**Azure + Azure Data Engineer Study Material:**

Table Of Contents

[Understanding of Different Cloud Deployment Models: 5](#_Toc158327081)

[Different cloud services/cloud service models: 6](#_Toc158327082)

[What is Microsoft Azure: 7](#_Toc158327083)

[Azure Resources/Services: 7](#_Toc158327084)

[Azure Resources/Services: 8](#_Toc158327085)

[Azure Resource Groups (RG) & Configuration and management of Azure Resource groups for hosting Azure services: 8](#_Toc158327086)

[What is Azure Storage Account/Azure Storage Services: 9](#_Toc158327087)

[Container/Blob (Binary large object): 9](#_Toc158327088)

[Page blob: 10](#_Toc158327089)

[Append blob: 10](#_Toc158327090)

[Block Blob: 10](#_Toc158327091)

[Redundancy/Replication: 10](#_Toc158327092)

[File Share Storage: 12](#_Toc158327093)

[The Difference: Azure Blob Storage vs File share Storage vs Disk Storage: 13](#_Toc158327094)

[Azure Storage Explorer(ASE): 15](#_Toc158327095)

[Add and azure account: 15](#_Toc158327096)

[Use a Storage Account Name & Key: 16](#_Toc158327097)

[Use a connection String: 16](#_Toc158327098)

[Shared Access Signature (SAS): 17](#_Toc158327099)

[Azure Table Storage Services: 17](#_Toc158327100)

[What is Table storage: 18](#_Toc158327101)

[Implementation of Table Storage Services: 18](#_Toc158327102)

[Soft Delete: 19](#_Toc158327103)

[How soft delete works 19](#_Toc158327104)

[Enabling or disabling soft delete: 20](#_Toc158327105)

[Retention period: 20](#_Toc158327106)

[Implementation Steps of Soft Delete: 20](#_Toc158327107)

[Azure Data Lake Storage Gen2 Storage Accounts: 22](#_Toc158327108)

[Benefits and challenges of Azure Blob vs. Data Lake storage 22](#_Toc158327109)

[Azure Blob Storage and Data Lake are well suited to specific situations and uses. 23](#_Toc158327110)

[Steps to Improvise & configure Azure Storage Accounts Authentication & Authorization 23](#_Toc158327111)

[Azure Logic Apps: 24](#_Toc158327112)

[Implementation Steps of Logic App Services: 24](#_Toc158327113)

[SQL DB as service in Azure: 26](#_Toc158327114)

[Azure SQL Database: 27](#_Toc158327115)

[Benefits of SQL DB as Service: 27](#_Toc158327116)

[SQL Database Deployment options: 27](#_Toc158327117)

[Azure SQL Database Purchasing Model: 27](#_Toc158327118)

[Implementation Steps of SQL DB as PAAS: 28](#_Toc158327119)

[Advantages of Cloud Computing: 29](#_Toc158327120)

[Advantages/Features of Azure Cloud Computing: 29](#_Toc158327121)

[Key points of Azure Data Engineering: 30](#_Toc158327122)

[Architecture of Azure Cloud Computing: 33](#_Toc158327123)

[Hierarchy of Microsoft Azure Cloud Platform: 35](#_Toc158327124)

[What is Azure Data Factory (ADF):: 36](#_Toc158327125)

[Process & Procedures to Load the Data from Source to Target using ADF: 37](#_Toc158327126)

[Integration Runtime (IR):: 38](#_Toc158327127)

[Building blocks of Azure Data Factory(ADF):: 38](#_Toc158327128)

[Pipelines in Azure Data Factory: 38](#_Toc158327129)

[Activities in ADF: 39](#_Toc158327130)

[Datasets in ADF:: 39](#_Toc158327131)

[Linked services in ADF:: 39](#_Toc158327132)

[Different types of Integration Runtime: 42](#_Toc158327133)

[Compute Infrastructure: 42](#_Toc158327134)

[Self-hosted Integration runtime: 42](#_Toc158327135)

[Use of Triggers: 43](#_Toc158327136)

[When to use ADF: 43](#_Toc158327137)

[Why to use ADF (or) Why ADF: 44](#_Toc158327138)

[Complete data flow end to end with ADF Process: 45](#_Toc158327139)

[Mapping Data Flows for Transformation & Aggregation: 46](#_Toc158327140)

[Implementation steps to create/deploy an ADF:: 46](#_Toc158327141)

[Networking (left side under ADF): 47](#_Toc158327142)

[Implementations steps for copying the data from Blob SA to ADL’s Gen2 SA using ADF: 48](#_Toc158327143)

[Implementation steps for copying the zip file from Blob SA to ADL’s Gen2 SA using ADF: 48](#_Toc158327144)

[Implementation steps to perform Metadata activity, Validation activity & If condition activity in Azure Data Factory (ADF): 50](#_Toc158327145)

[Implementation steps to perform Get Metadata control, Filter control, ForEach control in ADF pipelines: 54](#_Toc158327146)

[Implementation steps to copy the data from GitHub to Azure ADL Storage services (Using Parameters): 59](#_Toc158327147)

[Copying the data from http (GitHub) to ADL Gen2 StorageAccount: 61](#_Toc158327148)

[Allocating variables to ADF pipelines: 62](#_Toc158327149)

[Creating Dynamic Pipelines with lookup activity to get the count of files from .json: 63](#_Toc158327150)

[Copying data from multiple files: 65](#_Toc158327151)

[Copying the files from GitHub Dynamically with the use of Dynamic parameters allocation-AUTOMATION PROCESS: 67](#_Toc158327152)

[What is the use of Triggers in ADF: 72](#_Toc158327153)

[Types Triggers: 72](#_Toc158327154)

[Implementing schedule based trigger 73](#_Toc158327155)

[Copying the data from Azure SQL DB to ADL Gen2 Storage Account: 77](#_Toc158327156)

[Implementation steps to host a DB & DB Server in Azure cloud computing & loading the data from Blb SA(Source) to SqlDB(Destination/Target) in single table: 78](#_Toc158327157)

[Implementation steps for loading the data from Blb SA to SqlDB in multiple table: 81](#_Toc158327158)

[Implementation steps to copy the data from Blob Storage Account to SQLDB and then SQLDB to Gen2SA & executing multiple pipelines in One Go: 84](#_Toc158327159)

[Execution of above pipelines in sequence one after the other: 90](#_Toc158327160)

[When can we pick up schedule-based triggers??tumbling window base triggers?? & Event base triggers?? 90](#_Toc158327161)

[Tumbling window base triggers(TWBT): 90](#_Toc158327162)

[Implementation steps for Tumbling Window Based Trigger: 91](#_Toc158327163)

[Storage or Event base triggers: 92](#_Toc158327164)

[Implementation steps for Storage & Event based triggers: 92](#_Toc158327165)

[Azure Key Vault: 93](#_Toc158327166)

[Why we use Azure Key Vault: 93](#_Toc158327167)

[Implementation Steps for Deploying Azure key Vault: 93](#_Toc158327168)

[What is GitHub: 97](#_Toc158327169)

[Repository 97](#_Toc158327170)

[Branch 98](#_Toc158327171)

[Commits 98](#_Toc158327172)

[Pull Requests 98](#_Toc158327173)

[Creating an account in GitHub: 98](#_Toc158327174)

[Implementation steps to setup the code repository for ADF in GitHub Platform (or) Integrating ADF with GitHub Platform: 99](#_Toc158327175)

[Data flows in Azure Data Factory(ADF): 103](#_Toc158327176)

[Implementation steps for Dataflow in ADF: 104](#_Toc158327177)

[Inline Datasets in Dataflow Source Control in Source Settings tab: 105](#_Toc158327178)

[Implementation steps of Dataflow with Source, Filter & Sink Transformation in ADF with inline Dataset: 105](#_Toc158327179)

[Implementation of Dataflows using Source, Select, Sort, Sink Transformations: 109](#_Toc158327180)

[Implementation of Dataflows using Aggregate & Sink transformation: 111](#_Toc158327181)

[Implementation of Dataflow with conditional split & Sink transformation: 115](#_Toc158327182)

[Implementation of Dataflow with Exists & Sink transformation: 117](#_Toc158327183)

[Implementation steps of Azure Dataflows for Exist & Sink transformation: 119](#_Toc158327184)

[Implementation steps of Azure Dataflows for Source, Join & Sink transformation: 120](#_Toc158327185)

[Derived Column: 122](#_Toc158327186)

[Implementation steps of Azure Dataflows for Derived column transformation with Source & Sink transformation: 122](#_Toc158327187)

[Implementation steps of Azure Dataflows to connect SQL DB with Source & Sink transformation: 124](#_Toc158327188)

[Union & Union All: 129](#_Toc158327189)

[Windows Functions: 129](#_Toc158327190)

[RANK() 129](#_Toc158327191)

[DENSE\_RANK() 129](#_Toc158327192)

[Row\_Number() 129](#_Toc158327193)

[Implementation of Dataflows with Window & Sink transformations 129](#_Toc158327194)

[What are the dis-advantages of using traditional frameworks: 133](#_Toc158327195)

[Apache Spark 134](#_Toc158327196)

[Apache Spark Architecture: 135](#_Toc158327197)

[Spark Components: 137](#_Toc158327198)

[Spark Core 137](#_Toc158327199)

[Spark SQL 137](#_Toc158327200)

[Spark Streaming 137](#_Toc158327201)

[MLlib 138](#_Toc158327202)

[GraphX 138](#_Toc158327203)

[Resilient Distributed Dataset(RDD): 138](#_Toc158327204)

[Azure Databricks: 139](#_Toc158327205)

[Features of Azure Databricks: 139](#_Toc158327206)

[Implementations steps for Azure Databricks: 140](#_Toc158327207)

[Use Photon Acceleration: 142](#_Toc158327208)

[We can create 2 types of clusters in Azure Databricks as mentioned below. 144](#_Toc158327209)

[(i)All-purpose clusters/Interactive based cluster: 144](#_Toc158327210)

[(ii)Job clusters/Instant clusters: 144](#_Toc158327211)

[Implementations steps to read .csv file from Python notebook in Azure Databricks cluster: 147](#_Toc158327212)

[Connecting to Blb SA from Azure Databricks cluster for mounting the directory: 148](#_Toc158327213)

[Reading .csv file from Blob SA with Windows Azure Storage Blob Service(WASBS) METHOD from Azure Databricks cluster: 149](#_Toc158327214)

[Pyspark: 152](#_Toc158327215)

[Importing Apache Spark libraries & writing the code in Databricks cluster in %Scala: 152](#_Toc158327216)

[Azure AZ copy: 153](#_Toc158327217)

[(i)Migration From Private cloud to Public cloud(Forward Migration): 153](#_Toc158327218)

[Doing the Concatenation (as shown below) 154](#_Toc158327219)

[Migration from Public cloud to Private cloud (Reverse Migration): 154](#_Toc158327220)

[Migration of Data from One storage Account to Another (Platform or Cloud to Cloud Migration) 155](#_Toc158327221)

[DataBase Migration Assistant (DMA): 155](#_Toc158327222)

[Assess on-premises SQL Server Instances migrating to Azure: 155](#_Toc158327223)

[Use the below script to change the collation of the DB in our local Server/SSMS: 156](#_Toc158327224)

[To change the collation of DB first convert it to to Single User Operations with below script: 156](#_Toc158327225)

[Again covnert it back to Multi User Operations with below script: 156](#_Toc158327226)

[Migration Process/Procedure of SQL DB/DB Objects from On-prem(Private cloud) to Azure (Public cloud) using Database Migration Asissitant(DMA) 156](#_Toc158327227)

[Steps for taking the DB Backup: 157](#_Toc158327228)

[Restoring Adventure Works DB in Private Cloud(local SSMS): 157](#_Toc158327229)

[Structured Query Language (SQL) Queries 158](#_Toc158327230)

[What are the components of a SQL system? 159](#_Toc158327231)

[Refer the link below to practice SQL queries: 159](#_Toc158327232)

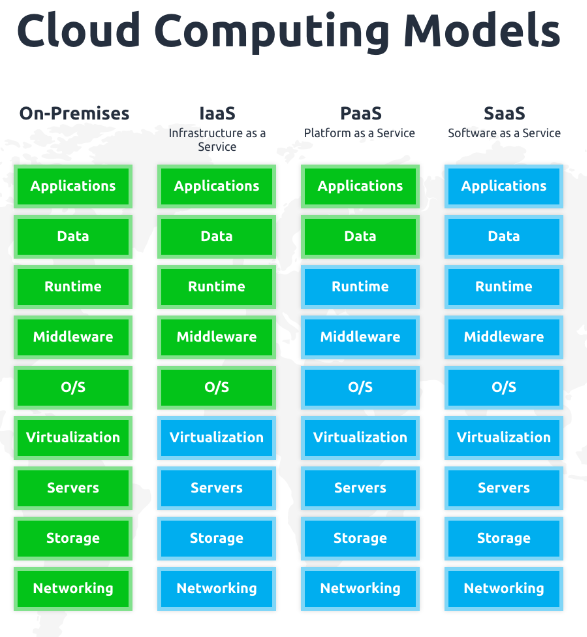
# **Understanding of Different Cloud Deployment Models:**

There are 3 ways that we can deploy our cloud application (Public, Private, Hybrid)

1. **Public Cloud:** Now the entire IT industry is moving and migrating their applications/data to Public cloud, we are having many cloud vendors (ex; Microsoft Azure, AWS, GCP, Oracle cloud, Salesforce, IBM…etc.) these cloud providers are offering their cloud computing platform with public cloud concept, till now the entire IT industry, the organizations/IT firms were having their own cloud (private cloud) and now the public cloud is available for all. **Public cloud means the platform is available, browsable and accessible for all, not our resources/services & data**. We must create an account/subscription in the vendor’s platform and then will get an identity & with authentication/authorization process will happen to login to our subscription in the cloud platform then it will allow us to create the resources/services, which all we need as per our application and project requirements.
2. **Private Cloud/On-premises:** Till now the entire IT industry (from decades), the IT firms/Organizations were having their own cloud and that is called a private cloud, the IT firms were maintaining these infrastructures /Datacenters on their own expenses and responsibilities, and this was **accessible only for them**. And they are responsible and accountable if some misshaped/disaster happened either natural disaster or manmade disaster have occurred.
3. **Hybrid Cloud:** It is a combination of Public and Private cloud. (Ex : let us say our frontend application servers are hosted in cloud as public and DB servers are in on-prem, now using VPN will connect from our back end server(i.e.. On-prem) to front end server (i.e. On cloud) this type of Setup/Scenarios is called Hybrid cloud)

# **Different cloud services/cloud service models:**

In cloud service model, the cloud providers/cloud vendors give it services in three different ways (i.e. IAAS, PAAS, SAAS)

* IAAS >>Infrastructure as a Service (Ex of IAAS is: VM, SA, Vnet, Azure Entra ID…etc.)
* PAAS >>Platform as a Service (Ex of PAAS is Azure SQL DB, Cosmo DB, Storage Account, ADL Gen2 SA, Logic Apps, ADF, Azure Databricks, Azure Synapse.……. etc.)
* SAAS >> Software as a Service (Ex of SAAS is: Skype, Gmail, FB, WhatsApp…etc.)

**What is Microsoft Azure:** Microsoft Azure is a cloud computing service is created and launched by Microsoft for building, testing, deploying, and managing applications (Business Applications) or applications data and services through **Microsoft-managed data centers.**

# **Azure Resources/Services:**

A public cloud computing platform, [Microsoft Azure](https://www.whizlabs.com/blog/introduction-to-microsoft-azure/) offers infrastructure as a service (IaaS), software as a service (SaaS), platform as a service (PaaS), and a server less model. A constant hybrid cloud, Microsoft Azure is growing in demand with approximately 90% of the Fortune 500 companies using Azure services.

The Azure cloud services are trained and created to deploy and manage even complex apps, through virtual infrastructure. It supports various programming languages, devices, databases, operating systems, and extensive frameworks. Therefore, Azure services intended for the professionals and enterprises offer all-around alternatives to the traditional means of organizational processes, with top Azure services greatly improving the performance.

(or)

Azure Resources/Services:

Anything that you are creating/deploying as part of your application need or project requirement is called as Resources or services.

|  |  |
| --- | --- |
| **Resources/Services** | **Category** |
| Storage Accounts(Standard & Premium) | IAAS |
| Data Lake Gen2 Storage Services | IAAS |
| SQL Database | IAAS/PAAS |
| SQL Servers | IAAS/PAAS |
| Azure Data Factory (ADF) | PAAS |
| Azure Storage Explorer (ASE) | PAAS |
| Az copy | PAAS |
| Azure Data Bricks | PAAS |
| Azure Synapse Analytics | PAAS |
| Azure Stream Analytics | PAAS |
| Azure File share servers | IAAS |

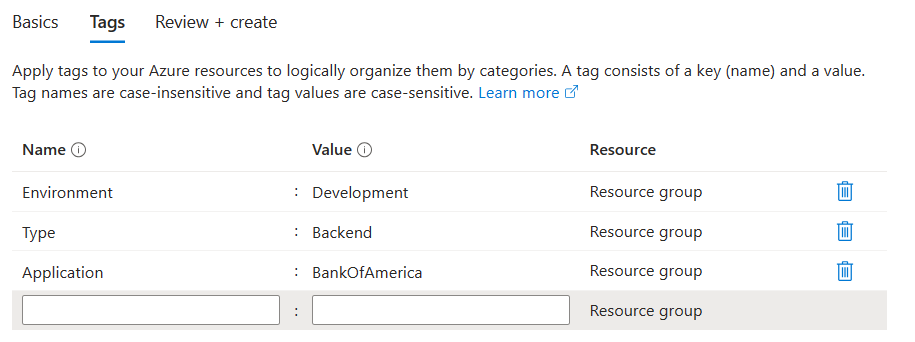
# **Azure Resource Groups (RG) & Configuration and management of Azure Resource groups for hosting Azure services:**

It is a place holder/name/folder basically which holds all our resources in azure. It is a logical container, which holds all our resources in Azure portal (or) in Azure resource manager, for each resource that you are creating in Azure **must and should** be in any of the resource groups. **We cannot create any resource in Azure without a resource group.**

**Why we create a Resource Group(RG):**

* RG's helps us to manage our resources in Azure cloud platform
* With RG we can logically divide the resources in Azure cloud platform.
* RG helps us to properly organized and manage the resources in the cloud platform.

**Tags Examples for RG:**

****

1. We can also move the resources from one resource group to another resource group (or) from one subscription to another subscription, but the tools & Scripts associated with moved resources will not work until we update them to use new resource ID’s.
2. If there are plenty of resources or big size resources in resource group then it might lead to a downtime (15-20 mins), so planning of moving resources should be planned in non-business hours.
3. If an operation of moving resources is already in-progress, then at the same time we cannot proceed to move further resources to move to the same resource group. We must wait until this operation has been completed.

# **What is Azure Storage Account/Azure Storage Services:**

Azure Storage is Microsoft's cloud storage solution for modern data storage scenarios.

(or)

Azure storage account contains all our storage data objects like blobs, files, queues & table storage account provides a unique namespace for our Storage data that is accessible from anywhere in the world over HTTP or HTTPS.

* Container/Blobs and File storage are the main concepts on which Azure Data Engineers works on.
* Table and Queue storage are used and worked by Developers.

Container/Blob (Binary large object): It is used to store binary large objects, In a blob we can store unstructured data and it is a part of storage service.

1. In Blob we have different types of storage. i.e.: (i)Page blob. (ii)Append blob (iii)Block blob.
2. Page blob: it is used to keep the VM disks, the data which we are using very frequently keep under page blob, its pricing is cheaper than block blob, basically we are storing here UN-structure data (ex: video files (2-3 hrs.), VM disks, DB files, Unstructured DB files.... etc.)

* Append blob: Used for logging purposes such as VM logs, diagnostics logs...etc.
* Block Blob: It gives us the URL access of the data which helps us to keep the data such as docs, videos, images, pdf’s…etc.
* Please refer to the link below to find the difference between the blobs.

D[ifferences Between Block Blobs, Append Blobs, and Page Blobs (smikar.com)](https://www.smikar.com/differences-between-blob-types/#:~:text=Block%20blobs%20are%20optimized%20for%20streaming%20and%20storing%20large%20amounts,random%20read%20and%20write%20operations)

* Storage account (SA) is just a name space (or) place holder once we have or created a Storage Account (SA) then we will get the access of Blob/Queue/Table/File share storages…etc.
* When we are creating a Blob & file in Storage Account then we create a container, which is nothing but a folder.
* Normally we upload .vhd files in page blob. When we create a page blob or block blob and if we go into it then will get the URL, this URL is private by default, and it won’t open in browser.
* When we create a SA then we basically fills the below details.

(i)Subscription (ii)Resource Group (iii) Storage account name (iv)Location (v)performance: standard & premium (vi)Account kind (vii)Replication (viii) Access tier

* Cluster technology is being used by MS in azure to make sure our data is being replicated, means if one server is down then in other regions/zones, the other servers are there as backup to provide the data to us.
* Depending on the requirement we can choose our offering and later on we can upgrade as well.
* Redundancy/Replication:

While creating a Storage account we have an option called replication and in replication we have 4-5 different types i.e.:

**(i)Locally Redundant Storage (LRS) >>**Here the data is going to kept in same region which we have selected. Here data is replicated 3 times and it gives SLA of 99.9 times 9. Here data gets replicated in the same region what we have choose. In LRS it will keep all the 3 copies of the data in the same region.

**(ii)zone redundant storage (ZRS) :** It gives an SLA of 99.12 times 9. It does the replication of data 3 times, and this replication could be in same region or different region

**(iii)Geo Redundant Storage (GRS) :** Here the data will get replicated 3 times in primary region and secondary region. It gives an SLA of 99.16 times 9.

**(iv)Read access geo redundant storage (RA-GRS) :** Here the data is replicated 3 times both in primary region and secondary region, but secondary region data is a ready only data.

* We never get to know in which regions or zones the data centers are located by MS.
* If we select Blob storage in account kind, then we won’t get the option of ZRS in replication.
* If we select standard in performance, then we won’t get the option of ZRS in replication and will get access tier as Cool or Hot
* Blob is more advanced than Files share/Table/Queue storages.
* Click on Next: Advance >>Secure transfer required : (i)disabled (ii)enabled
* If we select enable then we can’t upload/download the data using HTTP, but with HTTPS we can.
* If there is an HTTP connection means Non-secure and if its HTTPS connection means that is secured.
* Storage account(SA) will keep the data of our VM.
* **Virtual Network:** (i)all networks (ii)Selected networks.
* If we want to download/upload the data from all the networks then we can choose this option as All networks else if we want to restrict for any network then we can choose as selected networks.
* **Data Protection:** (i)Disable (ii)enable
* It allows us to recover our blob data to save when blobs and blob snapshots are deleted, if we override the blob data by any chance then by enabling this option it saves our data for specific time like (7 or 9 or 20 or 50 days based on our choice).
* When we are creating a folder/container in blob then we have 3 options
* **:** for public access level and i.e.: (i)Private (ii)Blob (iii)container.

**(i)Private** Only accessible to the owner/subscriber who has created the blob storage or accessible to the users whom the access has been granted by owner.

**(ii)Blob:** Means any person can read (blobs only, blobs means files only)blob is under the container, blob is just one file and container can have many blobs under it.

**(iii)container:** Anonymous read access for containers and blobs.

* If we open the link in browser then after we upload the files/docs in Storage account(SA) for BLOB then the link is somewhat like this.

<https://Practice11.blob.core.windows.net/manish/1st%20March(5).jpg>

Practice11 >> Storage account name

Blob.core.window.net >> blob storage

Manish >>folder name that we gave in the blob

1st March>>file name that we have uploaded.

**File Share Storage:** We create a file share and then we can help the users to map this file share with the team

* We can create a directory or folder in file share that we have created to any machine then it will give us an option to which o/s or m/c we want to connect like(windows, Linux, MacOS) and this file share will map to our m/c.
* The letter we choose for Drive then accordingly for every letter will get a different script to run in PowerShell to map the file share to our M/c or VM.
* To Map the file share to our VM we need a port No 443(this is an SMB port no or CIFS) this port No should be open in our environment by our internet service provider, if this port is open then only we can able to map it .
* In Azure VM this 445 port No is open or not, if its open then we can map our file share to any VM
* When we are working with files under the SA and when we click on files, here we have to create a +Fileshare and this is nothing but a folder/directory how we have in blob and under this FileShare we can place/upload files and folders what all we want.

<https://practice11.file.core.windows.net/shared11/wasay%20files/azure%20admin%20job%20description>

practice11 >>Storage account name

file.core.windows.net >> file storage

shared11 >>file share name

Wasif files >> folder name

Azure admin job description >> file name that we have uploaded under wasif files folder.

* While creating a SA if we create account kind as Blob Storage then in performance, we can find only Standard as enabled option and premium will get disappeared.
* While creating a SA if we select performance as standard and account kind as Blob storage then after deploying the SA we can only see Blobs under Storage Account means we can only see blobs service that’s it.
* While creating a SA if we select Account kind as General purpose then we can see all kind of storage services(like containers, Files, Table & Queue storage)
* Azure storage has 4 separate storage offerings (i.e: Containers, Files, Table & Queue) in which Blob is one of the storages offering, Blob is the most used by infrastructure specialist, platform eggs and Azure admins.

# The Difference: Azure Blob Storage vs File share Storage vs Disk Storage:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| | **Storage Category** | **Azure Blob Storage** | **Azure File Storage** | **Azure Disk Storage** | | --- | --- | --- | --- | | **Object Storage** | Primarily used for storing unstructured data such as text, images, videos, and backups | Not suitable for storing unstructured data; mainly used for sharing files across multiple machines in a distributed environment | Not suitable for storing  unstructured data; mainly  used for hosting VM disks and persistent storage for Kubernetes | | **Durability and Availability** | Provides high durability and availability of data with built-in redundancy and replication across multiple regions | Provides durability and availability of data with built-in redundancy across multiple nodes in the same region | Provides durability and  availability of data with  built-in redundancy within the same availability set or availability zone | | **Performance** | Provides high throughput and scalability for read-intensive workloads | Provides low latency and high throughput for read/write operations on small files | Provides low latency and  high throughput for  read/write operations on  large disks | | **Cost** | Cost-effective for storing large amounts of data that is not accessed frequently.  Cost depends on the amount of data stored per month, the number and type of operations performed on the data, and the data transfer costs. Blob Storage also offers different tiers of storage for different access frequencies and performance needs:**hot, cool, and archive**. | Cost-effective for sharing files across multiple machines; may be more expensive than Blob storage for large amounts of data,  Cost depends on whether you’re deploying [premium](https://learn.microsoft.com/en-us/azure/storage/files/understanding-billing#provisioned-model)  or [standard](https://learn.microsoft.com/en-us/azure/storage/files/understanding-billing#pay-as-you-go-model) file shares. | More expensive than Blob  and File storage; suitable  for performance-critical  workloads.  Pricing depends on the size  of the disks per month, the  number of transactions  performed on the disks, and  the data transfer costs. Disk  Storage also offers different  types of disks for different  performance and cost  requirements: ultra-disks,  premium SSDs, standard  SSDs, and standard HDDs. | | **Disaster recovery capability** | Supports geo-redundant storage (GRS) or read-access geo-redundant storage (RA-GRS), which replicates your data to a secondary region that is hundreds of miles away from the primary region. You can also initiate a storage account failover to switch the secondary endpoint to the primary endpoint in case of a regional outage. | Supports GRS or RA-GRS for disaster recovery. However, you cannot initiate a storage account failover for file shares. Instead, you have to use Azure Backup or Azure Site Recovery to protect your file shares from regional disasters. | Uses GRS or RA-GRS to  replicate your disks to a  secondary region. You can  also initiate a storage  account failover for disks.  Additionally, you can use  Azure Backup or Azure  Site Recovery to back up  and restore your disks in  case of a disaster. | | **Integration** | Provides integration with various Azure services such as Azure Functions, Logic Apps, Azure Stream Analytics, and Azure Data Factory. This makes it easy to build data processing pipelines in the cloud. | Provides integration with various Azure services such as Azure VMs and Azure Kubernetes Service and on-premise applications. Easy to migrate on-premises applications to the cloud and share files across different platforms. | Provides integration with  Azure VMs and Azure  Kubernetes Service.  This makes it easy to  attach high-performance  storage to VMs running on  Azure. | | **Reliability** | Data stores in Blob Storage are highly durable and reliable with very low chances of data loss.  Blob storage automatically creates multiple replicas of your data, and each replica is stored in a different storage scale unit. | The data stored in File Storage are highly durable and reliable with very low chances of data loss. The service automatically stores multiple replicas of your data in different storage scale units to ensure data durability. | Data stores in Disk Storage  are not durable and reliable  as compared to Blob Storage  and File Storage. Azure Disk  Storage provides durability  based on the type of disk you  choose. For example: Premium SSD disks offer an  SLA of 99.9% and provide  durability by replicating your  data within the same data  centre. Standard SSD disks and  Standard HDD disks offer an  SLA of 99.5% and store three  replicas of your data within  the same region, but across  different fault domains. | |  |  |  |

**Azure Storage Explorer(ASE):**

Azure storage explorer is an additional feature that will allow us to access or perform the operations of Storage Account (SA) content (like Blob/Files/Table/Queues). We can download and install this in our VM, we can install this software on any O/s, and using this tool, we can access/configure/manage our SA content.

This is an additional software which helps us to manage the azure storage, instead of giving Azure subscription access for each and every user, with this tool we can perform the Upload/download which ultimately effects or stores the data in our storage account, it is bit easy for us to manage our storage content from Azure storage explorer.

**Key Features of Azure Storage Explorer Software:**

* it is easy to manage the data from Azure Storage Explorer(ASE)
* this software is available at free of cost
* this software will work on any O/S
* this software is very user friendly even for non-tech people
* this software also helps us to provide the security to our storage account data(ex: files, videos, images, folders….etc)
* In ASE we can see all the storage accounts that we have created like containers, files…etc. everything basically. Will get login into ASE with our Azure subscription details (like how we login to Azure portal)
* If we login-in to ASE, then only we can see or access our SA content in ASE.
* When we click on Add an account in ASE then will get multiple option to login in to our ASE…the different options we have to login-in to our ASE are as follows

**(i)Add an Azure account (ii)Use a connection string (iii) (iv)Use a Storage account name and key (iv) use a shared access signature**

**(i)Add and azure account:** Here we pass our azure subscription creds which is a global admin access. After login-in we can perform many operations in ASE Ex: we can upload/download/Open/new folder/Rename/Delete/create snapshot…. etc.…if we are not able to see any of the SA in ASE then it means we don’t have access for that particular SA, but with Global admin access creds we can see all the contents of SA and all SAs in ASE.

1. In ASE we can perform operations for all offerings of SA like Blob/File/Table/Queue storages.
2. Connecting to multiple SA in ASE is possible.

