Microsoft Azure

Cloud computing

Delivery of computing services over the internet.
It is : economic and flexible

- Computing
- Networking
- Storage: to store data (such as Blobs..)
- Analytics : by looking at current trend and historical data

Types of cloud computing:

Public,Private and Hybrid cloud models

Public cloud: can be accessed by public/group of users for services
Private cloud: can be accessed by a single organisation over a private network (disadv.: high cost)
Hybrid cloud:combines both public and private clouds with the ability to sharing data between them (for the data that is meant to be accessed publicly can be shared with the public cloud and sensitive datas in private cloud)
Community cloud: when the infrastructure/resources/softwares services are shared between organisations with the same business goals like universities or colleges and these community clouds are run by the organisation members or by a third-party provider . it can be a public or private cloud

Cloud benefits

- **Economical:**
  you pay for only the services you use
  Instead of having your own IT infrastructure,hardware & third-party data centre which can be costly (high Capital expenditure CapEX),Azure handles this and the client only handles the operating expenditures (OpEX)
  CapEx: is the fixed assets the organisation buys such as laptops,hardware..
  OpEx: is the monthly fees the organisation pays for particular services.
  For example: instead of purchasing MS licence (fixed cost;CapEX)for each user,you pay monthly fee(like a rent payment) to spread the cost throughout the year(for security ,operating systems,network)
- **Scalability & elasticity:**

Scalability: to add/eliminate a computing resource to adjust to the business demand;scaling up or scaling down.
Horizontal scaling:

For ex: during the peak periods you need to increase the servers to handle the increased traffic during this period(scaling up)
Then you can eliminate these additional servers later when the peak period is over(scaling down)

Serverless Computing

Serverless computing in Azure refers to a cloud computing model where you can run code without having to manage the underlying infrastructure. In this model, you only pay for the amount of compute time used, without having to worry about the infrastructure or the cost of maintaining it.
In Azure, serverless computing is achieved through **Azure Functions and Azure Logic Apps.**

Azure functions: driven function to handle the IT infra whenever you run your code as you don't have to worry about the infra.
Azure logic apps: is a cloud based service integrated with the other cloud services (PAAS) and named as IPAAS; integration platform as a service.allows you to create your workflow.
Azure Logic App workflows are sequences of tasks that are automated and orchestrated by Azure Logic Apps. Workflows can be created using a visual designer that allows developers to drag and drop connectors and actions to define the sequence of tasks that make up the workflow.

A workflow in Azure Logic Apps can include a wide range of tasks and actions, such as sending an email, updating a database, processing a file, or triggering a function. Workflows can also include conditional logic, error handling, and retry mechanisms to ensure that the workflow runs smoothly and handles exceptions gracefully.

Azure Logic App workflows are triggered by events or schedules. Triggers are defined by connectors that are used to interact with other systems and services. For example, a trigger can be defined to start the workflow when a new file is uploaded to a specific folder in Dropbox, or when a new email is received in Outlook.

Once a workflow is triggered, Azure Logic Apps orchestrates the sequence of tasks defined in the workflow. Each task is executed in order, and the output of one task can be passed as input to the next task. Workflows can also include branching and looping structures to create more complex sequences of tasks.

Azure Logic App workflows can be monitored and managed using the Azure portal. Developers can view the status of workflows, check for errors, and debug issues in real-time. They can also use Azure Logic Apps to create alerts and notifications to keep track of important events or issues related to workflows.

Overall, Azure Logic App workflows provide a powerful and flexible way to automate and orchestrate tasks across multiple systems and services. With its visual designer, wide range of connectors, and extensibility options, Azure Logic Apps is a popular choice for developers looking to streamline and automate their workflows.

Data centre regions

Data centre regions: is distributed data centres in many areas geographically.
A region has one or more data centres within zones(they work as a backup for each other in case of emergencies)
Each zone has one DC.
Each data centre:

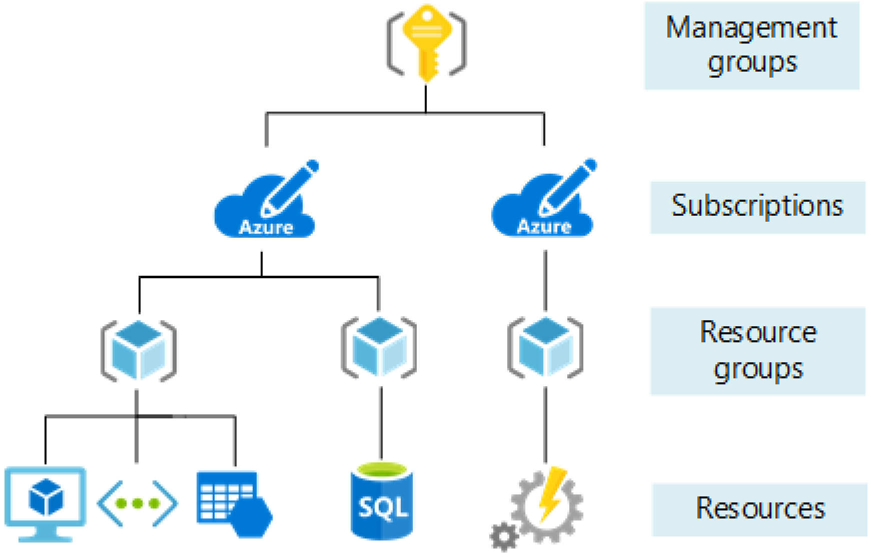
- has thousands of servers
- is independent (in its power,cooling,IT infra..)

