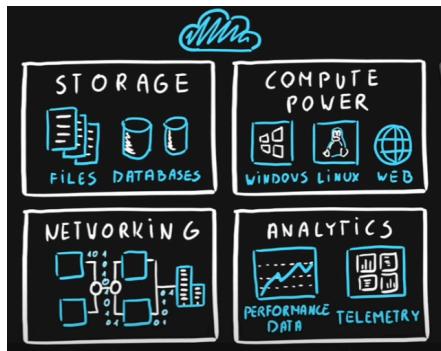


AZURE BASICS

- **Azure:** Microsoft's cloud computing platform



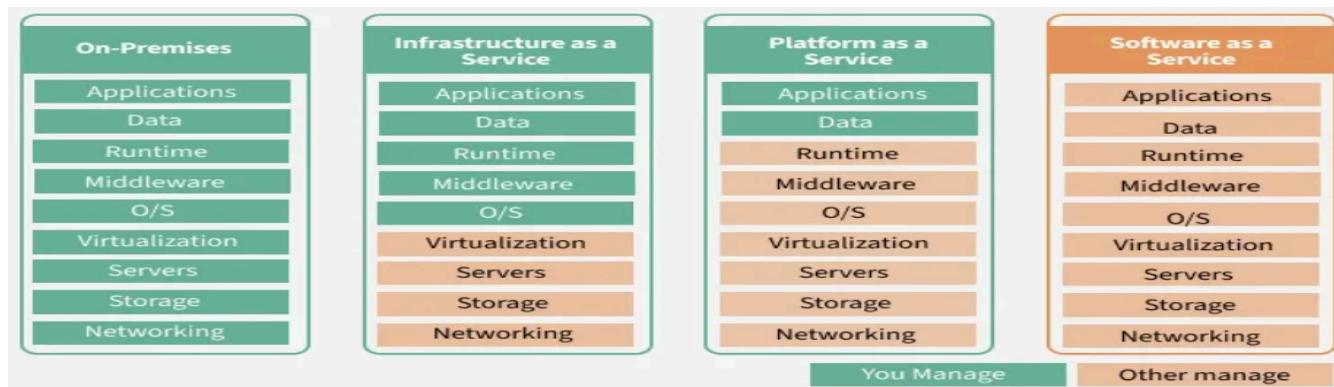
4 main Cloud services :

- *storage*(ex: Amazon S3/BlobStorage),
- *computational* (ex: OS, virtual servers) ,
- *networking* (ex: Azure Virtual N/w),
- *analytics* (ex: Google Big Query)

- **Cloud Characteristics:**

- *Scalability*: Vertical Scaling (▲ / ▼ memory/storage of existing resource- existing cpu,ram..) & Horizontal Scaling (▲ / ▼ no.of resources-cpu,linux,db.....for my application)
- *Elasticity* : allocate/de-allocate resources dynamically as per need.
- *Agility*: How fast to react? Min,sec... V/S "**On Prem(you own everything)**"-manually control,hrs, days,mths
- *Fault tolerance* : have racks & disks....**Recovery Soln**: Restart or Replication Factor = 3
- "**Pay as u GO**" : **actual usage** (e.g., per hour, per GB, per API call).

- **Service Models:**



IaaS(Infrastructure as Service)	PaaS(Platform as Service)	SaaS(Software as Service)
<ul style="list-style-type: none"> Resources provided related to hardware/virtualisation (process of creating virtual env, virtual mem, virtual servers, virtual storage-virtual layer b/w host os & multiple instances of guest os) Platform & software services managed by you! 	<ul style="list-style-type: none"> Resources provided app development, hosting, and runtime management Application & software services managed by you! 	<ul style="list-style-type: none"> Azure manages infrastructure, platform, & software You manage nothing! 😊
Ex: storage (ssd/hdd), servers(mem/cpu), vms(ubuntu), networking tools(routers) <ul style="list-style-type: none"> Azure Blob Storage Azure Virtual Network (VNet) Azure Virtual Machines (VMs) 	ex:  for rdbms, & web hosting , docker	ex: outlook, one-note

- 3 Famous Cloud Models:** Public, Private, Hybrid
- Infrastructure:**



Data Availability Zones: zoneA & zoneB : multiple datacentres, each with different configuration,power,cooling

