Migration Using SQL Managed Instance

**Introduction**

Azure SQL Database Managed Instance is a new deployment model of Azure SQL Database, providing near 100% compatibility with the latest SQL Server on-premises (Enterprise Edition) Database Engine. Managed Instance allows existing SQL Server customers to lift and shift their on-premises applications to the cloud with minimal application and database changes. At the same time, Managed Instance preserves all PaaS capabilities (automatic patching and version updates, automated backups, high-availability ), that drastically reduces management overhead and TCO. This lab walks you through the steps to create a Managed Instance.

### For more information: <https://docs.microsoft.com/en-us/azure/sql-database/sql-database-managed-instance-get-started> Log into the Azure Portal

1. In the **Email, phone, or Skype** box, type **<inject key="AzureAdUserEmail" />** and click **Next**
2. In the **Password** box, type **<inject key="AzureAdUserPassword" />** and click **Sign in**
3. In the **Stay signed in?** pop-up window, click **No**
4. In the **Welcome to Microsoft Azure** pop-up window, click **Maybe Later**

**Note:** If you receive a pop-up for Azure Advisor, click the **X** in the top right corner of the pop-up to close it

**Database Migration Service**

The 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. This service streamlines the tasks required to move existing third party and SQL Server databases to Azure. Deployment options include Azure SQL Database, Managed Instance, and SQL Server in Azure VM at Public Preview.

### Create an Instance of Azure DMS - (some) prep steps

1. In the menu on the left side of the Azure portal, click **All services**
2. On the **All services** blade under **GENERAL**, click **Subscriptions**
3. On the **Subscriptions** blade, click the **<inject key="subscriptionName" copy="false" />**subscription
4. On the **<inject key="subscriptionName" copy="false" />** blade in the left pane under **Settings**, click **Resource providers**
5. In the **Search** box in the right pane, enter <copy>**migration**</copy>
6. To the right of **Microsoft.DataMigration** click **Re-register**

**Note:** Due to security requirements in the Hands-On Labs environment, the HOL lab user does not have sufficient permissions to create Azure DMS. The Microsoft.DataMigration Resource provider has already been registered. If the Resource provider had not been previously registered, the option would have been **Register** instead of **Re-register** and **Unregister**.

### Create an Instance of Azure Data Migration Services

1. In the menu on the left side of the Azure portal, click **Create a resource**
2. On the **New** blade in the **Search** box, enter <copy>**azure database migration service**</copy> and press **Enter**
3. On the **Everything** blade in the results list, click **Azure Database Migration Service**
4. On the **Azure Database Migration Service** blade, click **Create**
5. On the **Create Migration Service** blade under **Service Name**, enter **<inject key="services\_DMS\_serverice\_name"/>**
6. In the **Subscription** dropdown menu, select **<inject key="subscriptionName" copy="false" />**
7. Under **Resource group**, select **<inject key="resourceGroupName" copy="false" />** from the dropdown
8. In the **Location** dropdown menu, select **South Central US**

**Note:** Due to security requirements in the Hands-On Labs environment, The **Migration Service** has already been created for the **Hands-On Labs** environment

1. Click the **X** near the top right corner of the **Create Migration Service** blade to close the form
2. In the popup window that appears, click **OK** to discard your unsaved edits

**Note:** You may need 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. Lastly, 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**.

### Create Storage SAS URI for DMS Migration Project

1. In the **Favorites** menu, click **All resources**
2. Click the storage account **sa<inject key="resourceGroupName" copy="false" />**
3. On the **sa<inject key="resourceGroupName" copy="false" />** blade, click **Storage Explorer (preview)**
4. On the **sa<inject key="resourceGroupName" copy="false" /> - Storage Explorer (preview)** blade flyout, expand **BLOB CONTAINERS**

**Note:** The Blob container **backup** has already been created for you.

**Note:** When creating a blob container in this situation, retrieve it's **SAS URI** by using the steps in the article [Manage Azure Blob Storage resources with Storage Explorer](https://docs.microsoft.com/en-us/azure/vs-azure-tools-storage-explorer-blobs#get-the-sas-for-a-blob-container) , and be sure to select all permissions (Read, Write, Delete, List) on the policy window while creating the **SAS URI**. This detail provides the Azure DMS with access to your storage account container for uploading the **backup files** used for migrating databases to Azure SQL Database Managed Instance.

1. Right click the blob container **backup**, select **Get Shared Access Signature**
2. On the **Shared Access Signature** blade flyout, under permissions select **Read, Write, Delete, and List** permissions
3. Click **Create**
4. Copy and paste the SAS **URL:** into notepad on your computer. You will need this later in the lab.

**Note:** This SAS URI will be used later in the lab for the SQL Data Migration Project

### Create and Run a Migration Project

1. In the **Favorites** menu, click **All resources**
2. On the **All resources** blade, click **<inject key="services\_DMS\_serverice\_name" copy="false" />**
3. On the **<inject key="services\_DMS\_serverice\_name" copy="false" />** blade, click **New Migration Project**
4. On the **New migration project** blade under **Project name**, enter <copy>**My-DMS-migration**</copy>
5. In the **Source server type** dropdown menu, select **SQL Server**
6. In the **Target server type** dropdown menu, select **Azure SQL Database managed Instance**
7. In the **Choose type of activity**, leave the settings of **Offline data migration**
8. Click **Create and run activity**
9. On the **Migration source detail** blade under **Source SQL Server instance name**, enter <copy>**<inject key="sqlvmName" copy="false" />.southcentralus.cloudapp.azure.com**</copy>.
10. In the **Authentication type** dropdown menu, select **SQL Authentication**

**Note:** Managed Instances support Authentication and Authorization through integrating Azure Active Directory. **Authentication** supports; SQL Authentication, which uses a username and password; Azure Active Directory Authentication, which uses identities managed by Azure Active Directory and is supported for managed and integrated domains, and more. **Authorization** supports the same capabilities as SQL Server 2017.