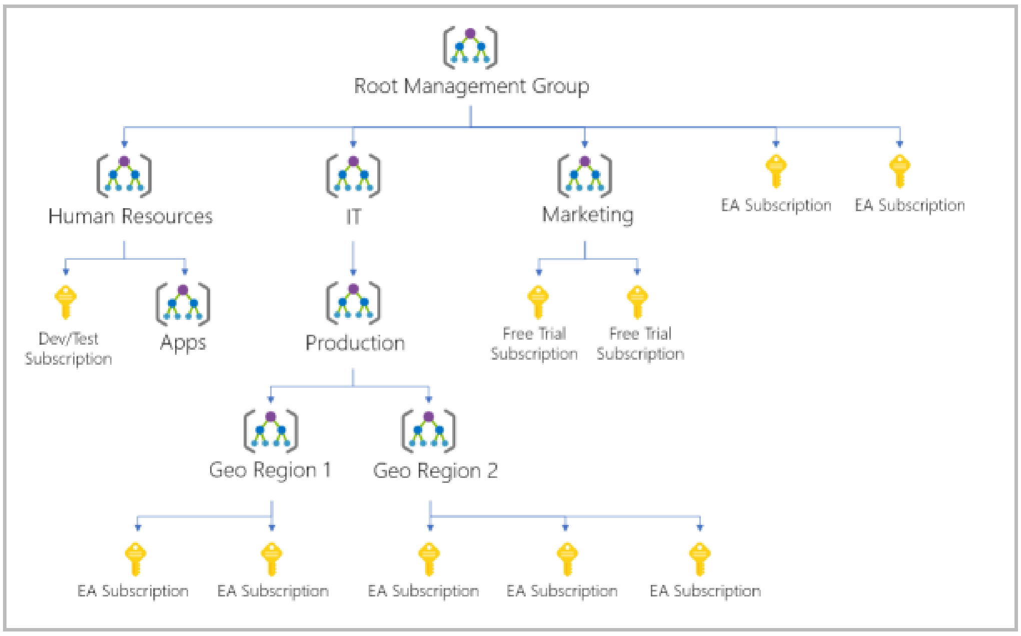
Azure recently has its DC in 60 regions

Benefits for customers:

- **Low latency:** by having data centres close to the users ,it improves the performance of your applications
- **High availability (happens within one region; between zones):** to achieve it for the users (less fault tolerance), multiple data centres in different areas are installed within one region for the backups.
- **Data sovereignty:** the user get to choose there datacenter regions
- **Disaster recovery(happens between regions; regions pairing):** as mentioned in the high availability having both the primary data centre and the back up one but between regions paired is great to improve the disaster recovery (when one region’s Data centre with all servers fails), the other region paired with it acts like a backup).

How to organise your Azure generally





- Creating management group for specified subscriptions
- Creating resource groups for the specified management group
- Creating resources/services such as sql,VM,Storage accounts,networking,analytics..
- Creating containers to store the data in.

  Blobs: Binary Large Objects that is used to store/access large unstructured data

  Types of blobs:

1- Block blobs: to store large files such as images,videos..as it breaks into smaller blocks to store it in parallel

2- Append blobs: allows you to add new data to an existing file without having to rewrite the entire new blob.

3- Page blobs:allows you to read or write from fixed-size pages
Note: you can download the Microsoft Azure storage Explorer to connect all your resources created to the App.
Download link: Azure Storage Explorer – cloud storage management | Microsoft Azure
Pricing calculator Link: https://azure.microsoft.com/en-us/pricing/calculator
AWS Link:
https://aws.amazon.com/about-aws/global-infrastructure/regions_az/

Connections between Python and Azure Resources

1- Connection between python and Azure sql database

Pushing data to it& read it using pyodbc library

Pyodbc: python open database connectivity

Code 1:

Connect to azure sql to open single thread connection/one sql table

```
pip install pyodbc
# to connect the python to azure sql
import pyodbc
server='trainingtestc339.database.windows.net'
database='trainingtest'
username='c339'
password='Rovy0191654747.'
driver= '{ODBC Driver 17 for SQL Server}'  # here make sure that you have
the driver version, otherwise install it from microsoft
#
https://learn.microsoft.com/en-us/sql/connect/odbc/download-odbc-driver-for-
sql-server?view=sql-server-ver16
connection_string =
f"DRIVER={driver};SERVER={server};DATABASE={database};UID={username};PWD={pa
ssword}"
conn = pyodbc.connect(connection_string)
cursor = conn.cursor()
cursor.execute("SELECT * FROM student7")
rows = cursor.fetchall()
for row in rows:
    print(row)
cursor.close()
conn.close()
```

Code 2: connecting to multiple threads/connections :