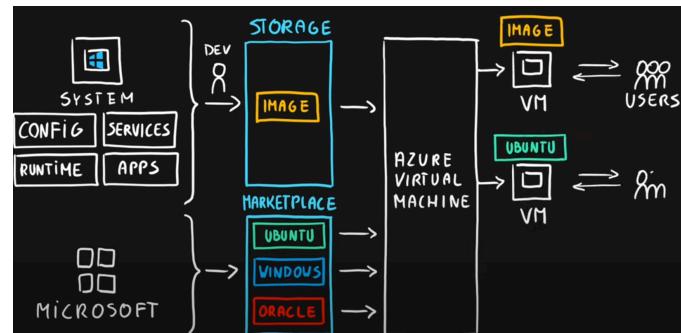
Region Pairs: regions-> having multiple availability zones

If 1 fail -> others are there to help!!! -> **Replication Factor !!**

How they are connected?? **Zonal Services** { specific to zone , ex:vms} + **Zone-redundant Services** {replicated across zones, ex: sql}

- Resources, Resource Groups, Resource Manager(access controls)
- **Storage Browser:** tool that allows you to manage everything that's in your azure account.
- **Compute Services:**

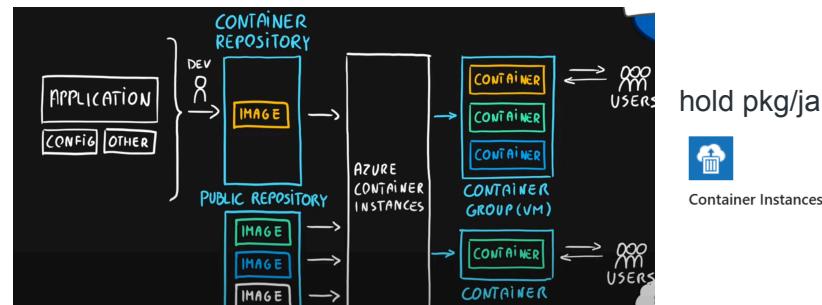
→ **AVM (Azure Virtual Machine):** Vms emulate hardware , each has its own os



- Images: read only templates, specific config(ram/storage/env/dependencies) , instr needed for setup of appln/container
- Provides market-place vm images & enables customised images to be built , stored (hubs/container repository) & used....
- Just select img->download img ->connect to it..
- Images are auto-scaled among vms ,

distributing traffic across vms ->using Load Balancer.

→ **Azure Container Instances:** Containers emulate os, same os but isolating apps based on req & config & services they use & offer(ex: python package versions)

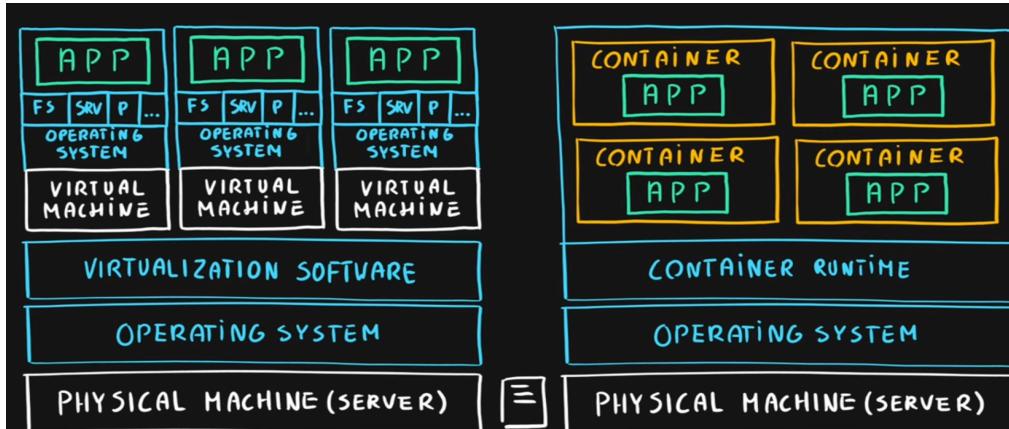


- Containers are instances of images that hold pkg/jar files for running the application(ex: sok8),

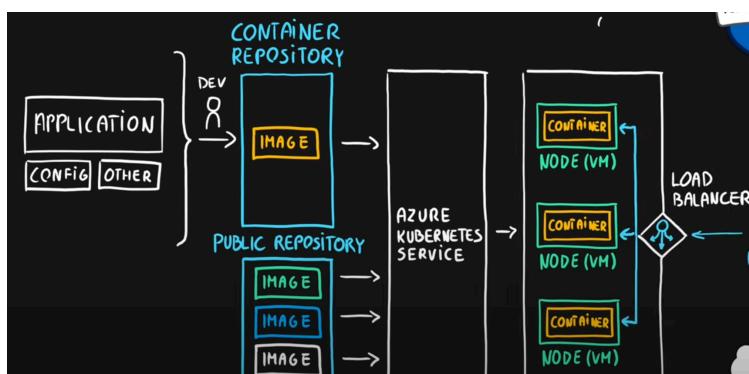


Select container image->& deploy -> container instances

Mainly used for microservices..



