Azure Database Migration Service, see the article [What is the Azure Database Migration Service](#).

FAQ about using Azure Database Migration Service

2/26/2020 • 7 minutes to read • [Edit Online](#)

This article lists commonly asked questions about using Azure Database Migration Service together with related answers.

Overview

Q. What is Azure Database Migration Service? Azure Database Migration Service is a fully managed service designed to enable seamless migrations from multiple database sources to Azure Data platforms with minimal downtime. The service is currently in General Availability, with ongoing development efforts focused on:

- Reliability and performance.
- Iterative addition of source-target pairs.
- Continued investment in friction-free migrations.

Q. What source/target pairs does Azure Database Migration Service currently support? The service currently supports a variety of source/target pairs, or migration scenarios. For a complete listing of the status of each available migration scenario, see the article [Status of migration scenarios supported by the Azure Database Migration Service](#).

Other migration scenarios are in preview and require submitting a nomination via the DMS Preview site. For a complete listing of the scenarios in preview and to sign up to participate in one of these offerings, see the [DMS Preview site](#).

Q. What versions of SQL Server does Azure Database Migration Service support as a source? When migrating from SQL Server, supported sources for Azure Database Migration Service are SQL Server 2005 through SQL Server 2019.

Q: When using Azure Database Migration Service, what's the difference between an offline and an online migration? You can use Azure Database Migration Service to perform offline and online migrations. With an *offline* migration, application downtime starts when the migration starts. With an *online* migration, downtime is limited to the time to cut over at the end of migration. We suggest that you test an offline migration to determine whether the downtime is acceptable; if not, do an online migration.

NOTE

Using Azure Database Migration Service to perform an online migration requires creating an instance based on the Premium pricing tier. For more information, see the Azure Database Migration Service [pricing](#) page.

Q. How does Azure Database Migration Service compare to other Microsoft database migration tools such as the Database Migration Assistant (DMA) or SQL Server Migration Assistant (SSMA)? Azure Database Migration Service is the preferred method for database migration to Microsoft Azure at scale. For more detail on how Azure Database Migration Service compares to other Microsoft database migration tools and for recommendations on using the service for various scenarios, see the blog posting [Differentiating Microsoft's Database Migration Tools and Services](#).

Q. How does Azure Database Migration Service compare to the Azure Migrate offering? Azure Migrate assists with migration of on-premises virtual machines to Azure IaaS. The service assesses migration suitability and performance-based sizing, and it provides cost estimates for running your on-premises virtual machines in Azure. Azure Migrate is useful for lift-and-shift migrations of on-premises VM-based workloads to Azure IaaS.

VMs. However, unlike Azure Database Migration Service, Azure Migrate isn't a specialized database migration service offering for Azure PaaS relational database platforms such as Azure SQL Database or Azure SQL Database Managed Instance.

Setup

Q. What are the prerequisites for using Azure Database Migration Service? There are several prerequisites required to ensure that Azure Database Migration Service runs smoothly when performing database migrations. Some of the prerequisites apply across all scenarios (source-target pairs) supported by the service, while other prerequisites are unique to a specific scenario.

Azure Database Migration Service prerequisites that are common across all supported migration scenarios include the need to:

- Create a Microsoft Azure Virtual Network for Azure Database Migration Service by using the Azure Resource Manager deployment model, which provides site-to-site connectivity to your on-premises source servers by using either [ExpressRoute](#) or [VPN](#).
- Ensure that your virtual network Network Security Group rules don't block the following communication ports 443, 53, 9354, 445, 12000. For more detail on virtual network NSG traffic filtering, see the article [Filter network traffic with network security groups](#).
- When using a firewall appliance in front of your source database(s), you may need to add firewall rules to allow Azure Database Migration Service to access the source database(s) for migration.

For a list of all the prerequisites required to complete specific migration scenarios using Azure Database Migration Service, see the related tutorials in the Azure Database Migration Service [documentation](#) on docs.microsoft.com.

Q. How do I find the IP address for Azure Database Migration Service so that I can create an allow list for the firewall rules used to access my source database for migration? You may need to add firewall rules allowing Azure Database Migration Service to access to your source database for migration. The IP address for the service is dynamic, but if you're using ExpressRoute, this address is privately assigned by your corporate network. The easiest way to identify the appropriate IP address is to look in the same resource group as your provisioned Azure Database Migration Service resource to find the associated Network Interface. Usually the name of the Network Interface resource begins with the NIC prefix and followed by a unique character and number sequence, for example NIC-jj6tnztnmarpsskr82rbndyp. By selecting this network interface resource, you can see the IP address that needs to be included in the allow list on the resource overview Azure portal page.

You may also need to include the port source that SQL Server is listening on the allow list. By default, it's port 1433, but the source SQL Server may be configured to listen on other ports as well. In this case, you need to include those ports on the allow list as well. You can determine the port that SQL Server is listening on by using a Dynamic Management View query:

```
SELECT DISTINCT
    local_tcp_port
FROM sys.dm_exec_connections
WHERE local_tcp_port IS NOT NULL
```

You can also determine the port that SQL Server is listening by querying the SQL Server error log:

```
USE master
GO
xp_readerrorlog 0, 1, N'Server is listening on'
GO
```

Q. How do I set up a Microsoft Azure Virtual Network? While multiple Microsoft tutorials that can walk you through the process of setting up a virtual network, the official documentation appears in the article [Azure Virtual](#)

Network.

Usage

Q. What is a summary of the steps required to use Azure Database Migration Service to perform a database migration? During a typical, simple database migration, you:

1. Create a target database(s).
2. Assess your source database(s).
 - For homogenous migrations, assess your existing database(s) by using [DMA](#).
 - For heterogeneous migrations (from compete sources), assess your existing database(s) with [SSMA](#). You also use SSMA to convert database objects and migrate the schema to your target platform.
3. Create an instance of Azure Database Migration Service.
4. Create a migration project specifying the source database(s), target database(s), and the tables to migrate.
5. Start the full load.
6. Pick the subsequent validation.
7. Perform a manual switchover of your production environment to the new cloud-based database.

Troubleshooting and optimization

Q. I'm setting up a migration project in DMS, and I'm having difficulty connecting to my source

database. What should I do? If you have trouble connecting to your source database system while working on migration, create a virtual machine in the virtual network with which you set up your DMS instance. In the virtual machine, you should be able to run a connect test, such as using a UDL file to test a connection to SQL Server or downloading Robo 3T to test MongoDB connections. If the connection test succeeds, you shouldn't have an issue with connecting to your source database. If the connection test doesn't succeed, contact your network administrator.

Q. Why is my Azure Database Migration Service unavailable or stopped? If the user explicitly stops Azure Database Migration Service (DMS) or if the service is inactive for a period of 24 hours, the service will be in a stopped or auto paused state. In each case, the service will be unavailable and in a stopped status. To resume active migrations, restart the service.

Q. Are there any recommendations for optimizing the performance of Azure Database Migration

Service? You can do a few things to speed up your database migration using the service:

- Use the multi CPU General Purpose Pricing Tier when you create your service instance to allow the service to take advantage of multiple vCPUs for parallelization and faster data transfer.
- Temporarily scale up your Azure SQL Database target instance to the Premium tier SKU during the data migration operation to minimize Azure SQL Database throttling that may impact data transfer activities when using lower-level SKUs.

Next steps

For an overview of the Azure Database Migration Service and regional availability, see the article [What is the Azure Database Migration Service](#).