**Managing Azure with Windows PowerShell**

<https://learn.microsoft.com/en-us/powershell/azure/what-is-azure-powershell?view=azps-11.3.0>

<https://learn.microsoft.com/en-us/powershell/azure/?view=azps-11.3.0>

**Overview of Azure Resource Manager**

Azure Resource Manager is the deployment and management service for Azure. It provides a management layer that enables you to create, update, and delete resources in your Azure account. You use management features, like access control, locks, and tags, to secure and organize your resources after deployment.

**What are the advantages of the Azure Resource Manager?**

Azure Resource Manager enables users to manage their usage of application resources. Few of the advantages of Azure Resource Manager are:

ARM helps deploy, manage and monitor all the resources for an application, a solution or a group

Users can be granted access to resources they require

It obtains comprehensive billing information for all the resources in the group

Provisioning resources is made much easier with the help of templates

<https://learn.microsoft.com/en-us/azure/azure-resource-manager/management/overview>

**Azure management services**

[**https://learn.microsoft.com/en-us/azure/governance/management-groups/azure-management**](https://learn.microsoft.com/en-us/azure/governance/management-groups/azure-management)

**What are the Various Azure Services and How does Azure Work?1. Azure Database service**

Azure provides more than 200 services, are divided into 18 categories. These categories include computing, networking, storage, IoT, migration, mobile, analytics, containers, artificial intelligence, and other machine learning, integration, management tools, developer tools, security, databases, DevOps, media identity, and web services. Let’s take a look at some of the major Azure services by category:

**Compute Services**

Virtual Machine

This service enables you to create a virtual machine in Windows, Linux or any other configuration in seconds.

Cloud Service

This service lets you create scalable applications within the cloud. Once the application is deployed, everything, including provisioning, load balancing, and health monitoring, is taken care of by Azure.

Service Fabric

With service fabric, the process of developing a microservice is immensely simplified. Microservice is an application that contains other bundled smaller applications.

Functions

With functions, you can create applications in any programming language. The best part about this service is that you need not worry about hardware requirements while developing applications because Azure takes care of that. All you need to do is provide the code.

The basic fundamental building block that is available in Azure is the SQL database. Microsoft offers this SQL server and SQL database on Azure in many ways. We can deploy a single database, or we can deploy multiple databases as part of a shared elastic pool.

**Networking**

Azure CDN

Azure CDN (Content Delivery Network) is for delivering content to users. It uses a high bandwidth, and content can be transferred to any person around the globe. The CDN service uses a network of servers placed strategically around the globe so that the users can access the data as soon as possible.

Express Route

This service lets you connect your on-premise network to the Microsoft cloud or any other services that you want, through a private connection. So, the only communications that will happen here will be between the enterprise network and the service that you want.

Virtual network

The virtual network allows you to have any of the Azure services communicate with one another privately and securely.

Azure DNS

This service allows you to host your DNS domains or system domains on Azure.

**Storage**

Disk Storage

This service allows you to choose from either HDD (Hard Disk Drive) or SSD (Solid State Drive) as your storage option along with your virtual machine.

Blob Storage

This service is optimized to store a massive amount of unstructured data, including text and even binary data.

File Storage

This is a managed file storage service that can be accessed via industry SMB (server message block) protocol.

Queue Storage

With queue storage, you can provide stable message queuing for a large workload. This service can be accessed from anywhere in this world.

**Azure Database Service Architecture**

Microsoft introduced a managed instance that is targeted towards on-premises customers. So, if we have some SQL databases within our on-premises datacentre and we want to migrate that database into Azure without any complex configuration, or ambiguity, then we can use managed instance. Because this is mainly targeted towards on-premises customers who want to lift and share their on-premises database into Azure with the least effort and optimized cost. We can also take advantage of licensing we have within our on-premises data center.



**Azure Database service**

Microsoft will be responsible for maintenance patching and related services. But, in case if we want to go for the IaaS service for the SQL server, then we can deploy SQL Server on the Azure Virtual machine. If the data have a dependency on the underlying platform and we want to log into the SQL Server, in that case, we can use the SQL server on a virtual machine.

We can deploy a SQL data warehouse on the cloud. Azure offers many other database services for different types of databases such as MySQL, Maria DB, and also PostgreSQL. Once we deployed a database into Azure, we need to migrate the data into it or replicate the data into it.