1. In the **User Name** box, enter <copy>**LabUser<inject key="resourceGroupName" copy="false" /></copy>**
2. In the **Password** box, enter **<inject key="AzureAdUserPassword" />**
3. Check the boxes next to **Encrypt connection** and **Trust server certificate**

**Note:** 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.**

**Note:** Azure SQL Database Managed Instance combines advanced security features provided by Azure cloud and SQL Server Database Engine. Managed Instance provide **additional** security isolation from other tenants in the Azure cloud. Security isolation includes:

* [**Native virtual network implementation**](https://docs.microsoft.com/en-us/azure/sql-database/sql-database-managed-instance-vnet-configuration)  and connectivity to your on-premises environment using Azure Express Route or VPN Gateway.
* SQL endpoint is exposed only through a private IP address, allowing safe connectivity from private Azure or hybrid networks.
* Single-tenant with dedicated underlying infrastructure (compute, storage).

**In addition, SQL Database Managed Instance provides a set of advanced security features that can be used to protect your data:**

* [**Managed Instance auditing**](https://docs.microsoft.com/en-us/azure/sql-database/sql-database-managed-instance-auditing)  tracks database events and writes them to an audit log file placed in your Azure storage account. Auditing can help maintain regulatory compliance, understand database activity, and gain insight into discrepancies and anomalies that could indicate business concerns or suspected security violations.
* **Data encryption in motion** - Managed Instance secures your data by providing encryption for data in motion using Transport Layer Security.
* [**Threat Detection**](https://docs.microsoft.com/en-us/azure/sql-database/sql-database-managed-instance-threat-detection)  complements [Managed Instance auditing](https://docs.microsoft.com/en-us/azure/sql-database/sql-database-managed-instance-auditing)  by providing an additional layer of security intelligence built into the service that detects unusual and potentially harmful attempts to access or exploit databases.
* [**Dynamic data masking**](https://docs.microsoft.com/en-us/sql/relational-databases/security/dynamic-data-masking?view=sql-server-2017)  limits sensitive data exposure by masking it to non-privileged users.
* [**Row-level security**](https://docs.microsoft.com/en-us/sql/relational-databases/security/row-level-security?view=sql-server-2017)  enables you to control access to rows in a database table based on the characteristics of the user executing a query (such as by group membership or execution context).
* [**Transparent data encryption (TDE)**](https://docs.microsoft.com/en-us/azure/sql-database/transparent-data-encryption-azure-sql?view=sql-server-2017)  encrypts Azure SQL Managed Instance data files, known as encrypting data at rest. TDE performs real-time I/O encryption and decryption of the data and log files. The encryption uses a database encryption key (DEK), which is stored in the database boot record for availability during recovery. You can protect all your databases in Managed Instance with transparent data encryption.

1. Click **Save**.
2. Under **Target server name**, enter <copy>[**sqlmi-shared-01.2ad734213499.database.windows.net**](http://sqlmi-shared-01.2ad734213499.database.windows.net/)</copy>
3. In the **Authentication type** dropdown menu, select **SQL Authentication**
4. In the **User Name** box, enter <copy>**LabUser<inject key="resourceGroupName" copy="false" /></copy>**
5. In the **Password** box, enter **<inject key="AzureAdUserPassword" />**
6. Click **Save**
7. On the **Select source databases** blade, select **<inject key="resourceGroupName" copy="false" />-adventure-works**
8. Click **Save**
9. On the **Select logins** blade, select none of the users
10. Click **Save**
11. On the **Configure migration settings** blade dropdown menu, select **I will let Azure Database Migration Service create backup files**

##### On the **Configure migration settings** blade, **Backup settings**

1. Network share location, enter <copy>**\\sql<inject key="resourceGroupName" copy="false" />\Backup</copy>**
2. Windows User\Domain, enter <copy>**localhost\LabUser</copy>**
3. Password, enter **<inject key="AzureAdUserPassword" />**

##### On the **Configure migration settings** blade, **Storage account settings**

1. SAS URI, This is the SAS URI you created earlier in the lab. Open Notepad and copy it

**Note:** This is the SAS URI you created earlier in the lab

1. Click **Save**
2. On the **Migration summary** blade under **Activity name**, enter <copy>**My-SQL-Migration-Activity**</copy>
3. On the **Migration summary** blade, select **Validation option**
4. On the **Choose validation option** blade, select **Do not validate my database(s)** and click **Save**
5. Click **Run migration**
6. On the **My-SQL-Migration-Activity** blade, you can monitor the status of the migration
7. Click **Refresh**
8. Click the **Databases** to see the detailed migration status.
9. Click **Refresh** until the **STATUS** of the migration is **Completed**

### Navigate the SQL Managed Instance Database

1. In the **Favorites** menu, click **All resources**
2. On the **All resources** blade, click **sqlmi-shared-01** SQL managed instace
3. On the **sqlmi-shared-01** blade, you can see a database named **<inject key="resourceGroupName" copy="false" />-adventure-works**, here you can manage your newly migrated database

# Conclusion

You have now migrated the data of the on-premises legacy SQL Server instance to an Azure SQL Managed Instance Database.