**(ii)Use a Storage Account Name & Key:** To login to ASE, we pass the SA name (exactly same) and key including display name, this key value will get from Access Keys (left side in settings tab, after getting into SA). By login-in with this option we have to pass

Display name (it can be anything we can give) SA name (It should be exactly the same)

Account key (have to pass the key value exactly the same), if the key has been shared by global admin to users in Europe/USA…etc., still they can able to connect to ASE & can access the SA content

1. **If we connect to ASE using above options, then we can able to see only the given storage account content not all the SA contents (very important to remember this point)**
2. The user who got login-in by passing the key value will have full right to delete/add/modify the data into the SA, after we delete any files/folders from blob there is a retention period we will keep in recycle bin for about 7/15/30/60 days as per our choice.
3. Once we share this key with anyone in the world then they will get the complete access of our SA and if we believe they are not in our organization anymore & we don’t want them to access or authorize our SA content then just if we refresh the key then the old key value will get expired, but yes again we have to share this new key after refresh with all the users across the globe
4. If we want to remove the SA that we have logged in to ASE, then simply select the SA>>right click >>De-attach
5. If we login-in to ASE for one SA by passing the Account name and key and if we pass the other account details like name and key then we can see the other SA details as well…. likewise goes on, we can see as many SA details we can.
6. **(iii) Use a connection String:** Here we give the display name as we want and pass the string value from SA Access key. If we see in Access key we have 2 keys values and 2 connection string values why we have 2 ??
7. If is for backup (or) we have a key1 & key2 , if we refresh the key1 then the access will gone, so what we can do is we can give key1 to priority level user, & key2 to less prioritize user, & if we experience the key has been compromised either key1 or key2 , so whatever the key has been compromise we can refresh it, so that the other users are not impacted, we can categorized the users and can share the keys.
8. When we are uploading a file in ASE will get an option under Blob type saying which blob we can store this file as Block blob/page blob/append blob.

# **(iv) Shared Access Signature (SAS):**

1. With this feature we can grant access to the user for our SA to a specific period of time, when we grant access to users using SA’s then that access is a restricted access(means with some limitations not like connection string (or) SA name & key)
2. **Use case of SAS:**

(i) if we want to give file share access to some ABC set of users and don’t want to give file share access to some set of users BCD in SA, then we go with SAS concept.

(ii) if we have some users who are going to be in our team for 3 months then we can go with this concept for a specific period of time.

1. When we are setting SAS(left side) under SA then we get below options to

Allowed services >> (i)Blob (ii)File (iii)Queue (iv)Table

Allowed Resource Type >> (i)service (ii) container (iii) object.

Allowed permissions >>(i) Read (ii)Write (iii)Delete (iv)List (v)Add (vi)Create (vii)Update…etc.

Start & Expiry Date/time >> Here we can give 1 day or 1 hr even

Allowed protocols >> (i)HTTP (ii)HTTPS…etc.

1. If we are selecting only blob on top then only we get the Blob services in SAS URL.

# **Azure Table Storage Services:**

Azure Table storage is a service that stores non-relational structured data in cloud computing, providing a key/attribute store with a schema less design. Because Table storage is schema less, it's easy to adapt our data as the needs of our application evolve. Access to Table storage data is fast and cost-effective for many types of applications and is typically lower in cost than traditional SQL for similar volumes of data.

We can use Table storage to store flexible datasets like user data for web applications, address books, device information, or other types of metadata our service requires. We can store any number of entities in a table, and a storage account may contain any number of tables, up to the capacity limit of the storage account.

# **What is Table storage:**

Azure Table storage stores large amounts of structured data. The service is a NoSQL datastore which accepts authenticated calls from inside and outside of Azure cloud. Azure tables are ideal for storing structured, non-relational data. Common uses of Table storage include:

* Storing TBs of structured data capable of serving web scale applications
* Storing datasets that don't require complex joins, foreign keys, or stored procedures and can be denormalized for fast access
* Quickly querying data using a clustered index
* Accessing data using the OData protocol and LINQ queries with WCF Data Service .NET Libraries
* We can use Table storage to store and query huge sets of structured, non-relational data, and your tables will scale as demand increases.
* When we are storing the data in tables storage services in our cloud storage account then those table are non-relational that tables are not having any connection with each other.
* These tables act like a separate/independent entity

# **Implementation of Table Storage Services:**

**Step 1:** Will create a Storage Account first then will create a table storage service inside a Storage Account

**Step 2:** Open Azure storage Explorer and connect to the respective Storage Account

**Step 3:** Expand tables>>double click the table we created in our Storage Account>>+Add>>Pass value to Partition Key and Row Key (these are the default columns will get created in table storage service in Azure Storage Account)>>

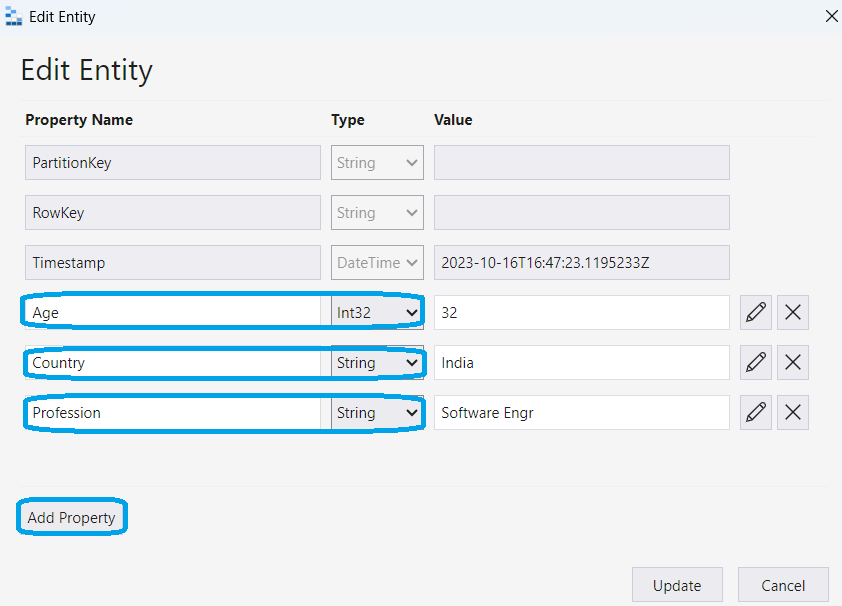
Pass the values for Partition Key and Row Key as an example below

Partition Key: Rahul

Row Key: Mehra

**Step 4:** Click on Add Property>>Pass the Property Name: City; Datatype: string and Value: Hyderabad

**Step 5:** Click on Insert finally

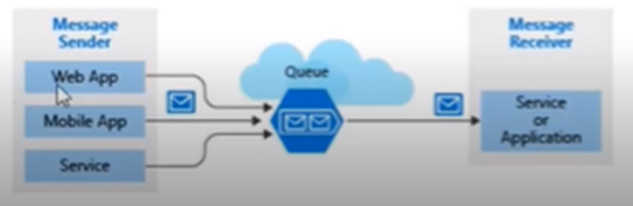


Like this we can explore many other options in Azure Storage Explorer….like Query/Import/Export/Add/Edit/Select All/Column options/Delete/Refresh…etc.

**Queue Storage Services:**

A single queue can be upto 5TB in size, so it can potentially store millions of messages, the target throughput for single queue is 2000 messages per second which allows it to handle high volume scenarios.

Messages in Storage queue are typically first-in-first-out, A message in Queue storage service can be in any format of up to 64KB and the maximum time that a message can remain in the queue is up to 7 days



Each queue maintains an inventory of messages, Queue is a collection of messages to read from any application from the cloud, through any rest API or any azure supplied client library we can consume the queues.

A diagram of a storage queue

Description automatically generated

* If we click on Deque then it will delete the messages in our queue one by one and with first in first out pattern.
* If we click on clear queue, then it will delete all the messages from our queue in one go.
* We can also access and browse this messages and queues in Azure storage explorer.
* We can also set the messages as never expire inside the queue.

**Soft Delete:** Azure Files share storage services and Blob storage services offers soft delete so that we can more easily recover our data when it is mistakenly deleted by an application or other storage account user.

Soft delete allows us to recover our Files share storage services and Blob storage services in case of accidental deletes

# How soft delete works

When soft delete for Azure file share & Blob storage is enabled, if a file share or Blob storage is deleted, it transitions to a soft deleted state instead of being permanently erased. You can configure the amount of time soft deleted data is recoverable before it's permanently deleted and undelete the share anytime during this retention period. After being undeleted, the share and all of contents, including snapshots, will be restored to the state it was in prior to deletion. Soft delete only works on a file share level - individual files that are deleted will still be permanently erased.

Soft delete can be enabled on either new or existing file shares. Soft delete is also backwards compatible, so you don't have to make any changes to your applications to take advantage of the protections of soft delete.

To permanently delete a file share in a soft delete state before its expiry time, you must undelete the share, disable soft delete, and then delete the share again. Then you should re-enable soft delete, since any other file shares in that storage account will be vulnerable to accidental deletion while soft delete is off.

For soft-deleted premium file shares, the file share quota (the provisioned size of a file share) is used in the total storage account quota calculation until the soft-deleted share expiry date, when the share is fully deleted.

# Enabling or disabling soft delete:

Soft delete for file shares is enabled at the storage account level, because of this, the soft delete settings apply to all file shares within a storage account. Soft delete is enabled by default for new storage accounts and can be disabled or enabled at any time. Soft delete is not automatically enabled for existing storage accounts unless [Azure file share backup](https://docs.microsoft.com/en-us/azure/backup/azure-file-share-backup-overview) was configured for a Azure file share in that storage account. If Azure file share backup was configured, then soft delete for Azure file shares are automatically enabled on that share's storage account.

If you enable soft delete for file shares, delete some file shares, and then disable soft delete, if the shares were saved in that period you can still access and recover those file shares. When you enable soft delete, you also need to configure the retention period.

### Retention period:

The retention period is the amount of time that soft deleted file shares are stored and available for recovery. For file shares that are explicitly deleted, the retention period clock starts when the data is deleted. Currently you can specify a retention period between 1 and 365 days. You can change the soft delete retention period at any time. An updated retention period will only apply to shares deleted after the retention period has been updated. Shares deleted before the retention period update will expire based on the retention period that was configured when that data was deleted.

* In Blob Storage services we can retain the container or the individual files
* But in File share storage services we can retain the file share but we cannot retain the individual files

# Implementation Steps of Soft Delete:

1. Create a Storage Account (When creating a SA soft delete will be automatically enabled this we can see in Data protection tab while creating the SA, we can also change No of retain days for the file share/blob storage)
2. Create a 2-file share and upload a file inside the file share.
3. Delete one of the file share by clicking on 3 dots(extreme right) and click on Delete share>>check the check box I agree to the deletion of my file share….and finally click on Delete
4. Now if we click on Refresh under file share then will see there is only one file share available.
5. Click on Show deleted shares then here we see the deleted file share also.
6. Now click on Deleted file share and click on three dots (extreme right) and click on Undeleted.

**Note:** soft delete is enabled/workable on file share **not on inside the file share files** that we upload, if we deleted a file that we have uploaded inside the file share then that file cannot be retain back.

**Business case scenario 1:** How to delete a file share permanently when the soft delete is enabled

If we want to delete a file share permanently when a file share is enabled then we have to disable the soft delete first then we can delete the file share and then this time the file share will be deleted permanently.

**Implementation of above point:**

1)Click on Soft delete: 7 days and choose Soft delete for all file shares as Disabled and finally click on Save button(@ bottom)

2)Now if you delete a file share then this will be deleted permanently bcoz the soft delete is disabled and if we click on Show deleted shares it will not show the deleted file share bcoz this time the file share got deleted permanently.

**Business case scenario 2:** How to delete a file share/container in a soft delete state before its expiry time.

Step1: Firstly delete the share (before doing this step1 ensure the soft delete is enabled) and click on Show deleted shares

Step2: Undelete the file share which we have deleted

Step2: disable soft delete

Step3: Delete the same file share again

Step4: Now if we click on Show deleted shares then now we won’t see the file share which we have deleted.

Hence, we can also delete a file share in a soft delete state before its expiry time.

# Azure Data Lake Storage Gen2 Storage Accounts:

Azure Data Lake Storage Gen2 (ADLS Gen2) is a cloud-based repository/Storage account for both structured and unstructured data. For example, we could use it to store everything from documents to images to social media streams. Data Lake Storage Gen2 is built on top of Blob Storage. This gives us the best of both worlds.

Azure Blob Storage is one of the most common Azure storage types. It's an object storage service for workloads that need high-capacity storage. Azure Data Lake is a storage service intended primarily for big data analytics workloads.

Azure Data Lake Gen1 is a storage service that's optimized for big data analytics workloads. Its hierarchical file system can store machine-learning data, including log files, as well as interactive streaming analytics. It is performance-tuned to run large-scale analytics systems that require massive throughput and bandwidth to query and analyse large amounts of data.

Azure Data Lake Gen2 converges the features and capabilities of Data Lake Gen1 with Blob Storage. It inherits the file system semantics, file-level security and scaling features of Gen1 and builds them on Blob Storage. This results in a low-cost, tiered-access, high-security and high availability big data storage option.

# Benefits and challenges of Azure Blob vs. Data Lake storage

Azure blobs are a durable storage option, with appropriate redundancy options to keep data safe. All data is encrypted, and there is fine-grained access control. Azure blobs are also massively scalable for text and binary data.

# Azure Blob Storage and Data Lake are well suited to specific situations and uses.

One challenge of Azure blobs is when customers use it, they can incur lots of data transfer charges. Along with the typical data transfer read/write charges at the [various tiers](https://www.techtarget.com/searchstorage/tip/Choose-the-best-Azure-Blob-Storage-tier) -- Premium, Hot, Cool and Archive -- there are iterative read/write operation charges, indexing charges, SSH FTP transfers, fees for data transfers for geo replicated data and more. Each transfer type may only cost fractions of cents, but when doing hundreds of thousands of transactions, these costs can add up quickly.

Azure Data Lake enables users to store and analyze petabytes (PB) of data quickly and efficiently. It centralizes data storage, encrypts all data and offers role-based access control. Because Data Lake storage is highly customizable, it is economical. Users can independently scale storage and computing services and use object-level tiering to optimize costs.

# Steps to Improvise & configure Azure Storage Accounts Authentication & Authorization

1) Login to Azure portal and create a storage account and a blob storage service inside the storage account.

2) Provide the reader access to resource group in which the storage account is present to the user.

3) Then provide reader access to Storage account

4) And then login with that user credentials again and you should be able to see the storage account

5) Up to here you can see a storage account with container inside in it and if we try to upload any of the file then will get an error saying don’t have permissions

6) Now come inside the storage account>>go inside the container>>and on left side click on Access control (IAM) to provide access to this storage container>>+Add>>Add role assignment>>search for storage account contributor access>>select it>>Next>>+Select member>>Next>>Review+assign

7)Wait for some 10-15 mins and login with the user credential whom we have provided the access, then here we can see the user can see the storage account and container in it and can upload the files and folders as per the access provided.

**Note:** To know the user has got what kind of access on Storage account (or) container then go inside the storage account or a container then click on Access control(IAM) and in Find type the user name whom we gave the access and click the user.

# **SQL DB as service in Azure:**

Basically Azure gives us two options to run SQL Server workloads.

**PAAS:** Azure SQL Database(DBAAS):: When are deploying SQL DB directly in Azure cloud platform then it will be considered as PAAS

**IAAS:** SQL Server on Azure VM’s i.e. SQL Server inside fully managed VM.:: When we are deploying SQL DB in an Azure VM in Azure cloud platform then it will be considered as IAAS

**Azure SQL Database:** Azure SQL DB is a cloud based relational database service that is built on SQL Server Technologies, it supports T-SQL commands, tables, indexes, views, primary keys, store procedures, triggers, roles, functions…etc.

SQL Database delivers predictable performance, scalability with no downtime, business continuity and data protection with almost zero administration, with which we can focus rapid app development and accelerating our time to market rather than managing virtual machines and infrastructure as it is based on SQL server engine, SQL DB supports existing SQL server tools, libraries and API’s which makes it easier for us to move and extend to cloud.

# Benefits of SQL DB as Service:

* High availability: For each SQL DB created on Azure, there are three replicas/backups of that Database.
* On Demand: One can quickly provision the DB when needed with a few mouse clicks.
* Reduced Management Overhead: It allows you to extent your business applications into the cloud by building on core SQL server functionality.

# SQL Database Deployment options:

* **(i)Single Database:** it is an isolated single DB, it has its own guaranteed compute, memory, and storage.
* **(ii)Elastic pool:** Collection of single DB’s with fixed set of resources such CPU. Memory shared by all DB’s in pool.
* **(iii)Managed Instances:** Set of Databases which can be used together.

# Azure SQL Database Purchasing Model:

There are two purchasing models or service tier **DTU & vCore**

1. **vCore:**

vCore are virtual core provide higher compute, memory, and storage limits and gives us the great control over the compute and storage resources that we create and pay for.

1. **Database Transaction Unit(DTU):**

DTU stands for Database Transaction Unit and it is a combined measure of compute, storage, & IO resources. This DTU based model is not supported for managed instance.

In Azure SQL DB’s are available in two purchasing models DTU & vCore. SQL Databases is available in

(i)Basic,

(ii)Standard/General Purpose,

(iii)Premium (Business critical & Hyper scale service tiers) each service tier offers different level of performance and capabilities to support lightweight to heavyweight database workloads, we can build our first app on a small database for few months and then we can change the service tier manually or programmatically at any time based upon our convenience without any downtime to our apps and customers.

# **Implementation Steps of SQL DB as PAAS:**

Search for SQL server in Azure portal>>Create and deploy the SQL server in Azure.

If we click on SQL databases (left side) then will see no DB’s available inside this SQL server, now will provision a new DB in Azure portal.

Search for SQL DB in azure portal>>create>>fill the details accordingly and create a new DB in azure portal.

Goto SQL servers (that we have created)>>Backups (left side)>>and here will see the DB backup is already available for us in azure portal which is been taken by Azure and from here we can Restore this DB if required based upon the need.

On the extreme right it is giving us an option to restore the DB backup

In Retention policies tab we can see that the Long term retention (LTR) that we can set for weeks, months and years, select the DB and click on configure policies (if we want to change the configure policies) and here we can change

1)Point-in-time-restore to max of 35 days.

2)Take a differential backup every 12 hrs or 24 hrs.

3)Weekly LTR Backups

4)Monthly LTR Backups

5)Yearly LTR Backups

After making the changes and finally click on Apply and then yes.

After connecting to Cloud DB in our Laptop we can create and insert the data into table of cloud DB to check, the sample queries attached below

# Advantages of Cloud Computing:

* **24/7 availability & accessibility:** It is available & accessible from anywhere & at any time we only need to have device and internet connectivity to access the Azure resources/services.
* **Scalability:** The face of transformation is very simple and ease in cloud computing (ex; resizing of resources/workloads/services…etc.) and that can be done just with click of a mouse.
* **Security:** it is using a very high security algorithms and Hash functions to protect our data & resources.
* **Enhanced collaboration:** Just in one platform (portal.azure.com) you are going to get all the resources/services that what you need for your project or application requirement.
* **Cost effective**: It is very much cheap an economical as compared to private cloud or an old legacy system.
* **Reliable:** Cloud services are consistently good in quality with equal performance even if we perform multiple enhancements on them.

# Advantages/Features of Azure Cloud Computing:

* 1)As It is a product of Microsoft as Microsoft has launched many frameworks, tools, IDE’s, languages for the applications development and all the applications are doing great business from decades, hence clients in the market has got that faith & trust saying Microsoft products are reliable, reasonable, efficient and even economical for software applications development.
* 2)Compare to AWS the learning/working curve of Microsoft Azure Cloud Computing is small. Azure is easy to work, easy to learn, easy to manage, there is no such pre-requisites required to learn Azure, no programming language understanding is required to work in Azure cloud computing.
* 3)Azure is cheap as compare to other cloud providers (4-12%)
* 4)If you are making Azure as your cloud computing partner then it is offering you MS office, WPS office, Lync, skype, share point...etc. and other platform available at cheaper cost which ultimately needed for our applications/project developments.
* 5)As compared with other cloud providers Azure is offering you many regions/places to deploy/provisioning/creating your resources (VM's, SA's, DB's, Vnet, NSG’s, Backup’s...etc.)for our software applications.
* 6)Azure is using a very high security algorithms and Hash functions to protect your data and resources what all is been provisioned in different regions.
* 7)Azure is providing default encryption for all your services that you are provisioning in cloud computing platform, with which it is not at all easy for any ethical hacker to hack / hijack the resources which are hosted in Azure Cloud Computing Platform.

# Key points of Azure Data Engineering:

* If we see these days the data volume is very high bcoz the data is getting generated from variety of sources like laptops, smart phones, streaming platforms, printers, sensors, social media platforms, Medical records, banking transactions, reels, memes, videos, posts, check-in & check outs, pod casts, audio and video platforms (ex: Spotify, Netflix. Prime videos…etc.).
* It could be a structure data or unstructured data or may be semi structured data from different domains like health care domain, retail domain, energy domain, banking domain, manufacturing domain…etc., even lot of lot of data is generating from social media platforms (like: WhatsApp, twitter, FB, Insta, snapchat, Vchat, TikTok, telegram…. etc. etc.)
* Being an Azure Data Eng. we must collect the data from variety of sources, we must load the data into different cloud services, we have to do certain transformations, we have to prepare the data for the customers as per the needs and finally load the data into cloud storage services, or cloud Databases or cloud file share storage services…etc.
* To process and segregate the data we have to make use of variety of different Azure resources in Azure cloud platforms, which offers us 200+ resources/services. Here some services which are relevant for Azure Data Engineers which helps us to process, load and transform the data into different targets.
* Here will discuss each service with architecture with a detail explanation from basic to advance level.
* Now a day’s data is considering as a global/universal currency which will be important and matters to any country or company/firms as it adds a value to the business.
* When we start a new business **ex: Restaurant** and we are offering variety of different menus to our customers for Breakfast, lunch, dinners, snacks. And to setup the restaurant we need a lot of infrastructure, like building place, tables, chairs, menus, Kitchen crockery, manpower all these are required to setup the business, hence lot of cost is involved and when the business is success again, we have to increase/expand the business.
* To grow the business again like to open a new branch in another city or place then we need lot of data for the analysis, research to take a final decision for the new branch.
* If we purchase a smart phone from Flipkart or Amazon first they will take our complete data(by creating an account in their platform) whether we purchase it or not, and now here you will search the phone you want, and for this phone we can see now the ratings, feedbacks, reviews of that phone and this is all nothing but data which is help us to take the decision to purchase this phone or not, like this way its helps the other customers to decide whether to purchase this product or not, even after delivery it sends you mail and reminders to provide the ratings and feedback about the product(this is also a type of data that helps us and other customers too)
* Hence in this way data is a key role now a days and in future also there are plenty of opportunities that will get, and the volume of data is growing enormously these days.
* If we observed previously, we were having data in KB, MB & GB, but now all the business applications or Enterprise applications are having data in Gigabytes(GB) and Terabytes(TB), bud down the line after some years we may expect these data grows to

(i)PB>>Petabyte

(ii)EB>>Exabyte

(iii)ZB>>Zettabyte

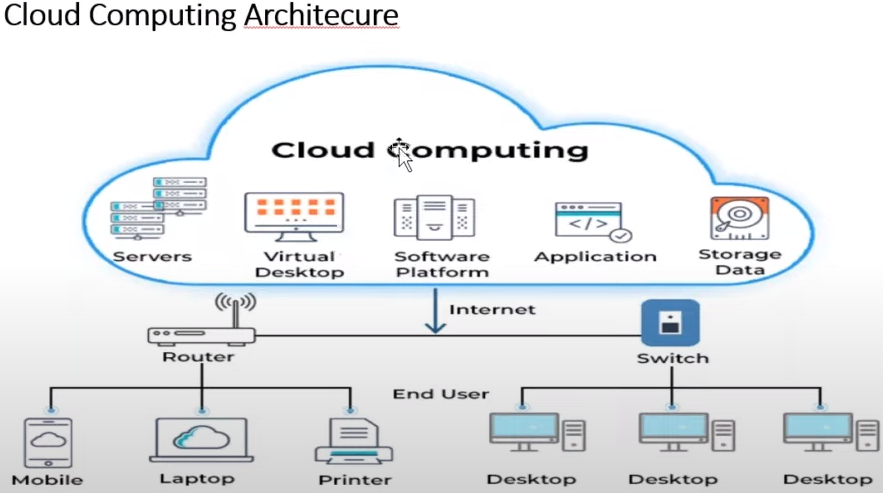
(iv)YB>>Yottabyte

(v)BB>>Brontobyte

(vi)GB>>Geopbyte

* We may also experience the Enterprise applications may contain, structured, semi structure and un-structured data and to process, transform, execute and load this data Azure cloud computing is offering a variety of different services for Azure Data Engineers…like (i)Blb SA, (ii)ADL Gen2 SA, (iii)MS-SQL DB, (iv)Cosmos DB, (v)ADF, (vi)Azure Data bricks, (vii)FTP Servers, (viii)Json, (ix)GitHub portal, (x)Azure Devops portal…. etc.…. etc.
* We are processing, segregating and executing this data to add a value to the Business…like if you are doing a frequent shopping on Flipkart or Amazon…then they will keep sending you new products launch, attractive discounts on special events, similar products that you have searched in previous records, you have just done the search and left the shopping in between then they will keep all the records of your browsing history and interest and they keep sending us the promotion events and messages.
* If the sales of the products are decreasing quarter wise or week wise then by capturing all the data and records we can do the analysis on top of the data and we can realize why the sales got decrease/increase whether the advertisements was less, or the sales team was on leave, or the market was down, any recession occurred…etc.
* These days if you are a first time customer like you are using Zomato, swiggy or redubs or abhibus mobile apps they are giving 50% discounts for first timers, with this they are attracting the customers and getting all our information and also doing advertisements for their business applications and a value add, this all can be possible if we collect the data from the customers on their visit to our web application.
* These data will collect from the variety of sources and load the data into cloud computing by doing some transformations, analysis on top of the data and load finally to some SA and from these the Data scientist will use the power BI reports, Tableau, and different visualization tools that are available in the market.
* We received the data from many sources and that data could be structured, semi-structured or unstructured data to process, transform and load these types of data Azure offers us variety of different activities, controls and data flows as part of Azure Data engineering.
* We can save cost almost 6 times if we process, extract, transform and load the data via Azure Cloud Resources and storage into Azure Storage services and with this we can improve the performance, productivity, time, manpower, cost…etc. with cloud computing.
* There are many tools in the market, like SSIS, Informatica, Oracle BI…etc. but how benefit and different the cloud Data Engineering resources when it compares to traditional data transformation tools (SA, Gen2 SA, ADF, Data bricks, SQL DB’s in cloud…etc.).
* Now many organizations are moving their jobs and ETL tools from SSIS, Informatica to Azure Data Engineering resources and services offered by Azure cloud computing

# Architecture of Azure Cloud Computing:

****

* Whatever the services that we are using as Azure Data Engineer, Microsoft is offering the best security standards for it, and ADF/Azure Data Bricks is an in-built or a native tool to Microsoft Azure with this the same security standard is provided to all the 200+ services in Microsoft Azure cloud computing.
* With these Azure Data Engineering services, we can move our personal data, structured data, semi structured data, unstructured data, data in any format can be processed, extract, transform and load into different destination using Azure Data Engineering services.
* When the data is in rest mode (means loaded into the SA) or while the data is in transit mode (means moving from one place to another) the data is encrypted by default from Azure cloud computing end.
* Why many businesses are using Azure Data Engineering services is bcoz it’s providing the best of

(i)Cost.

(ii)Productivity.

(iii)Performance.

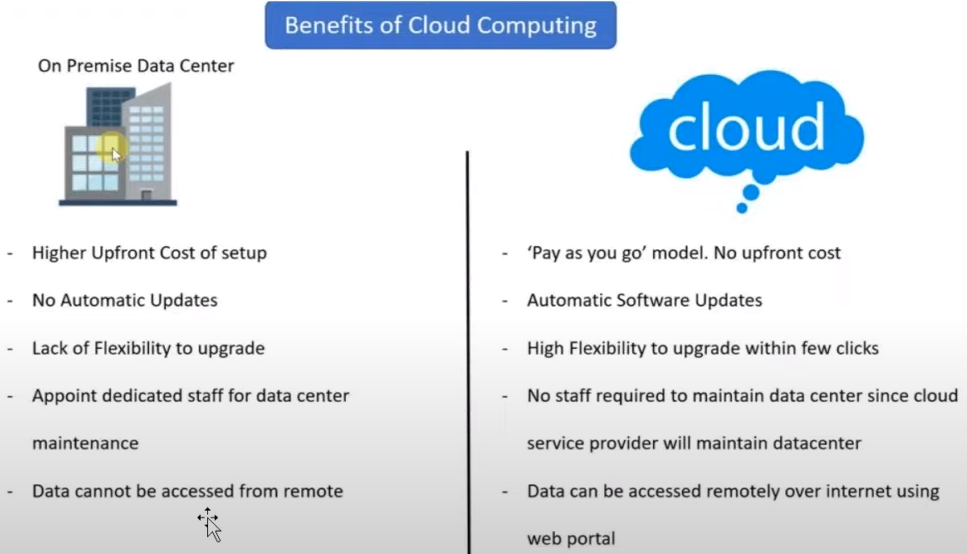
(iv)Efficiency.

(v)Security(encryption).

(vi)Reliability.

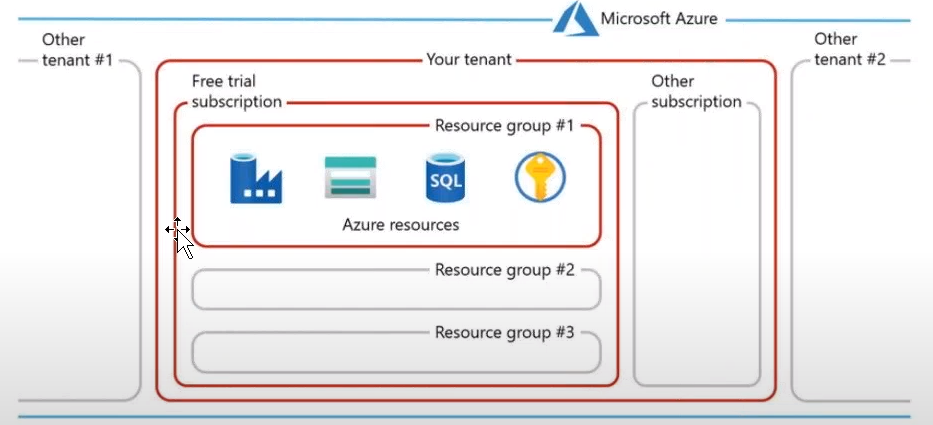
(vii)Easy to use.