- Defining the max connections to be opened at the same time
- Creating ‘for’ loop to create a connection and add it to the queue using connection string auth.
- Create get_connection() function used to open the connection between the azure sql and python
- Create release_connection() function to ensure the connection is closed/released properly

To open the connection:

- get_connection()
- Creating a cursor object; connection.cursor()
- Executing your query; cur.execute(‘Query’)
- Using ‘for’ loop to go through each row of the table
- You can open another connection and many ones but based on the max queue defined earlier in step 1
- Don't forget to release the connection at the end for each connection.

```
import pyodbc
from queue import Queue
from threading import Lock
from getpass import getpass
#password = getpass()

server='trainingtestc339.database.windows.net'
database='trainingtest'
username='c339'
#password = getpass('Rovy0191654747.')
password='Rovy0191654747.'
driver= '{ODBC Driver 17 for SQL Server}'  # here make sure that you have the driver version, otherwise install it from microsoft
# https://learn.microsoft.com/en-us/sql/connect/odbc/download-odbc-driver-for-sql-server?view=sql-server-ver16
connection_string = f"DRIVER={driver};SERVER={server};DATABASE={database};UID={username};PWD={password}"
max_con=10
connection_pool=Queue(max_con)
lock=Lock()
```

```
def create_connection():# developing a function to get the connection from 
queue
    with lock:
        if not connection_pool.empty():
            return connection_pool.get()
    connection = pyodbc.connect(connection_string)
    return connection

def get_connection():  # adding a new connection to the queue
    with lock:
        if not connection_pool.empty():
             return connection_pool.get()
        connection = pyodbc.connect(connection_string)
        return connection
     #       return connection_pool.put(connection)


def release_connection(connection): # closing the connection before adding it 
back to the queue
    with lock:
        connection.close()
        connection_pool.put(connection)

connection=get_connection()
cursor=connection.cursor()
cursor.execute('select * from student7','select * from vehicle')
results=cursor.fetchall()  # this line is optional
for cur in results:  # or for row in cursor: print (row)
    print(cur)

release_connection(connection)
# This code should correctly create a connection pool and allow you to reuse 
connections to the database.
#  Note that you may need to modify the connection_string variable to match 
your own database credentials.
```

The main difference between these two code snippets is that the second one is designed to be used in a multithreaded environment where multiple threads need to execute SQL queries concurrently. By creating a connection pool, the code can reuse existing connections rather than creating new

```
connections each time a query is executed, which can improve
performance.
The first code snippet is simpler and designed for a
single-threaded environment where only one connection is
needed. It creates a connection, executes a query, fetches
the results, and then closes the connection.
If you only need to execute a single SQL query in a
single-threaded environment, the first code snippet is
probably sufficient. However, if you need to execute multiple
queries concurrently in a multithreaded environment, the
second code snippet with a connection pool would be a better
choice.
```

```
#connection queue
from queue import Queue
from threading import Lock
import pyodbc
from getpass import getpass


max_con=2
connection_pool=Queue(max_con)
lock=Lock()

# create connections and add them to the queue
for i in range(max_con):
    connection_pool.put(pyodbc.connect(connection_string))


# define a function to get connection from queue
def get_Connection():
    with lock:
        if not connection_pool.empty():
            return connection_pool.get()
    # if queue is empty, raise an exception
    raise Exception("Maximum number of connections reached")

def release_connection(connection):
    with lock:
        connection_pool.put(connection)

connection=get_Connection()
cursor=connection.cursor()
cursor.execute("select * from student7")
```

```
for row in cursor:
    print(row)
connection1=get_Connection()
cursor=connection.cursor()
cursor.execute('select * from student7')
for row in cursor:
    print(row)

connection1=get_Connection()
cursor=connection.cursor()
cursor.execute('select * from student7')
for row in cursor:
    print(row)

release_connection(connection)

# output ; will give the first two connections queries and then detect the
max connection limit

(1, 'rovan', 'f', Decimal('5000.53'))
(2, 'rovan', 'f', Decimal('5000.53'))
(3, 'rovan', 'f', Decimal('5000.53'))
(4, 'rovan', 'f', Decimal('5000.53'))
(5, 'rovan', 'f', Decimal('5000.53'))
(6, 'rovan', 'f', Decimal('5000.53'))
(7, 'rovan', 'f', Decimal('5000.53'))
(1, 'rovan', 'f', Decimal('5000.53'))
(2, 'rovan', 'f', Decimal('5000.53'))
(3, 'rovan', 'f', Decimal('5000.53'))
(4, 'rovan', 'f', Decimal('5000.53'))
(5, 'rovan', 'f', Decimal('5000.53'))
(6, 'rovan', 'f', Decimal('5000.53'))
(7, 'rovan', 'f', Decimal('5000.53'))

---------------------------------------------------------------------------
Exception                                 Traceback (most recent call last)
Cell In[60], line 43
     40 for row in cursor:
     41     print(row)
---> 43 connection1=get_Connection()
     44 cursor=connection.cursor()
     45 cursor.execute('select * from student7')

Cell In[60], line 25, in get_Connection()
     23         return connection_pool.get()
     24 # if queue is empty, raise an exception
---> 25 raise Exception("Maximum number of connections reached")
```

```
Exception: Maximum number of connections reached
```