→ Azure Kubernetes Service(AKS):

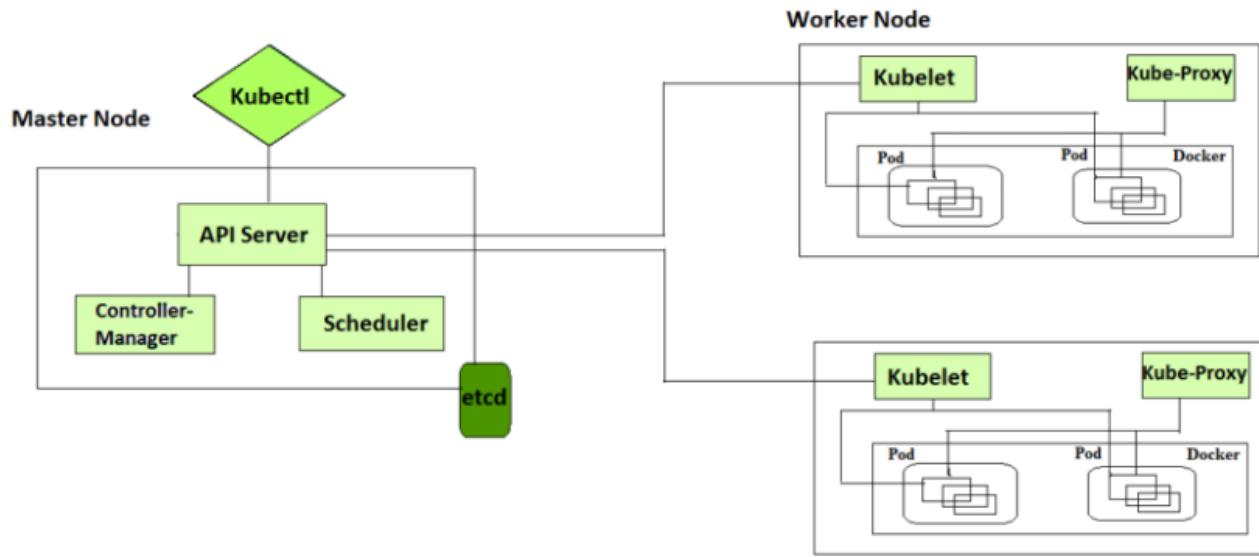


- *Orchestration service*: allocate/deallocate & schedule containers automatically , to run on pods(gp of container)
- Some cmd's:
 - kubectl get nodes
 - kubectl get pods
 - kubectl apply -f file.yaml
 - Kubectl logs <pod-name>
 - Kubectl describe pod <pod-name>
 -



Azure Kubernetes Service (AKS)

Microsoft | Azure Service



Kubectl-entry point to kubernetes

Api servers- that listens to requests passed via commands

schedular- schedule nodes to run

Control manager- manages worker nodes (monitors their status & tries to match to desired state- 3 replica -> 4replica)

Etcdb- db that logs metadata & all info of pods

Kubelet- entry point in worker node, tracks pod health, container state

Pod- gp of containers, *temporary-disposable* unit(deletion) ,IP addr for each pod to communicate with other pod, container are co-located(share resources)

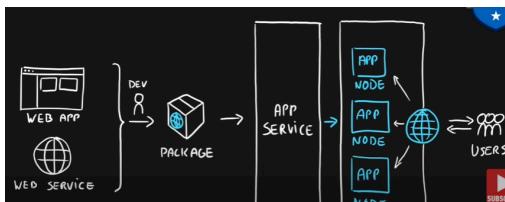
Kube proxy- manages networking across pods via IP

Service/LoadBalancer- routes traffic across pods

Deployment kind - “Hey Kubernetes, create and manage this set of pods for me.”