(viii)native to many source and destination platforms…. etc. etc.



* Azure Data Engineering services are very much cost effective when it comes to the comparison of other tools available in the market and even bcoz of this also all the tech firms and clients are preferring to go for Azure Data Engineering services to process, extract, transform and load the data.
* If we see one of important services of Azure Data Engineering i.e.: Azure Data Factory(ADF) as this is server less infrastructure as we no need to worry about the underlying infrastructure everything will be taken care by Microsoft, it is offered as a PAAS Service, SSIS, Informatica, Data Stage, all these ETL jobs are getting migrated to Azure Data Factory.
* With Microsoft(Msft) Azure we only pay what we need, lets us say for an example we have created an ADF to move the data from source to target then we have to pay only for the processing time which has taken to move the data from source to target…. ex: if there is 100GB of data and if want to move to cloud from on-prem and if it is taking some 20 mins of time to load this data then I am ending paying only for 20 mins to Azure cloud computing (i.e.: to Msft)
* Azure Data Engineering services are feasible to access, process and load the data from any resource and from any region and even at any time…there is no need for us to create a VPN for security point of view bcoz Msft is providing the default encryption for all of its resources that we are using in Azure cloud computing, we just need an access (linked service) to connect to the different sources

# Hierarchy of Microsoft Azure Cloud Platform:

****

* We work and manage different clients in Azure cloud computing and each client have different business functions like supply, trading, logistics & transports, finance, HR, sales and product management.
* In one tenant we can have multiple subscriptions and here tenant represents as client, and

(i)we can have separate subscription for HR,

(ii)separate subscription for Finance

(iii)separate subscription for Logistics and transports…etc

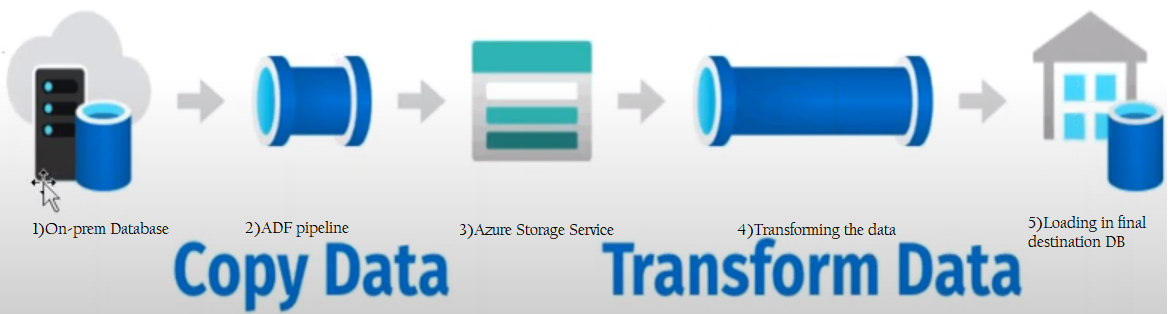
(iv) likewise for each dept. we can create a different subscription.

* ADF Version1 is deprecated and now we are using ADF Versioin2, Version1(V1) launched in 2016 and V2 launched in 2017 and they enhanced V2 in 2018 and so on….
* In V2 of ADF the biggest change Microsoft has made is we can execute the SSIS packages directly.
* As per the documentation of Microsoft ADF is charging $0.4 to process and load 100GB of data into target

# **What is Azure Data Factory (ADF):**

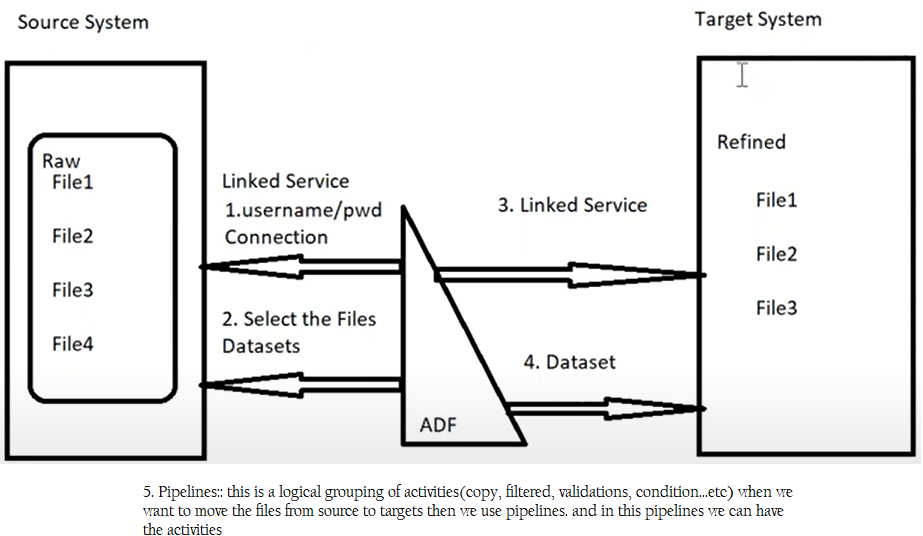
A fully managed, server less data integration solution for ingesting/inserting preparing and transforming all our data at scale.

* When Compare to other ETL tools like SSIS, Informatica, Data stage talent there are many ETL tools in the current market, but in these tools we have to setup everything like infrastructure, cost, performance, productivity, security and compare all these parameters Azure Data Factory(ADF) is much more efficient than these tools and ADF is fully managed my Microsoft(MS) and we need not to worry about underlying infrastructure, cost, performance, security…etc. everything will be taken care by Microsoft for ingesting/loading the data into different destinations/targets.
* We were doing some SQL queries for doing the transformation but from 2019 onwards, we have a mapping data flow came into picture where we need, or we can do all kind of transformations easily without writing the code in ADF.
* We can do many things in ADF (as shown in below image), we can simplify two main tasks, we can copy the data, we can transform the data and like these kinds of tasks can be automated and scheduled.



* In ADF we can fetch/extract the data from variety of different sources the data could in any size, any format, any shape we can fetch/extract the data from source, transform and load it into destinations.
* Based upon the business needs we can insert/dump the data into destination in any different format as compare with source (Ex: if the data in source in excel format we can extract transform and load the data into .csv format)
* We can even load the data from multiple sources to one single destination (like SQL DB single Table, or in one single file or in parquet format varies business to business requirements).
* Microsoft has inbuilt many source and destination as Datasets as native to Azure Data Factory studios for Azure Data Engineers.

# **Architecture for processing & procedures to Load the Data from Source to Target using ADF:**



* If we want to move the files (file1, file2 & file3…etc) from my source to target then we need some compute infrastructure, and this compute infrastructure is taken care by integration runtime(IR) and this IR is automatically managed by ADF based upon the data volume which has to move from source to target

# **Integration Runtime (IR)::**

It basically provides compute infrastructure and it is used by ADF and this compute infrastructure is like network, storage, memory…etc. and all these things can be taken care by Integration Runtime, by default will have an IR while creating an ADF service. If we want to move the data from cloud to cloud or want to move the data from public network to cloud, then we cannot install any external integration runtime and by default will have this IR.

* When performing the transformation means we are performing here like Joints, Unions, Select, Where Aggregations (Max, Min, Avg, Sum…etc.) Having…. etc.
* When client don’t want to load the data as it is…. they need some transformations to be performed before loading the data in target then these transformations are helpful.
* ADF is complete ETL (Extract Transform Load) or ELT (Extract Load Transform) tool in which we can extract, transform, and load the data into destination.

# **Building blocks of Azure Data Factory(ADF)::**

A picture containing text, font, diagram, electric blue

Description automatically generated

# **Pipelines in Azure Data Factory:**

Pipeline is a logical grouping of activities that together performs a task. A data factory can have one or more pipelines. The activities we defined in pipeline performs the action on our data. 40 activities we can define in one pipeline

# **Activities in ADF:**

The activities in a pipeline defined actions to perform on our data. ADF supports below types of activities. i.e.:

(i)Data movement activities

(ii)Data transformation activities

(iii)Data control activities.

# **Datasets in ADF::**

Datasets identify data within different data stores such as tables, files, folder & documents…etc., before we create a dataset, we must create a linked service to link our data store to the data factory.

(or)

After the link service confirmed the sources system and if we are having 10 files in the source system to which we are connected then the Data set will carry the information to pick which of the files in the source systems

# **Linked services in ADF::**

Linked services are much like connection strings which defines the connection information needed for ADF to connect to external resources.

(or)

It contains the information about the source systems, means if we are having 10 source system then link service will carry the information to which source system it has to connect.

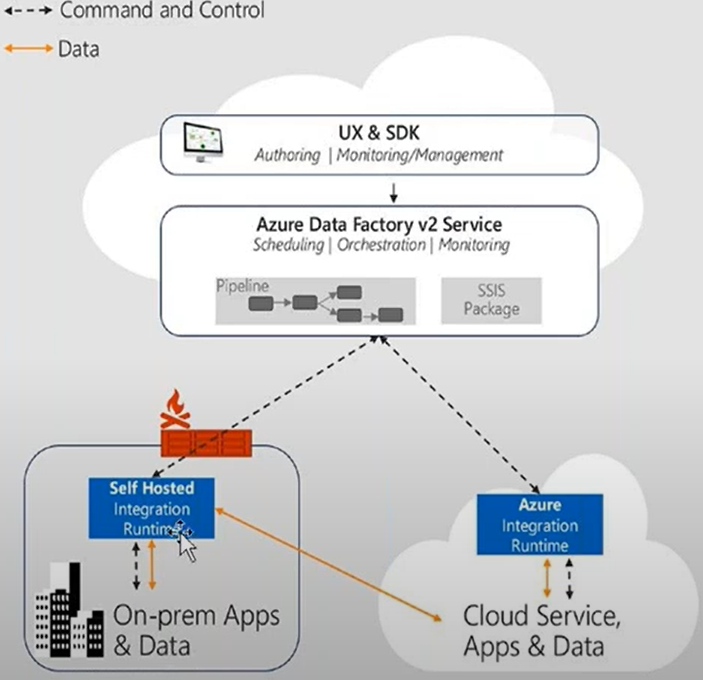
**Example:** If we want to copy the data from Blob storage to SQL Database then we must create 2 linked services. One for Azure Blob storage and one for SQL Database.

* We need contributor access in Azure portal to load the data from source to target with ADF in Azure subscription.
* If we want to move the data, then there are 2 ways i.e.:

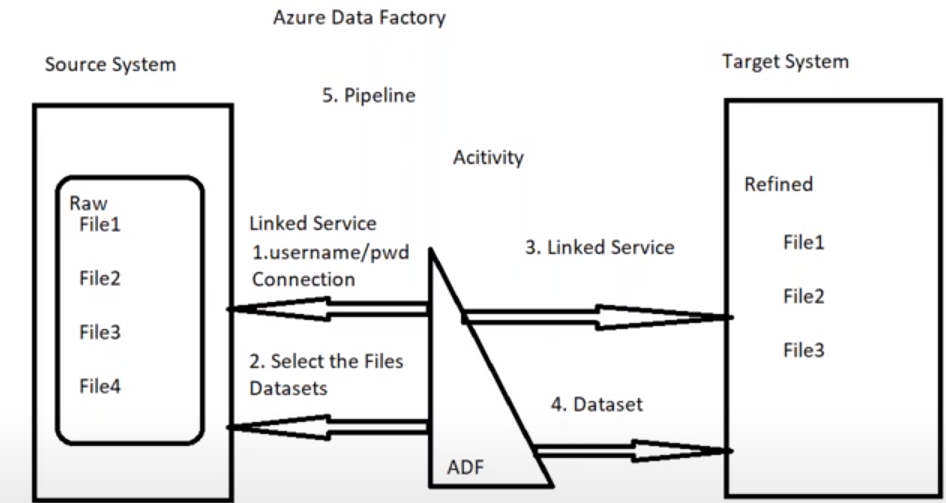
(i)we can connect to the on-prem system and then we can move the data to the cloud

(ii)we can also move the data from cloud to cloud, it might be from AWS or might be from public network also, and for this we can use Azure Integration runtime to connect to the source and then we can transfer the data from one place to an another.

* The data is available in AWS, and we want to move the data to Azure cloud or inside the Azure cloud and this can be done with integration run time.
* This integration run time concept is very useful and it provides a compute infrastructure. If we wanted to move the data from one place to an another, then we must install a software an executable file (i.e.: SHIR>>Self Hosted Integration Runtime) as shown in below image.
* Azure Data Factory Version 1 were having lot of limitations and problem, due to which Version 1 is deprecated, and a 2nd version of Azure Data Factory launched in the year 2019 and now we are using everywhere ADF Version 2 only.
* When we want to move the data from On-prem to Azure cloud DB then we must install Self Hosted Integration Runtime(SHIR) in On-prem server (where our DB is) bcoz the on-prem servers are connected to a private network.
* Whatever the data type is like unstructured, semi-structured or structured when we want to move from source(On-prem) to target (Azure cloud DB) then we have to install SHIR in on-prem
* When we want to move the data from one cloud DB to another then we need Linked services.



* If we don’t want to move our data to other regions, then here we can create our own integration runtime and by default will have the Azure Integration runtime.
* In ADF itself we are having a data flows, maybe our source system hosted on a virtual network and here we can create Azure integration runtime and we can enable the virtual network option and we can connect the security to our system (source system)
* ADF is completely code free tool, most of the things we can configure and setup using drag & drop.
* If we want to automate our workflow and we want to schedule our pipeline, then in ADF itself we have a different kind of triggers available.



* As shown in above pic if my client wants me to schedule/run every day 4 AM IST, he doesn’t want this to be run manually every time instead it should schedule/automate at a specific time and whatever the new data has been loaded into my source that should be loaded into my Target system using Azure Data Factory(ADF) service
* So, in order to work on this schedule mechanism, we have a concept called triggers in ADF, we have different types of triggers

(i)Scheduled triggers.

(ii)Tumbling window trigger.

(iii)Event Trigger

* If we want to schedule our pipeline with a future date and time, then we generally used this scheduled trigger.
* If we want to run this pipeline with the past window slices or if we want to run pipelines, generally there are lot of concepts associated with tumbling trigger, if there is a huge volume of data, large files small files then in such scenarios we can use the properties in tumbling trigger like window Max concurrency.
* If we want to schedule the pipelines only on weekdays no weekends and wants to load the data into target and those types of schedules like every day 5AM IST or for every 4 hrs, or for every Monday 7AM IST…. etc. then we can achieve this with schedule base trigger.
* **Data flow:** Here will do the transformations, we are having different types of transformations in SQL (i.e.: sum of the values, Avg, min, max, count of values) all these comes under aggregations, like this we have different types of transformation that we need to perform.
* There are also different types of transformation that we can do in data flow like joins, unions, conditional split.

**Different types of Integration Runtime:** Integration runtime(IR) is the compute infrastructure used by Azure Data Factory. There are 3 different types of IR available.

(i)Azure IR (ii)Azure-SSIS IR (iii)Self hosted IR

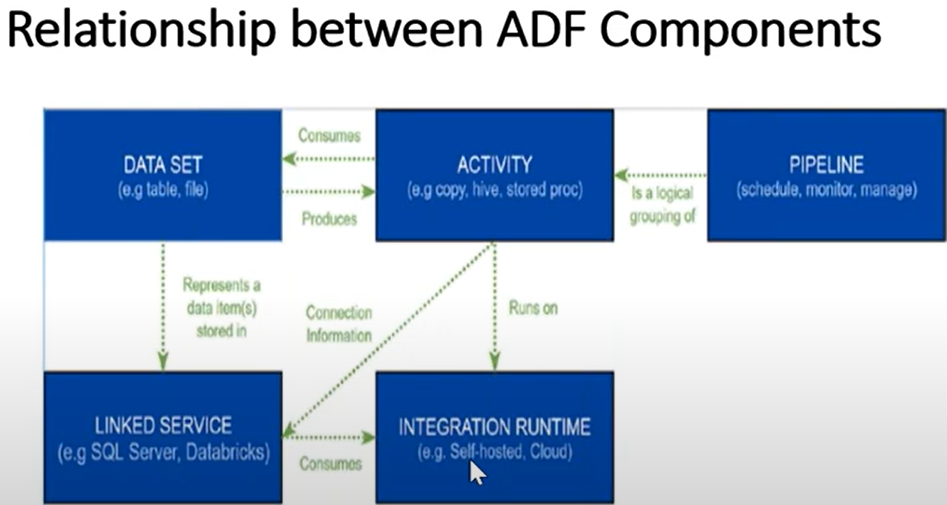
# Compute Infrastructure:

When we want to move data from source to target then the compute infrastructure is needed for ADF to move the data from one place to another.

* Without writing even a single piece of code we can execute SSIS packages in ADF but for this we should run the Azure SSIS integration runtime, so here we must create this Azure SSIS integration content. This Azure SSIS integration runtime we have to connect when we want on-prem SSIS package and we want to execute those SSIS packages in ADF

**Self-hosted Integration runtime:**

if the data is in on-premises and hosted on a private network and when we have to connect to their network then in this case we use the self-hosted integration run time as it is a secured network and when we want to connect to that secured system and need to move the data to the target (data lake storage…. etc.). Hence when we want to move the data from on-prem to cloud then we can use self-hosted agent.



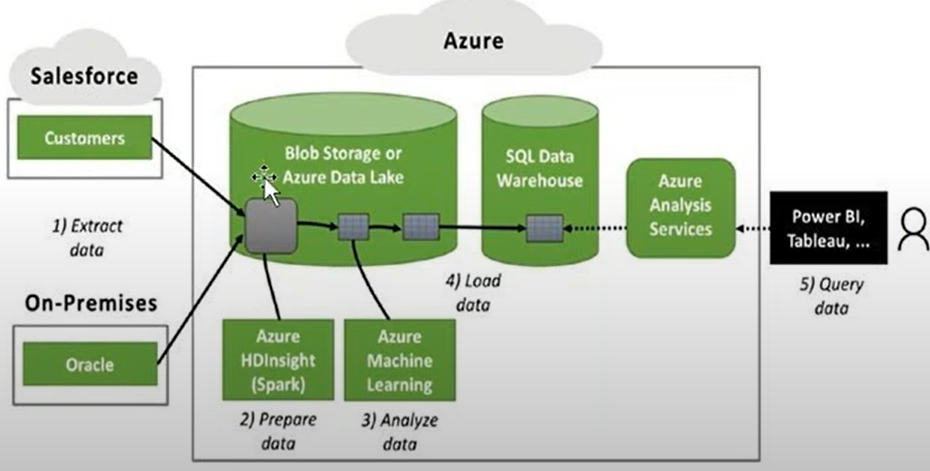
* Here we can see how the components of ADF are clearly dependent on each other.

# **Use of Triggers:**

* + Triggers are used to schedule execution of pipeline.
  + Pipelines and triggers have many to many relationships, ex: multiple triggers Kick off a single pipeline or a single trigger can Kick off multiple pipelines.

# **When to use ADF:**

* We can use ADF When we are building a big data analytics solution on Microsoft Azure
* We can use ADF When we are building a modern data warehouse solution that relies on technologies such as (i)SQL Server. (ii)SSIS and (iii)SQL Server Analysis Services
* ADF also provides the ability to run SSIS packages on Azure or build a modern ETL/ELT pipeline and letting us access both on-premises and cloud data services.
* We can use ADF to migrate or copy the data from physical server to the cloud or from a non-Azure cloud to Azure(blob storage, data lake storage, SQL Cosmos DB)
* ADF can be used to migrate both structured and binary data.
* When it comes to any ETL product in the market what customer currently looking is for cost, productivity, performance, security and ADF is providing these all features.
* Compare to other ETL tools ADF is very effective and building big data analytics solution in Microsoft Azure and building a modern data warehouse solution with lot of benefits and features for Azure Data Engr.
* The underlying infrastructure everything is managed by ADF even when we are running SSIS packages the underlying infrastructure everything is managed by Azure this is the reason for which companies and clients are moving to ADF.
* When we don’t want to maintain your underlying infrastructure, and everything should be managed by Microsoft and we want to move to any PAAS services then everything is managed by the cloud vendor (Microsoft Azure)
* With ADF we can connect to any public network and we can move the data to the cloud we can move either structured or unstructured data or binary data (like audio files, video files, image files, sensor data, streaming data…etc.) using ADF services.



# **Why to use ADF (or) Why ADF**:

* **Cost Effective:** ADF is server less and the billing is based on factors such as the Number of activities run the data movement duration.

**Ex:** If we run our ETL/ELT pipeline hourly, which also involved data movement (assuming 100GB data movement per hour which should take around 8 mins with 200 MBPS bandwidth) then ADF will bill us not more than $12 for the monthly execution.

**Cloud Scale:**ADF being a PAAS offering can quickly scale if needed, for the big data movement with data sizes from terabytes to petabytes, we will need the scale of multiple nodes to chunk data in parallel.

**Enterprise grade Security:** The biggest concern around any data integration solution is the security, as the data may well contains sensitive personally identifiable information(PII)

**High Performance Hybrid Connectivity:** ADF supports more than 90+ connectors the connectors support on-premises sources as well, which helps us to a data integration solution with our on-premises sources.

**Easy Interaction:** As ADF supports so many connectors and that makes it easy to interact with all kinds of technologies.

**Visual UI authoring and monitoring tool:** it makes us super productive as we can go with drag and drop development. The main goal of the visual tool is to allow us to be productive wit ADF by getting pipelines up and running quickly without requiring us to write a single line of code.

**Schedule pipeline execution:** Every business have different latency requirement (hourly, daily, weekly, monthly…and so on) and jobs can be schedule as per the business requirements.

# 

# **Complete data flow end to end with ADF Process:**

****

**Data Flow:**

* Data flow allows data engineers to develop graphical data transformation logic without writing code,
* Data flows are executed as activities within azure data factory pipelines using scaled-out Azure Data bricks clusters.
* Within ADF, Integration runtime(IR) are the compute infrastructure use to provide data integration capabilities such as data flows & data movement. ADF has the following three IR types

1. **Azure integration runtime:** All patching, scaling, and maintenance of the underlying infrastructure are managed by Microsoft, and the IR can only access the data stores and services in public networks.
2. **Self-hosted integration runtime:** The infrastructure and hardware are managed by us, and we will need to address all the patching, scaling and maintenance, the IR can access the resources in both public and private networks.
3. **Azure-SSIS integration runtimes:** VM’s running the SSIS engine allow us to natively execute SSIS packages. All the patching, scaling and maintenance are managed by Microsoft, the IR can access resources in both public & private networks.

# Mapping Data Flows for Transformation & Aggregation:

Mapping data flows are visually designed data transformation in Azure Data Factory, it allows data engineers to develop data transformation logic without writing code, the resulting data flows are executed as activities with ADF pipelines and that use scaled out Apache Spark Cluster.

There are three different cluster types available in mapping Data Flows i.e.:

**General Purpose:** We use the default general purpose cluster when we intend to balance the performance and cost, this cluster will be ideal for most data flow workloads.

**Memory Optimized:** more costly per core memory-optimized clusters if our data flow has many joins and lookups since they can store more data in memory and will minimize any out of memory errors we may get. If we experience any out of memory errors when executing data flows, switch to a memory optimized Azure IR configuration.

**Compute Optimized:** Use the cheaper per-core priced compute-optimized clusters for non-memory-intensive data transformations such as filtering data or adding derived columns.

**Schedule & Monitor:** We can schedule and monitor all of our pipelines runs natively in ADF user experience for triggered pipelines, additionally we can create alerts and received texts or emails related to failed, succeeded or custom pipeline execution statuses.

**Steps on high level to work with Azure Data Factories:**

* First will create an ADF
* Will launch the ADF studios
* Will create the link services for Source systems
* Will create the dataset for source system
* Will create the link services for Destination systems
* Will create the dataset for Destination system
* Will design the ADF pipelines(in this ADF pipelines will use the different controls/activities as per the need of the business)
* Will do publish all & publish(build/saving all the changes whatever you have done in ADF)
* Will Debug(run/deploy) the ADF pipelines(running the ADF pipelines)
* Finally, the data will get loaded into our targets/destination systems.

# **Implementation steps to create/deploy an ADF:**

**Step 1:** Create a Resource Group

**Step 2:** Search for Data factories in Azure portal>>Create and fill the below details.

(i)Subscription: any/Free trial

(ii)Resource Group: any

(iii)Name: any

(iv)Region: any

(v)Version: V2

**Step 3:** Click on Next: Git configuration>>check the box Configure Git later>>click on Networking>>click on Next: Advanced>>Next: Tags>>Next: Review + Create>>Create>>wait for some 4-5 minutes until the ADF gets deployed.

* After ADF gets deployed click on **Launch studio (in the centre)** then will navigate to a new page wherein we can see all the features of ADF studios like
* (i)Home (ii)Author (iii)Monitor (iv)Manage (v)Learning Centre (vi)Updates (vii)Switch to another Data Factory (viii)Notifications (ix)Settings…etc.

# Networking (left side under ADF):

Here we are having public endpoint and private endpoint and securely connect to the data source we use the private endpoint. Public endpoint in the sense on the internet everyone can be able to access our storage account, data lake storage, SQL DB if it is in public endpoint, but it is recommended to keep the private endpoint with which we can securely connect to our data tools from our ADF service.

* In real time projects will create a separate storage account for each environment like for Atest environments data will create a separate SA. For Dev environments data will create a separate SA.

For Pro environments data will create a separate SA…. etc.

* We use Azure storage explorer to configure all the different environments SA’s at one place to avoid going to Azure portal every time and opening the SA for each and every time for different environments.
* When we don’t want to move the data particularly to specific regions then at the time of defining the integration runtime we can do this configuration here, mentioning these are the regions in which data should not get transfer if someone Azure Data Engr trying to do it so then it should get failed automatically.
* To securely connecting the storage account, we can use Rest API
* There is no restriction to load the data into Storage account we can load even 100 GB, 200GB, 1TB or 5TB…etc. no limit as such.
* Create a Data Lake Gen2 storage account and while creating a SA in Advanced tab under Data Lake Storage Gen2 check the checkbox for Enable hierarchical namespace (when we are checking this checkbox it means we are trying to create a Data Lake Gen2 SA, the only change between this blob SA and Data lake Gen2 SA is this check only). Data lake storage Gen2 will accelerate big data analytics workloads if we are trying to perform any analytics on top of the data then we should load the data into data lake storage Gen2.
* The main difference between blob and data lake storage Gen2 is to load the data in a data lake in hierarchical folder structure (Year. Month. Week. Day. Hr…etc.)
* When we want to perform analytics on data then we should load the data into data lake storage Gen2. Blob storage is not meant for doing big data analytics. Azure Data Lake Storage(ADL’s)Gen2 is meant for big data analytics. we can also do Access Control List up to five level in ADL’s Gen2 but not on blob storage.
* Blob Storage service supports only object-based storage and ADL’s Gen2 supports both file and object based storage.
* ADL’s Gen2 has a great security when compared to blob storage service.
* In Advance tab of Azure Storage Account when Enable hierarchical namespace is checked then it is ADL’s Gen2 and when it is not checked then it is blob storage service.

# **Implementations steps for copying the data from Blob SA to ADL’s Gen2 SA using ADF:**



# **Implementation steps for copying the zip file from Blob SA to ADL’s Gen2 SA using ADF:**

A diagram of a link between two lines

Description automatically generated with medium confidence

**Step1:** Create Blob SA(source)>>create a container/folder in it and upload below zip folder.

  
**Step2:** Create ADL’s SA(destination)>>create container inside the SA.

**Step3:** Deploy Azure Data Factory>>Launch the ADF Studio>>

**Creating Link service for Source:**

**Step4:** Click Manage (left side)>>Linked services>>+New>>a window will open on right side and in search type blob storage click on it and then click on continue and fill the details as mentioned below.

**Name:** LS\_BlbSA/any name

**Azure subscription:** select the subscription here accordingly

**Storage account name:** 1925blbsa/any name (here carefully choose the blbsa)>>Click on Test connection>>Apply

**Creating Link service for Destination**

**Step5:** Click on Manage (left side under ADF studio)>>Linked services>>+New>>>>a window will open on right side and in search type Gen2 click the Gen2 storage>>continue and fill the details as mentioned below.

**Name:** LS\_AdlSA/any name

**Azure subscription:** select the subscription here accordingly

**Storage account name:** 1923adlsa/any name (here carefully choose the blbsa)>>Click on Test connection>>Create.

**Creating Dataset for Source:**

**Step6:** Click on Author(left side under ADF Studio)>>Dataset>>click on extreme 3 dots of Azure Dataset>>New Dataset>>a window will open on right side and in search type blob storage click on it and then click on continue>>Now here we have to select what files format(csv, excel, Json…etc.) our zip folder is containing(in blob SA) and select Delimited Text>>continue>>Name:DS\_Zipfile>>Linked service:LS\_BlbSA>>click on folder(extreme right)>>click on container>>choose the zip file(this we have uploaded in our blbsa)>>ok>>import schema: None>>ok>>compression type:ZipDeflate (.zip)>>click on preview data(there we can see the data from all the files that our .zip folder is containing)>>Compression level: optimal

**Creating Dataset for Destination:**

**Step7:** Click on Author (left side under ADF Studio)>>Dataset>>click on extreme 3 dots of Azure Dataset>>New Dataset>>a window will open on right side and in search type Gen2 storage click on it and then click on continue>>Now here we have to select what files format(csv, excel, Json…etc.) our zip folder is going to copy (in Gen2 SA) and select DelimitedText>>continue>>Name:DS\_ZipfileDest>>Linked service: LS\_AdlSA>>click on folder(extreme right)>>click on container>> >>ok>>import schema: None>>ok>>Publish all>>Publish

**Step8:** In Author only>>click on Pipelines>>click on 3 dots and say new pipeline>>Name: Process\_ZipFolderPractice and then click on properties Icon on top.

**Step9:** Click on Move & transfer (top under Activities)>>then drag and drop the Copy data activity to middle pane(pipeline canvas)>>Name: CopyDataFromBlbSAToAdlSA>>click on source tab>>Source dataset:DS\_Zipfile>>click on Sink tab>>Sink dataset:DS\_ZipfileDest>>File extension: .csv

**Step10:** click on publish All on top>>Publish>>wait for some time until the publish completes successfully

**Note: For every single change, we must do publish without fail**

**Step11:** Click on Debug>>wait for some time until the Debug completes (it should be succeeded) and copy the .zip folder files from source SA to ADL SA.

**Step12:** Now come to the destination SA then will see here the zip folder should be copied with all the files in it.

Hence copied the .zip folder which contains multiple files from one SA to an another.

**Note:** If we want to copy the data from one SA to another and both the SA’s are Blob SA, then there is no need to have 2 separate linked services to be created.

If we want to copy the files, which are in the form of videos, clips, reels...etc. basically unstructured data then we use binary format files.