2- Connection between python and Azure storage/container Reading&downloading the file Using azure.storage.blob library;two classes of (BlobServiceClient,BlobClient)

To provide:

- Connection string
- Container name
- Blob name =’file path’
- BlobClient.from_connection_string(‘connection string’,’container name’,’blob name’,)
- .download_blob()
- .content_as_text

```
from azure.storage.blob import BlobClient
import json
import io
connection_string =
'DefaultEndpointsProtocol=https;AccountName=relgendy;AccountKey=YTQ1cy1bOeU5
9C/5cp+Fjk+/DKHvr9LlimYMwuB+IU9yUtMzmvAISJy9SH4sRC9z87AyOM5ZTCyX+AStInnX4g==
;EndpointSuffix=core.windows.net'
container_name = 'historicaldata'
blob_name= 'Project/data analysis project/DE_category_id.json'

blob_client = BlobClient.from_connection_string(
    connection_string,container_name,blob_name)

with blob_client:
  blob_data = blob_client.download_blob()
  data = blob_data.content_as_text()
  categories=json.loads(data)
```

```
# Check if blob exists in container
if blob_client.exists():
    print(f"Blob '{blob_name}' exists in container '{container_name}'")

    # Get blob URL
    blob_url = blob_client.url
    print(f"Blob URL: {blob_url}")
else:
    print(f"Blob '{blob_name}' does not exist in container
'{container_name}'")
```

```
#output
Blob 'test.csv' exists in container 'historicaldata'
Blob URL: https://relgendy.blob.core.windows.net/historicaldata/test.csv
```

Pushing data/file to azure storage

```
from azure.storage.blob import BlobServiceClient, BlobClient,
ContainerClient
import os

# Set the connection string for your Azure Blob Storage account
connect_str
='DefaultEndpointsProtocol=https;AccountName=relgendy;AccountKey=YTQ1cy1bOeU
59C/5cp+Fjk+/DKHvr9LlimYMwuB+IU9yUtMzmvAISJy9SH4sRC9z87AyOM5ZTCyX+AStInnX4g=
=;EndpointSuffix=core.windows.net'


# Create a BlobServiceClient object
blob_service_client = BlobServiceClient.from_connection_string(connect_str)

# Set the name of the container where you want to upload the file
container_name ='historicaldata'
# Create a ContainerClient object for the container
container_client = blob_service_client.get_container_client(container_name)

# Set the path of the file you want to upload
local_path = "from Jinesh\movies.csv"

# Set the name of the blob that will be created in the container
blob_name = "movies.csv"

# Get the BlobClient object for the blob
blob_client = container_client.get_blob_client(blob_name)

# Upload the file to the blob
with open(local_path, "rb") as data:
    blob_client.upload_blob(data)
```

Azure Storage Blob library's classes

BlobServiceClient and BlobClient are both classes in the azure.storage.blob module of the Azure Storage SDK for Python.

BlobServiceClient represents a client to interact with the Blob service at the account level, allowing you to perform operations on containers and blobs at the account level, like creating or deleting containers, getting properties of the account or container, or listing the blobs in a container.

On the other hand, BlobClient represents a client to interact with a specific blob within a container, allowing you to perform operations on a specific blob, like uploading, downloading, deleting, or getting properties of the blob.

In summary, BlobServiceClient is used for account-level operations and BlobClient is used for blob-level operations.

3- Connection between Azure Storage and Azure sql Using Pandas library and azure.storage.blob lib;BlobServiceClient,BlobClient,ContainerClient classes

Azure Data Factory (ADF)

ADF:
Azure Data Factory can be used to move and transform data between Azure Storage and Azure SQL, and many other data sources and destinations. In fact, Azure Data Factory supports a wide range of data sources and destinations including on-premises and cloud-based data stores, such as Blob storage, Azure Data Lake Storage, Azure SQL Database, Azure Synapse Analytics, and many others.
In order to transfer data between these sources and destinations, you can create data pipelines in Azure Data Factory.
. A data pipeline consists of one or more activities that define the data movement and transformation operations that need to be performed on the data.
To create a data pipeline in Azure Data Factory, you will need to define the source and destination data stores
So in short, Azure Data Factory acts as a connector that allows you to transfer and transform data between various data sources and destinations, including Azure Storage and Azure SQL.

Definitions:

Data Flows
Activities within Azure Data Factory pipelines that use scaled-out Apache Spark clusters.
ADF data flow
Azure Blob Storage is a general-purpose object storage solution that is designed to

store unstructured data such as images, videos, documents, and backups. It provides hot, cool, and archive tiers for storing data based on its access frequency, and allows you to easily manage access control, data protection, and data retention policies.

Azure Data Lake Gen 2, on the other hand, is a scalable and secure data lake storage and analytics service that is designed to handle large amounts of structured and unstructured data. It is built on top of Blob Storage and provides additional capabilities such as hierarchical file systems, batch analytics, and big data processing tools such as Apache Spark and Azure HDInsight. Data Lake Gen 2 supports multiple file formats such as Parquet, Avro, and ORC, and allows you to store data in a way that enables faster processing and analysis.