<code>apiVersion: apps/v1</code>	<code>apiVersion</code>	Which Kubernetes API version to use
<code>kind: Deployment</code>	<code>kind</code>	The type of resource (Deployment)
<code>metadata:</code>	<code>metadata.name</code>	Name of the Deployment
<code> name: nginx-deployment</code>		
<code>spec:</code>	<code>spec.replicas</code>	Desired number of pods
<code> replicas: 2</code>		
<code> selector:</code>	<code>spec.selector</code>	Which pods belong to this Deployment
<code> matchLabels:</code>		
<code> app: nginx</code>		
<code>template:</code>	<code>template</code>	Pod blueprint
<code> metadata:</code>	<code>template.metadata.labels</code>	Labels assigned to pods
<code> labels:</code>		
<code> app: nginx</code>		
<code>spec:</code>	<code>template.spec.containers</code>	List of containers inside each pod
<code> containers:</code>	<code>container.name</code>	Name of the container
<code> - name: nginx</code>		
<code> image: nginx:latest</code>	<code>container.image</code>	Container image to run
<code> ports:</code>	<code>container.ports</code>	Exposed ports inside the container
<code> - containerPort: 80</code>		

→ Azure App Service: host enterprise apps



- **Networking Services:** VPN, Gateways

- **Azure Storage Services:**

Data Types: Structured(rdbms/sql) , Semi-Structured (key-value,json/NoSQL) , Unstructured (audio,video/blob)

3 storage tiers: HOT 😱 , COOL 😌 , ARCHIVE

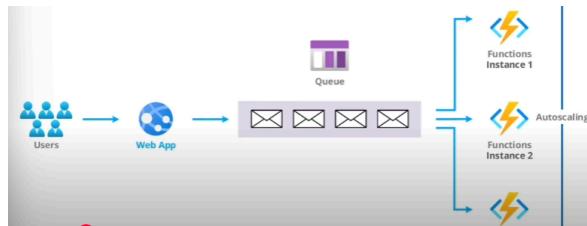
→ Azure Blob Storage

Containers of Blobs (general purpose blob/file storage)

→ Azure Queue Storage

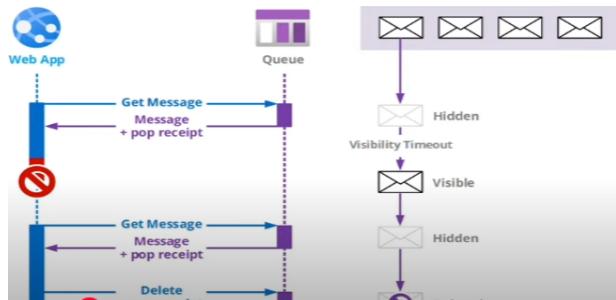
Store msg in queue for asynchronous processing(ex:iot/real time data)

msg have **TTL(TimeToLive)** for expiration>>exceeded: msg auto deleted



Consumers apps/services consume queue msg

Ex: “**Retry Pattern**” [hide/lock the msg when get req received so that cant be used by other process ->make visible asap once get response sent with ack ->later used by other process]

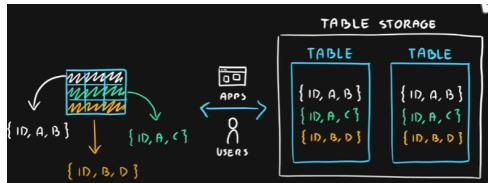


→ Azure Table Storage

Group of Tables (can store structured &semi-structured data)

Used when: proper schema, json serialisable data

Ex: used for simple logging, metadata &configuration store

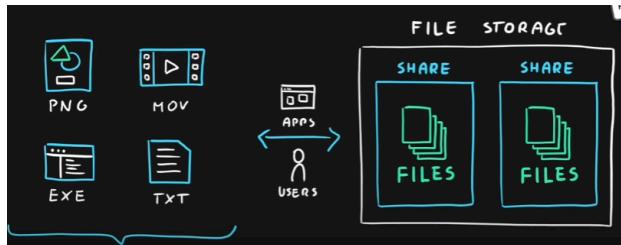


Follows CompositePrimaryKey = PartitionKey(groping all items having that partition(column field)) + RowKey(uniquely identify each record) + Timestamp

→ Azure File Storage

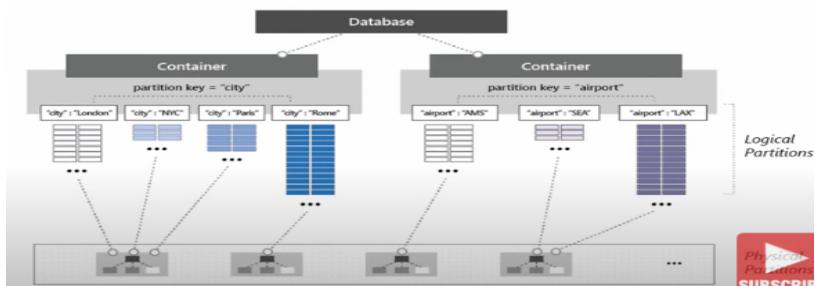
Shares/Folder of Files (shared directories & sync with local directory)

“Lift & shift operation” : “**lifting**” the app/workload from your current local env and “**shifting**” it to the cloud.