# **Implementation steps to perform Validation activity, Metadata activity, & If condition activity in Azure Data Factory (ADF):**

A screenshot of a computer

Description automatically generated

**Step1:** Create a Blob Storage Account(Src SA) and blob container inside it and place/upload the below .csv file.



**Step2:** Create a ADL Gen2 Storage Account(Dest SA) and container inside it.

**Step3:** Create ADF>>Launch ADF studio>>Author>>pipelines>>click 3 dots of pipelines and say New pipeline>>Name: Dynamic Pipeline>>in Activities pane(@ center) type Get Metadata>>drag and drop the **Get Metadata** control to center>>click on settings tab(below center)>>+New>>search blob storage>>click on Azure blob storage>>Continue>>choose Delimited Text(csv) file>>Continue>>Name: DS\_Input>>click on Linked service>>+New>>Name: LS\_BlbStorage>>Select Azure Subscription & Blob Storage account(carefully)>>Test connection>>create>>click on folder ikon>>myblobcon>>select the file here>>ok>>ensure the First row as header checkbox is selected>>Import schema as None>>ok

**Step4:** In setting tabs for Field list click on +New>>click the drop box every time and click on +New every time and select the below options for everyone box

column count

Content MD5(Message Digest)

Item name

Item type

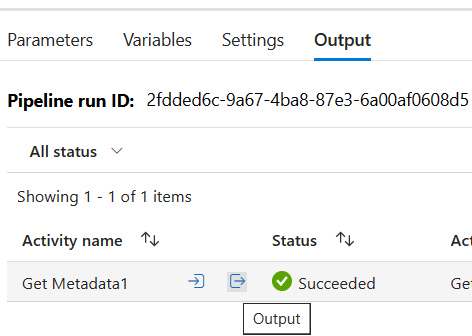
Exists

Last modified

Size

Structure

**Step5:** Click on Publish all(at top)>>Publish>>close>>click on Debug>>wait for some 5 mins till it’s get deployed and in output tab see the arrows for inputs and outputs(as shown below)



**Metadata** means when the file got created, on which date and time it got modified, what is the file size, what is the file type (like excel, csv, notepad…. etc.), how many columns are there in the file, what are the columns names and the header column is called as Meta data (or) schema.

**Step6:**

in Activities pane(@ center) type validation>>drag and drop the **validation control** to center before Get Metadata activity>>Name: Validate if file exist>>click on settings tab>>Dataset: click the nob and choose DS\_Input(this Dataset we have created in above demo, so above demo is related to this demo)>>Timeout:7.00:00:00(means this control will keep on validating whether this file is there or not till 7 days)>>Sleep: 30(means for every 30 seconds it will check)>>Minimum size: 10 Bytes (means here I want if the file size is less than 10 bytes then don’t pick the file)>>Now establish a connection between validation control and Get Metadata control(this we have done in above demo) by dragging the green line from validation control to Get Metadata control>>Publish all(this should get succeeded)>>Publish>>Debug(this should get succeeded)>>Wait for some 1-2 mins(depends upon the file size) then it will check whether the file exists or not by validation control in the blbsa(as this is source) if yes then from Get Metadata control it will give us the Metadata details of the file.

**Step7:** Now remove the .csv file from blbsa (as this is our source)>> in settings tab of validation control for Timeout set: 0.00:00:30(30seconds or any time we can set)>>Publish All>>Publish>>Debug the Pipeline>>Now here we should get Status as timeout bcoz we have removed the .csv file from source storage account and Timeset as 30 seconds

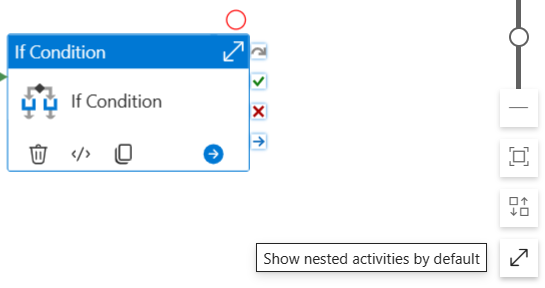
**Note: keep the .csv file back into the Src SA and change the Timeout: 7.00:00:00 for Validation control settings tab, the above step7 is just for testing purpose**

**Step8:** Search for If condition control(generally we use this if condition control to check whether the files content is exists or not, file exist or not, file content is as expected or not) in Activities and drag and drop if condition activity after Get Metadata control>>click on the if condition control>>In General tab give the name accordingly>>Establish a connection between Get Metadata control and If condition control>>click on Activities tab>>click on Expression box>>Add dynamic content, then right side window will open in the same windows & below we can see all the expressions option>>just make a single click on **GetMetadata columnCount** and then Add @equals method as shown below in the expression box>>after we write the expression as shown below then finally click on ok(at below)

A screenshot of a computer

Description automatically generated

**Step9:** Now click on Show nested activities by default(extreme right as shown in below image)>>then will find 2 boxes in If condition controls as True & False and in that click on True box pencil(as shown in below image)>>Now here in Activities search for copy data control and then drag and drop>>in General tab pass Name: Copy data from source to target>>Now click on source tab>>+New>>in search type Azureblobstroage>>click on Azure blob storage>>continue>>Delimited text>>continue>>Name: DS\_inputforcopycontrol>>Linked service: LS\_BlbStorage(this we have created in above demo)>>click on folder ikon>>click on container(myconblb)>>select the file>>Ok>>ensure the check box First row as header checked>>Import schema as None>>Ok

A screenshot of a computer

Description automatically generated with medium confidence

**Step10:**Click on sink tab>>+New>>in search type gen2 and select Azure Datalake Storage Gen2>>continue>>Delimited text>>Continue>>Name:DS\_Output>>Linked service:+New>>Name: LS\_Adlstorage>>Azure subscription: choose accordingly>>Storage account name:DestinationSA(this SA we have created at the top and gave this name to SA, choose the storage accordingly whatever the name you gave)>>Create

**Step11:** click on folder ikon (under File path)>>myconadl (this container we created and gave name like this, that’s what we are selecting here the same)>>ok>>ensure the box first row as header has selected>>Import schema: none>>ok>>now in same sink tab only scroll down and change the File extension: .csv>>Publish All>>Publish>>click on pipeline

**Step12:** Click on Publish (this should be successfully published)>>Debug>>Now all the controls which we placed in pipelines should get succeeded and we can see the file copied from source blbstorageAccount to ADLstorageAccount.

# **Implementation steps to perform Get Metadata control, Filter control, ForEach control in ADF pipelines:**

A screenshot of a computer

Description automatically generated

**Step1:** Create a Blob Storage Account(Ex:1174srcsa) and blob container inside it and place/upload the multiple .csv files and also zip folder/files.



**Step2:** Create a ADL Gen2 Storage Account(1174destsa) and container inside it.

**Linked service for Source Storage Account**

**Step3:** Create ADF>>Launch ADF studio>>Click on Manage ikon(right side)>>Linked services>>+New>>in search type Azure blob storage>>click on Azure blob storage>>Continue>>Name: LS\_Source>>choose the subscription and Source Storage account(1959blbsa this we created in above step)>>Testconnection>>Create

**Linked service for Destination Storage Account:**

**Step4:** Now creating one more linked service for Destination Storage Account>>Manage ikon (right side)>>Linked services>>+New>> in search type Azure Data Lake Storage Gen2>>Click on it>>Continue>>Name: LS\_Dest>>choose the subscription and storage account name (1960adlsa this we created in step2)>>Testconnection>>Create

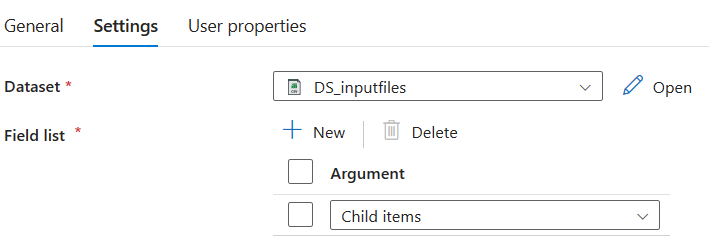
**Creating Dataset for source Linked Service:**

**Step5:** Click on Author ikon(right side)>>Datasets>>click on 3 dots>>New dataset>>in search type Azure blob storage>>click on Azure blob storage>>Continue>>Delimited Text>>Continue>>Name: DS\_Inputfiles>>Linked service: LS\_Source(this we have created in above step)>>click on folder ikon under Filepath>>click myblobcon(this container we have created in step1)>> and say Ok(here don’t click on any one file particularly bcoz here our plan is to copy all the files to our ADL storage account)>>Ok>>Ensure the check box is checked for First row as header>>Import schema as none>>ok

**Creating Dataset for Destination Linked Service:**

**Step6:** Click on Author icon (right side)>>Datasets>>click on 3 dots>>New dataset>>in search type Azure data lake storage Gen2>>click on Azure data lake storage Gen2>>Continue>>Delimited Text>>Continue>>Name: DS\_Outputfiles>>Linked service: LS\_Dest (this we have created in above step)>>click on folder ikon under Filepath>>click myadlcon (this container we have created in step1)>> and say Ok>>Ensure the check box is checked for First row as header>>Import schema as none>>ok

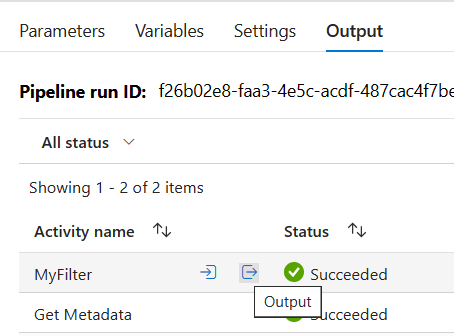
**Step7:** Click on author ikon>>click on 3 dots of pipelines>>new pipeline>>give name: PL\_For\_BulkCopy>>in activities search Get Metadata and drag and drop this activity/control into pipeline from activities pane>>In General tab give Name: Get Metadata>>Click settings tab>>For Dataset click the nob and choose Ds\_inputfiles>>Field list: +New>>from the dropdown box select child items(as shown in below image)>>publish and Debug>>after this is completed check the input and output.



**Step8:** Now in activities pane search for filter activity>>drag and drop the filter activity/control to pipelines>>establish a connection with green line extension>>in General tab>>give Name: MyFilter>>click on settings tab>>click on Items box>>Add dynamic content>> a window will get open on right side and in that select **Get Metadata childitems**>>ok> click on condition box>>Add dynamic content>> a window will get open on right side and in that select Activity outputs tab>>and write the **below expression 1)** in pipeline expression builder box>>and then finally click on ok

1. @endswith(item().name,'.csv')>>This expression is to pick all the files who has .csv extension
2. @startswith(item().name,'Sales\_')>>This expression is to pick all the files whose name starts with Sales\_

**Step9:** Now click on publish and Debug and if we see in Input and Output of Filter activity (as shown in below image) after Debug succeeded then will see in output it is **picking only .csv files extension to copy from the source to target**, and here in our source Storage Account we kept files of .zip extension and .csv extension both.



**Step10:** Now drag and drop ForEach control/activity from Activities pane to pipeline>>Establish a connection (by dragging the green line) from Filter control to ForEach control>>Click on ForEach activity>>in General tab pass Name: ForEachFile>>in setting tab>>click on items box>>Add dynamic content>>in Activity output tab>>click on MyFilter (this name MyFilter we gave above for ForEach control) and type the below expression in pipeline expression builder box>>ok

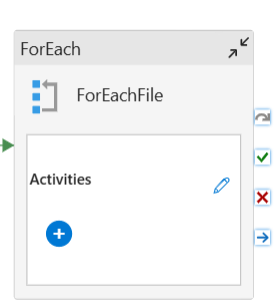


@activity('MyFilter').output.value

**Sequential:** when we are trying to process a file (just one file) which is of 50GB and there is no parallel process here then we click on sequential and then we can process the 50GB file easily and can load into the target/dest…**Here the files will get loaded one by one in sequential order** like up to 3 or 4 files we can go for this sequential option.

**Batch count:** When we have copy No of files (like 40-50 file or more of like 100MB or 200MB…etc.) then we use Batch count, this batch count will do the **parallel processing** **of files** and parallel processing always improves the performance, the default value for Batch count is 20

**Step11:** In ForEach control click on the pencil ikon(as shown below) and in activities pane search for copy data control and drag and drop in pipeline>>in General Tab give Name: CopyDataFromSourceToTarget>>in Source Tab click on +New>>in search box type blob storage>>click on it>>continue>>DelimitedText>>Ok>>Name: DS\_InputForMultiFiles>>Linked service: LS\_Source(this we have created above)>>For File path click on folder ikon>>select the container not any file(this we have created above)>>ok>>ensure the checkbox first row as header>>Import schema as None>>ok



**Step 12:** Now in source tab>>click on Open for source dataset(click on the header of copy data control)>>Parameters>>+New>>Name: SourceFiles>>click on connections tab>>click on File name box then click on Add dynamic content>>click on SourceFiles then an expression will get on pipeline expression builder>>ok

**Step13:** In source tab(for Copy data control) click on SourceFiles Textbox(click on the header of copy data control)>>Add dynamic content>>click on ForEachitem>>and type the below expression in Pipeline expression builder>Ok

@item().name

**Step14:** Click on Sink tab>>For Sink dataset: DS\_outputfiles>>open>>Parameters>>+New>>Name: SinkFileNames>>click on connection tab>>click on File name box>>Add dynamic content>>click on SinkFileNames>>an expression will get printed in pipeline expression builder>>ok

**Step15:** Come back and click on the header of copy data control>>click on SinkFileNames box>>Add dynamic content>>and type the expression below in pipeline expression builder>>and then click on ok

@item().name

**Step16:** Now click on Publish All>>Publish>>(this should get succeed)>>Debug>>wait for some 2-3 mins and will see the files which are having .csv extension only will get copied to ADLs Gen2 Storage Account.

Hence we have copied one particular extension file among multiple files from our source system to Destination system.

**Business Case Scenario 1:** Now delete the file from the destination storage account and keep multiple .csv files in your source storage account and debug the pipeline again, the ADF pipeline will copy all the .csv files from source to destination storage accounts.

**Business Case Scenario 2:** Now Delete all the files from the destination storage account>>come to filter activity>>settings tab>>in condition box in expression builder change from .csv to .zip>>do the publish and debug and this time we notice only .zip will be copied to our destination SA

# **Implementation steps to copy the data from GitHub to Azure ADL Storage services (Using Parameters):**

**Step 1:**

Create a ADL gen2 storage account and create a container inside the SA

**Step 2**

Create ADF>>Launch ADF studio’s

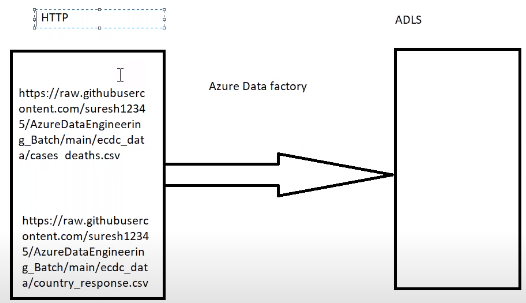
**Step 3:**

Create a linked service(LS\_ADLGen2) inside ADF studio under managed tab and this Linked service is for target/dest (i.e. ADL gen2 Storage Account)

**Step 4:**

Create a Dataset (as ds\_outpoutgen2) for ADL gen2 Storage account under ADF studios and this Dataset will be Target/dest Dataset>>Publish all>Publish

As per the below diagram if we see, we are trying to load the data of multiple files which are placed at GitHub(source)



(i) [AzureDataEngineering\_Batch/ecdc\_data/cases\_deaths.csv at main · suresh12345/AzureDataEngineering\_Batch · GitHub](https://github.com/suresh12345/AzureDataEngineering_Batch/blob/main/ecdc_data/cases_deaths.csv) >>click on View raw(centre in this link to see the complete source data)

(ii) [AzureDataEngineering\_Batch/ecdc\_data/country\_response.csv at main · suresh12345/AzureDataEngineering\_Batch · GitHub](https://github.com/suresh12345/AzureDataEngineering_Batch/blob/main/ecdc_data/country_response.csv)>>click on Raw(top right side in this link to see the complete source data)

**Note:** Instead of hard coding the values, we can pass the parameters for Datasets, linked services & at pipeline level and once we create a parameter, we cannot modify it.

**Linked Service for source:**

**Step 5:**

Launch the ADF studios>>Manage>>Linked services>>+New>>**In search type HTTP**>>click on Http>>continue>>Name: LS\_HTTP>> Authentication type: Anonymous >>Expand the parameters (in same window, just scroll down below)>>+New>>Name: BaseURL>>then scroll up a little click on Base URL box>>click on Add dynamic content>>click on baseURL>>ok>>click on create.

**Dataset for source:**

**Step 6:**

click on Author (left side inside ADF studios)>>click on 3 dots of Datasets>>New dataset>>in search type HTTP>>click on HTTP>>continue>>Delimited text(bcoz in our git hub all are .csv files)>>continue>>Name: ds\_Inputhttp>>Linked service: LS\_HTTP>>Ensure the text box is check for First row as header>>Import schema as None>>Ok

**Adding parameters to Source Dataset:**

**Step 7:**

Click on parameters tab>>+New>>pass Name: BaseURL>>click on +New again and pass Name: RelativeURL>>click on connection tab>>click on baseURL text box>>Add dynamic content>>click once on BaseURL in the opened window>>ok>>Now click on Relative URL box>>Add dynamic content>>click once on Relative URL>>ok

**Step 8:** just do publish here so that all the Linked services, Datasets, parameters, pipelines…etc. we have created will get saved.

**Creating Pipeline:**

**Step 9:**

click on Author ikon (left side 2nd Ikon in ADF studios)>>click on 3 dots for pipelines>>New pipelines>>Name: PL\_Parameterizedpipeline>> click on Parameters(below for pipeline)>>+New>>Name: SourceBaseURL>>click on +New again and pass Name: SourceRelativeURL>> in Activities pane in search type copy data and drag and drop the Copy data control from activities pane to pipeline>>Name: CopyData

# **Copying the data from http (GitHub) to ADL Gen2 StorageAccount:**

**Step 10:**

Click on Copy data control>>click on source tab>>click on Source dataset box knob and select ds\_Inputhttp(this dataset we have created above)>> click on BaseURL check box >>click on Add dynamic content>>just click once on SourceBaseURL the expression will get copied in the pipeline expression builder>>ok>>click on RelativeURL check box>>click on Add dynamic content>>just click once on RelativeBaseURL>> the expression will get copied in the pipeline expression builder>>ok>>click on Sink tab>>Click on sink dataset box knob and select ds\_outputgen2(this dataset we have created above)

**Step11:**

Publish (wait for some 2-3 mins till publish gets succeeded)>>Debug>>here it will ask to pass the values for SourceBaseURL & SourceRelativeURL pass below accordingly and click ok

**SourceBaseURL:** <https://raw.githubusercontent.com/>

**SourceRelativeURL:** suresh12345/AzureDataEngineering\_Batch/main/ecdc\_data/country\_response.csv

suresh12345/AzureDataEngineering\_Batch/main/ecdc\_data/cases\_deaths.csv

**Below is GitHub URL which contains multiple .csv files(as source)**

[AzureDataEngineering\_Batch/ecdc\_data at main · suresh12345/AzureDataEngineering\_Batch · GitHub](https://github.com/suresh12345/AzureDataEngineering_Batch/tree/main/ecdc_data)

Now from above URL path there are many files and we can pass the SourceBaseURL & SourceRelativeURL accordingly(as mentioned above) and can load the data of any file from GitHub to our Azure Storage services in cloud computing.

Hence here dynamically we are passing the URL values(i.e.: GitHub links) of the GitHub account from where we are directly loading the data to our Azure Datalake Gen2 Storage Services.

# **Allocating variables to ADF pipelines:**

When we create a variables in ADF pipelines then we can able to modify whenever we want, but while creating a parameters we cannot able to modify any value.

**Step 12:** Now delete the parameters of the pipelines>>go to pipeline>>parameters tab>>select the 2 parameters i.e: SourceBaseURL & SourceRelativeURL>>click Delete on top>>

**Step 13:**

Click on variables tab>>+New>>Name:SourceBaseURL>>click on +New again>>Name:SourceRelativeURL>>on same window for Default value box pass the URL values for SourceBaseURL & SourceRelativeURL and pass the below values to the SourceBaseURL & SourceRelativeURL

SourceBaseURL: <https://raw.githubusercontent.com/>

SourceRelativeURL: suresh12345/AzureDataEngineering\_Batch/main/ecdc\_data/cases\_deaths.csv

**Step 14:**

Click on copy data control>>click on Source tab>>click the box of BaseURL>>window will get open on right side>> remove the old expression >>click on variables tab in the newly opened window>>click on SourceBaseURL>>ok>>click the box of RelativeURL>>window will get open on right side>>remove the old expression>>click on variables tab in the newly opened window>>click on SourceRelativeURL>>ok>>Publish all>>Publish>>Debug>>To check whether the data/files is getting loaded into our destination with variables.

**Note:** We have pass the values to variables but if we want to overwrite the variables, values sometimes then we can use the set variables control.

**Step 15:** In activities pane search for **set variables**>>drag and drop this control in ADF pipelines before copy data control>>establish a connection with green line>>click on the set variable control>>in General tab give Name: SetVariable>>click on settings tab>>Name: SourceRelativeURL>> and pass value as

**Value:** suresh12345/AzureDataEngineering\_Batch/main/ecdc\_data/testing.csv

**Step 16:** Publish the pipeline>>Debug and now if we notice the variable value that what we have passed for pipeline has been overwritten by the variable value that we have passed for Set variable control.

Hence like this we can overwrite the value of the variable using Set variable control with ADF pipelines in ADF studios.

# **Creating Dynamic Pipelines with lookup activity to get the count of files from .json:**

Lookup activity can retrieve a dataset from any of the data sources supported by data factory and Synapse pipelines. We can use it to dynamically determine which objects to operate on in a subsequent activity, instead of hard coding the object name. Some object examples are files and tables.

Lookup activity reads and returns the content of a configuration file or table. It also returns the result of executing a query or stored procedure. The output can be a singleton value or an array of attributes, which can be consumed in a subsequent copy, transformation, or control flow activities like ForEach activity.

Lookup activity can be used to read a config files, to read a single row, to read a config table, we use this lookup activity to retrieve the data from multiple sources, it can read and return the contents of configuration file and table

**Step1:**Goto the below link

[GitHub - suresh12345/AzureDataEngineering\_Batch: Resources for the ADF For Data Engineers - Project on Covid19](https://github.com/suresh12345/AzureDataEngineering_Batch)

click on code>>Download zip>>the complete zip folder will get downloaded in our laptop(downloads)>>right click Extract All>>Extract>>double click on AzureDataEngineering\_Batch-main>>again double click on AzureDataEngineering\_Batch-main>>Goto config folder(in your laptop in downloads only)>>double click on section5 folder>>and open ecdc\_file\_list\_for\_2\_files.json file in Notepad++>>and now in config file(json file basically)make the changes as below

A picture containing text, line, font, screenshot

Description automatically generated

Now from above screen we can understand that we are reading the data from 2 different files. i.e.: 1)cases\_deaths.csv & 2)hospital\_admissions.csv

Else directly take the below file and directly upload it in Source Storage Account container(first put it in your desktop and then upload it in SA container)



**Step2:**

Create a storage account(as 1964blbsaconfig, this is blob storage account not ADL Gen2 storage account)>>create a container inside the storage account as config>>come inside the config folder and upload the config file(ecdc\_file\_list\_for\_2\_files.json)which we prepare in above step

So here whenever we want to modify or add an extra file then there is no need to touch the ADF pipelines, here directly we can go to the config file and add the new file details like how we have added for the above 2 files.

**Creating ADF & Linked service for the storage account:**

**Step 3:**

Create ADF>>Launch ADF studios>>click on manage ikon(left side)>>Linked services>>+New>>in search type blob storage>>click on Azure blob storage>>continue>>Name: LS\_BlbSA>>select the subscription & Storage account carefully>>test connection>>create

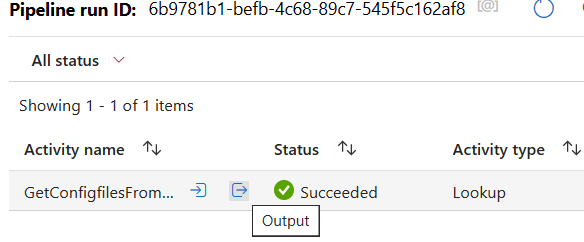
**Create a Dataset for StorageAccount:**

**Step 4:** Create a Dataset>>New Dataset>>in search type blob storage>>click on Azure blob storage>>continue>>**Json**>>continue>>Name: DS\_BlbJsonconfig>>Linked Service: LS\_BlbSA>>File path: click on folder ikon>>click on config>>click on ecdc\_file\_list\_for\_2\_files.json>>ok>>Import schema: None>>ok>>Publish all>>Publish

**Step 5:** Create a new pipeline>>Name: PL\_Dynamicpipeline>>in activities pane search for lookup activity>>drag and drop the lookup activity from activities pane to pipeline canvas>>Click the lookup control and in General tab>>Name: GetConfigfilesFromBlbSA>>click on settings tab>>click on Source dataset box knob>>and click on DS\_BlbJsonconfig>>uncheck First row only checkbox for sure

**Step6:**

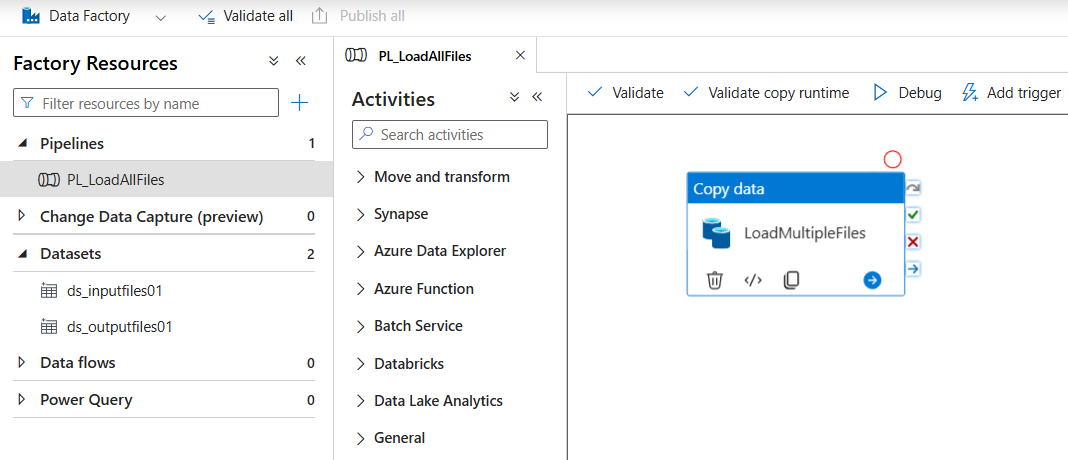
Now publish the pipeline>>Debug>>Now if we click at output and see the window open then count will be 2 bcoz in blob storage account config file we have mentioned **2 .csv files.**



Hence, like this the lookup control will read the files based upon the No of files we are passing in the config file(json file).

# 

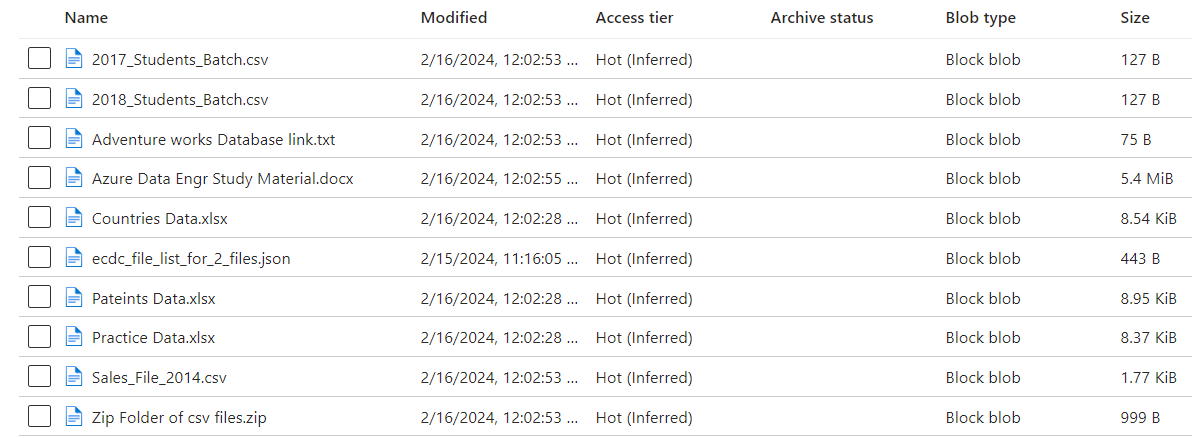
# **Copying data from multiple files:**

****

**Note:** In this use case we are picking/copying all the files from our source SA irrespective of the extensions to load it into our destination SA

**Step1:**

Create a Blob Storage Account(1971blbsa)>>create a container(myconfig) inside the SA>>upload multiple .csv files inside the container as shown in image below also upload other multiple files from your laptop



**Step2:**

Create an Azure Datalake storage Gen2 Storage account(1974adlsa)>>create container(mydestconfig) inside the SA and this is our target SA.

**Step3:**

Create ADF>>Launch ADF studios>>click on Author ikon(left side)>>pipeline>>new pipeline>>Name: PL\_LoadAllFiles>>In Activities pane search copy data>click on General tab>>Name: LoadMultipleFiles>>click on Source tab>>+New>>in search type blob storage>>click on Azure blob storage>>Continue>>Delimited text>>Continue>>Name: ds\_inputfiles01>> Linked service>>+New>>Linked service: LS\_BlbSA>>select the subscription>>Select the source storage account>>Test connections>>Create>>For File path: click on the folder ikon>>click on beside the folder(not on the folder)>>Ok>>ensure First row as header check box is checked>>Import schema: None>>Ok

**Step4:**

Click on the copy data control>>source tab>>For file path type select the radio button Wildcard file path(and this copies all the files from source storage account to adlSA and if we want to copy the data of only .csv files then we can mention for myconfig: \*.csv in 2nd box)>>click on Sink tab>>+New>>in search type Gen2>>click on Azure Datalake Storage Gen2>>Continue>>Delimited text>>Continue>>Name: ds\_outputfiles01>>Linked services:+New>>Name: LS\_destfiles01>>Select the subscription and storage account respectively>>test connection>>Create>>Ok>>For File path click on folder ikon and select the container(mydestconfig: this we have created in step2)Ensue check box first row as header is checked>>Import schema as None>>Ok

**Step5:**Publish all>>Publish>>Debug>>After its get succeeded we can see all the files of all types will get copied to ADL Gen2 SA.

**Note1:** if we want to copy only one file of any extension(like…. .csv, or .xlsx, or .doc or .zip, …etc) then click on the copy control>>source tab>>File path type: File path in dataset>>in source tab>>open>>then for File path click on Browse>>select the one file we want to copy(any one file ex: Zip Folder of csv files.zip)>>ok>>Publish All>>Publish>>Debug

Hence here will see only 1 file will get copies to our destination storage account instead of multiple files.

