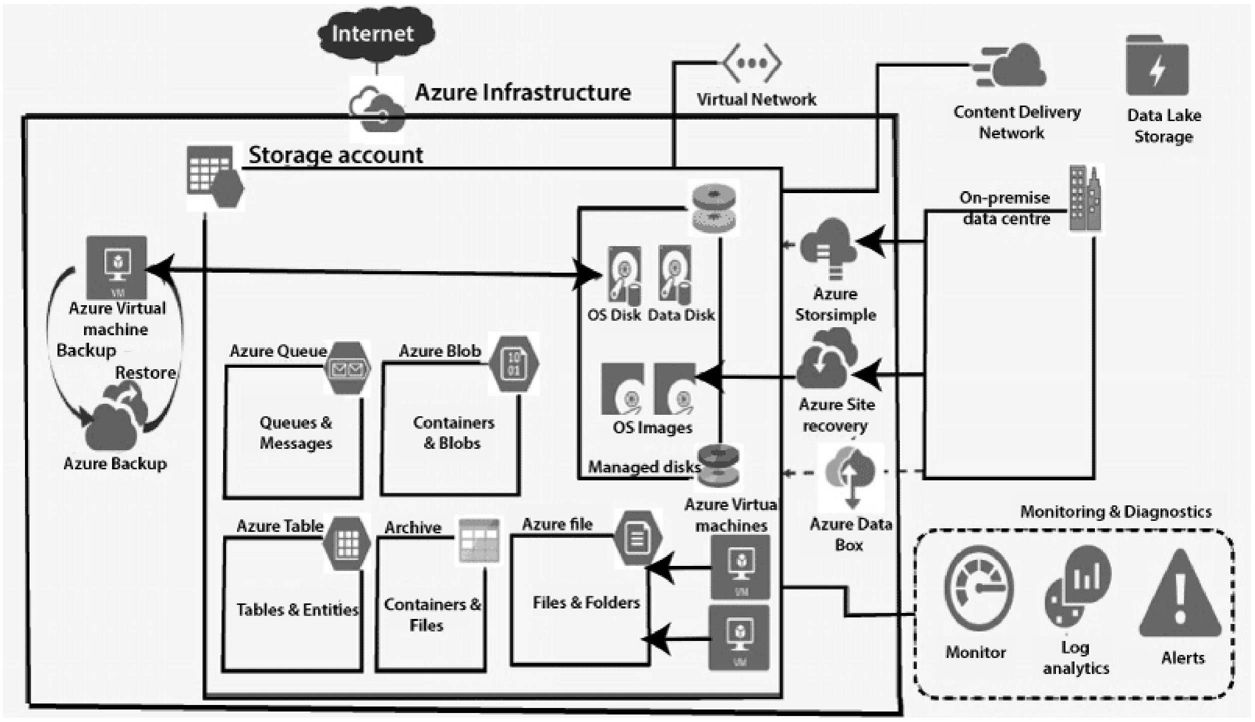
Azure Storage Building Blocks

The fundamental building block of Azure storage service is the ***Azure storage account***. The Storage account is more like an administrative container for most of the Azure storage services. All the storage services are explained below.

- **Azure Blob:** We can have *Azure Blob* storage within the storage account, which is used to store the unstructured data such as media files, documents, etc.



- **Azure file:** *Azure file* can be used in case if we want to share files between two virtual machines, then we can create an Azure file share and access it on both of the virtual machines. We can share the data between two or more VMs.
- **Archive:** *Archive* is recently introduced, and it is in preview. We can use the archive for cost optimization. So, we can move any infrequently accessed blobs or files into the archive to optimize the cost. However, once you move the data into an archive, it will take some time for the recovery of that data.
- **Azure Queues:** It can be used to store messages.

- **Azure Table:** It can be used to store entities. The *Azure Table* is a bit different from the SQL table. This is a NoSQL datastore where the schema within the table is not enforced.

And apart from all these services, there is one other key service which is:

- **Azure Disk Storage:** Any OS disk associated with the virtual machine in Azure will get stored in a disk storage account. And also, any OS image from which this OS disk is generated will get stored as a **.vhd** file within the disk storage.
- **Azure Storsimple:** In a hybrid cloud storage solution, Azure offers Storsimple. Storsimple is a hybrid storage solution that works at a SAN(Storage Area Network) level. It was used to be a separate company, but Microsoft brought it with them and is now offering the same services as a part of Azure and from DR (Disaster Recovery) perspective.
- **Azure Site Recovery:** In case if we want to use Azure as a DR data-center, then we can use Azure site recovery to replicate the workloads from our on-premises data-center into Azure. Replicated workloads will be stored as images within a storage account. Whenever our on-premises data-center is down, we can run some automated scripts which will consider that recent image and build a virtual machine.
- **Azure Data Box:** If we have terabytes of data which we want to transfer from the on-premises data center into Azure and we don't want to choose a network as an option because transferring the data over the network in terms of terabytes is not feasible. So, in that case, we can use the Azure data box. By using Azure Data Box, we can load the data into the data box and give that data box to Microsoft. Microsoft will load that data into Azure
- **Azure Backup:** We can use Azure backup to backup the disks of our virtual machine into a recovery service vault and restore the same using that image. We have to be aware that Azure backup doesn't utilize storage to store the disk image. They are stored in the ***recovery services vault***.
- **Azure Monitor:** It can be used for the monitoring of all these services. We can use *Azure Monitor* for simple monitoring, and we can use ***log analytics*** for advance monitoring and analysis. We can also use ***alerts*** in case if we want to get