→ Azure Cosmos Db : no-sql, schemaless , multiple apis supported(sql, mongodb, gemlin {graph db}, cassandra{columnar db})

Distributes data into partitions: LOGICAL{group all items having same key} & PHYSICAL{store multiple logical partitions into hardware storage,ex: disks,partition at hardware storage level}



Azure Cosmos DB for NoSQL

Azure Cosmos DB's core, or native API for working with documents. Supports fast, flexible development with familiar SQL query language and client libraries for .NET, JavaScript, Python, and Java.

Azure Cosmos DB for MongoDB

Fully managed database service for apps written for MongoDB. Recommended if you have existing MongoDB workloads that you plan to migrate to Azure Cosmos DB.

For noSQL: query {key:value} using simple sql,

For mongodb: {key:vales} with values having varying fields, use mongodb query language ,ex: find,insert..

RequestUnit(RU) : cost of api call being made

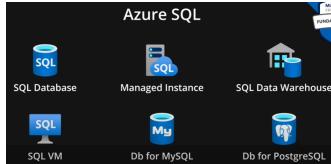
In below ex: partition key="category"

→ **Azure SQL Db:** mainly for structured data with constraints & relations

Elastic Pool: collection of single databases with shared resources

Managed Instance: fully managed instance of SQL Server

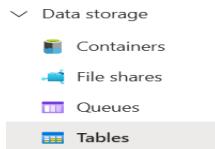
Sql Datawarehouse: cloud-based data warehouse service designed for storing and analyzing large volumes of data



Screenshot of the Azure Data Studio interface. The left sidebar shows navigation options like Overview, Activity log, Tags, Diagnose and solve problems, Query editor (preview), Mirror database in Fabric (preview), Resource visualizer, Settings, Data management, Integrations, Power Platform, Security, Intelligent performance, Monitoring, Automation, and Help. The main area has tabs for Query 1, Run, Cancel query, Save query, Export data as, and Show only Editor. The Query 1 tab contains T-SQL code for creating a table and inserting data into the movies table, followed by a select statement. The Results tab displays the resulting table:

id	title	genre
1	Inception	Science Fiction
2	The Godfather	Crime
3	Titanic	Romance
4	The Dark Knight	Action
5	Interstellar	Science Fiction

→ **Azure DataLake Storage**



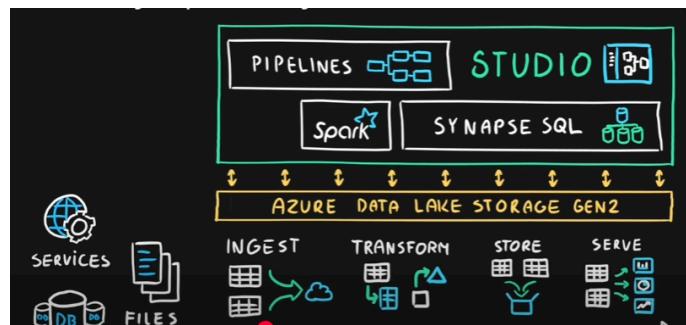
Enable hierarchical namespace: store files/data in directories, sub-directories, easy to access & organise

Better than blob storage: as all files will be in same level, just with unique path (files inside container)

Mainly used for big data analytics: integrated with synapse,data bricks, data factory

- **Big Data Services:**

- **Azure Synapse Analytics:**



Data warehouse & analytics platform

Spark

Sql pools

Synapse Pipelines(ETL pipelines)

All combined in Studio(unified experience)

Integrates **data ingestion, preparation, management, and analytics** in one workspace.

Integrate with adls

- **Azure HDInsight:**



Open source Big Data processing service

Managed Apache Big Data ecosystem

Work with open-source bigdata frameworks:hadoop,kafka..

Hadoop – for batch processing of big data.

Spark – for in-memory, fast data processing.