A screenshot of a computer

Description automatically generated

And if we see in above image we have mentioned .zip file then only this one zip file will get copied from source SA to Destination SA.

**Note2:**

If we want to increase the processing power(DIU’s) of the ADF pipelines then select the copy control>>settings tab>>for Maximum data integration unit:8 or 16 or 32 whatever the value we want we can pass as we increase the value the processing power will increase/performance will increase and quickly the files will get copied from source SA to destination SA (by default it is **Auto** means the No of files we are having in source and to copy it to destination Storage Account it will **increase automatically** based upon the volume of files)

# **Copying the files from GitHub Dynamically with the use of Dynamic parameters allocation-AUTOMATION PROCESS:**

**Note:** In this use case we are going to keep 2 files in our Json in source SA then these 2 files which are there in our Source SA will copied to the destination SA and if I put 2 more files in my Json then without touching my Pipelines the other 2 new files will also get loaded into my destination SA.

**Step1:**

Create a Blob storage account>>create a container inside the storage account as srccon>>come inside the container folder and upload the below .json config file



So here whenever we want to modify or add an extra file then there is no need to touch the ADF pipelines, here directly we can go to the config file and add the new file details like how we have added for the above 2 files.

**Step2:**

Create ADL Storage Gen2 StorageAccount, create one container inside it, and create one more folder (mypracticedata) inside the container

**Creating ADF & Linked service for the storage accounts:**

**Step 3:**

Create ADF>>Launch ADF studios>>click on manage ikon(left side)>>Linked services>>+New>>in search type blob storage>>click on Azure blob storage>>continue>>Name: LS\_BlbSA>>select the subscription & Storage account carefully>>test connection>>create

**Step4:**

Create a linked service(LS\_ADLSGen2Connection) for ADL Storage Gen2 same as above step but this is for Destination Storage account.

**Create a Dataset for Lookup activity:**

**Step 5:** Create a Dataset>>New Dataset>>in search type blob storage>>click on Azure blob storage>>continue>>**Json**>>continue>>Name: DS\_BlbJsonconfig>>Linked Service: LS\_BlbSA>>File path: click on folder ikon>>click on config>>click on ecdc\_file\_list\_for\_2\_files.json>>ok>>Import schema: None>>ok

**Step 6:**

Create a new pipeline>>Name: PL\_DynamicPipeline>>in activities pane search for **lookup activity**>>drag and drop the lookup activity from activities pane to pipeline canvas>>Click the lookup control and in General tab>>Name: GetFilesCount>>click on settings tab>>click on Source dataset box knob>>and click on DS\_BlbJsonconfig>>uncheck First row only checkbox for sure>>Publish all >> Publish >>Debug >> and here in output we see 2 files count as we have kept 2 files in our Json file(in Src Storage Account)

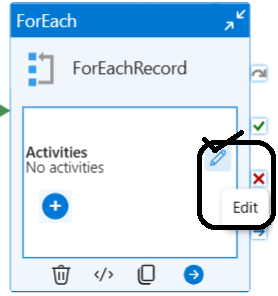
**Step7:** Launch the ADF studios>>Manage>>Linked services>>+New>>In search type HTTP>>click on Http>>continue>>Name: LS\_HTTPConnection>> Authentication type: Anonymous >>Expand the parameters(in same window, just scroll down below)>>+New>>Name: BaseURL>>then scroll up click on Base URL box>>click on Add dynamic content>>click on baseURL>>ok>>click on create>>Publish all >> Publish

**Step8:**

Drag and drop **ForEach activity** after lookup activity from Activities pane to Pipeline canvas and establish a connection between the 2 activities with a green line >>In General tab>> Name: For Each Record>> In settings tab>>click on Items box>>click on Add dynamic content>> just click once on GetFilesCount then will see an expression will get in pipeline expression builder and we have to concatenate/add **.value** in the expression as shown below >> ok

@activity('GetFilesCount').output.value

**Step9:** Click on Edit configuration mark or Pencil Icon on ForEachRecord activity control as shown below then it will navigate inside the control and now in activities pane search CopyData activity and drag and drop in pipeline canvas>>



**Step10:** in General tab>>Name: CopyDataFromHTTPToADLSA>>In Source tab>>+New>>In search type http>>click on http>>continue>>delimited>>continue>>Name: ds\_inputfiles>> Linked service: LS\_HTTPConnection>>Ok

**Step11:**

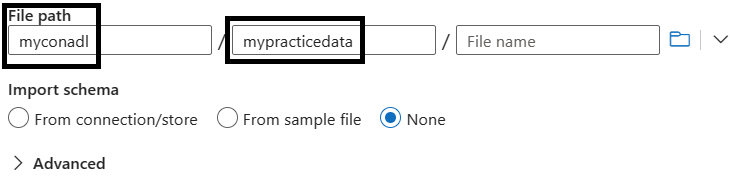
In Source tab>> Click on Open>>Linked service: LS\_HTTPConnection >>click on parameters tab>>+New>>Name:baseURL>>click again on +New>>Name:relativeURL>>click on connection tab>>click on BaseURL text box>>Add dynamic content>>Just click once on BaseURL the expression will get printed in Pipeline expression builder>>ok>> click on RelativeURL text box>>Add dynamic content>>just click once on RelativeUrl >> expression will get printed>>ok

**Step12:**Come to copy data control(CopyDataFromHTTPToADLSA) or activity in the pipeline inside For Each activity>>inside source tab for Source dataset text box change the datasets frequently one or the other just to refresh and finally ds\_inputfiles is our correct dataset only we are doing this refresh bcoz we are unable to see the RelativeURL, to get the RelativeURL we are refreshing by changing the datasets and then we see BaseURL & relativeURL>>click on BaseURL text box>>Add dynamic content>>just click once on For Each Record and add **.sourceBaseURL** as shown below and do the same for relativeURL and add

**.sourceRelativeURL**

@item().sourceBaseURL>>For sourceBaseURL  
@item().sourceRelativeURL>>For relativeBaseURL

**Step13:** Click on Sink tab>>Sink dataset box>>+New>>in search type Gen2>>click on Storage Gen2>>continue>>**Click on delimited**>>Continue>>Name: ds\_outputdata>>Linked service: LS\_ADLSGen2Connection>>For File path pass the details as shown below>>Import schema: None>>Ok



Myconadl>>container we create inside the destination SA

Mypracticedata>>Folder created inside the above container in destination SA.

**Note:** Here we are converting from .csv(source) format to Parquet(destination) format, here if we load the data in Parquet format then we can save lot of space, it is a columnar based format and .csv is row based format, columnar format gives better performance while reading the data while loading the data and it saves lot of space, for big data analytics we use parquet format only, so **Here the data at source is @ .csv but when it get loaded in destination then it will be in Parquet format**, here the process of converting is called analytics, and we are adding the value to the business while changing the formats from source to destination.

**Step14:**

In Sink tab click on open>>click on parameters tab>>+New>>Name: Filename>>click on connections tab>>For File path click on the file name text box>>Add dynamic content>>click on Filename>>Ok

**Step15:**

Click on the pipeline come to CopyData activity(inside For Each activity)>>click on Sink tab>>click on Filename text box>>Add dynamic content>>Click on For Each Record and add .sinkFileName in the pipeline expression builder as shown below>>and finally click on OK>>Publish.

@item().sinkFileName

**Step16:**

Publish>>Debug>>Now if we see in blbsa config file what filenames we have mentioned the same files will get copied to our Destination Storage account and here we have copied the files dynamically with the help of parameters.

**Below is GitHub URL which contains multiple .csv files(as source)**

[AzureDataEngineering\_Batch/ecdc\_data at main · suresh12345/AzureDataEngineering\_Batch · GitHub](https://github.com/suresh12345/AzureDataEngineering_Batch/tree/main/ecdc_data)

Now if we want to copy multiple files like 3, 4, 5, 6….n files like that then there is no need to touch the ADF pipelines or any of the activities/controls we have designed, directly we can add the baseURL, relativeURL & files names in the config file present in the source Storage Account and run the ADF pipeline what all the .csv files we mentioned in config file(Json file) will get copied to our ADLSA Gen SA(destination storage account)



To add the code in the config file in Source SA>>go inside the blob Source SA>>container>>click on the config file>>click on Edit and add the code as shown in above image and then **click on Save above for sure**

Come to the pipeline>>publish (if required)>>Debug>>and here will see all the Four files we have mentioned in the **.json** config file has been copied to the destination Storage Account(ADL SA).

**Hence, we proved here dynamically we are loading or copying the data(files) from source to destination storage account without touching the ADF pipelines again and again**.

**Note:** Click on For Each activity >>Settings tab>>Batch count:2(then it will process 2 files at a time, if pass 4 it will process 4 files at a time whatever the No we are passing accordingly it will process that many files to copy from source to target and the default size for batch count if we won’t mention anything then it is 20)

# **What is the use of Triggers in ADF:**

We use triggers to automate our ADF pipelines, means when we want our ADF pipelines to run at a particular time or at a particular interval without any human intervention then we use triggers.

# Types Triggers:

Basically, we are having 3 types of triggers for ADF pipelines and i.e.:

1) schedule-based triggers

2) Tumbling windows triggers

3) Event based triggers

**Implementing schedule based trigger :**

**(This Demo is same as above demo upto step No: 14):**

**Step1:**

Create a storage account (as 1964blbsaconfig, this is blob storage account not ADL Gen2 storage account)>>create a container inside the storage account as config>>come inside the config folder and upload the below .json config file



**Step2:**

Create ADL Storage Gen2 StorageAccount and create one container inside it and also create one folder(mypracticedata) inside the container

**Creating ADF & Linked service for the storage accounts:**

**Step 3:**

Create ADF>>Launch ADF studios>>click on manage ikon(left side)>>Linked services>>+New>>in search type blob storage>>click on Azure blob storage>>continue>>Name: LS\_BlbSA>>select the subscription & Storage account carefully>>test connection>>create

**Step4:**

Create a linked service(LS\_ADLSGen2Connection) for ADL Storage Gen2 same as above step but this is for Destination Storage account.

**Create a Dataset for Lookup activity:**

**Step 3:** Create a Dataset>>New Dataset>>in search type blob storage>>click on Azure blob storage>>continue>>**Json**>>continue>>Name: DS\_BlbJsonconfig>>Linked Service: LS\_BlbSA>>File path: click on folder ikon>>click on config>>click on ecdc\_file\_list\_for\_2\_files.json>>ok>>Import schema: None>>ok>>Publish all>>Publish

**Step 4:**

Create a new pipeline>>Name: PL\_DynamicPipeline>>in activities pane search for **lookup activity** >> drag and drop the lookup activity from activities pane to pipeline canvas>>Click the lookup control and in General tab>>Name: GetFilesCount>>click on settings tab>>click on Source dataset box knob>>and click on DS\_BlbJsonconfig>> uncheck First row only checkbox for sure>>Publish all >> Publish>>Debug>> and here we see in the output count as 2 bcoz we kept 2 .csv files in our Json config file in Source Storage account

**Step5:** Launch the ADF studios>>Manage>>Linked services>>+New>>In search type HTTP>>click on Http>>continue>>Name: LS\_HTTPConnection>> Authentication type: Anonymous >>Expand the parameters (in same window, just scroll down below)>>+New>>Name: BaseURL>>then scroll up click on Base URL box>>click on Add dynamic content>>click on baseURL>>ok>>click on create>>Publish all>>Publish

**Step6:**

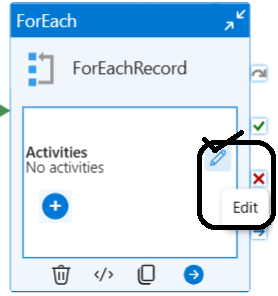
Now publish the pipeline>>Debug>>Now if we click at output and see the window open then count will be 2 bcoz in blob storage account config file we have mentioned 2 csv files.

**Step7:**

Drag and drop ForEach activity after lookup activity from Activities pane to Pipeline canvas and establish a connection between the 2 activities with a green line >>In General tab>>Name: For Each Record>>In settings tab>>click on Items box>>click on Add dynamic content >> just click once on GetFilesCount then will see an expression will get in pipeline expression builder and we have to concatenate/add **.value** in the expression as shown below>>ok

@activity('GetFilesCount').output.value

**Step8:** Click on Edit configuration mark on ForEachRecord activity control as shown below then it will navigate inside the control and now in activities pane search CopyData activity and drag and drop in pipeline canvas>>



**Step9:** in General tab>>Name: CopyDataFromHTTPToADLSA>>In Source tab>>+New>>In search type http>>click on http>>continue>>delimited>>continue>>Name: ds\_inputfiles>> Linked service: LS\_HTTPConnection>>Ok

**Step10:**In Source tab>>Click on Open>>Linked service: LS\_HTTPConnection >>click on parameters tab>>+New>>Name:baseURL>>click again on +New>>Name:relativeURL>>click on connection tab>>click on BaseURL text box>>Add dynamic content>>Just click once on BaseURL the expression will get printed in Pipeline expression builder>>click on RelativeURL text box>>Add dynamic content>>just click once on relative >>expression will get printed>>ok

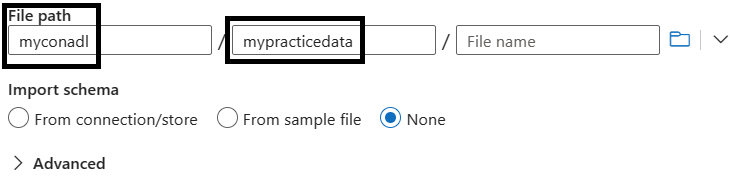
**Step11:**

Come to copydata(CopyDataFromHTTPToADLSA) activity in the pipeline inside For Each activity>>inside source tab for Source dataset text box change the datasets frequently one or the other just to refresh and finally ds\_inputfiles is our correct dataset only we are doing this refresh bcoz we are unable to see the RelativeURL, to get the RelativeURL we are refreshing by changing the datasets and then we see BaseURL & relativeURL>>click on BaseURL text box>>Add dynamic content>>just click once on For Each Record and add **.sourceBaseURL** as shown below and do the same for relativeURL and add

**.sourceRelativeURL**

@item().sourceBaseURL>>For sourceBaseURL  
@item().sourceRelativeURL>>For relativeBaseURL

**Step12:** Click on Sink tab>>Sink dataset box>>+New>>in search type Gen2>>click on Storage Gen2>>continue>>**Click on Parquet**>>Continue>>Name: ds\_outputdata>>Linked service: LS\_ADLSGen2Connection>>For File path pass the details as shown below>>Import schema: None>>Ok



Myconadl>>container we create inside the destination SA

Mypracticedata>>Folder created inside the above container in destination SA.

**Step13:**

In Sink tab click on open>>click on parameters tab>>+New>>Name: Filename>>click on connections tab>>For File path click on the file name text box>>Add dynamic content>>click on Filename>>Ok

**Step14:**

Click on the pipeline come to CopyData activity(inside For Each activity)>>click on Sink tab>>click on Filename text box>>Add dynamic content>>Click on For Each Record and add .sinkFileName in the pipeline expression builder as shown below>>and finally click on OK>>Publish all >> Publish

@item().sinkFileName

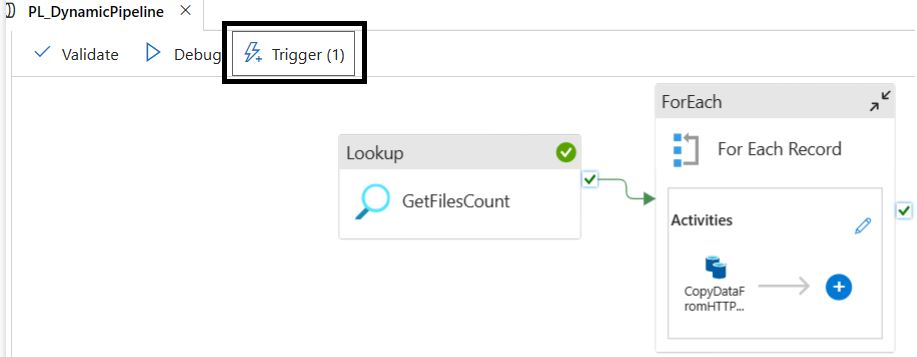
**Step 15:**

Click on Manage ikon (left side under ADF studios)>>Triggers>>+New>>Name: TR\_ScheduleTirgger>>Type: Schedule>>Start date: mention any suitable date & time(ex: 7/5/2023, 4:00:00 AM)>>Time zone: select any suitable time zone>>Recurrence: Every 10 Hour(s) or **Every 1 minute(s)[set this as Recurrence]** or Every Day…etc(we can set whatever we want like weekly, daily, hourly, minute…etc. based upon project requirement)>>click on specify an end date check box and mention the end date>>check the check box to Start trigger on creation>>ok

**Note:** schedule trigger is many to many relationships and this trigger we can allocate to multiple pipelines if we want

**Step16:**

Click on Author ikon(left side)>>Add trigger(center middle on the pipeline)>>New/Edit>>click on choose trigger box>>and here will find the trigger what we have created in above step>>click on that trigger>>Ok>>Ok>>if we notice now then on top center middle at Trigger will find as (1)>>this means we have setup one trigger to this pipeline as shown in below image.



**Step 17:**

Publish>>Click on Monitor ikon (left side) and this time we wont click on Debug manually means(run the pipeline) this time it should get executed/run/debug **by itself** bcoz we have set up a schedule trigger, click on Refresh (on top center) and if we see now the trigger is getting executed automatically by itself and we have not Debug it manually.

**Note:** Every time the pipeline gets trigger then Microsoft will charge us, so we can even stop this trigger>>click manage ikon (left side)>>Triggers(stop)>>click on Stop button(middle as shown in below image), once we stop the trigger we can see the status of the trigger as stopped and finally click on Publish to save the changes.

A screen shot of a computer

Description automatically generated

# **Provisioning of Azure SQL Data Base in Cloud Platform:**

Azure SQL Database is a fully managed platform as a service (PaaS) database engine that handles most of the database management functions such as upgrading, patching, backups, and monitoring without user involvement. Azure SQL Database is always running on the latest stable version of the SQL Server database engine and patched the OS with [99.99% availability](https://azure.microsoft.com/support/legal/sla/azure-sql-database). PaaS capabilities built into Azure SQL Database enable us to focus on the domain-specific database administration and optimization activities that are critical for our business.

With Azure SQL Database, we can create a highly available and high-performance data storage layer for the applications and solutions in Azure. SQL Database can be the right choice for a variety of modern cloud applications because it enables us to process both relational data and [non relational structures](https://learn.microsoft.com/en-us/azure/azure-sql/multi-model-features?view=azuresql), such as graphs, JSON, spatial, and XML.

Azure SQL Database is based on the latest stable version of the [Microsoft SQL Server database engine](https://learn.microsoft.com/en-us/sql/sql-server/sql-server-technical-documentation?toc=/azure/sql-database/toc.json). we can use advanced query processing features, such as [high-performance in-memory technologies](https://learn.microsoft.com/en-us/azure/azure-sql/in-memory-oltp-overview?view=azuresql) and [intelligent query processing](https://learn.microsoft.com/en-us/sql/relational-databases/performance/intelligent-query-processing?toc=/azure/sql-database/toc.json). In fact, the newest capabilities of SQL Server are released first to Azure SQL Database, and then to SQL Server itself. You get the newest SQL Server capabilities with no overhead for patching or upgrading, tested across millions of databases.

SQL Database enables us to easily define and scale performance within two different purchasing models: a [vCore-based purchasing model](https://learn.microsoft.com/en-us/azure/azure-sql/database/service-tiers-vcore?view=azuresql) and a [DTU-based purchasing model](https://learn.microsoft.com/en-us/azure/azure-sql/database/service-tiers-dtu?view=azuresql). SQL Database is a fully managed service that has built-in high availability, backups, and other common maintenance operations. Microsoft handles all patching and updating of the SQL and operating system code, we don't have to manage the underlying infrastructure.

# **Implementation steps to host a DB & DB Server in Azure cloud computing & loading the data from Blb SA(Source) to SqlDB(Destination/Target) in single table:**

A screenshot of a computer

Description automatically generated

**Step1:**

Create a blob Storage Account>>create container inside the SA and upload multiple .csv files inside the container.

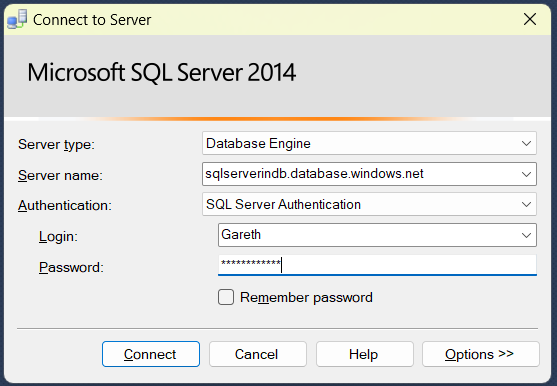
**Step2**

Search for SQL DB in Azure portal>>click on SQL DB>>Create>>in Basics tab fill the details accordingly>>Database name: azuresqldatabase001>>server:azuresqlserverdemo001>>Backup storage redundancy: Locally-redundant backup storage>>click on Next: Networking>>connectivity method: Public endpoint>>Allow Azure services and resources to access the server: Yes>>Add current client IP address: Yes>>click on Next: security>>Enable Microsoft Defender for SQL :Not now>>click on Next: Additional settings>>click on Next: Tags>>click on Review+Create>>Create.

**Step3:**

Deploy DB and DB server in Azure portal and after the DB & DB server got deployed in Azure portal>>go inside the DB or click on the DB(which we have deployed)>>click on Query editor(preview)>>pass the login(ex: Gareth) and password(ex:Shaikpet@123) this we gave at the time of DB server creation in Azure portal then here if we expand the tables, Store Procedures all seems blank>>Click on compute+Storage(left side)>>Ok then here will see all the details that we have considered at the time of DB deployment>>connection strings>>properties>>Locks…all these are the features of SQL DB.

**Step4:** Launch the SSMS in your laptop>>and pass the details as shown below and click on connect then here will login to the Azure DB server which we have created in Azure cloud portal.



**Step5:**

Create ADF>>Launch ADF studios>>Manage(left side)>>+New>>in search type blob>>click on Azure blob storage>>continue>>Name: LS\_FilesToSqlDB>>select the subscription and SA carefully>>test connection>>Create

**Step6:**

click on Author>>Pipeline>>New pipeline>>Name: PL\_BLOB\_TO\_SQLDB>>in activities pane search for copydata activity and drag and drop on pipeline canvas>>click on General tab>>Name: Copy\_data\_from\_blob\_to\_SqlDB>>click on Source tab>>+New>>In search type blob>>click on Azure blob storage>>Continue>>Delimited text>>Continue>>Name: ds\_inputdataset>>Linked service: LS\_FilesToSqlDB>>For file path: click on the folder ikon>>click on any one of the file>>say ok>>Ensure to check the box for first row as header>>Import schema as : None>>Ok

**Step7:**

Come to pipeline>>Click on Sink tab>>+New>> in search type sql>>click on Azure SQL Database>>Continue>>Name: ds\_SQLConnection>>Linked service: +New>>Name: LS\_CsvFilesToSqlDB>>Select the subscription, DB server name whatever we gave at the step1, and DB carefully>>Authentication Type: SQL authentication>>Username: Gareth>>Password: Shaikpet@123>>testconnection>>Create>>check the checkbox Enter manually>>Import schema: None>>Ok

**Step9:**

Goto pipeline>>click on the copy data activity/control>>click on Sink tab>>open>>check the check box Enter manually>>for schema name text box: dbo>>for table name text box: Product(we can give any name and this is going to be our table in SQLDB in Azureportal).

**Step10:**

Goto pipeline>>click on the copy data activity/control>>click on Sink tab>>For Table option: Auto create enable(with this it will automatically create a table for us in target(i.e.: SQLDB)>>Publish all >> Publish >> Debug

**Step11:**

**Performing the testing:**

Now come to SQLDB in Azure portal>>click on Query editor (left side)>>make a login with username and password>>expand the tables folder here will find a table got created with the name we pass along with a data. Just pass the below query to verify the data in SQL DB

Select \* from [dbo].[Product]

Hence here we have uploaded the data in SQLDB table from .CSV file using ADF Pipelines and Copy Activity.

# **Implementation steps for loading the data from Blb SA to SqlDB in multiple table:**

A screenshot of a computer

Description automatically generated

**Step1:**

Create an SQL DB in Azure portal as per the above demo process and procedure.

**Step2:**

Create a blob Storage Account>>create container inside the SA, upload multiple .csv files, and excel files inside the container.

**Step3:**

Create ADF>>Launch ADF studios>>Manage (left side)>>+New>>in search type blob>>click on Azure blob storage>>continue>>Name: LS\_FilesToSqlDB>>select the subscription and SA carefully>>test connection>>Create.

**Step4:**

Launch ADF>>Author>>Dataset>>New Dataset>>In search type blob>>Azure blob storage>>continue>>delimited>>continue>>Name: ds\_blbfiles>>Linked service: LS\_FilesToSqlDB>>For file path click on folder ikon>>click the container beside(don’t choose any specific file here)>>Ok>>check the checkbox as first row as header>>Import schema as none>>ok

**Step5:**

Click on Author>>Dataset>>click on 3 dots>>New Dataset>>in search type blob>>Click on Azure blob storage>>continue>>Delimited>>continue>>Name: ds\_blbfilestosql>>Linked service: LS\_FilestoSQLDB>>For File path click on Folder icon>>click on container (this we create inside the SA)>>don’t select any one particular file, just click on the container beside>>Ok>>check the checkbox as first row as header>>Import schema as none>>ok>>Publish all>> Publish

**Step6:**

Click on Author>>Pipeline>>New pipeline>>Name: PL\_BLOB\_TO\_SQLDB>>in activities pane search for Get Metadata activity/control and drag and drop in pipeline canvas>>In General tab>>Name: Get Files>>click on settings tab>>for Dataset: ds\_blbfiles

**Step7:**

Come to pipelines>>select the Get Metadata control>>click on settings tab>>For Field list click on +New>>and for newly appeared box select Child items>>Publish all>>Publish>>Debug and if we see in output it will show all the files that we have kept in our blob storage account

**Step8:**

Come to pipelines>>in activities pane search for filter activity>>drag and drop the filter activity after Get Metadata activity control>>Establish a connection with green line>>in General tab>>Name: Filter CSV Files>>click on settings tab>>click on items text box>>Add dynamic content>>just click once on Get Files(Get Files activity output) and the expression will get in the pipeline expression builder and add **.childitems** as shown below>>and finally click on ok

@activity('Get Files').output.childItems

Click on condition text box(just below)>>Add dynamic content>>and type the below expression>>and finally click on ok

@endswith(item().name,'.csv')

Now click on Publish All>>Debug and here will see the output for the 2 controls/activity i.e.: Get Metadata control and Filter control

**Step9:**

Now search for **ForEach** control and drop in pipeline canvas after Filter control>>establish a connection with green line b/w Filter control and ForEach control>>Click on ForEach activity>>in General tab>>Name: ForEachFile>>Click on settings tab>>click on items text box>>add dynamic content>>and click on Filter CSV Files(Filter CSV Files activity output) and add **.value** as shown below >>click Ok

@activity('Filter CSV Files').output.value

**Step10:**

Now double click on ForEach activity(to go inside the ForEach activity)>>in Activities pane search for copy data control/activity and paste it in pipeline canvas>>In General tab>>Name: CopyFilesFromBlbStorageToSQLDB>>click on source tab>>For Sourcedataset box choose ds\_blbfilestosql>>click on open(to go inside the dataset)>>Click on parameters tab>>+New>>Name: SourceFiles>>click on connections tab>>click on Filename text box>>add dynamic content>>Just click once on SourceFiles then an expression will get printed on Pipeline Expression Builder>>Ok

**Step11:**

Click on copyData activity>>Source tab>>click on SourceFile text box>>Add dynamic content>>click on ForEachFile and an expression will get printed and add **.name** as shown below>>click on Ok

@item().name

**Step12:**

Click on CopyControl>>Sink tab>>+New>>in search type SQL>>click on Azure SQL Database>>continue>>Name: ds\_sqltables>>click on Linked service box>>+New>>Name: LS\_Filesinsqltables>>select subscription, sql server name, DB name, Username, Password accordingly and then click on test connection>>click create>>click on Enter manually check box>>Import schema as None>>and click on Ok finally.

**Step13:**

Click on Open(to go inside the dataset) in Sink tab>>Click on parameters tab>> +New>>Name: TableName>>click back on connection tab>>click on Enter manually check box>>schema name text box type **dbo** and for table name text box click Add dynamic content>>click TableName the expression will get printed in pipeline expression builder>>Ok

**Step14:**

Come to pipeline at copy control>>in Sink tab>>click on TableName text box>>Add dynamic content>>Click on ForEachFile then expression will get printed as shown below and add **.name**>>Ok.

@item().name

**Step15:**

Come to copydata activity>>click on sink tab>>For Table option choose Auto create table radio button>>Publish All>>Publish>>Debug>>Now go inside the SQLDB in azure portal click on Query Editor>>pass the creds>>and here we can see all the tables got created along with data in it. We may see the same in SSMS in our laptop.

**Step16:**

If we see in the SQLDB all the tables got created and data got loaded into the tables but all the tables has an extension of .csv and if we want to remove the .csv extension for each and every table then login to SSMS delete all the tables with below command

Drop Table table\_name1,table\_name2,table\_name3,table\_name4;>>SQL Command

**Step17:**

Come to copy activity>>sink tab>>click on TableName text box>>Add dynamic content>>and put the below expression in pipeline expression builder>>and click ok finally>>Publish All>>Debug

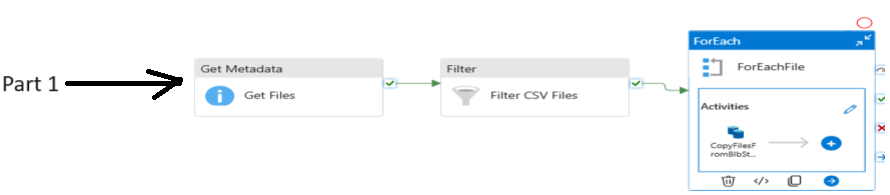
@replace(item().name,'.csv','')

Hence, now all the tables are created again in our DB without .csv extension with all the data inside the tables.

# **Implementation steps to copy the data from Blob Storage Account to SQLDB and then SQLDB to Gen2SA & executing multiple pipelines in One Go**:

**Part1:** copy the data from Blob Storage Account to SQLDB

**Part2:** copy the date from then SQLDB to Gen2SA



A screenshot of a computer

Description automatically generated

**Step1:**

Create a blob Storage Account>>create container inside the SA and upload multiple .csv files and excel files inside the container.

**Step2:**

Create an SQL DB in Azure portal as per the above demo process and procedure.