alerted on certain things, for example, if the file share capacity is reaching its limit, then we configure it in such a way that we will get alert about the same.

- **CDN (Content Delivery Network):** It is used for the delivery of the contents stored in the storage account. We can use a content delivery network to reduce the latency of the delivery. We'll create a CDN endpoint near to the users to reduce the latency.

Finally, the storage account will be connected to ***Virtual Network***. The storage account will have a ***storage firewall*** where we can configure that from which virtual network you want to accept connections. So we can specify a particular IP address from where we want to allow connections or a specific subnet within a virtual network.

Azure Storage Account

An Azure Storage Account is a secure account, which provides you access to services in Azure Storage. The storage account is like an administrative container, and within that, we can have several services like blobs, files, queues, tables, disks, etc. And when we create a storage account in Azure, we will get the unique namespace for our storage resources. That unique namespace forms the part of the URL. The storage account name should be unique across all existing storage account name in Azure.

Types of Storage Accounts

Storage account type	Supported services	Supported performance tiers	Supported access tiers	Replication options	Deployment model	Encryption
General-purpose V2	Blob, File, Queue, Table, and Disk	Standard, Premium	Hot, Cool, Archive	LRS, ZRS, GRS, RA-GRS	Resource Manager	Encrypted
General-purpose V1	Blob, File, Queue, Table, and Disk	Standard, Premium	N/A	LRS, GRS, RA-GRS	Resource Manager, Classic	Encrypted

Blob storage	Blob (block blobs and append blobs only)	Standard	Hot, Cool, Archive	LRS, GRS, RA-GRS	Resource Manager	Encrypted

Types of performance tiers

Standard performance: This tier is backed by magnetic drives and provides low cost/GB. They are best for applications that are best for bulk storage or infrequently accessed data.

Premium storage performance: This tier is backed by solid-state drives and offers consistency and low latency performance. They can only be used with Azure virtual machine disks, and are best for I/O intensive workload such as the database.

(So every virtual machine disk will be stored on a storage account. So, if we are associating a disk, then we will go for the premium storage. But if we are using storage account specifically to store blobs, then we will go for standard performance.)

Access Tiers

There are four types of access tiers available:

Premium Storage (preview): It provides high-performance hardware for data that is accessed frequently.

Hot storage: It is optimized for storing data that is accessed frequently.

Cool Storage: It is optimized for storing data that is infrequently accessed and stored for at least 30 days.

Archive Storage: It is optimized for storing files that are rarely accessed and stored for a minimum of 180 days with flexible latency needs (on the order of hours).

Advantage of Access Tiers:

When a user uploads the document into the storage, the document will initially be frequently accessed. During that time, we put the document in the *hot Storage tier.*

But after some time, once the work on the document is completed. Nobody generally accesses it. So it will become infrequently accessed document. Then we can move the document from *Hot storage* to *Cool storage* to save the cost because cool storage is built based on the number of times the document is accessed. Once the document is matured, i.e., once we stopped working on that document, the document becomes old. We rarely refer to that document. In that case, we put it in cool storage.

But for six months or one year, we don't want the document to be referred to in the future. In that case, we will move that document to archive storage.

So hot storage is costlier than cool storage in terms of storage. But cool storage is more expensive in terms of access. Archive storage is used for archiving the documents into storage, which is not accessed