**Azure Database Services for Data Migration**

The services that are available in Azure, which we can use to migrate the data from our on-premises SQL Server into Azure.

**Azure Data Migration Service:** It is used to migrate the data from our existing SQL server and database within the on-premises data center into Azure.

**Azure SQL data sync:** If we want to replicate the data from our on-premises database into Azure, then we can use Azure SQL data sync.

**SQL Stretch Database:** It is used to migrate cold data into Azure. SQL stretch database is a bit different from other database offerings. It works as a hybrid database because it divides the data into different types - hot and cold. A hot data will be kept in the on-premises data center and cold data in the Azure.

**Data Factory**

It is used for ETL transformation, extraction loading, etc. Using the data factory, we can even extract the data from our on-premises data center. We can do some conversion and load it into the Azure SQL database. Data Factory is an ETL tool that is offered on the cloud, which we can use to connect to different databases, extract the data, transform it, and load into a destination.

**Security**

All the databases that are existing in Azure need to be secured, and also we need to accept connections from known origins. For this purpose, all these database services come with firewall rules where we can configure from which particular IP address we want to allow connections. We can define those firewall rules to limit the number of connections and also reduce the surface attack area.

**Cosmos DB**

Cosmos DB is a NoSQL data store that is available in Azure, and it is designed to be globally scalable and also very highly available with extremely low latency. Microsoft guarantees latency in terms of reading and writes with Cosmos DB. For example - if we have any applications such as IoT, gaming where we get a lot of data from different users spread across globally, then we will go for Cosmos DB. Because Cosmos DB is designed to be globally scalable and highly available due to which our users will experience low latency.

Finally, there are two things, and one is we need to secure all the services. For that purpose, we can integrate all these services with Azure Active Directory and manage the users from Azure Active Directory also. To monitor all these services, we can use the security center. There is an individual monitoring tool too, but Azure security center will keep on monitoring all these services and provide recommendations if something is wrong.

**Azure SQL Database**

SQL database is the flagship product of Microsoft in the database area. It is a general-purpose relational database that supports structures like relation data - JSON, spatial, and XML. The Azure platform fully manages every Azure SQL Database and guarantees no data loss and a high percentage of data availability. Azure automatically handles patching, backups, replication, failure detection, underlying potential hardware, software or network failure, deploying bug fixes, failovers, database upgrades, and other maintenance tasks.



**Azure SQL Database**

There are three ways we can implement our SQL database

**Managed Instance:** This is primarily targeted towards on-premises customers. In case, if we already have a SQL server instance in our on-premises data-center and you want to migrate that into Azure with minimum changes to our application and the maximum compatibility. Then new will go for the managed instance.

**Single database**: We can deploy a single database on Azure its own set of resources managed via a logical server.

**Elastic pool**: We can deploy a pool of databases with a shared set of resources managed via a logical server.

We can deploy the SQL database as an infrastructure as a service. That means we want to use the SQL server on an Azure virtual machine, but in that case, we are responsible for managing the SQL server on that particular Azure virtual machine.

**Purchasing model**

There are two ways we can purchase the SQL Server on Azure.

**VCore purchasing model:** The vCore-based purchasing model enables us to independently scale compute and storage resources, match on-premises performance, and optimize price. It also allows us to choose a generation of hardware. It also allows us to use Azure Hybrid Benefit for SQL Server to gain cost savings. Best for the customer who values flexibility, control, and transparency.

**DTU model:** It is based on a bundled measure on compute, storage, and IO resources. Sizes of the compute are expressed in terms of Database Transaction Units (DTUs) for single databases and elastic Database Transaction Units (eDTUs) for elastic pools. This model is best for customers who want simple, pre-configured resource options.

**Azure SQL Database service tiers**

**General Purpose/ Standard model:** It is based on a separation of computing and storage service. This architectural model depends on the high availability and reliability of Azure Premium Storage that transparently copies database files and guarantees for zero data loss if underlying infrastructure failure happens.

**Business Critical/ Premium service tier model:** It is based on a cluster of database engine processes. Both the SQL database engine process and underlying mdf/ldf files are placed on the same node with locally attached SSD storage providing low latency to our workload. High availability is implemented using technology similar to SQL Server Always On Availability Groups.

**Hyperscale service tier model:** It is the newest service tier in the vCore-based purchasing model. This tier is a highly scalable storage and computes performance tier that leverages the Azure architecture to scale-out the storage and computes resources for an Azure SQL Database beyond the limits available for the General Purpose and Business Critical service tiers.