In the context of Azure Data Factory, Linked Services for Azure Blob Storage and Azure Data Lake Gen 2 are used to connect to the respective storage accounts and perform data integration tasks. So, if you want to process and analyze large amounts of data, and use big data tools such as Spark or HDInsight, then Azure Data Lake Gen 2 would be the better choice. If you just need to store and access unstructured data, then Azure Blob Storage would be sufficient.

Dataset
Represent data structures within your data stores.

Pipelines
A logical grouping of activities that perform a specific unit of work. These activities together perform a task.

Integration Runtimes
Provides the bridge between the activity and linked services.

Linked Services
Define the required connection information needed for Azure Data Factory to connect to external resources, such as a data source.

Activities
Azure Data Factory supports three types of ____: data movement, data transformation, and control.

<u>To create a data factory</u>

- create data factory:
name: DatafactoryTrainingtestc339
- go to this link
https://portal.azure.com/#view/AzureTfsExtension/OrganizationsTemplateBlade
https://aex.dev.azure.com/me?mkt=en-US
- add organisation , sign in and create new project
- add the project link to the create data factory
- to add repo name , go to the project page and in the left bar you will find repos , add new repository
- in azure the repository type is: Azure DevOps repo or GitHub repo.
- public endpoint
- enable creating virtual machine every time i run a task

- to manage repositories in azure DevOPS. repo .. manage repositories .. security

- go to your datafactory and then launch studio
- go to manage icon and then click on Git configuration

- to configure a repository using Github and connect it to my data factory
- create repo in github and then go with the steps in the DF
- after creating , you will be able to see many files in the github repo
- go to home icon + ingest + next

Copy Data tool steps/creating data pipeline from the source to the destination
- source type : azure blob storage name: azurestorageblobsource1
- adding new connection (name: azurestorageblobsource1 and copy the key of the azure storage account and paste it in the account key after selecting the storage account name)
- proceed with the steps and for the destination enter the name azureblobstoragedestination1
- once done. browse through your source and get the file you want to move
- then go to pipelines and copy it to the destination (you can edit the file directory and the name of the file in the destination)
- click save and debug.
- status will be queued then succeeded . Once successful, check your

destination for the file to make sure the process is a success.
- you can find all these updates in the JSON file in the Github pipeline folder in the repository.
- for other files you can press on copy again and start specifying the directories of the file source and destination(Sink).

<u>Quiz notes & engage materials:</u>

Azure Resource Manager 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 organise your resources after deployment.also manage your infrastructure through declarative templates rather than scripts.

Azure Resource Manager templates (ARM templates)/ Azure Resource Manager supports the use of declarative templates to define resources for deployment, enabling you to create a template based on existing resources.

The template is a JavaScript Object Notation (JSON) file that defines the infrastructure and configuration for your project . In the template, you specify the resources to deploy and the properties for those resources as an Infra-code for your project. (like your application code for the project to work).

Infra-code: use the practice of infrastructure as code. In code, you define the infrastructure that needs to be deployed. The infrastructure code becomes part of your project. Just like application code, you store the infrastructure code in a source repository and version it. Anyone on your team can run the code and deploy similar environments.

Resource groups are a way to organise and manage resources in Azure.

Although an **Azure geography** often aligns to a specific country, a geography can also align to a market, such as Europe or Asia. You can host resources in any region, so geographies by themselves do not determine where you can place resources. Geographies also do not correspond to physical data centers, but instead contain regions in which data centers reside.(Geographies typically contain two or more regions.)

Region pair: A region pair consists of two regions within the same geography.

Availability zones: An availability zone encompasses separate power, networking, and cooling, and it is intended to guard against data loss or outages caused by failures in any of those three categories.
Although a single data centre generally fits those criteria, a data centre is not an availability zone, and vice versa. Conceptually, however, they are much the same. Deploying services across availability zones enables you to achieve higher SLAs for those services.

Tags: You can apply tags to a resource group. For example, you might use tags to differentiate production from development or user acceptance testing (UAT) resources.
The tag applies only to the resource group and not to the resources inside the group.
Think of the tag as a label you add to the box, not to the contents of the box.
However, the resources in the resource group can have their own tags.
Applying a tag to a resource group applies the tag only at the container level.

Azure Tenants is a group of users.

Azure subscriptions:Using Azure requires an Azure subscription. A subscription provides you with authenticated and authorised access to Azure products and services
is an identity in Azure Active Directory (Azure AD)
Create additional Azure subscriptions:
- Environments: When managing your resources, you can choose to create subscriptions to set up separate environments for development and testing, security, or to isolate data for compliance reasons
-Organizational structures: For example, you could limit a team to lower-cost resources, while allowing the IT department a full range.
-Billing:you might want to create subscriptions to manage and track costs based on your needs. For instance, you might want to create one subscription for your production workloads and another subscription for your development and testing workloads.