**Step3:**

Create ADF>>Launch ADF studios>>Manage (left side)>>+New>>in search type blob>>click on Azure blob storage>>continue>>Name: LS\_FilesToSqlDB>>select the subscription and SA carefully>>test connection>>Create.

**Step4:**

Launch ADF>>Author>>Dataset>>New Dataset>>In search type blob>>Azure blob storage>>continue>>delimited>>continue>>Name: ds\_blbfiles>>Linked service: LS\_FilesToSqlDB>>For file path click on folder ikon>>click the container beside(don’t choose any specific file here)>>Ok>>check the checkbox as first row as header>>Import schema as none>>ok

**Step5:**

Click on Author>>Dataset>>click on 3 dots>>New Dataset>>in search type blob>>Click on Azure blob storage>>continue>>Delimited>>continue>>Name: ds\_blbfilestosql>>Linked service: LS\_FilestoSQLDB>>For File path click on Folder ikon>>click on container(this we create inside the SA)>>don’t select any one particular file, just click on the container beside>>Ok>>check the checkbox as first row as header>>Import schema as none>>ok>>Publish all

**Step6:**

click on Author>>Pipeline>>New pipeline>>Name: PL\_BLOB\_TO\_SQLDB>>in activities pane search for Get Metadata activity/control and drag and drop in pipeline canvas>>In General tab>>Name: Get Files>>click on settings tab>>for Dataset: ds\_blbfiles

**Step7:**

Come to pipelines>>select the Get Metadata control>>click on settings tab>>For Field list click on +New>>and for newly appeared box select Child items>>Publish all>>Debug and if we see in output it will show all the files that we have kept in our blob storage account

**Step8:**

Come to pipelines>>in activities pane search for filter activity>>drag and drop the filter activity after Get Metadata activity control>>Establish a connection with green line>>in General tab>>Name: Filter CSV Files>>click on settings tab>>click on items text box>>Add dynamic content>>just click once on Get Files(Get Files activity output) and the expression will get in the pipeline expression builder and add **.childItems** as shown below>>and finally click on ok

@activity('Get Files').output.childItems

Click on condition text box(just below)>>Add dynamic content>>and type the below expression>>and finally click on ok

@endswith(item().name,'.csv')

Now click on Publish All>>Debug and here will see the output for the 2 controls/activity i.e.: Get Metadata control and Filter control

**Step9:**

Now search for **ForEach** control and drop in pipeline canvas after Filter control>>establish a connection with green line b/w Filter control and ForEach control>>Click on ForEach activity>>in General tab>>Name: ForEachFile>>Click on settings tab>>click on items text box>>add dynamic content>>and click on Filter CSV Files(Filter CSV Files activity output) and add **.value** as shown below >>click Ok

@activity('Filter CSV Files').output.value

**Step10:**

Now double click on ForEach activity(to go inside the ForEach activity)>>in Activities pane search for copy data control/activity and paste it in pipeline canvas>>In General tab>>Name: CopyFilesFromBlbStorageToSQLDB>>click on source tab>>For Sourcedataset box choose ds\_blbfilestosql>>click on open(to go inside the dataset)>>Click on parameters tab>>+New>>Name: SourceFiles>>click on connections tab>>click on Filename text box>>add dynamic content>>Just click once on SourceFiles then an expression will get printed on Pipeline Expression Builder>>Ok

**Step11:**

Click on copyData activity>>Source tab>>click on SourceFile text box>>Add dynamic content>>click on ForEachFile and an expression will get printed and add **.name** as shown below>>click on Ok

@item().name

**Step12:**

Click on CopyControl>>Sink tab>>+New>>in search type SQL>>click on Azure SQL Database>>continue>>Name: ds\_sqltables>>click on Linked service box>>+New>>Name: LS\_Filesinsqltables>>select subscription, sql server name, DB name, Username, Password accordingly and then click on test connection>>click create>>click on Enter manually check box>>Import schema as None>>and click on Ok finally.

**Step13:**

Click on Open(to go inside the dataset) in Sink tab>>Click on parameters tab>> +New>>Name: TableName>>click back on connection tab>>click on Enter manually check box>>schema name text box type **dbo** and for table name text box click Add dynamic content>>click TableName the expression will get printed in pipeline expression builder>>Ok

**Step14:**

Come to pipeline at copy control>>in Sink tab>>click on TableName text box>>Add dynamic content>>Click on ForEachFile then expression will get printed as shown below and add **.name**>>Ok.

@item().name

**Step15:**

Come to copydata activity>>click on sink tab>>For Table option choose Auto create table radio button>> in sink tab only>>click on TableName text box>>Add dynamic content>>and put the below expression in pipeline expression builder>>and click ok finally>>Publish All>>Publish>>Debug

@replace(item().name,'.csv','')

Now do publish all>>publish>>Debug then her will see all the tables are created in our DB with all the data inside the tables.

**Step16:**

Create ADL Gen2 Storage account(ex: sqltoadlsa) and container(ex: destcon) inside the SA.

**Step17:**

Create a new pipeline>>Name: PL\_SQLDB\_TOADL>>drag and drop lookup activity>>in General tab>>Name: GetTables>>in Settings tab>>+New>>in search type Sql(bcoz we are pulling the data from SQL DB)>>continue>>Name: ds\_inputtables>>Linked service:LS\_Filesinsqltables(this Linked service we have created in step 12)>>check the checkbox Enter manually>>Import schema as None>>Ok

**Step18:**

Create a new Dataset>>in search type Gen2>>click on Azure Data Lake Storage Gen2>>Continue>>DelimitedText>>Continue>>Name: ds\_ADLSGen2>>Linked service: +New>>Name: LS\_ADLSGen2>>select the subscription, Storage Account(sqltoadlsa) carefully>>test connections>>create>>for Filepath:destcon(destination storage account container)>>Import schema: None>>Ok

**Step19:**

Come to Lookup activity>>In Settings tab only>>uncheck First row only checkbox>>for Use query click on query radio button then a box will appear for us>>click on that box>>click on Add dynamic content>>and paste the below query

SELECT

\*

FROM

database\_name.INFORMATION\_SCHEMA.TABLES

WHERE table\_type = 'BASE TABLE'

**Note:** Here in above query for database\_name pass the SQLDB name what we gave at the time of DB creation in azure portal.

Ex: database\_name as NareshDB1947

And after pasting the query with DB name changes click Ok>>Publish All>>Publish>>Debug

**Step20:**

Drag and drop(D&D) ForEach activity in pipeline after lookup activity establish a connection>>Click on ForEach activity>>In General tab>>Name: ForEachTable>>in Settings tab>>click on items check box>>Add dynamic content>>click on GetTables(GetTables activity output) and add **.value** as shown in below expression>>click ok

@activity('GetTables').output.value

**Step21:**

Double click on ForEach activity>>drag and drop copy activity/control>>In General tab>>Name: CopyDataFromSQLDBToADLSA>>In Source tab>>+New>>in search type sql>>click on Azure SQL Database>>Continue>>Name: ds\_inputsqltables>>Linked service: LS\_Filesinsqltables>>check the box Enter manually>>Import schema: None>>Ok>>Click on open(inside source tab only)>>click the Enter manually checkbox>>click on parameters tab>>+New>>For Name text box pass Table\_Schema>>click again on +New>>For Name text box pass Table\_Name

**Step22:**

Come to pipeline>>click on copy activity>>in Source tab>>For Source dataset text box drop down choose ds\_inputsqltables dataset(this is just a refresh we are doing to populate the parameters)>>click on Table\_schema txt box>>Add dynamic content>>click on ForEachTable and add **.Table\_SCHEMA** as shown below and say ok

@item().TABLE\_SCHEMA

Click on Table\_Name text box>>Add dynamic content>>click on ForEachTable and add **.TABLE\_NAME** as shown below and finally click on ok

@item().TABLE\_NAME

**Step23:**

Come to Copy Data activity>>Now click on Sink tab>>for Sink dataset choose ds\_ADLSGen2>>click on +New(in sink tab only)>>in search type gen2>>click on Azure Data Lake Storage Gen2>>Continue>>Delimited text>>continue>>Name:ds\_outputfiles>>Linked service:LS\_ADLSGen2>>click on folder ikon>>click on container>>in Directory box pass as OutputTables>>check first row as header>>Import schema as none>>Ok>>Now click on open(in sink tab only)>>click on parameters tab>>+New>>Name:Filename>>click back on connection tab>>click on filename text box>>Add dynamic content>>click on Filename>>ok

**Step24:**

Come to copy activity>>sink tab>>for Sink dataset carefully choose ds\_outputfiles>>click on Filename text box>>Add dynamic content and paste the below expression in pipeline expression builder and click OK finally.

@concat(item().TABLE\_SCHEMA,'\_',item().TABLE\_NAME,'.csv')

**Step25:**

Click on copy activity>>click on source tab>>click open>>click on connection tab>>click on schema name txt box>>add dynamic content>>click on Table\_ Schema>>Ok>>click on table name text box>>add dynamic content>>click on Table\_Name>>ok

Publish All>>Publish>>Debug>>Hence now we can see what all the tables we have in SQL DB will be loaded into our ADL Gen2 Storage account in the form of .csv files.

**Note:** Now delete the tables from DB and .csv files from destination storage account **for sure,** so that when we execute the pipelines again and can load the data as shown below

# Execution of above pipelines in sequence one after the other:

If we want to execute multiple pipelines one after the others based upon the project requirements then we can create a new pipeline>>Name:PL\_Executepipeline>>in activities pane search Execute pipeline activity>>drag and drop in pipeline canvas>>In General tab>>Name: ExecuteFirstPipeline>>In Settings tab>>For Invoked pipeline choose PL\_BLOB\_TO\_SQLDB>>drag and drop Execute pipeline activity again one more time from activities pane to pipeline canvas>>establish a connection between the two activities>>In General tab>>Name: ExecuteSecondPipeline>>In Settings tab>> For Invoked pipeline choose PL\_SQLDB\_TOADL>>Publish All>>Publish>>Debug.

# **When can we pick up schedule-based triggers??tumbling window base triggers?? & Event base triggers??**

**Tumbling window base triggers(TWBT):**

It is one-one relationship trigger, we cannot allocate multiple pipelines to tumbling window base trigger, we can allocate only one pipeline at a time for tumbling window base trigger, there are certain properties in tumbling window base trigger(TWBT), we can use this TWBT when we want to process huge volume of files or small volume of files and we want to process those sequentially and based upon the requirement we are trying to process those then in these cases we can use TWBT, we can schedule this TWBT at least for every 5 mins or more than 5 mins but not less than 5 mins, a TWBT can depend on a maximum of five other triggers.

# **Implementation steps for Tumbling Window Based Trigger:**

**Step1:**

Create a Blb Storage account>>container inside it and upload some .CSV files.

**Step2:**

Create ADF>>Launch ADF Studios

**Step3:**

Create a Linked service(LS\_BlbSA) and Dataset(ds\_blbfiles) inside the ADF for Blob Storage Account(Src SA).

**Step4:**

Create a pipeline >>Drag and Drop GetMetada control/activity in pipeline canvas>>In Genera tab>>Name: GetMetadataOfFiles>>In settings tab>>Dataset:ds\_blbfiles>>for Field list click on +New and in the box choose Child items >>Publish All>>Publish

**Step5:**

Goto Manage(inside ADF studios)>>Triggers(left side)>>+New>>Name: MyTWBT(any name)>>Type: click on the box and choose Tumbling window>>Start date: 7/11/2023, 5:05:30 PM(any time)>>Recurrence for Every : 5 Hours>check the box for Start trigger on creation>>ok

**Step6:**

Come to pipeline>>Add Triggers>>New/Edit>>Choose trigger: here we have to select the tumbling window trigger(MyTWBT) that we have created>>Ok>>Ok>Publish All>>Publish

**Step7:**

Come to monitor(left side) and here we see MyTWBT will be triggered for every 5 Minutes as we setup. Here we can see what all the different Trigger we have schedule in Trigger by name box, we can see the status of succeeded triggers, Failed trigger, waiting trigger, Running trigger…etc.

# **Storage or Event base triggers:**

Some time we don’t know when the file(source file….ex: .csv files) will be landed into a Storage Account (Blob SA-Source), may be every day 2PM as per UTC time zone, and the customer is also putting the file in Blob SA and at the same time we are running our pipelines or we are not sure at what time the customer is going to put the files in Blob SA then using the Event based trigger. We can pick the files and process them through pipeline and load it into the target. Hence for this scenario we create this event based trigger.

# **Implementation steps for Storage & Event based triggers:**

**Step1:** Create a Storage Account>>create a container in it>>place .csv files inside the container.

**Step2:**

Create an ADF>>Create a Linked service>>Create Dataset>>Publish All

**Step3:**

Create a Pipeline>>Drag and Drop GetMetada activity/control>>Give the name in general tab>>in settings tab pass the dataset, click on +New for Field list and select child items>>Publish All>>Debug

**Step4:**

Click on Manage ikon>>Triggers>>+New >>Name: MyEventBasedTrig>>Type: Storage events>>Select the subscription, Storage Account, all containers inside the SA>>Blob path ends with: .csv(if particularly we want to pick the csv files only)>>For Event: check the box Blob is created(means when the file is placed)>>check the box Start trigger on creation>>Continue>>Ok>>Publish All.

**Step5:**

Come to pipeline>>Add trigger>>New/Edit>>MyEventBasedTrig>>Continue>>Continue>>Ok>>Publish All>>Publish

Now whenever someone placed a new file of .csv extension in Blb SA container then this trigger will get initiated and execute the ADF pipeline.

**Step6:**

Now let us put one .csv file in our storage account and then come to monitor inside ADF studios>>Trigger runs(left side)>>refresh>then here will see a trigger is been initiated.

# **Azure Key Vault:**

Azure Key Vault is a cloud service for securely storing and accessing secrets(passwords). A secret is anything that we want to tightly control access to, such as API keys, passwords, certificates. Key Vault service supports two types of containers: vaults and managed hardware security module (HSM) pools. Vaults support storing software and HSM-backed keys, secrets, and certificates. Managed HSM pools only support HSM-backed keys.

# **Why we use Azure Key Vault:**

Centralizing storage of application secrets in Azure Key Vault allows us to control their distribution. Key Vault greatly reduces the chances that secrets/passwords may be accidentally leaked. When application developers use Key Vault, they no longer need to store security information in their application. Not having to store security information in applications eliminates the need to make this information part of the code. For example, an application may need to connect to a database. Instead of storing the connection string in the app's code, we can store it securely in Key Vault.

Our applications can securely access the information they need by using URIs. These URIs allow the applications to retrieve specific versions of a secret. There's no need to write custom code to protect any of the secret information stored in Key Vault….for further information about Azure Key Vault please refer the link [Azure Key Vault Overview - Azure Key Vault | Microsoft Learn](https://learn.microsoft.com/en-us/azure/key-vault/general/overview)

# **Implementation Steps for deploying & Integrating Azure key Vault with ADF:**

**Step1:**

Search for Key vault in Azure portal>>click on Azure Key vault>>+Create>>and fill in the details accordingly.

Subscription: Free trial(or choose accordingly)

Resource Group: NareshRG(or any name)

Key Vault Name: NareshKV1911(or any name)

Region: EastUS(or any)

Pricing tier: Standard

Purge Protection: Disable

Click on Next>>Permission model: Vault access policy>>click on Next>>click on Next>>Pass the tags(if needed else this is optional)>>Next>>Review+Create>>After all validations get passed click on Create

**Step2:**

Create SQL server & SQL DB and DB make the connection string accordingly to place it in an Azure keyvault as shown below.

**Standard connection string:**

Data Source=tcp:<servername>.database.windows.net,1433;Initial Catalog=<databasename>;User ID=<username>@<servername>;Password=<password>;Trusted\_Connection=False;Encrypt=True;Connection Timeout=30

**Make the connection string as below Example:**

Data Source=tcp: **sqlserverinazure1921.database.windows.net**,1433;Initial Catalog= **NareshDB**UserID=Gareth@**sqlserverinazure**;Password=Shaikpet@123;Trusted\_Connection=False;Encrypt=True;Connection Timeout=30

Data Source=tcp:sqlserverinazure4445.database.windows.net,1433;Initial Catalog=NareshDB1990UserID=Gareth@sqlserverinazure;Password=Shaikpet@123;Trusted\_Connection=False;Encrypt=True;Connection Timeout=30

**Step3:**

Go inside the Keyvault(in another tab)>>secrets>>+Generate/Import>>Name: SQLConnectionString>>For Secret value: pass the complete connection that we have created above>>finally click on Create.

**Step4:**

Create an ADF

**Step5:**

Go to keyvault>>Access configuration(left side)>>Go to access policies>>+Create>>click on Configure from a template box and choose Key & Secret Management>>click Select all for Key permissions/Secret permissions/Certification permissions>>Next>>in search box type NareshADF1911(this is the ADF name we gave/created so give the ADF accordingly) and click on it>>Next>>Next>>Create.

**Step6:**

Come to ADF and Launch ADF Studios(in other tab)>>click on Manage(left side)>>Linked services>>+New>>in search type SQL>>click on Azure SQL Database>>continue>>Name: LS\_SQLKeyVault>>scroll down a little and then click on Azure Key Vault>>click AKV linked service box>>+New>>Name:LS\_Keyvault>>click on Enter manually radio button>>for Base URL pass Vault URI value(this will get in Keyvault overview)>>test connection>>Create>>Refresh the Secret name carefully>>select SQLConnectionString>>Give the Server name, subscription name, DB Name>>Create>>Publish All>>Publish>>Debug

**Note:** If we noticed here, 2 linked service we have created

**LS\_Keyvault:** this linked service we have created for Keyvault.

**LS\_SQLKeyVault:** this linked service we have created to access the SQL DB via Keyvault

If we click on LS\_SQLKeyVault then in right side window open we can see 2 options i.e: Connection string & Azure Key Vault and if we click on Connection string and select the radio button From Azure Subscription, server name we must pass, DB name we have to pass and for **Authentication type** we can choose any one option like

(i)SQL authentication (here we have pass the user name and password we gave at the time of DB and DB server creation)

(or)

(ii)System Assigned Managed Identity (here will get all the things by default) for secure purpose we use this option

**To view the access on Azure Key vault:**

Go inside the Key vault>>Access control IAM>>view my access>>then here will find Service Administrator access that we have for our KeyVault

**Vault URI:**

If we are having a web application and DB and the DB password is stored in our Key Vault and we want our application to get the DB password, connect and communicate with the DB and for this in our application setting will give the Vault URI.

Access policies (left side)>>Here will see the global admin user details of our subscription who has created the Key and have admin access rights on our subscription.

Access Configuration (left side)>>Here we can see in permission model as

(i)Vault access policy>>By default Key vault will get created with this permission.

(ii)Azure role-based access control.

**Keys:** Here we can store keys example if we want to encrypt the disk of our VM then we can store such keys here to do the VM disk encryptions.

**Secrets:** Here we can store the passwords, example a DB password. Profile password, SharePoint site passwords…etc.

**Purge Protection:** When we Enable the purge protection while creating the key vault then no one can delete the secrets, keys or certificates from our Azure key vault.

**To store the secrets in key vault:**

Click on the Secrets(left side)>>Generate/Import>>Name: NareshDBPassword (this is just a secret name not a userid or username or DB server name)>>Secret value: Shaikpet@123>>Create

**Certificates:** If we have applied SSL certificates on our webapps either on App services or on Azure VM’s and if we want to retrieve that certificate from key vault then such certificates, we can store it here in Azure key vault.

After creating a keyvault and adding a secret in the KV and if we login to the Azure Subscription with different user credentials the user unable to see the keyvault and other resources in the subscription and now let us give the owners permission to that user at subscription after that if we login with the user credentials then the user can see the resources in that subscription along with key vault but not the secrets in the key vault.

**Now in order to access the secrets/keys/certificates from key vault the user must be added in Access Policies with some permissions** then after that if we login with that user access then we can see the secrets with that user credentials.

**Azure Virtual Machine for Deployment:** If we want to retrieve any secret/key/certificate from Virtual machine then we can use this option.

**Azure Resource manager for template deployment:** If we are deploying a resource through ARM templates and integrating the key vault in ARM template and wanted from key vault that any key/secret/certificate to retrieve from ARM template then we can use this option.

**Azure Disk encryption for Volume encryption:** At the time of disk encryption for Virtual machine and we want the key to retrieve from the key vault for disk encryption for our VM then we can use this option.

# **What is GitHub:**

GitHub is a website and cloud-based service that helps developers store and manage their code/data, as well as track and control changes to their code. To understand exactly what GitHub is, we need to know two connected principles:

* Version control
* Git

**Essentials features of GitHub are:**

* Repositories
* Branches
* Commits
* Pull Requests
* Git (the version control software GitHub is built on)

## **Repository**

* A GitHub repository can be used to store a development project.
* It can contain folders and any type of **files** (HTML, CSS, JavaScript, Documents, Data, Images).
* A GitHub repository should also include a **license** file and a **README** file about the project.
* A GitHub repository can also be used to store ideas, or any resources that you want to share.

## **Branch**

* A GitHub branch is used to work with different versions of a repository at the same time.
* By default, a repository has a master branch (a production branch).
* Any other branch is a copy of the master branch (as it was at a point in time).
* New Branches are for bug fixes and feature work separate from the master branch. When changes are ready, they can be merged into the master branch. If you make changes to the master branch while working on a new branch, these updates can be pulled in.

## **Commits**

At GitHub, changes are called commits.

Each commit (change) has a description explaining why a change was made.

## **Pull Requests**

* Pull Requests are the heart of GitHub **collaboration**.
* With a pull request you are **proposing** that your changes should be **merged** (pulled in) with the master.
* Pull requests show content **differences**, changes, additions, and subtractions in **colors**(green and red).
* As soon as you have a commit, you can open a pull request and start a discussion, even before the code is finished.

# **Creating an account in GitHub:**

Got to the below link to create and account in GitHub

<https://github.com/signup>

1)Enter an email address (any mail is fine Gmail, outlook, yahoo mail, Hotmail…etc. But a GitHub account should not have been created previously with the same mail id)

2)Password

3)Username

4)will get an OTP on our mail pass the OTP and will get login to the GitHub platform.

# **Implementation steps to setup the code repository for ADF in GitHub Platform (or) Integrating ADF with GitHub Platform:**

**Step1:** Come to Azure portal and create a Blob SA, ADF, and then Linked service in ADF, Dataset in ADF and a pipeline, just put one simple activity/control GetMetada then publish all>>publish and debug the pipeline.

**Step2:** If already have an account in GitHub platform, then go to the below link

<https://github.com/signup>

click on Sign in on top right, pass the mail ID & Password and make sign in.

3)Click on black color ikon(top left)>>click on New (available in green color left side)>>Repository name: ADF-Repo>>scroll down a little>>Description: My repo creation for Nareshit Students for ADF artifacts >>choose public/private option (as per project requirements). Here I am choosing public>>check the box Add a README file>>Create repository.

4)come to Azure portal>>click on Azure Data factory>>choose Set up code repository as shown in below image>>

A screenshot of a computer

Description automatically generated

Repository type: GitHub>>GitHub repository owner: Khidash>>Continue>>>Repository name: ADF-Repo (this repo we have created in GitHub portal)>>Collaboration branch: Master>>Publish branch: adf\_publish>>Import resource into this branch: Master>>Apply>>wait for some time till the configuration established>>Save

* Now if we see on top in ADF then we can see the Master branch and we never make the changes like Adding new pipelines or adding new activities in pipelines in the Master branch. Bcoz this branch is a production ready branch, and many people will take reference of this branch.
* Master branch is a master copy, and we never do any testing or direct implementation on this Master branch instead we create a new branch will do our development in another branch and when we are confident that everything is implemented properly and correctly then will merge our newly created branch with Master/Main branch.
* If we click on the main or master branch knob then will get an option to create a New branch as shown in image below.

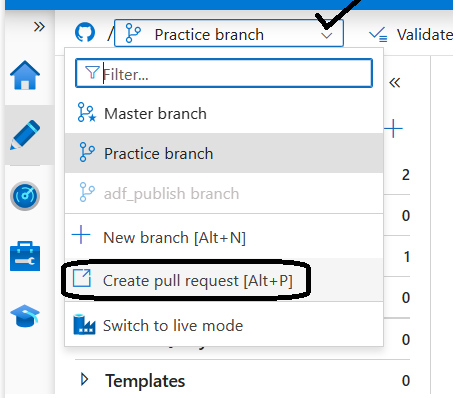
A screenshot of a computer

Description automatically generated

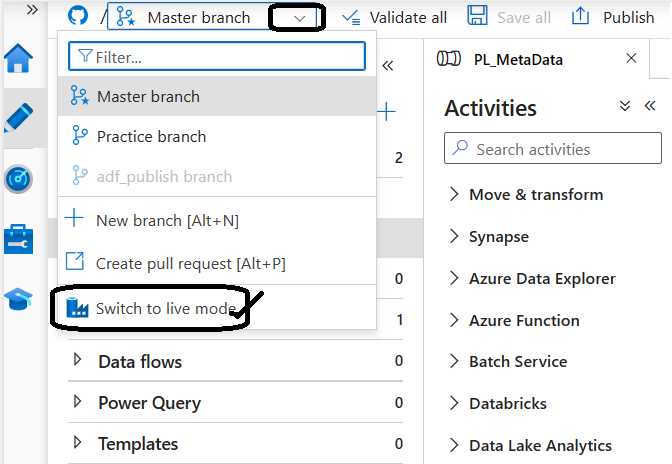
* Click on +New branch>>Branch name: Practice>>Create>>Now if we notice a practice branch has been created which is the replication of Master branch carries all the Pipelines, Datasets, Linked services…etc. as of master branch and now in this branch we can do our development as per the project requirements.
* Now create a new pipeline in practice branch>>Name: PL\_BranchPractice>>Drag and Drop(D&D) Wait activity/control in pipeline canvas>>in General tab>>Name: WaitFor10Seconds>>in Settings tab>>Wait time in seconds:10(any value we can pass)>>click on Save all(on top)>>ok
* We never do publish from other branches that we are creating, will do Publish always from Master branch(same like Azure Devops), will merge this branch with our master branch with which whatever the development or activities that we have done then it will get merge with the master branch then will do the publish.
* If we login to the GitHub portal and see we are having 2 branches((i)Master branch & (ii)Practice branch) in ADF-Repo that we have created above the same is shown below



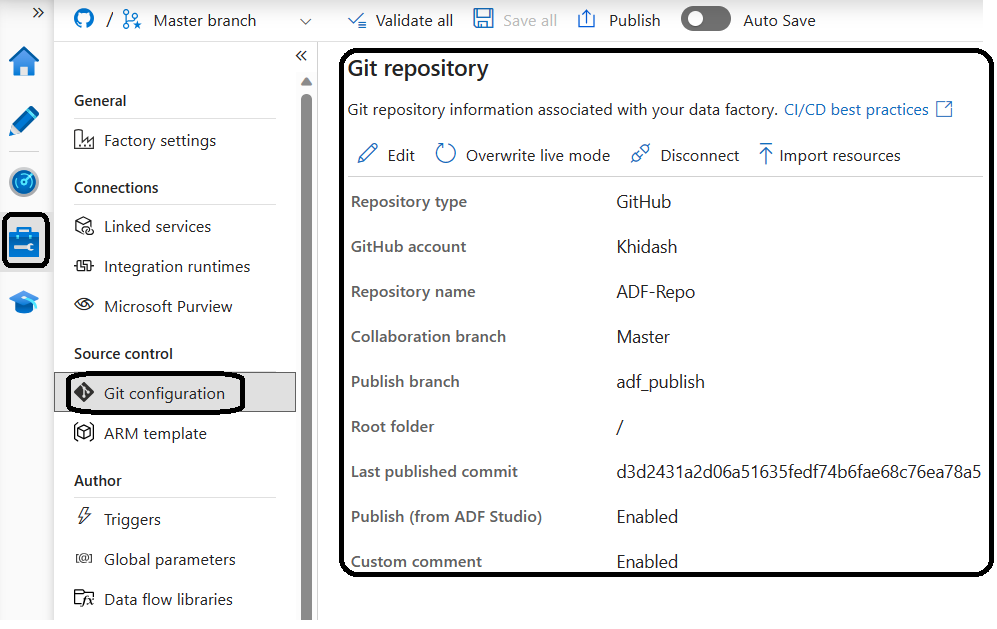
* Now if we want to make a pull request in order to merge the new activities to master branch then click on Branch knob and below will get an option to do the pull request (as shown in image below)



* Click on the Create pull request then it will redirect us to GitHub portal again click here on Create pull request (right side green color)>>Pass the comments in text box(ex: Merging the 2 branches)>>Create pull request>>Merge pull request>>Confirm merge>>Delete branch
* Come to Azure portal in ADF>>click on Master/Main branch and now here we can see the new pipeline(PL\_BranchPractice) in our Master branch>>click on Publish>>Ok>>wait for some time till Publishing is completed>>Click on Main/Master branch knob(shown below)>>click on Switch to live mode(as shown below) and here we can see the new pipeline(PL\_BranchPractice and other activities we have created in this pipeline) what we have created



* Inside ADF studios>>click on Manage>>Git configuration>>and here we can see the complete GitHub portal configuration details(as shown in below image).



# **Execution of on-prem SSIS Packages from ADF:**

To run the SSIS packages which are established in our on-prem via Azure Data Factory(ADF) then we use Execute SSIS Package Activity in ADF pipeline.

Execute SSIS package allows us to assess the cloud compatibility of our SSIS packages and run them on Azure-SSIS Integration Runtime (IR) in Azure Data Factory (ADF). With the use of this feature we can attach a newly created/existing Azure-SSIS IR to SSIS projects and then execute our packages on it.

**Implementation steps to execute the on-prem SSIS packages via ADF:**

**Preq-requisites:** To perform this automation of executing the SSIS packages via ADF we have to install

i)VS 2022 Enterprise Edition with Microsoft.DataTools.IntegrationServices(Add ON)

ii)SQL Server

iii)SSMS

iv)Active Azure Subscription(Free trial is also fine)

**Step1:**

Install Visual Studio 2022 Enterprise addition in your laptop

and then install Microsoft.DataTools.IntegrationServices(Add ON)

**Step 2:**

Create Azure SQL Server/Azure SQL Instance & Azure SQL DB in Azure portal as usual

**Step3:**

Open SSMS in our local laptop>>connect to Cloud SQL Server and run the below table creation script on cloud SQL DB

Create Table SSISLog

(

ID int identity(1,1),

LogDateTime datetime

)

GO

**Step 4:**

Create a folder(as NareshSSISProj) in your local laptop in any Drive(E) then>>Open Visual Studio 2022>>New Project>>create a new project>>search Sql server integration services(on top)>>click on Integration services project>>Next>>Set the path where we want to create the project(ex: E:\NareshSSISProj)>>Create>>will see a new SSIS Project will be created.