**SQL database logical server**

It acts as a central administrative point for multiple single or pooled database logins, firewall rules, auditing rules, threat detection policies, and failover groups.

It must exist before we can create the Azure SQL database. All databases on a server are created within the same region as the logical server.

The SQL database service makes no guarantees regarding the location of the database in relation to their logical servers and exposes no instance-level access or features.

An Azure database logical server is the parent resource for databases, elastic pools, and data warehouses.

**Elastic pools**

It is a simple and cost-effective solution for scaling and managing more than one database. The databases inside an elastic pool are on a single Azure SQL Database server and share a group of resources at a fixed price.

We can configure resources for the pool based either on the DTU- based purchasing model or the vCore-based purchasing model.

The size of a pool always depends on the aggregate resource needed for all databases in the pool. It determines the following options:

The maximum resources utilized in the pool by the databases.

The maximum storage bytes utilized in the pool by the databases.

* **Introduction to Azure Data Factory**

**What Is Azure Data Factory?**

Azure Data Factory is a cloud-based integration service that orchestrates and automates the movement and transformation of data. It works heavily on the data that you store.

Let us discuss the process followed in the Azure Data Factory**.**



**Input Datasets**

This represents the collection of data within the data stores. The data passes through a pipeline for processing.

**Pipeline**

A pipeline consists of a group of activities, such as:

**Data movement activity**

Data transformation activity using:

SQL

Stored procedures

Hive

Output Datasets

After the data is transformed into the pipeline, we get an output dataset. Here, we get a structured form of data.

**Linked Services**

The data from output datasets passes to linked services, such as:

* **Azure Data Lake**

**Block storage**

**SQL**

Linked services contain information needed to connect to external sources. This is similar to the concept of a connection string in an SQL Server, where you define the source and destination of your data.

**Gateway**

This connects your on-premises data to the cloud. It consists of a client agent that is installed on the on-premises data system, which then connects to the Azure data.

**Cloud**

The data is analyzed and visualized using a number of analytical frameworks, like Apache Spark, R, Hadoop, and so on.

**What Is Azure Data Lake?**

Azure Data Lake is a highly scalable, distributed, parallel file system in the cloud that is specifically designed to work with multiple analytics frameworks.

The data in output datasets (collected from mobile, the web, social platforms, etc.) is sent into the Azure Data Lake Store. It is then provided to external frameworks, like R and Apache Spark.

Data Lake works on two main concepts: storage and analytics.

**Storage**

Storage is unlimited, allowing users to save very large files. A variety of data (like unstructured or structured data) can be stored here.

**Analytics**

Through analytics, you can monitor and diagnose real-time data from connected devices, such as vehicles, buildings, or machinery to initiate actions such as generating alerts, responding to events, and optimizing operations.

You can also monitor financials such as:

Financial transactions in real-time to detect fraudulent activity

The use of a credit card across geographic locations

The number of transactions on a single credit card

**What is ETL and when it's required ?**

[**https://www.geeksforgeeks.org/etl-process-in-data-warehouse/**](https://www.geeksforgeeks.org/etl-process-in-data-warehouse/)

**What is Data warehouse?**

[**https://www.beyondkey.com/blog/what-is-data-warehouse/**](https://www.beyondkey.com/blog/what-is-data-warehouse/)

**Planning and implementing Azure Non-Relational Databases**

**• Planning and deploying Cosmos DB**

[**https://learn.microsoft.com/en-us/azure/cosmos-db/nosql/quickstart-portal**](https://learn.microsoft.com/en-us/azure/cosmos-db/nosql/quickstart-portal)

**Implementing and managing Cosmos DB**



Azure Cosmos DB is a NoSQL data store. It is different from the traditional relational database where we have a table, and the table will have a fixed number of columns, and each row in the table should adhere to the scheme of the table. In the NoSQL database, you don't define any schema at all for the table, and each item or row within the table can have different values, or different schema itself.

**Advantages of Cosmos DB**

No Schema & Index management: The Azure database engine is fully schema-agnostic. Therefore no schema and index management are required. We also don't have to worry about application downtime while migrating schemas.

Industry-leading comprehensive SLAs: Cosmos DB is the first and only service to offer industry-leading full 99.99% high availability, read and write latency at the 99th percentile, guaranteed throughput, and consistency.