Azure Storage Replication

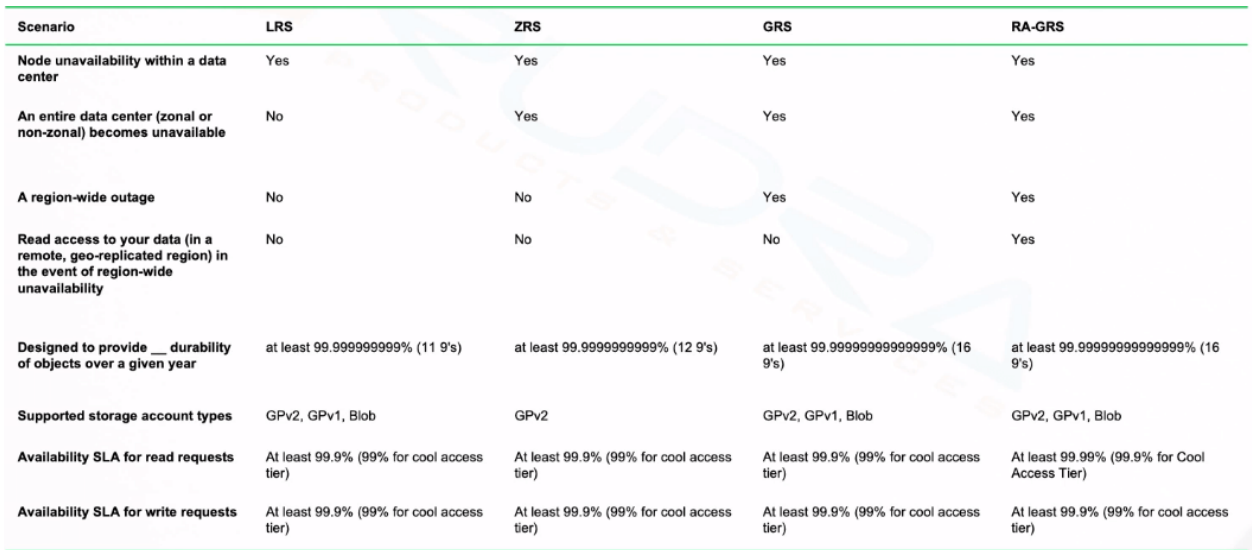
Azure Storage Replication is used for the durability of the data. It copies our data to stay protected from planned and unplanned events, ranging from transient hardware failure, network or power outages, and massive natural disasters to man-made vulnerabilities.

Azure creates some copies of our data and stores it at different places. Based on the replication strategy.

LRS (Local Redundant Storage): So, if we go with the local-redundant storage, the data will be stored within the data center. If the data center or the region goes down, the data will be lost.

ZRS (Zone-Redundant Storage): The data will be replicated across data centers but within the region. In that case, the data is always available within the data center, even if one node is not available. OR we can say that the data will be available also if the entire data center goes down because the data is already copied in another data center within the region. However, if the region itself is gone, then you will not get the data access.

GRS (global-redundant storage): To protect our data against region-wide failures. We can go for global-redundant storage. In this case, the data will be replicated in the paired region within the geography. And in case if we want to have read-only access to the data that is copied to another region, then, in that case, we can go for ***RA-GRS (Read Access global-redundant storage)***. We can get different things in terms of durability, as we can see in this table below.



Scenario	LRS	ZRS	GRS	RA-GRS
Node unavailability within a data center	Yes	Yes	Yes	Yes
An entire data center (zonal or non-zonal) becomes unavailable	No	Yes	Yes	Yes
A region-wide outage	No	No	Yes	Yes
Read access to your data (in a remote, geo-replicated region) in the event of region-wide unavailability	No	No	No	Yes
Designed to provide __ durability of objects over a given year	at least 99.999999999% (11 9's)	at least 99.9999999999% (12 9's)	at least 99.99999999999999% (16 9's)	at least 99.99999999999999% (16 9's)
Supported storage account types	GPv2, GPv1, Blob	GPv2	GPv2, GPv1, Blob	GPv2, GPv1, Blob
Availability SLA for read requests	At least 99.9% (99% for cool access tier)	At least 99.9% (99% for cool access tier)	At least 99.9% (99% for cool access tier)	At least 99.99% (99.9% for Cool Access Tier)
Availability SLA for write requests	At least 99.9% (99% for cool access tier)	At least 99.9% (99% for cool access tier)	At least 99.9% (99% for cool access tier)	At least 99.9% (99% for cool access tier)

Storage account endpoints

Whenever we create a storage account, we will get an endpoint to access the data within the storage account. So each object that we stored in Azurestorage has an address, which includes your unique account name and the combination of an account name, and service endpoint, which forms the endpoint for your storage account.

For example, if your general-purpose account name is *mystorageaccount* then generally the default endpoints for different services looks like:

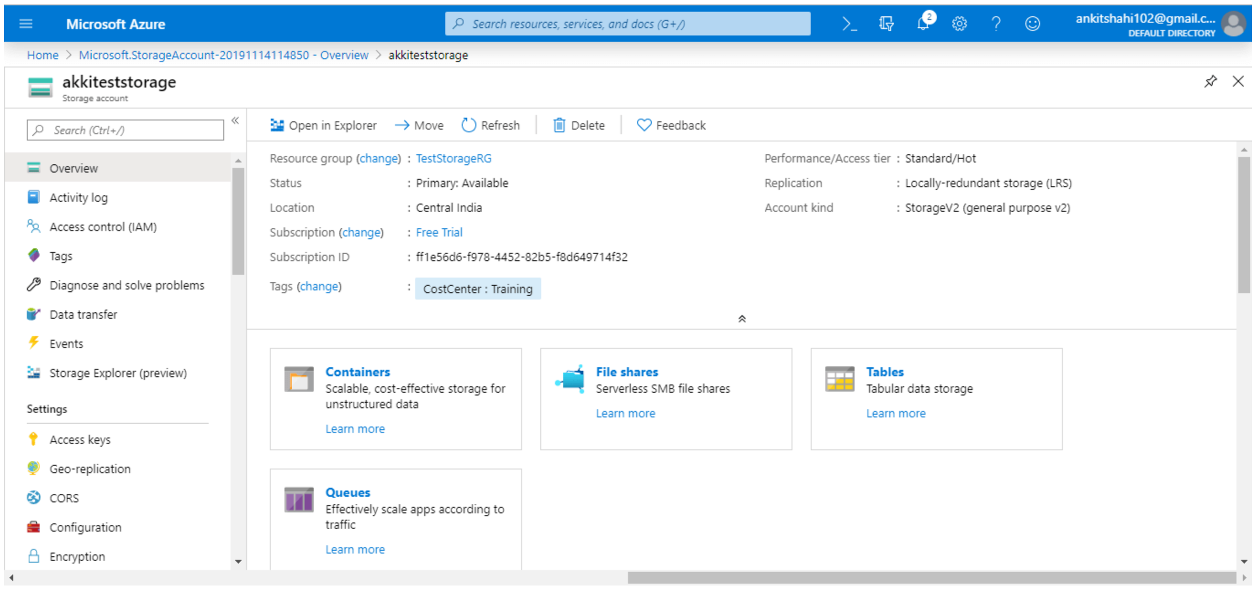
Azure Blob storage: *http://mystorageaccount.blob.core.windows.net.*

Azure Table storage: *http://mystorageaccount.table.core.windows.net*

Azure Queues storage: *http://mystorageaccount.queue.core.windows.net*

Azure files: *http://mystorageaccount.file.core.windows.net*

In case if we want to map our custom domain for these, we can still do that. We can use our custom domain in reference to these storage service endpoints.



Resource group (change) : TestStorageRG
Status : Primary: Available
Location : Central India
Subscription (change) : Free Trial
Subscription ID : ff1e56d6-f978-4452-82b5-f8d649714f32
Tags (change) : CostCenter : Training

Performance/Access tier : Standard/Hot
Replication : Locally-redundant storage (LRS)
Account kind : StorageV2 (general purpose v2)

You can see all the values that you have selected for different configuration setting when creating the storage account.

Let's see some key configuration settings and key functionality of the storage account

Activity Log: We can view an activity log for every resource in Azure. It provides the record of activities that have been carried out on that particular resource. It is common for all the resources in Azure.

Access Control: Here, we can delegate access for the storage account to different users.

Tags: We can assign new tags or modify the existing tags here. We can also diagnose and solve the problems in case if we have any problems.

Events: We can subscribe to some of the events that are happening within this storage account, it can be either logic app or function. For example, a blob is created in a storage account. That event will trigger a logic app with some metadata of that blob.

Storage explorer: This is where you can explore the data that is residing in your storage account in terms of blobs, files, queues, and tables. Again there is a desktop version of this storage Explorer which you can download and connect also, but this is more of a web version of it.

Access Keys: We can use it to develop applications that will access the data within the storage account. However, we might not want to give access to this access key directly.

We may wish to create SAS keys. Here, we can generate specific SAS keys for a limited period, with limited access. Then provide that SAS signature to our developers. Another way is the access keys. Access key gives blanket access. So we recommend not to give access of the access keys to anyone other than the one who created that storage account.

CORS (Cross-Origin Resource Sharing): Here, we can mention the domain name and what operations are allowed.

Configuration: If we want to change any configuration values, then there are certain things that we can't change once the storage account is created - for example, performance type. But we can change the access tier, and secure transport required or not, the replication strategy, etc.

Encryption: Here, we can specify our own key if we want to encrypt the data within the storage account. We need to click on the check box, and we can select a key vault URI where the key is located.

(SAS) Shared access signature: Here, we can generate the SAS keys with the limited access and for the limited period, and provide that information to developers who are developing applications using the storage account. SAS is used to access data that is stored in the storage account.

Firewalls and Virtual network: Here, we can configure the network in such a way that the connections from certain virtual networks or certain IP address ranges are allowed to connect to this storage account.

And we can configure advanced threat protection and can make the storage account compatible to host a static website

Properties: Here we can see the properties related to the storage account

Locks: Here, we can apply locks on the services.

So these are the different settings we can configure, and the rest of the settings are related to different services within the storage account - for example, blob, file, table, and queue.