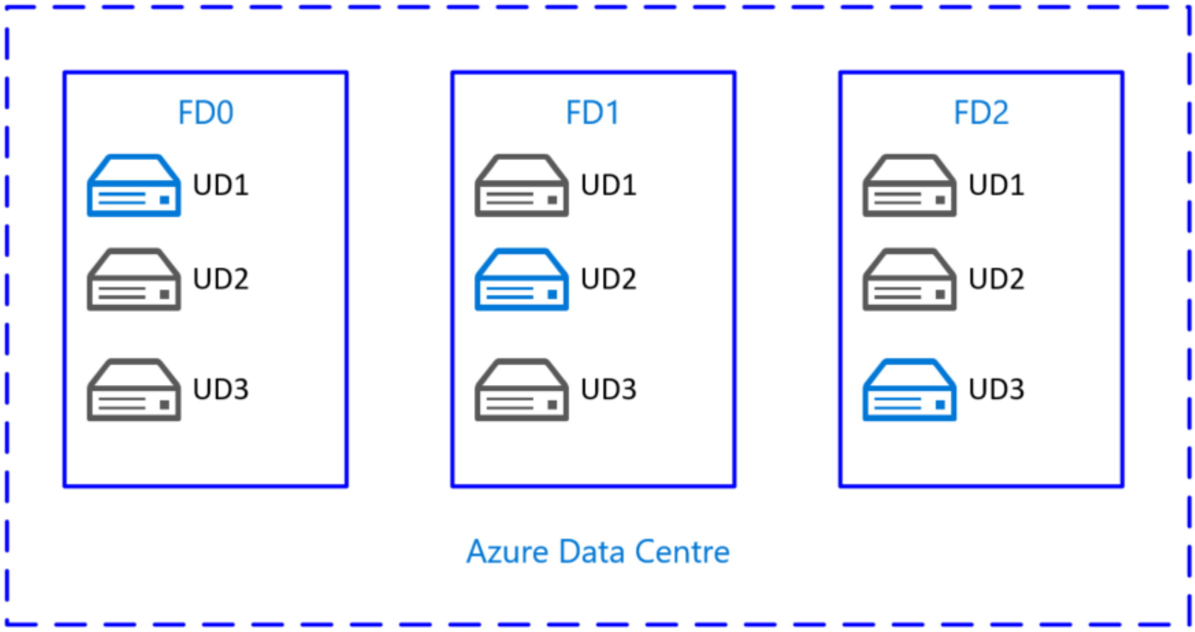
<u>Availability Sets</u>

Availability sets are another feature of Azure that help you avoid potential outages caused by hardware issues, updates, or other events. Two elements that enable availability sets are update domains and fault domains.

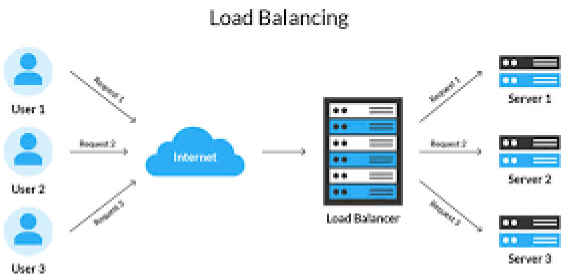
A fault domain is a logical grouping of hardware that shares a power source and network switch, similar to a physical rack in a data center containing VMs.(if the physical hardware fails, all the VMs will fail in this rack)

An update domain is a logical group of VMs that undergoes maintenance activities or reboot events at the same time.

Benefits:An availability set distributes VMs across multiple fault domains and update domains (see below). Distributing the VMs in this way helps guard against outages caused by a power or networking event in a fault domain and also enables the VMs to be updated or otherwise maintained within their respective update domains without causing the set as a whole to be unavailable



Here for ex: if UD1 in FD0 is being updated/rebooted. UD2 in FD1 and UD3 in FD2 are still available. So, the load balancer will manage the traffic to UD2. And if FD0 fails because of power outage,network failure.FD1 & FD2 are still available.



load balancer: manages the traffic.

An Azure Load Balancer is a **Layer-4 (TCP, UDP)** load balancer that **distributes incoming traffic among healthy instances of services defined in a backend pool**. The load balancer can be used to improve the availability and scalability of applications or services that run in the cloud. It can **distribute incoming traffic based on rules and algorithms you define**, such as round-robin, least connections, or source IP hash, and can automatically scale your applications or services up or down based on demand.

Azure Load Balancer supports both inbound and outbound scenarios and **can be used with both public and private IP addresses.** It also provides health probes to monitor the health of the backend instances and can detect and remove unhealthy instances from the pool.

In addition to the basic Load Balancer, Azure also provides **an Application Gateway, which is a Layer-7 (HTTP, HTTPS) load balancer that provides additional features** such as SSL termination, URL-based routing, session affinity, and web application firewall (WAF) capabilities.

Azure provides a suite of fully managed load-balancing solutions for your scenarios.