The low total cost of Ownership: Since Cosmos DB is a fully managed service, we no longer need to manage and operate complex multi-datacenter deployment, and upgrades of our database software pay for the support, licensing, or operations.

Developing application using NoSQL APIs: Cosmos DB implements Cassandra, MongoDB, Gremlin, and Azure Table Storage wire protocol directly on the service.

Global distribution: Cosmos DB allows us to add or remove any of the Azure regions to our Cosmos account at any time, with a click of a button.

**Cosmos Database Structure**

Database: We can create one or more Azure Cosmos database under our account. A database is analogous to a namespace, and it is the unit of management for a set of Azure Cosmos containers.



Cosmos Account: the Azure Cosmos account is the basic unit of global distribution and high availability. For globally distributing our data and throughput across multiple Azure regions, we can add or remove Azure regions from our Azure Cosmos at any time.

Container: An Azure Cosmos container is the unit of scalability for both provisioned throughput and storage of items. A container is horizontally partitioned and then replicated across multiple regions.

**Global distribution and Partitioning**

Cosmos DB works differently from the traditional relational database where we have a table, and all the rows in the table will sit in one physical place. When it comes to Cosmos DB, we will create logical partitions within a container so that we can have a certain amount of items with one partition key and a certain number of items with another partition key. They are called logical partitioning, and each logical partitioning can reside in a physical partition.



A container can contain millions of items, and we can divide these millions of items using partition key and make logical partitions, and each logical partition will reside in a physical partition. This is how the load of the container will be distributed across the board locally and also the data from here.

**Request Units**

We pay for the throughput we provision, and the storage we consume on an hourly basis with Azure Cosmos DB.

The cost of all the database operations is normalized by Azure Cosmos DB and is expressed in terms of Request Units (RUs). The price to read a 1-KB item is 1 Request Unit (1 RU). All other database operations are similarly assigned with a cost in terms of RUs.

The number of RU's consumed will depend on the type of operation, item size, data consistency, query patterns, etc.



For the management and planning of capacity, Azure Cosmos DB ensures that the number of RUs for a given database operation over a given dataset is deterministic.

**Azure Data Factory**

Azure Data Factory is a data-integration service based on the Cloud that allows us to create data-driven workflows in the cloud for orchestrating and automating data movement and data transformation. Data Factory is a perfect ETL tool on Cloud. Data Factory is designed to deliver extraction, transformation, and loading processes within the cloud. The ETL process generally involves four steps:



Connect & Collect: We can use the copy activity in a data pipeline to move data from both on-premises and cloud source data stores.

Transform: Once the data is present in a centralized data store in the cloud, process or transform the collected data by using compute services such as HDInsight Hadoop, Spark, Data Lake Analytics, and Machine Learning.

Publish: After the raw data is refined into a business-ready consumable form, it loads the data into Azure Data Warehouse, Azure SQL Database, and Azure Cosmos DB, etc.

Monitor: Azure Data Factory has built-in support for pipeline monitoring via Azure Monitor, API, PowerShell, Log Analytics, and health panels on the Azure portal.

**Components of Data Factory**

Data Factory is composed of four key elements. All these components work together to provide the platform on which you can form a data-driven **workflow with the structure to move and transform the data.**

Pipeline: A data factory can have one or more pipelines. It is a logical grouping of activities that perform a unit of work. The activities in a pipeline perform the task altogether. For example - a pipeline can contain a group of activities that ingests data from an Azure blob and then runs a Hive query on an HDInsight cluster to partition the data.

Activity: It represents a processing step in a pipeline. For example - we might use a copy activity to copy data from one data store to another data store.

Datasets: It represents data structures within the data stores, which point to or reference the data we want to use in our activities as I/O.

Linked Services: It is like connection strings, which define the connection information needed for Data Factory to connect to external resources. A Linked service can be a data store and compute resource. Linked service can be a link to a data store, or a computer resource also.

Triggers: It represents the unit of processing that determines when a pipeline execution needs to be disabled. We can also schedule these activities to be performed at some point in time, and we can use the trigger to disable an activity.

Control flow: It is an orchestration of pipeline activities that include chaining activities in a sequence, branching, defining parameters at the pipeline level, and passing arguments while invoking the pipeline on-demand or from a trigger. We can use control flow to sequence certain activities and also define what parameters need to be passed for each of the activities.