**Step 5:**

Right click on below connection manager>>New OLEDB Connection>>New>>ok>>On top for **Provider:** Microsoft OLE DB Driver for SQL Server>>ok>>For Server or **file name:** sqlserverinazure145.database.windows.net(this DB and DB server we have created in Azure portal pass accordingly)>>click on Radion button Use a specific username and password. Pass the username and password we gave at the time of SQL Server & SQL DB creation in Azure portal>>Initial Catalog:Select the DB here we created in Azure portal>>Test connection>>Ok>>Ok>>right click on connection manager that we created>>Rename>>Give name as Myconnection>>click Save & Save All on top

**Step 6:**

Drag and Drop execute SQL Task in SSIS Package from SSIS Toolbox(left side)>>double click on Execute SQL Task>>**For connection:** click the knob and choose Myconnection>>For SQL Statement: Pass the below SQL Query

insert into SSISLog values(getdate())

and say Ok>>Save>>Save All>>right click on the solution>>Build

**Step 7:**

Right click on the project>>Properties>>

Protection Level: Dontsavesensitive>>Ok>>Apply>>Ok

Again, right click beside execute SQL Task>>Properties>>scroll down>>

Protection Level: Dontsavesensitive>>Ok>>Save & Save All

**Step 8:**

Right click beside ExecuteSQLTask>>Variables>>click on the first ikon>>give Name: ConnectionString>>For Data type change the value to string>>click on cross to close the window.

**Step 9:**

Now right click on Myconnection>>Properties>>ConnectionString>>copy the complete connection string value and paste it in a notepad as shown below

Data Source=sqlserverinazure1914.database.windows.net;User ID=Gareth;Initial Catalog=NareshDB;Provider=MSOLEDBSQL.1;Application Name=SSIS-Package-{6A10F167-939B-42A4-9822-162E0D5835BE}sqlserverinazure1914.database.windows.net.NareshDB.Gareth;Auto Translate=False;

Now concatenate password to the above connection string as shown in below(as Example)

Data Source=sqlserverinazure1914.database.windows.net;User ID=Gareth;Initial Catalog=NareshDB;Provider=MSOLEDBSQL.1;Application Name=SSIS-Package-{6A10F167-939B-42A4-9822-162E0D5835BE}sqlserverinazure1914.database.windows.net.NareshDB.Gareth;Auto Translate=False;Password=Shaikpet@123;

& now copy the complete connection string(from above)>>right click beside ExecuteSQLTask>>Variable>> and for Value paste the complete connection string with password>>close the window

**Step 10:**

Right click on Myconnection>>Properties>>Click on Expressions 3 dots button>>For Property: ConnectionString>>For Expression click on 3 dots>>Expand variables and parameters folder and then drag and drop ConnectionString variable in Expression box>>Ok>>Ok>>Build>>Build Solution

**Step 11:**

Create an ADF>>Launch ADF studios>>Manage>>Integration runtimes>>+New>>Azure-SSIS>>continue>>

**Name:** SSIS-IR>>

**Description:** Creating this SSIS Integration Runtime(IR) for on-prem SSIS Packages.

**Node size:** D4\_v3(4 core(s), 16384 MB)>>

**Node number:** 2>>Continue>>uncheck the box Create SSIS catalog(SSISDB)>>Continue>>Continue>>Create>>Wait for some 4-5 mins untill the Status changed to running for SSIS Integration runtime that we have created.

**Step 12:**

Click on Author Ikon>>Create a new ADF pipeline>>in activities pane search for Execute SSIS Package Activity>>click on the control>>in genera tab give name as Execute SSIS package>>click on settings tab>>For Azure SSIS-IR: Select NareshSSIS-IR(this integration runtime we have created in above steps)>>For Package location choose Embedded package>>click on upload>>Select the package(Navigate to the path where we create the SSIS project)>>Open>>Publish all>>publish>>Debug

**Step 13:**

After Debug gets succeeded>>we can see the SSIS package execution inserted a record inside the SSISLog table in SSMS.

Hence using ADF Execute SSIS Package control/activity we are running the on-prem SSIS packages in Azure portal ADF and inserting the data in SQL Databases.

**Note:** If we want to Automate the Execution of SSIS Package then we can setup the Schedule based trigger where in the package run by itself as per the schedule and load the data into SQL DB's.

# **Data flows in Azure Data Factory(ADF):**

Mapping data flows are visually designed for data transformations in Azure Data Factory. Data flows allow data engineers to develop data transformation logic without writing code. The resulting data flows are executed as activities within Azure Data Factory pipelines that use scaled-out Apache Spark clusters. Data flow activities can be operationalized using existing Azure Data Factory scheduling, control, flow, and monitoring capabilities.

Mapping data flows provide an entirely visual experience with no coding required. Your data flows run on ADF-managed execution clusters for scaled-out data processing. Azure Data Factory handles all the code translation, path optimization, and execution of your data flow jobs, below is the link for for Azure Data flow

[Mapping data flows - Azure Data Factory | Microsoft Learn](https://learn.microsoft.com/en-us/azure/data-factory/concepts-data-flow-overview)

* Based upon the client requirements we can choose the DataFlows(DF)and in Dataflow we use the transformations(like Joins, conditional split, Exists, Union, Lookup…etc.). the limitation we have in Dataflow is we cannot connect to Dataflows from on-premises.
* We are having 2 types of Dataflows i.e.: (i)Mapping dataflows & (ii)Wramling dataflow

# **Implementation steps for Dataflow in ADF:**

**Step1:**

Create a Storage Account of type Blob SA, create a container inside the SA and place .csv file in it.

**Step2:**

Create ADF, Linked service(LS\_BlbSA), pick here delimited text

**Step3:**

In ADF Studios click on Author(left side)>>click on 3 dots of Data flows>>New data flow>>click on the arrow in pipeline canvas>>Add source

**Source Transformation:** A source transformation configures our data source for the data flow or from our source systems, to fetch the data from our source systems will use the source transformation in ADF data flows When we design data flows, our first step is always configuring a source transformation.

**Step4:**

Click on Source1>>click on Source settings tab>>For Dataset click on +New>>in search type Blob>>click on Azure Blob Storage Account>>continue>>Delimited text>>continue>>Name: ds\_blbsa>>Linked service: LS\_BlbSA>>for File path click on folder ikon, go inside the container and click on the .csv file>>ok>>check the box first row as header>>Ok

**Step5:**

In source setting tab>>Output stream name: DetailsData>>For options: check the box for Allow schema drift and Infer drifted column types>>Make Validate option Enabled for Data flow debug(as shown in below image)>>click ok>>Now we can notice here green check mark beside data flow debug option.

A screenshot of a computer

Description automatically generated

**Step6:**

First click on DetailsData>>Data preview tab>>Refresh (to see the source data)>>Click on Projection tab>>Detect data type>>and here we can see the source columns data type can change if needed(like for years passed column as Integer, Name column as string, Marks column as string)

Hence this way we can prepare our source transformation accordingly as per the input type that we have for Dataflows. Here we can relate our source file to source transformation and can analyze or see the data in Data preview tab.

# **Inline Datasets in Dataflow Source Control in Source Settings tab:**

Inline datasets are spark, like whatever the transformations we are doing inside the dataflows they are internally converted to spark scalar code and it will run on top of Data bricks spark cluster since it is a driver and cluster code, cluster code is a group of machines instead of running in a single node it will run on group of machines parallelly and it will provide the output for us.

Inline datasets are native to spark and when we create a source transformation is whether your source information is defined inside a dataset object or within the source transformation. Most formats are available in only one or the other.

When a format is supported for both inline and in a dataset object, there are benefits to both. Dataset objects are reusable entities that can be used in other data flows and activities such as Copy. These reusable entities are especially useful when we use a hardened schema. Datasets aren't based in Spark. Occasionally, you might need to override certain settings or schema projection in the source transformation.

Inline datasets are recommended when you use flexible schemas, one-off source instances, or parameterized sources. If your source is heavily parameterized, inline datasets allow you to not create a "dummy" object. Inline datasets are based in Spark, and their properties are native to data flow.

To use an inline dataset, select the format you want in the **Source type** selector. Instead of selecting a source dataset, you select the linked service you want to connect to.

# **Implementation steps of Dataflow with Source, Filter & Sink Transformation in ADF with inline Dataset:**

**Source Transformation:**

A source transformation configures our data source for the data flow or from our source systems, to fetch the data from our source systems will use the source transformation in ADF data flows When we design data flows, our first step is always configuring a source transformation.

**Filter Transformation:**

Filter transformation allows row filtering based upon a condition. The output stream includes all rows that match the filtering condition. The filter transformation is like a WHERE clause in SQL and when we fetch the data from the source systems and want to filter the data based upon some business requirements then will use the filter transformation before loading the data into our destination.

**Sink Transformation:**

After we finish transforming our data, write it into a destination store by using the sink transformation. Every data flow requires at least one sink transformation, but we can write to as many sinks as necessary to complete our transformation flow. After we have done with all our transformation and business logic as per the project requirements will finally use Sink transformations to load the data into the destination systems. Each sink transformation is associated with exactly one dataset object or linked service. The sink transformation determines the shape and location of our data to load it finally into our destination system.

A screenshot of a computer

Description automatically generated

**Step1:**

Create a Storage Account of type Blob SA(Source Storage Account), create a container inside the SA and place the below .csv file in it.



**Step2:**

Create a Storage Account of type ADL Gen2(Destination Storage Account), create a container inside the SA

**Step3:**

Create ADF,>>Create 2 Linked services

i.e: LS\_BlbSrc and

LS\_Gen2Dest here pick delimited text while creating the linked services and also Create 2 Dataset in ADF for BlobSA (ex: DS\_BlbSrc) & ADL Gen2 SA(ex: DS\_Gen2Dest)

**Step4:**

In ADF Studios click on Author>>click on 3 dots of Data flows>>New data flow>>Name: df\_dataflow>>click on the arrow>>Add source>>in source settings tab>>Output stream name: SourceData>>Source type: click on inline>>Inline dataset type: Delimited text(bcoz here in SA we have kept .csv file and also in Dataset we created using delimited text only)>>Linked service: LS\_BlbSrc.

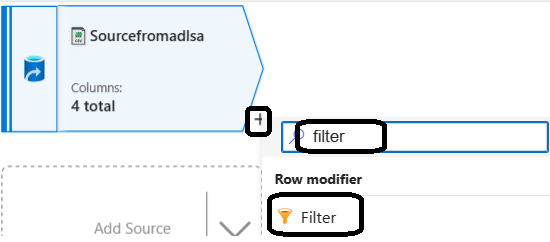
**Step5:**

Click on Source option tab>>For File path click on Browse and pick anyone of the .csv file(if we are having multiple files in the SA container) and if we want to load the data from multiple files then select the radio button wildcard and for filename text box pass \*.csv>>scroll down a little in source option tab also check the checkbox for First row as header

**Step6:** Enable Data flow debug>>Click on source control transformation>>Projection tab>>Import schema>>window open right side and click on Import

**Step7:**

Click on + symbol below the source control a window will pop below and in that search type filter and select filter transformation-as shown in below image (if we want to filter any rows then will use this filter transformation)>>



**Step8:**

Click on filter transformation>>In Filter settings tab>>Output stream name: FilterTheRows>>click on **Filter on** box>>click on open expression builder and type the below expression>>Save and finish

Year == 2020 && Product\_Type == 'electronics'

**Note:** This expression can be written purely on the .csv file columns that we are uploading, here Year, Product\_Type are columns we are having in .csv that we have uploaded/placed in Step1. Always ensure the column names we are mentioning in the expression should have the same column in .csv files.

**Step9:**

Click on filter transformation>>Data preview tab>>Refresh(to see the data in filter transformation as per the expression we have written above)

**Step10:**

Click on + symbol below the filter transformation control a window will pop below and in that search type sink>>click on Sink transformation (as shown in image below) A screenshot of a computer

Description automatically generated

**Step11:**

Click on Sink transformation>>Sink tab>>Output stream name: SinkData>>For Dataset: DS\_Gen2Dest (this dataset we have created above)>>Publish All>>Publish

**Note:** By default ADF will take AutoResolveIntegrationRuntime to copy the data from source to target and if we want we can even set the IR of our choice as shown in below step

**Step12:**

Launch ADF studios in another tab>>click on Manage(left side)>>Integration runtimes>>+New>>Azure, Self-Hosted>>Continue>>Azure>>Continue>>Name: integrationRuntime1>>create>>Publish All

**Step13:**

Click on Manage>>Integration runtimes>>click on integrationRuntime1>>Data flow runtime then here we can decide the compute size like small, medium, large or we can even customize accordingly as per project requirements>>Apply>>Publish All

**Step14:**

Create a new pipeline>>Name: Run\_Dataflow>>drag and drop Data flow activity/control from activities pane to pipeline canvas>>click on Dataflow activity>>Name: RunDataFlow>>In Settings tab>>Data flow: df\_dataflow>> Publish All>>Refresh the page if Publish All is succeeded>>Run on (Azure IR):integration Runtime1(this we have created above)>>come to pipeline>>click the data flow control>>Settings tab>>and here for Logging level choose Basic radio button>>Publish All>>Publish

**Step15:**

If we want the file name in our destination Gen2 SA as per our choice, then come to Dataflow>>click on sink transformation>>in Settings tab>>File name option: Name file as column data>>column data: choose any column>>Publish All>>Come to pipeline>>Debug and now we can see our files in destination Gen2 SA as per the file name we want.

Hence using Dataflow transformations, we are inserting/dumping the data into different targets by application filters as per the business requirements.

# **Implementation of Dataflows using Source, Select, Sort, Sink Transformations:**

**Source Transformation:**

A source transformation configures our data source for the data flow or from our source systems, to fetch the data from our source systems will use the source transformation in ADF data flows When we design data flows, our first step is always configuring a source transformation.

**Select Transformation:**

We use the select transformation to rename, drop, or reorder columns. This transformation doesn't alter row data, but chooses which columns are propagated to downstream…for example whatever the column that we want to drop or don’t want to consider in our destination then we can remove those columns from our source data before loading it to our destination.

In a select transformation, users can specify fixed mappings, use patterns to do rule-based mapping, or enable auto mapping. Fixed and rule-based mappings can both be used within the same select transformation. If a column doesn't match one of the defined mappings, it will be dropped.

**Sort Transformation:**

The sort transformation allows us to sort the incoming rows on the current data stream. We can choose individual columns and sort them in ascending or descending order. The Sort transformation sorts input data in ascending or descending order and copies the sorted data to the transformation output or to our destination.

**Sink Transformation:**

After we finish transforming our data, write it into a destination store by using the sink transformation. Every data flow requires at least one sink transformation, but we can write to as many sinks as necessary to complete our transformation flow. After we have done with all our transformation and business logic as per the project requirements will finally use Sink transformations to load the data into the destination systems. Each sink transformation is associated with exactly one dataset object or linked service. The sink transformation determines the shape and location of our data to load it finally into our destination system.

A diagram of a diagram

Description automatically generated

**Step1:**

Create a Blb SA Storage Account and a container inside it and upload the blow CSF file**—Source**



**Step2:**

Create ADL Gen2 Storage Account, container init**—Destination**

**Step3:**

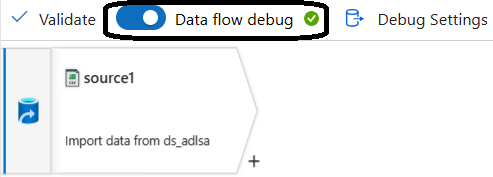
Create ADF, 2 Linked Services as LS\_BlbSA\_Src & LS\_Gen2SA\_Dest for both Blob Storage Account & ADL Gen2 Storage Account using delimited text

**Step4:**

Create Datasets for both Blb SA (ds\_blbsa\_src) & ADL Gen2 Storage Account(ex: ds\_adlsa\_dest) using delimited text>>Publish All>>Publish

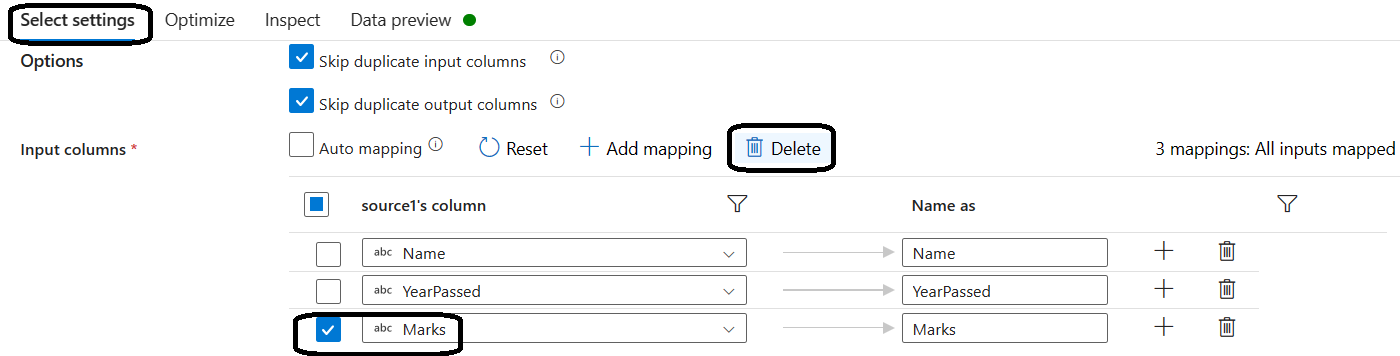
**Step5:**

Create a new data flow>>Name: df\_dataflow1>>Click on Add source box>>in Source settings tab>>For Dataset click the knob and choose ds\_blbsa\_src >>Enable data flow debug(on top) as shown below



**Step6:**

click on + >>a window will appear at the bottom>>in search type select>>click on Select transformation>>click on Select settings tab>>scroll down and here will see what all the columns that are going to appear in our destination if we don’t need some columns then we can select that column and delete as shown in below image, we can do this based upon the project requirements.



**Step7:**

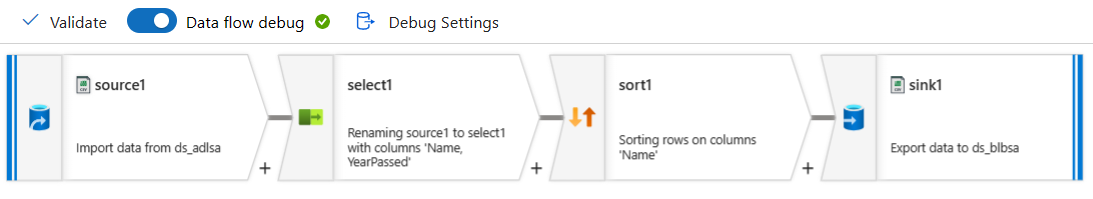
Click on + on Select transformations>>in below window search for Sort and click on Sort transformation>>in Sort settings tab scroll down and for Sort conditions click the knob and select the column name(ex: Country, means here we are inserting the data in destination based upon names in Ascending order)

**Step8:**

Click on + on sort transformation>> in below window search for sink and click on sink transformation>>click on Sink transformation>>In Sink tab>>for Dataset click the knob and choose ds\_adlsa\_dest>>Enable dataflow debug at the top

**Step9:**

Click on Sort transformation>>click on Data preview tab>>Refresh>>here we can see what all the columns we are going to insert in our destination and we can also notice here that last columns is not getting inserted in our destination.



**Step10:**

Click on Sink transformation>>Optimize tab>>click on single partition radio button>>click on settings tab>>File name option: Output to single file>>Output to single file: Details.csv(this file will get in our destination Gen2 SA)>>Publish All>>Publish

**Step11:**

Create a pipeline>>Name: Run\_DataflowAgain>>Drag and drop Data flow activity into pipeline canvas>>in General tab>>Name:df\_dataflow1>>in settings tab>>Data flow:df\_dataflow1(this data flow we have created in above steps)>>Publish All>>Publish>>Debug.

**Note:** Now if we notice here we can see in the destination storage account(i.e.: Blb SA) the details files will be present with only 2 or 3 columns where as in our Source Storage Account(adlSA) this same file is having multiple columns(may 3-4 columns).

# **Implementation of Dataflows using Aggregate & Sink transformation:**

**Source Transformation:**

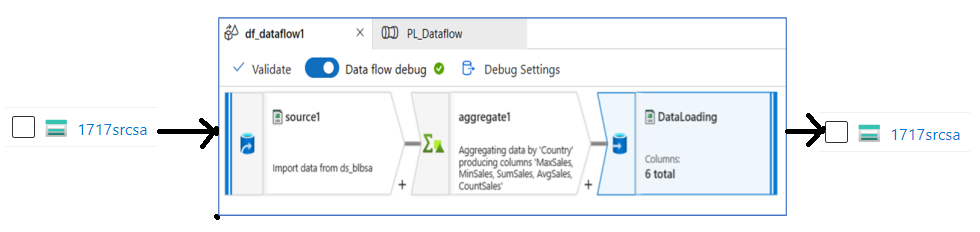
A source transformation configures our data source for the data flow or from our source systems, to fetch the data from our source systems will use the source transformation in ADF data flows When we design data flows, our first step is always configuring a source transformation.

**Aggregate Transformation:**

Aggregate transformation defines aggregations of columns on top of our source data or in our data streams. Using the Expression Builder, we can define different types of aggregations such as SUM, MIN, MAX, and COUNT and can also be grouped by existing or computed columns.

**Sink Transformation:**

After we finish transforming our data, write it into a destination store by using the sink transformation. Every data flow requires at least one sink transformation, but we can write to as many sinks as necessary to complete our transformation flow. After we have done with all our transformation and business logic as per the project requirements will finally use Sink transformations to load the data into the destination systems. Each sink transformation is associated with exactly one dataset object or linked service. The sink transformation determines the shape and location of our data to load it finally into our destination system.



**Note:** Here we are using one Storage account only as both source and destination.

**Step1:**

Create a Blb SA Storage Account and a container inside it and upload below .csv file inside the container.



**Step2:**

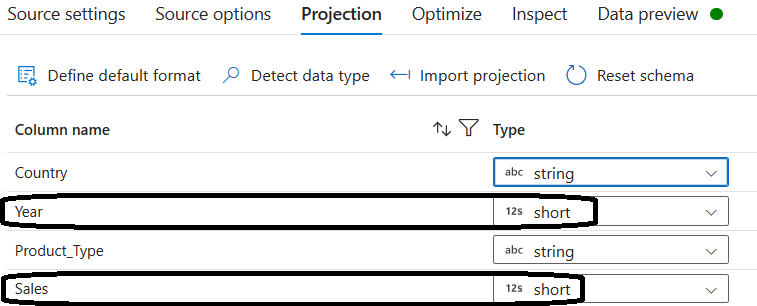
Create ADF, Linked Services(LS\_BlbSA\_Src) for Blb SA choose delimited text while creating a Dataset bcoz we have uploaded .csv file in the storage account.

**Step3:**

Create a new data flow>>Name: df\_dataflow2>>Click on Add source box>>in Source settings tab>>For Dataset: +New>>in search type blob storage>>continue>>delimited text>>continue>>Name:ds\_blbsa\_src>>For File path: click on folder ikon and select the file>>check First row as header>>Import schema: Form connection/store>>ok

**Step4:**

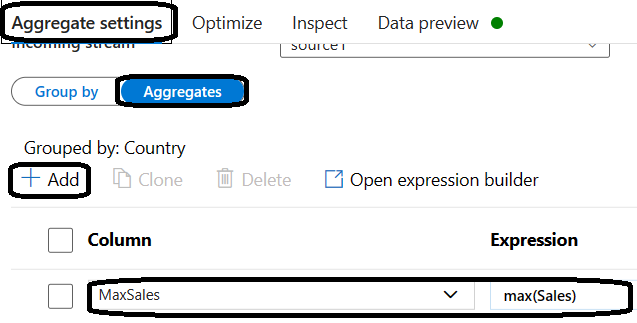
Enable Data Flow Debug>>click on Source transformation>>click on Projection tab and change the data type for Sales and Year column to Short as shown in below image.



**Step5:**

Enable data flow debug>>Ok>>click on Data preview tab>>Refresh(to check the data)>>click on + on source transformation>>in search type Aggregate>>click on Aggregate transformation>>in Aggregate settings tab scroll down and for columns select Country>>click on Aggregates(as shown below)>>For Column type/say **MaxSales**>>click on ANY for Expression then an expression builder will be opened and there type the below expression>>Save and finish

max(Sales)



**Step6:**

Click on +Add>> For Column type/say **MinSales**>>click on ANY for Expression then an expression builder will be opened and there type the below expression>>Save and finish

min(Sales)

**Step7:**

Click on +Add>> For Column type/say **SumSales**>>click on ANY for Expression then an expression builder will be opened and there type the below expression>>Save and finish

sum(Sales)

**Step8:**

Click on +Add>> For Column type/say **AvgSales**>>click on ANY for Expression then an expression builder will be opened and there type the below expression>>Save and finish

avg(Sales)

**Step9:**

Click on +Add>> For Column type/say **CountSales**>>click on ANY for Expression then an expression builder will be opened and there type the below expression>>Save and finish

count(Sales)

**Step10:**

Click on Aggregate transformation>>Data preview tab>>Refresh (to see the data in aggregated values as shown below)

A screenshot of a computer

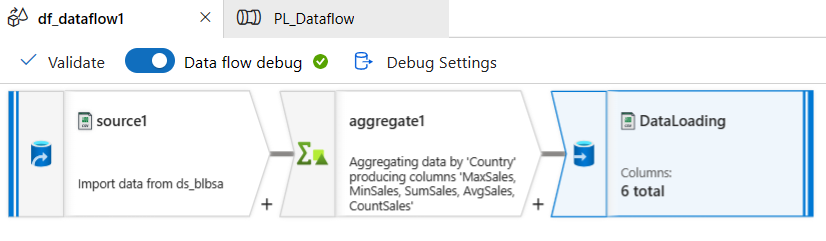
Description automatically generated

**Step11:**

Click on + on Aggregate transformation>>In search type sink>>click on sink transformation>>in Sink tab>>Output stream name: DataLoading>>Dataset: ds\_blbsa\_src >>in Settings tab>>File name option: Name file as column data>>Column data: Country>>Publish All>>Publish

**Step12:**

Create a new pipeline>> Name: PL\_Dataflow>>Drag and drop the Data flow activity in pipeline canvas>>in General tab>>Name: df\_dataflow1>>in Settings tab>>Data flow:df\_dataflow1>>Publish All>>Publish>>Debug



**Note1:** Hence with all above transformations we can see that for single file data we have divided into multiple aggregations using aggregate transformations and divided the data into multiple files and loaded in our Blb SA(this we have considered as both source and destination in this demo).

**Note2:**

If we want all the details in one single files instead of multiple files then click on Sink transformation>>Click on Optimize tab>>click the radio button as single partition>>click on Settings tab>>For File name option: Output to single file>>For Output to single file : Aggregates.csv(this file name we are giving we can give any name as per the project requirements)

# **Implementation of Dataflow with conditional split & Sink transformation:**

**Source Transformation:**

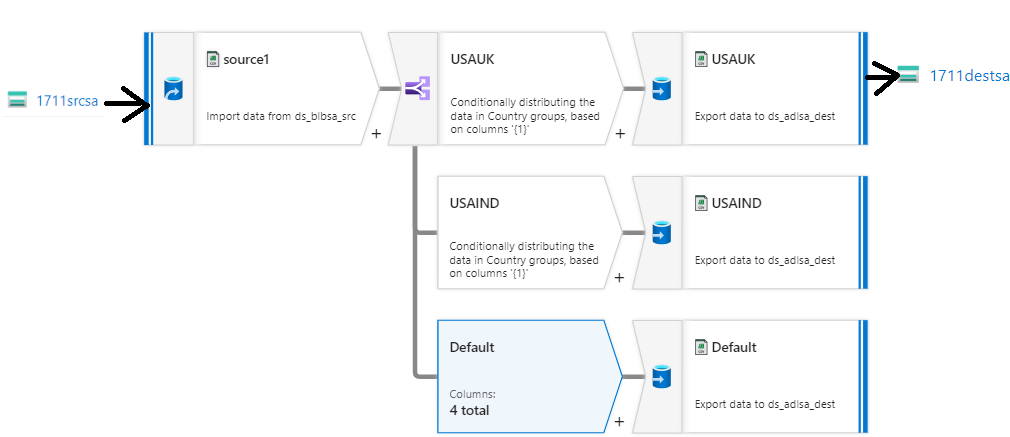
A source transformation configures our data source for the data flow or from our source systems, to fetch the data from our source systems will use the source transformation in ADF data flows When we design data flows, our first step is always configuring a source transformation.

**Conditional Split Transformation:**

The conditional split transformation routes data rows to different streams based on matching conditions. The conditional split transformation is like a CASE decision structure in a programming language. The transformation evaluates expressions, and based on the results, directs the data row to the specified stream. With conditional split transformation we can split the data based upon the business conditions and load it into our destination systems.

**Sink Transformation:**

After we finish transforming our data, write it into a destination store by using the sink transformation. Every data flow requires at least one sink transformation, but we can write to as many sinks as necessary to complete our transformation flow. After we have done with all our transformation and business logic as per the project requirements will finally use Sink transformations to load the data into the destination systems. Each sink transformation is associated with exactly one dataset object or linked service. The sink transformation determines the shape and location of our data to load it finally into our destination system.



**Step1:**

Create a Blb SA, container inside the SA and upload below .csv file inside the Blb SA



**Step2:**

Create ADL Gen2 Storage account, create container and a folder inside the container (Ex: conditional split) inside the SA,

**Step3:**

Create an ADF>>Launch ADF>>

(i)Create a Linked Service (LS\_Blbsa\_Src) for Blb SA & Create a dataset(ds\_blbsa\_src) for Blb SA

(iI)Create a Linked Service (LS\_Adlsa\_Dest) for ADL Gen2 SA & Create a dataset(ds\_adlsa\_dest) for ADL Gen2 SA

**Step4:**

In ADF studio create new Dataflow>>Name: df\_dataflow12>>click on source transformation>>For Dataset: ds\_blbsa\_src>>Enable Data flow debug option(on top).