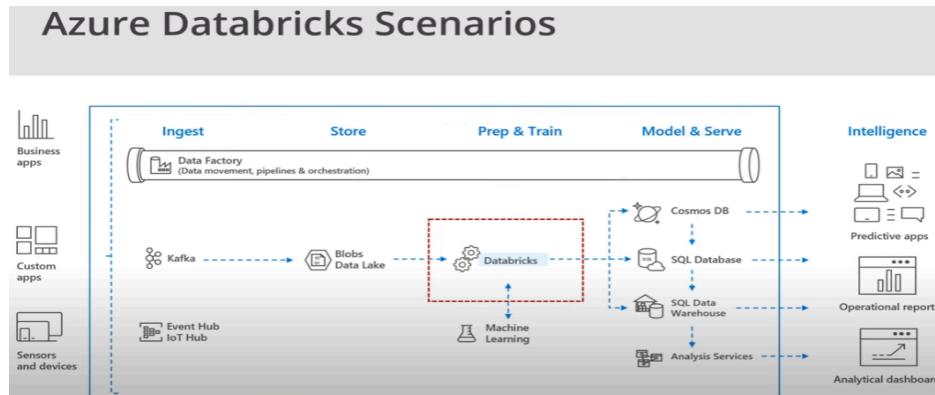
Hive – for SQL-like querying on big data.

Kafka – for real-time data streaming.

HBase – for NoSQL data storage.

→ **Azure Databricks**: analytics platform only for **spark based applications**

Allows to create clusters, write scripts(py,scala,write spark code(apply actions & transformations)) that will run on clusters



→ **Azure Data Factory**: ETL(Extract Transform Load) pipeline building tool

Source: input files to fetch/extract ; Sink: target/destination where we load the data into

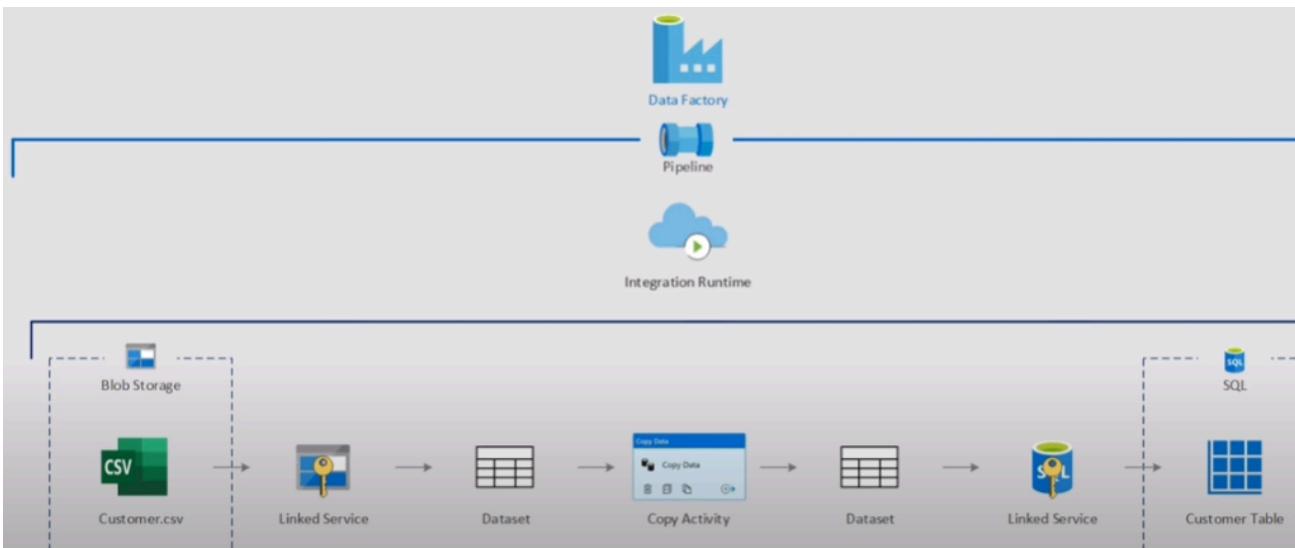
Linked Service: establish connection to source & sink

Dataset: what specific data we need to use (ex:which table,file, apply transformation to get enriched data)

Main Activity: transformation(**Copy Activity**" or "**Data Flow Activity**")

IntegrationRuntime: handles entire job running, runs the activities by integrating multiple tools

Pipeline: logical container of workflow: defining sequence of activities



Screenshot of the Microsoft Azure Data Factory Author interface, showing the configuration of a pipeline named **blobtosql**:

Factory Resources:

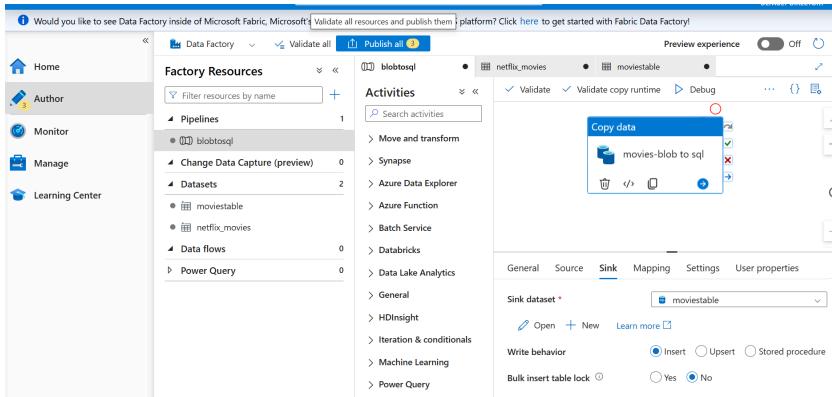
- Pipelines:** Contains **blobtosql**.
- Datasets:** Contains **moviestable** and **netflix_movies**.
- Data flows:** Contains **netflix_movies**.
- Power Query:** Contains **netflix_movies**.

Activities:

- Copy data:** Task named **movies-blob to sql**.

Copy data Task Configuration (General tab):

- Source dataset:** **netflix_movies**
- File path type:** File path in dataset
- Filter by last modified:**
- Start time (UTC):** [Blank]
- End time (UTC):** [Blank]



etl-demosearche-database (amdemosqlserver...)

SQL database

Service menu

Login + New Query ⚡ Open query Feedback Getting started

etl-demosearche-database (hemaganta1...)

Opening linked object explorer here. For full capability please click here to open Azure Data Studio.

Tables > dbo.movies ...

Views > sys.database_firewall_rules ...

Stored Procedures

Query 1

Run Cancel query Save query ...

```
1 select * from movies;
```

Results Messages

show_id	title
70234439	Transformers Prime
80117401	Jandino: Whatever it
81145628	Norm of the North: I

→ Azure Event Hub:

real time data ingestion service

Events sent by producers via HTTP or Kafka ...into Event Hub(contains partitions of event data enable parallel processing)

Partitionkey: used to land events into specific partition

Namespace: collections of logical eventhubs

Consumers: Blob Storage, Azure Stream Analytics



Offset: position of event in partition

Each partition maintains its own offset sequence.

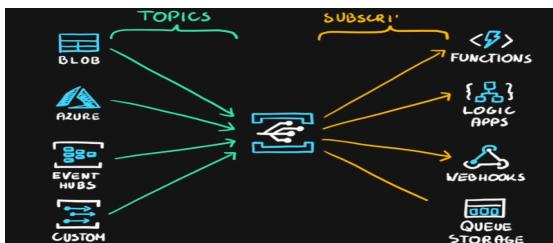
Each consumer group tracks its own offset per partition.

Helps each consumer know how many partitions read so far -If that consumer restarts, it can continue from next offset -no need to reprocess old events

→ Azure Event Grid (~Kafka):

Follows pub-sub model: providers register themselves -> publish messages as topics ->send to event-grid ->event grid sends the specific topics being subscribed to by the subscribers

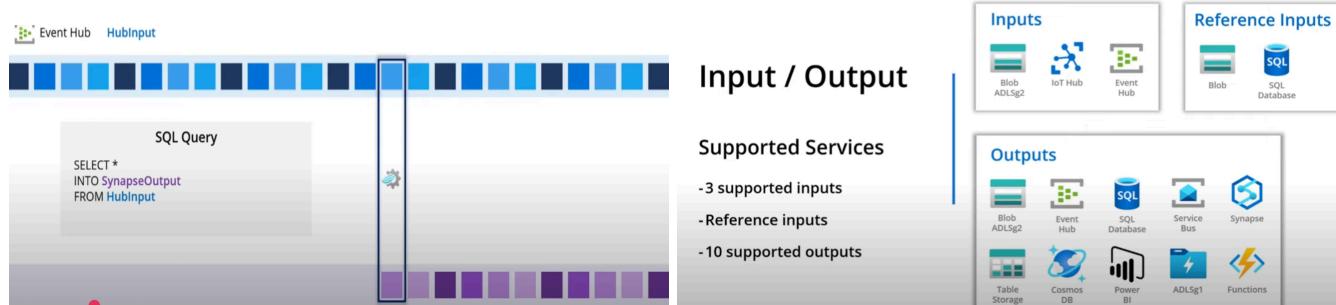
Used for : real time events/data routing b/w different sources & consumers.