- If you are looking to do DNS based global routing and do not have requirements for Transport Layer Security (TLS) protocol termination ("SSL offload"), per-HTTP/HTTPS request or application-layer processing, **review Traffic Manager.**
- If you want to load balance between your servers in a region at the application layer**, review Application Gateway.**
- If you need to optimise global routing of your web traffic and optimize top-tier end-user performance and reliability through quick global failover, **see Front Door.**

Azure Database Migration Service

Azure Database Migration Service supports a variety of database migration scenarios for both one-time (offline) and continuous synchronisation (online) migrations. In an offline migration, the source is offline while the migration takes place, making the application(s) supported by that data unavailable. In an online migration, the data is synchronised from the live source to the target and then the application is cut over to the new instance of the database.

what is the difference between azure lake storage gen 1 and gen 2?

ADLS Gen1 is built on top of Hadoop Distributed File System (HDFS), whereas ADLS Gen2 is built on top of Azure Blob Storage. This means that ADLS Gen2 leverages the scalability, durability, and security features of Azure Blob Storage, such as geo-redundancy and hierarchical namespace.

IOT

IoT Central is a pre-built SaaS application platform that is designed for rapid development of IoT solutions, and it is not optimized for handling large-scale telemetry data coming from thousands of sensors. It is more suitable for simpler IoT solutions where users do not have to manage the underlying infrastructure or build a custom application. Therefore, for a solution that requires monitoring and analyzing telemetry data from thousands of sensors, IoT Hub would be the more appropriate Azure solution.

IoT Hub allows devices to send telemetry data, such as sensor readings, to the cloud, where it can be processed and analyzed in real-time. It also enables cloud-to-device messaging, allowing applications to send commands or updates back to the devices, such as firmware updates or configuration changes.
(creating an iot hub in azure)

Azure Sphere provides a highly secure environment for running IoT applications, protecting devices from malicious attacks and ensuring that data is transmitted securely.

APIs enabling software applications to communicate and exchange data with each other in a standard and consistent way. It can be used for many purposes, such as enabling access to a web service, integrating different software systems, building mobile apps, and automating business processes.
It defines the rules for accessing and exchanging data between different systems or components, typically over a network connection.

Azure Artifact

Azure Artifacts is a service provided by Microsoft Azure that allows you to create, host, and share packages with your team and across your organization. Packages can include various types of files, such as libraries, executables, and other types of code artifacts. Azure Artifacts supports multiple package formats and package management tools, such as NuGet, npm, Maven, and Python packages. The service provides various features, including versioning, permissions, and caching, to help you manage and control access to your packages. Azure Artifacts can be used in conjunction with other Azure services, such as Azure DevOps, to facilitate collaboration and streamline your development workflow.

Artificial intelligence

AI falls into two broad categories: deep learning and machine learning.

- Deep learning uses a system modelled on the human mind to enable the service to discover information, learn, and grow.
- Machine learning is a data science technique that uses data to train a data model, test the model for relative accuracy, and then apply the model to new data. A properly trained model should then be able to accurately forecast behaviours, events, and outcomes based on its analysis of past data elements.

Which of the following describe Azure Machine Learning Studio?

a.
It provides the ability for developers to create no-code and code-first machine learning solutions.

b.
It provides a web portal through which developers can use drag-and-drop to create machine learning solutions.

c.
It enables you to deploy machine learning models as web services.

Azure Data bricks

- Create a cluster
- Import data in workspace

To import a file from azure storage in databricks notebook:

```
storage_account_name = "your_storage_account_name"
storage_account_access_key = "your_storage_account_access_key"
container_name = "your_container_name"
file_name = "your_file_name.csv path"
```

```
csv_uri =
f"wasbs://{container_name}@{storage_account_name}.blob.core.windows.net/{file_name}"
spark.conf.set(f"fs.azure.account.key.{storage_account_name}.blob.core.windows.net", storage_account_access_key)

df = spark.read.csv(csv_uri, header=True, inferSchema=True)
```

Ms security and compliance

link:
https://learn.microsoft.com/en-us/compliance/regulatory/offering-home?view=o365-worldwide
Microsoft compliance offerings refer to a set of tools, solutions, and services designed to help organizations meet various regulatory and compliance requirements. These offerings cover a range of compliance areas, including data protection, privacy, security, and risk management. Some examples of Microsoft compliance offerings include:

Microsoft Compliance Manager: A tool that helps organizations assess and manage their compliance with various regulations and standards, including GDPR, HIPAA, ISO 27001, and more.

Microsoft 365 Compliance Center: A centralized hub for managing compliance across Microsoft 365 services, including Exchange Online, SharePoint Online, and OneDrive for Business.

Microsoft Azure Compliance: A set of tools and services that help organizations meet regulatory requirements when using Azure, Microsoft's cloud computing platform.

Microsoft Information Protection: A suite of tools that helps organizations protect sensitive data, including encryption, access controls, and data loss prevention.

Microsoft Cloud App Security: A tool that helps organizations monitor and control access to cloud applications, including identifying and blocking risky activities.

These Microsoft compliance offerings help organizations protect sensitive data, comply with regulatory requirements, and reduce their overall risk of data breaches and other security incidents.

- ms privacy statement

The Microsoft Privacy Statement outlines how Microsoft collects, uses, and protects personal data that it processes through its products, services, and websites. The statement applies to all Microsoft products and services that link to the statement, as well as any other Microsoft-controlled online or offline experiences that collect personal data.

The statement explains what types of personal data Microsoft collects, such as name, contact information, and device data, and how Microsoft uses this data for different purposes, such as providing and improving its products and services, personalizing experiences, and promoting safety and security.

The statement also explains the choices users have regarding their personal data, such as the ability to access, correct, or delete their data, and how Microsoft works to keep personal data secure, including through encryption and other security measures.

Additionally, the statement outlines how Microsoft may share personal data with third parties, such as service providers and business partners, and how Microsoft complies with legal requirements and responds to government requests for personal data.

Overall, the Microsoft Privacy Statement aims to be transparent and informative about how Microsoft handles personal data, and to provide users with the tools and information they need to make informed decisions about their privacy.

- thier online service terms :

he Microsoft Online Service Terms (OST) are a set of legally binding terms that govern the use of Microsoft's online services, including Microsoft 365, Dynamics 365, Azure, and other online services. These terms define the rights and responsibilities of customers who use Microsoft's online services, as well as the terms under which Microsoft provides those services.

The OST cover a variety of topics, such as customer data, service availability and support, security and privacy, intellectual property rights, and liability and indemnification. They also include specific provisions related to the particular online service being used, such as licensing terms and service level agreements.

By agreeing to the OST, customers agree to comply with Microsoft's acceptable use policy, which outlines the types of behavior that are prohibited when using Microsoft's online services. The OST also include provisions related to data protection, and customers are responsible for ensuring that their use of Microsoft's online services complies with applicable data protection laws.

Overall, the Microsoft Online Service Terms aim to provide a clear and consistent set of terms that govern the use of Microsoft's online services, and to ensure that customers understand their rights and obligations when using these services.

Defense in depth:

defense in depth is a comprehensive security strategy that involves implementing multiple layers of protection to reduce the risk of security breaches and data loss. By using a combination of physical, technical, and administrative security measures, organizations can create a more robust and effective security posture that can withstand a variety of threats and attacks.

- Physical Security: This can include measures such as surveillance cameras, access control systems, and physical barriers like fences or walls to prevent unauthorized access to facilities and sensitive areas.

- Perimeter Security: This involves implementing security controls at the network perimeter to prevent unauthorized access, such as firewalls, intrusion detection and prevention systems (IDPS), and virtual private networks (VPNs).

- Network Segmentation: This involves dividing the network into smaller segments or subnets to isolate critical systems and limit the spread of malware or attacks.

- Application Security: This includes implementing security controls and best practices for software development and deployment, such as secure coding practices, regular security testing, and application firewalls.

- User Education and Awareness: This involves providing training and education for users to help them recognize and avoid common security threats, such as phishing and social engineering attacks.

Firewall: serves from L3 to L7, A firewall is a device or service that inspects network traffic flowing through it and applies actions to that traffic based on rules that you specify. For example, if the only endpoints that you need to serve to your users are HTTP (port 80) and HTTPS (port 443), you would create a rule in your firewall to block all traffic inbound for ports other than 80 and 443.
It filters traffic based on protocol, source IP address, source port, destination IP address, and destination port.
overall types of firewalls are network perimeter firewall,application firewall,network security group firewall between azure VN.
Web Application Firewall does not provide traffic filtering ,Web Application Firewall protects web applications against common web-based attacks.

An application security group (ASG) enables you to group servers based on the applications running on them and then manage security for them as a group.

The most widely used protocol layer model is the Open Systems Interconnection (OSI) model, which consists of seven layers:

- Physical Layer: This layer defines the physical media and electrical characteristics used for transmitting data over the network, such as cables, connectors, and wireless signals.

- Data Link Layer: This layer provides the means to transfer data between devices over the physical layer. It handles the framing of data, error detection and correction, and flow control.

- Network Layer: This layer is responsible for routing and forwarding data packets between different networks. It determines the best path for data to travel between source and destination devices.

- Transport Layer: This layer provides end-to-end communication between applications or processes running on different hosts. It ensures that data is delivered reliably, and provides flow control and error recovery mechanisms.

- Session Layer: This layer establishes and manages sessions between applications, which allows them to exchange data in a synchronized and organized manner.

- Presentation Layer: This layer provides a common format for data representation, such as text, images, and video. It also handles data compression and encryption.

- Application Layer: This layer provides application-level services, such as file transfer, email, and web browsing. It interacts directly with the user or application, and is responsible for initiating and terminating communication sessions.

User Defined Route UDR

UDR:
In Azure, user-defined routes (UDRs) are used to override the default routing behavior for network traffic. By default, Azure routes traffic between subnets and virtual networks using system-defined routes. However, in certain scenarios, it may be necessary to use UDRs to customize the routing behavior for specific traffic flows.

Here are some situations where you might want to use Azure UDRs instead of the default route:

- Virtual Network Peering: When you have two virtual networks connected through peering, you can use UDRs to control the traffic flow between them. For example, you can configure UDRs to route traffic between specific subnets in one virtual network to specific subnets in the other virtual network.

- Network Virtual Appliance: When you deploy a network virtual appliance (NVA) such as a firewall or load balancer, you can use UDRs to route traffic through the NVA. This allows you to inspect and filter traffic before it reaches its destination.