→ Azure Streaming/Synapse Analytics: ADF+SPARK+SQL DATAWAREHOUSE

Input sources: get real time streaming data from event-hub

Perform analytics by writing sql queries & send the enriched data to output sources(cosmosdb, adls)



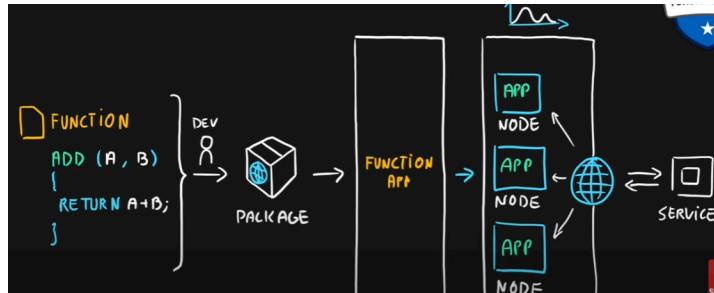
- **Serverless Services:** You write code or workflows, Azure runs them without you worrying about servers(build,manage,scaling,deployment)

→ Azure Functions:

Small pieces of code written->devs deploy to pkgs->registered in function apps & exposed

Mainly designed for nano-services

Run the code snippets in the cloud (serverless)

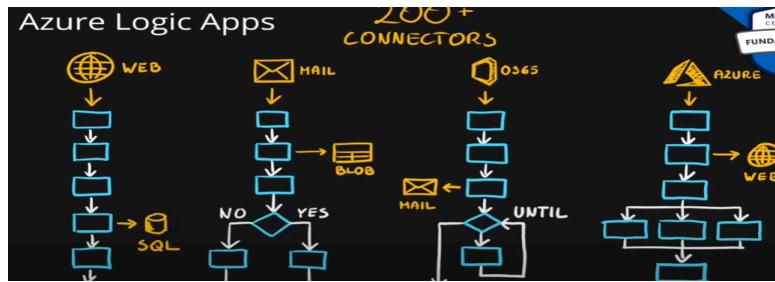


→ Azure Logic Apps

Automate workflows using a visual designer (low-code)

Build step-by-step logical workflows(conditions, loops) & trigger them using microsoft products(outlook,O365)

Ex: When a new row is added to an Excel sheet in OneDrive → send an email → update a SQL Database.



- **ML Service:** Azure ML Lab (to write nb, train models, deploy,test, monitor, re-learn form observation/pattern)

- **Security Services: Authorisation & Authentication**

Authentication: Who You Are??

Authorisation: What you can access??

→ **Azure Active Directory(AAD):**

Identity & access management service

Manages- users,resource groups,roles,role assignments,

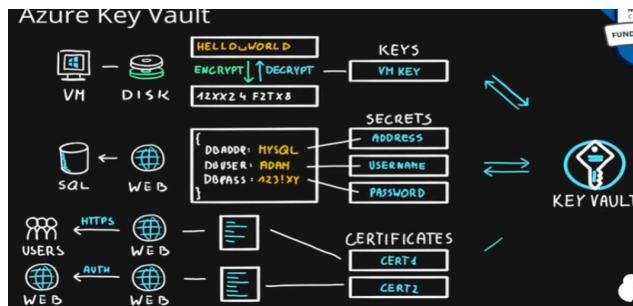
Can be used by other microsoft apps(office 365,outlook)

→ **SLA (Service Level Agreement):** measure of guarantee of expected level of service

availability,performance,reliability provided by Microsoft Azure for the services they provide

→ **Azure KeyVault:**

Secure storage for encrypted keys, configuration secrets/credentials , certificates(SSL/TLS-identifies service & encrypts communication b/w client and server)



→ **Azure Role-Based Access Control(RBAC):**

Role: set of actions assigned to users/user groups..

Scope: where the access can be applied to..follows a hierarchy (access to parent object(ex: resource gp) can be extended and applied to child object(ex:resources))

Resource Locks: assign locks to resources to prevent accidental modification/deletion

(*ReadOnly Lock*- only read allowed)

(*Delete Lock*)- all actions except delete allowed

→ **SAS(Shared Access Signature):** temporary pass(temporary url & pswd that is short lived) allowing others to access my storage account with specific permissions without sharing my storage account key

Storage Account Key: master credentials for your entire Azure storage account