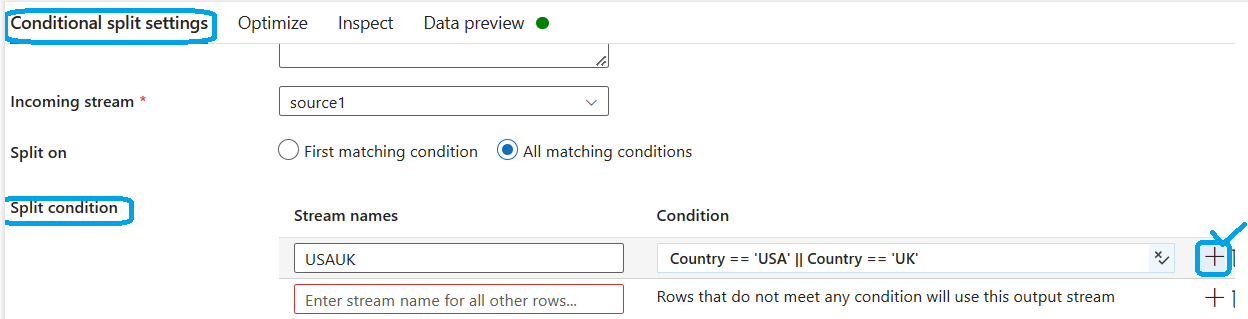
**Step5:**

Click on + Source transformation>>in search type conditional split>>click on conditional split>>in conditional split settings tab>>Split on: All matching conditions>>For Stream names(text box): USAUK>>For condition box click on **ANY** and in expression builder type the below expression>>Save and finish

Country == 'USA' || Country == 'UK'

**Step6:**

Click on + on extreme right(as shown below) to add a new row condition>>For Stream names(text box):USAIND>> For condition box click on **ANY** and in expression builder type the below expression>>Save and finish



Country == 'USA' || Country == 'IND'

**Step7:**

For last text box of Steam names: Default and write the expression as

Country == 'Default'

**Step8:**

Click on +(1st) of conditional split transformation>>in search type sink>>click on Sink transformation>>in sink tab>>Output stream name: USAUK>>For Dataset: ds\_adlsa\_dest(created in above steps)>>click on Optimize tab: single partition>>click on Setting tab>>For Filename option: Output to single file>>For Output to single file: USAUK.csv

Click on +(2nd) of conditional split transformation>>in search type sink>>click on Sink transformation>>in sink tab>>Output stream name: USAIND>>For Dataset: ds\_adlsa\_dest (created in above steps)>>in Optimize tab: single partition>>click on Setting tab>>For Filename option: Output to single file>>For Output to single file: USAIND.csv.

Click on +(last) of conditional split transformation>>in search type sink>>click on Sink transformation>>in sink tab>> Output stream name: Default>>For Dataset: ds\_adlsa\_dest (created in above steps)>>in Optimize tab: single partition>>click on Setting tab>>For Filename option: Output to single file>>For Output to single file: Default.csv

**Step9:**

Enable Dataflow Debug>>Publish All>>Publish

**Step10:**

Create a new pipeline>>Name: PL\_df\_dataflow12>>Drag and drop the dataflow activity in pipeline canvas>>in General tab>>Name: DataFlowExe>>in Settings tab>>Data flow:df\_dataflow12>>Publish All>>Publish>>Debug

Hence here we can see in destination ADL Gen2 Storage Account 3 .csv files with respective data in it.

**Note:** If we are having a huge volume of data in files like 10 Lakhs plus records and we want to load the data in destination in short time then in click on Dataflow activity in pipeline>>In Settings tab>>expand sink properties>>check Run in parallel check box.

# **Implementation of Dataflow with Exists & Sink transformation:**

**Source Transformation:**

A source transformation configures our data source for the data flow or from our source systems, to fetch the data from our source systems will use the source transformation in ADF data flows When we design data flows, our first step is always configuring a source transformation.

**Exists Transformation:**

The exists transformation is a row filtering transformation that checks whether our data exists in another source or stream. The output stream includes all rows in the left stream that either exist or don't exist in the right stream. The exists transformation is like SQL WHERE EXISTS and SQL WHERE NOT EXISTS. The exists transformation makes a comparison between 2 data sources based upon the condition and if the same records are exists within then it will load the data into the destination systems.

**Sink Transformation:**

After we finish transforming our data, write it into a destination store by using the sink transformation. Every data flow requires at least one sink transformation, but we can write to as many sinks as necessary to complete our transformation flow. After we have done with all our transformation and business logic as per the project requirements will finally use Sink transformations to load the data into the destination systems. Each sink transformation is associated with exactly one dataset object or linked service. The sink transformation determines the shape and location of our data to load it finally into our destination system.

A screenshot of a computer

Description automatically generated

**Step1:**

Create a Blb SA, container inside the SA and upload below 2 .csv file inside the Blb SA>>**Source**

**Step2:**

Create ADL Gen2 Storage account, container inside the SA>>**Destination**

**Step3:**

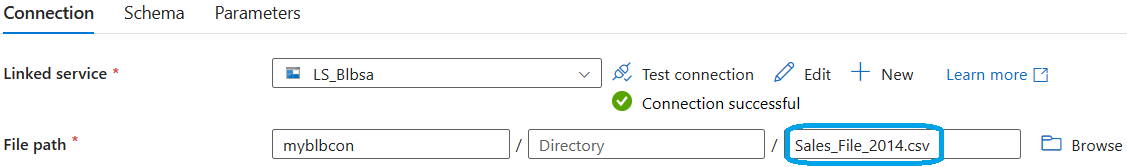
Create an ADF>>Launch ADF>>

(i)Create a Linked Service (LS\_BlbSA) for Blb SA & Create a dataset((i)ds\_blbsa\_2014 & (ii)ds\_blbsa\_2020) for Blb SA

(ii)Create a Linked Service (LS\_AdlSA) for Adl SA & Create a dataset(ds\_adlsa) for Adl Gen2 SA

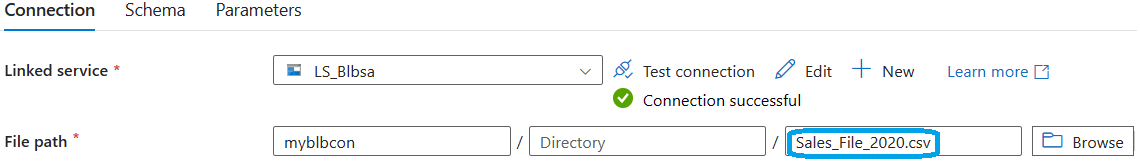
**Step3:**

Create a new dataflow>>Enbale data flow debug>>Name: df\_dataflow5>>Click on Add Source for Source transformation>>in Source settings tab>>Output stream name: source1>>Dataset: ds\_blbsa\_2014(this we have created at the top and for this source1 we have set Sales\_File\_2014.csv)>>For Dataset: click on open and here we see the Sales\_File\_2014.csv(as shown in below image) and if we are not seeing this file click on Browse and select Sales\_File\_2014.csv file



**Step4:**

Click on Add Source(below) for Source transformation>>in Source settings tab>>Output stream name: source2>>Dataset: ds\_blbsa\_2020(this we have created at the top and for this source2 we have set Sales\_File\_2020.csv and if we are not seeing this file click on Browse and select Sales\_File\_2020.csv file)



**Step5:**

Click on + for Source1 transformation>>in search type Exists>>in Exists settings tab>>Output stream name: Exists>>Left stream: source1>>Right stream: source2>>for Left: source1’s column: Year>>for Right: source2’s column: Year

**Step6:**

Click on **+** on Exists transformation>>in search type Sink>>select Sink>>in Sink tab>>Dataset: ds\_adlsa(this dataset for target we have created above)>>in optimize tab choose single partition>>in settings tab>>File name option:Output to single file>>Output to single file: OnlyYears.csv>>Publish All>>Publish

**Step7:**

Create a new pipeline>>PL\_df\_dataflow5>>Drag and drop dataflow activity in pipeline canvas>>Name: df\_dataflow5>>in settings tab>>Data flow:df\_dataflow5>>Publish All>>Publish>>Debug.

**Note:** Here we won’t find any data in our destination .csv file bcoz there is not even a single matching record in both the source files if we look carefully.

# **Implementation steps of Azure Dataflows for Exist & Sink transformation:**

**Source Transformation:**

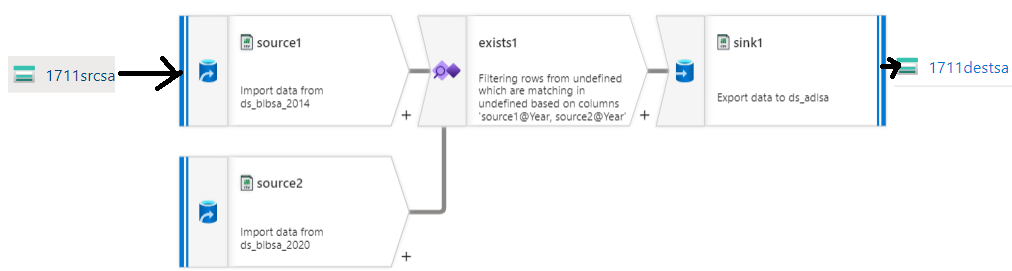
A source transformation configures our data source for the data flow or from our source systems, to fetch the data from our source systems will use the source transformation in ADF data flows When we design data flows, our first step is always configuring a source transformation.

**Exists Transformation:**

The exists transformation is a row filtering transformation that checks whether our data exists in another source or stream. The output stream includes all rows in the left stream that either exist or don't exist in the right stream. The exists transformation is like SQL WHERE EXISTS and SQL WHERE NOT EXISTS. The exists transformation makes a comparison between 2 data sources based upon the condition and if the same records are exists within then it will load the data into the destination systems.

**Sink Transformation:**

After we finish transforming our data, write it into a destination store by using the sink transformation. Every data flow requires at least one sink transformation, but we can write to as many sinks as necessary to complete our transformation flow. After we have done with all our transformation and business logic as per the project requirements will finally use Sink transformations to load the data into the destination systems. Each sink transformation is associated with exactly one dataset object or linked service. The sink transformation determines the shape and location of our data to load it finally into our destination system.



**Step1:**

Create a Blb SA, container inside the SA and upload below 2 .csv file inside the Blb SA

**Step2:**

Create an ADF>>Launch ADF>>

(i)Create a Linked Service (LS\_BlbSA) for Blb SA & Create a dataset(ds\_blbsa) for Blb SA

(ii)Create a Linked Service (LS\_AdlSA) for Adl SA & Create a dataset(ds\_adlsa) for Adl Gen2 SA

**Step3:**

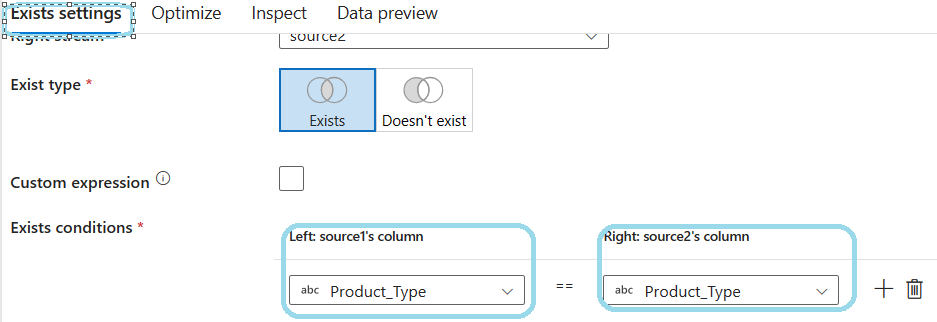
In ADF create a Dataflow>>Name: df\_dataflow7>>click on Add source>>click on Source transformation(source1)>>in Source settings tab>>Dataset: ds\_blbsa>>For Dataset click on open and click on Browse to keep **Sales\_Files\_2014.csv.**

**Step4:**

Click on Add source again>>click on source transformation(source2)>>in Source settings tab>>Dataset: ds\_blbsa>>For Dataset click on open and click on Browse to keep **Sales\_Files\_2020.csv**

**Step5:**

Click on + of source1>>in search type Exists>>click on Exists transformation>>in Exists settings tab>>Right stream: source2>>For Left: source1’s column: Product Type>>Right: source2’s column: Product Type(as shown in below image)>>Enable dataflow debug>>ok



**Step6:**

Click on + on Exists transformation>>in search type sink>>click on sink transformation>>in Sink tab>>For Dataset: ds\_adlsa>>in Settings tab>>File name option: Name file as column data>>click Refresh(@ top right of the page as shown below)

**Step7:**

Create a new pipeline>>Name: PL\_df\_dataflow7>>Drag and drop the Dataflow entity>>Name: Dataflow7>>in settings tab>>Data flow:df\_datflow7>>Publish All>>Publish>>Debug.

# **Implementation steps of Azure Dataflows for Source, Join & Sink transformation:**

**Source Transformation:**

A source transformation configures our data source for the data flow or from our source systems, to fetch the data from our source systems will use the source transformation in ADF data flows When we design data flows, our first step is always configuring a source transformation.

**Join Transformation:**

We use join transformation to combine data from two sources or streams in a mapping data flow. The output stream will include all columns from both sources matched based on a join condition. Here in this transformation will join multiple data sources based upon the condition by considering one of the common columns and load the data into the target/destination.

Mapping data flows currently supports five different join types.

### Inner Join

Inner join only outputs rows that have matching values in both tables/data sources.

### Left Outer:

Left outer join returns all rows from the left stream and matched records from the right stream. If a row from the left stream has no match, the output columns from the right stream are set to NULL. The output will be the rows returned by an inner join plus the unmatched rows from the left stream.

### Right Outer:

Right outer join returns all rows from the right stream and matched records from the left stream. If a row from the right stream has no match, the output columns from the left stream are set to NULL. The output will be the rows returned by an inner join plus the unmatched rows from the right stream.

### Full Outer:

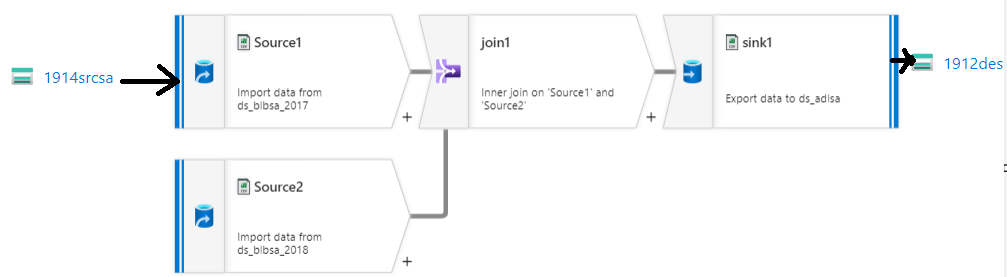
Full outer join outputs all columns and rows from both sides with NULL values for columns that aren't matched.

### Custom/Cross join:

Cross join outputs the cross product of the two streams based upon a condition. If you're using a condition that isn't equality, specify a custom expression as your cross-join condition. The output stream will be all rows that meet the join condition.

**Sink Transformation:**

After we finish transforming our data, write it into a destination store by using the sink transformation. Every data flow requires at least one sink transformation, but we can write to as many sinks as necessary to complete our transformation flow. After we have done with all our transformation and business logic as per the project requirements will finally use Sink transformations to load the data into the destination systems. Each sink transformation is associated with exactly one dataset object or linked service. The sink transformation determines the shape and location of our data to load it finally into our destination system.



**Step1:**

Create a Blb SA**(Source)**, container inside the SA and upload below 2 .csv file inside the Blb SA

**Step2:** Create an ADL Gen2 Storage Account**(Destination)**

**Step3:**

Create an ADF>>Launch ADF>>

(i)Create a Linked Service (LS\_BlbSA) for Blb SA & Create a dataset(ds\_blbsa\_Emp & ds\_blbsa\_Dept) for Blb SA

(ii)Create a Linked Service (LS\_AdlSA) for Adl SA & Create a dataset(ds\_adlsa) for Adl Gen2 SA

**Step4:**

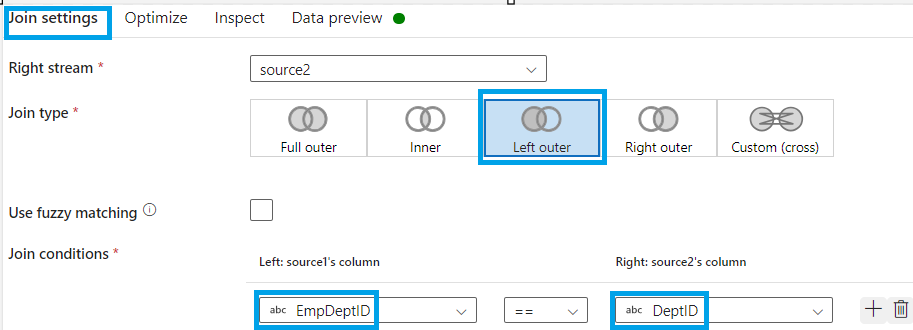
In ADF create a Dataflow>>Name: df\_dataflow11>>Enable Data Flow Debug>>click on Add source>>click on Source transformation(source1)>>in Source settings tab>>Dataset: ds\_blbsa\_Emp>>click on Data preview tab>>Refresh.

**Step5:**

Click on Add source again>>click on source transformation(source2)>>in Source settings tab>>Dataset: ds\_blbsa\_Dept>>click on Data preview tab>>Refresh.

**Step6:**

Click on + of source1>>in search type Join>click on Join transformation>>in Join settings tab>>Right stream: source2>>For Join type: Left (choose which type of join we want to consider like Inner join, Left outer join, Full outer join….etc)>>For Join conditions and For Left: source1’s column: EmpDeptID and For Right: source2’s column: DeptID(as shown in below image for both Left: source1’s column and Right: source2’s column)>>



**Step7:**

Click on + of Join transformation>>in search type sink>>click on sink transformation>>in Sink tab>>Dataset: ds\_adlsa>>in Settings tab>>File name option: Output to single file>>Output to single file: Leftjoinresults.csv>>in Optimize tab>>single partition>>Publish All>>Publish.

**Step8:**

Create a new pipeline>>Name: PL\_df\_dataflow11>>Drag and drop the Data flow activity>>In Settings tab>>Data flow: dataflow1111>>compute size: medium>>Publish All>>Publish.

**Step9:**

Here in destination storage account we can see .csv file which contains the Join records whatever join we gave(ex: Innerjoin, Left join….etc.)

# **Derived Column:**

When creating a derived column, we can either generate a new column or update an existing one. The Derived Column transformation creates new column values by applying expressions to transformation input columns.We use the derived column transformation to **generate new columns** in our data flow or to modify existing fields In the Column textbox, enter in the column we are creating. To build the derived column's expression, we click on the **Enter expression** textbox. We can either start typing our expression or open the expression builder to construct our logic.

# **Implementation steps of Azure Dataflows for Derived column transformation with Source & Sink transformation:**

A close up of a document

Description automatically generated

**Step1:**

Create a Blb SA, container inside the SA and upload below .csv file inside the Blb SA



**Step2:**

Create an ADL Gen2 Storage Account, and container inside the Storage Account.

**Step3:**

Create an ADF>>Launch ADF>>

(i)Create a Linked Service (LS\_BlbSA) for Blb SA &

Create a dataset(ds\_blbsa) for Blb SA

(ii)Create a Linked Service (LS\_AdlSA) for Adl SA & Create a dataset(ds\_adlsa) for Adl Gen2 SA

**Step4:**

Create a new dataflow>>Name:df\_dataflow55>>Enable : Data flow debug(for sure every time)click on Add source>>in Source settings tab>>Dataset: ds\_blbsa>>Datapreview tab>>Refresh

**Step5:**

Click on + source transformation>>in search type Derived Column>>Click on Derived column>>in Derived column’s settings>>For Column mentioned Year (as shown below) and click on ANY(as shown below) for expression builder and write the below expression accordingly

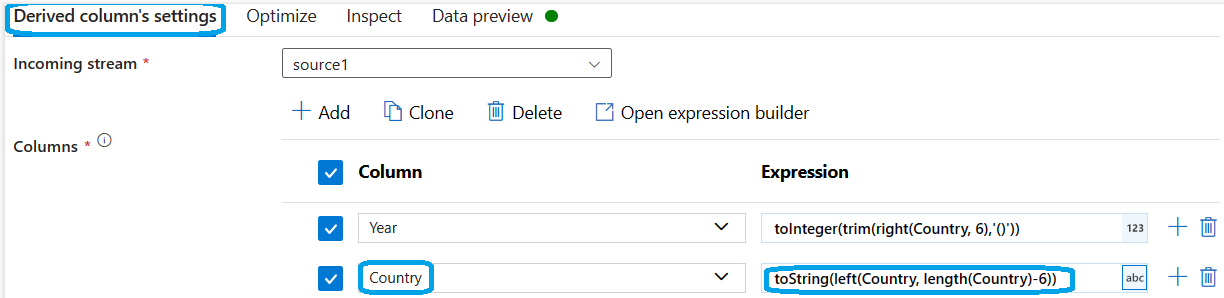
A screenshot of a computer

Description automatically generated

toInteger(trim(right(Country, 6),'()'))

**Step6:**

Click on +(shown below)>>click on Add column(as shown below)>>in the 2nd column which generated just now mentioned as Country>>click on ANY on 2nd column>>write the below expression in expression builder.



toString(left(Country, length(Country)-6))

**Step7:**

Click on Derived column transformation>>Enable Data flow debug>>click on Data preview tab>>Refresh>>Now here we can see the country column carrying only countries in it and a new Derived column Year has been emerged which is carrying Years only(as shown in image below)

A screenshot of a data analysis

Description automatically generated

**Step8:**

Click on + on Derived column>>in search type sink>>click on sink transformation>>in Sink tab>>Dataset: ds\_adlsa>>in Settings tab>>File name option:Output to single file>>Output to single file:Derivedcolumnresults.csv>>in Optimize tab>>click on Single partition>>In Data preview tab>>Refresh(to see how the data is getting loaded in our destination ADL Gen2 Storage account)>>Publish All>>Publish

**Step9:**

Create a new pipeline>>Name:PL\_df\_dataflow55>>Drag and drop the data flow activity in pipeline canvas>>in settings tab>>Dataflow:dataflow55>>Compute size: Medium>>Publish All>>Publish>>Debug

**Step10:**

Now come to destination Storage Account i.e.: ADL Gen2 SA and we can see a Derivedcolumnresults.csv file in destination SA

# **Implementation steps of Azure Dataflows to connect Azure SQL DB with Source & Sink transformation:**

**Source Transformation:**

A source transformation configures our data source for the data flow or from our source systems, to fetch the data from our source systems will use the source transformation in ADF data flows When we design data flows, our first step is always configuring a source transformation.

# **Derived Column:**

When creating a derived column, we can either generate a new column or update an existing one. The Derived Column transformation creates new column values by applying expressions to transformation input columns. We use the derived column transformation to generate new columns in our data flow or to modify existing fields In the Column textbox, enter in the column we are creating. To build the derived column's expression, we click on the **Enter expression** textbox. We can either start typing our expression or open the expression builder to construct our logic.

**Pivot Transformation:**

We use pivot transformation to create multiple columns from the unique row values of a single column. Pivot is an aggregation transformation where we select group by columns and generate pivot columns using an aggregate function.

**Sink Transformation:**

After we finish transforming our data, write it into a destination store by using the sink transformation. Every data flow requires at least one sink transformation, but we can write to as many sinks as necessary to complete our transformation flow. After we have done with all our transformation and business logic as per the project requirements will finally use Sink transformations to load the data into the destination systems. Each sink transformation is associated with exactly one dataset object or linked service. The sink transformation determines the shape and location of our data to load it finally into our destination system.

A diagram of a diagram

Description automatically generated

**Step1:**

Create SQL DB(Ex:AdventureWorks) in Azure portal following the same steps as regular and in Additional settings tab for Use existing data click on Sample as shown below and rest of the steps and procedures are same**>>This will be as Source**

A screenshot of a computer screen

Description automatically generated

**Step2:**

Create Gen2 Storage Account, container inside the SA**>>This will be as Destination**

**Step3:**

Create an ADF>>Create a Linked service(LS\_Sql) for SQL DB create this LS for Adventure Works DB>>Create a Dataset(ds\_Sql) for [SalesLT].[Product] table present in AdventureWorks SQL DB

Also create a Linked service(LS\_Adlsa) and a Dataset(ds\_adlsa)

>>Publish All>>Publish

**Step4:**

Create a new dataflow>>Name:df\_dataflow1>>Click on Add source>>in Source settings tab>>Dataset:ds\_sql>>Enable data flow debug option>>In Data Preview tab>>Refresh.

**Step5:**

Click on + on source transformation>>in search type Derived column>>click on Derived column>>in Derived column’s settings tab>>For columns: Color>>click on ANY (shown below) to open the expression builder and type the below expression>>Save and finish

A screenshot of a computer

Description automatically generated

iif(isNull(Color) || Color == 'null', 'NA', Color)

**Step6:**

Click on +Add >>Add column(as shown below)>>for newly added column pass the name as size(as shown below)>>double click on ANY>>and write the below expression>>Save and finish

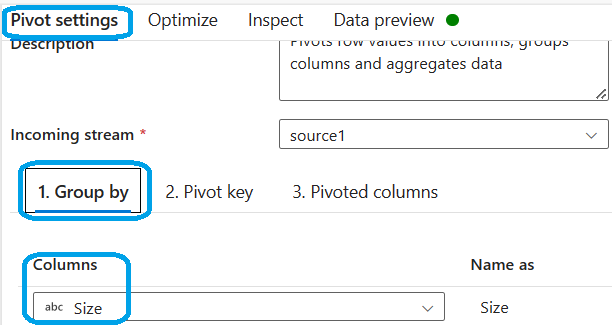
A screenshot of a computer

Description automatically generated

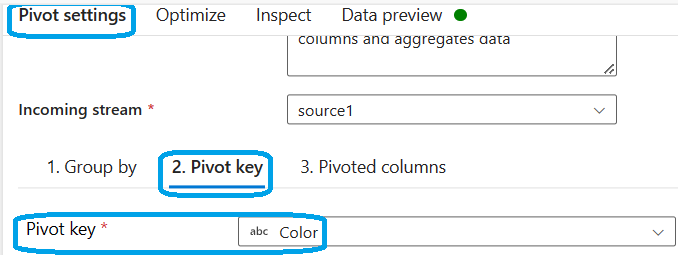
iif(isNull(Size) || Size == 'NULL', 'NA', Size)

**Step7:**

Click on + on Derived column>>in search type Pivot>>click on Pivot>>in Pivot settings tab>>scroll down>>click on Group by >>For columns: Size(as shown below)



**Step8:** Now click on Pivot key>>scroll down>>For Pivot Key: Color(as shown below)



**Step9:**

Now click on Pivoted columns tab(as shown below)>>double click on ANY to open the expression builder and write the below expression for avg of standard cost>>save & finish>>give name as Avg for next text box(as shown in image below).

toDecimal(avg(StandardCost),10,2)

A screenshot of a computer

Description automatically generated

**Step10:**

Click on Pivot transformation>>Data preview>>Refresh>> To see the data reflection in which the columns turned to rows as we have used Pivot transformation.

**Step11:**.

Click on + on pivot transformation>>in search type for sink transformation>>click on Sink transformation>>in Sink tab>>Dataset: ds\_adlsa>>in Settings tab>>File name option: Output to single file>>Output to single file:pivotresults.csv>>in Optimize tab>>click on Single partition>>In Data preview tab>>click on Refresh and finally we can see here how and what data is going to inserted in our destination i.e: ADL Gen2 SA from SQL DB table(i.e: [SalesLT].[Product])>>Publish All>>Publish…finally our all transformations looks like below(shown in image)



**Step12:**

Create a new pipeline>>Name: PL\_df\_dataflow77>>Drag and drop the data flow activity into pipeline canvas>>in Settings tab>>Data flow: df\_dataflow77>>Compute size: Medium>>Publish All>>Publish>>Debug

Hence here we can see the data got exported from Adventure Works DB(source) to ADL Gen Storage Account( destination SA)

# Union & Union All:

**UNION** and **UNION ALL** in SQL are used to retrieve data from two or more tables. UNION returns distinct records from both the table, while UNION ALL returns all the records from both the tables.

# Windows Functions:

A *window function* performs a calculation across a set of table rows that are somehow related to the current row. This is comparable to the type of calculation that can be done with an aggregate function. But unlike regular aggregate functions, use of a window function does not cause rows to become grouped into a single output row — the rows retain their separate identities. Behind the scenes, the window function can access more than just the current row of the query result.

* RANK()  
  As the name suggests, the rank function assigns rank to all the rows within every partition. Rank is assigned such that rank 1 given to the first row and rows having same value are assigned same rank. For the next rank after two same rank values, one rank value will be skipped.
* DENSE\_RANK()  
  It assigns rank to each row within partition. Just like rank function first row is assigned rank 1 and rows having same value have same rank. The difference between RANK() and DENSE\_RANK() is that in DENSE\_RANK(), for the next rank after two same rank, consecutive integer is used, no rank is skipped.

# Row\_Number()

It assigns consecutive integers to all the rows within partition, within a partition, no two rows can have same row number.

**Implementation of Dataflows with Window & Sink transformations:**

**Step1:**

Create SQL DB(Adventure Works) in Azure portal following the same steps as regular and in Additional settings tab for Use existing data click on Sample as shown below and rest of the steps and procedures are same.

A screenshot of a computer screen

Description automatically generated

**Step2:**

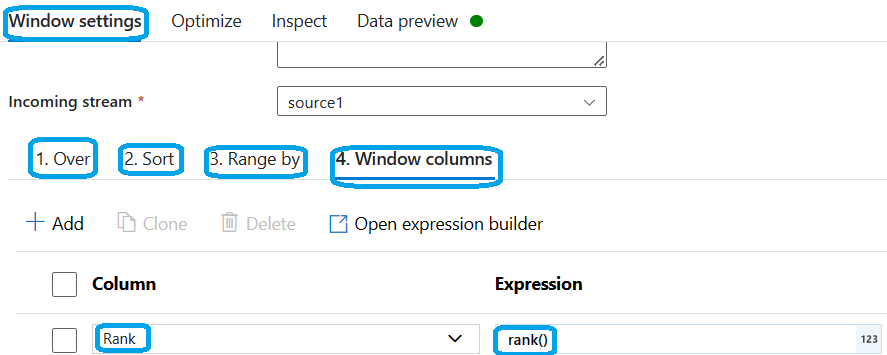
Create an ADF>>Create a Linked service(LS\_Sql) for SQL DB, create this LS for Adventure Works DB>>Create a Dataset(ds\_Sql) for [SalesLT].[Product] table present in AdventureWorks SQL DB>>Publish All>>Publish

**Step4:**

Create a new dataflow>>Name:df\_dataflow99>>Click on Add source>>in Source settings tab>>Dataset:ds\_sql>>Enable data flow debug option>>goto projection tab>>import projection>>In Data Preview tab>>Refresh.

**Step5:**

Click on + on source transformation>>in search type Window>>click on Window transformation>>in Windows settings tab>>click on 1.Over(as shown below)>>for source1’s column: choose Size>>click on 2.Sort(as shown below)>> for source1’s column: Standard Cost>>click on 4.Window columns>>for Column: Rank(on left side) and in expression box type rank()(on right side as shown below).



**Step6:**

Click on Window transformation>>Data preview>>Refresh>>if we navigate to extreme right then we can see Rank(as shown below) which shows the same rank 3 for 3 rows bcoz the standard cost is having the same value and next rank it took 6 not 4. Here if the particular row is having the same value then it gives the same rank to that rows(ex: Standard cost)A screenshot of a computer

Description automatically generated

**Step7:**

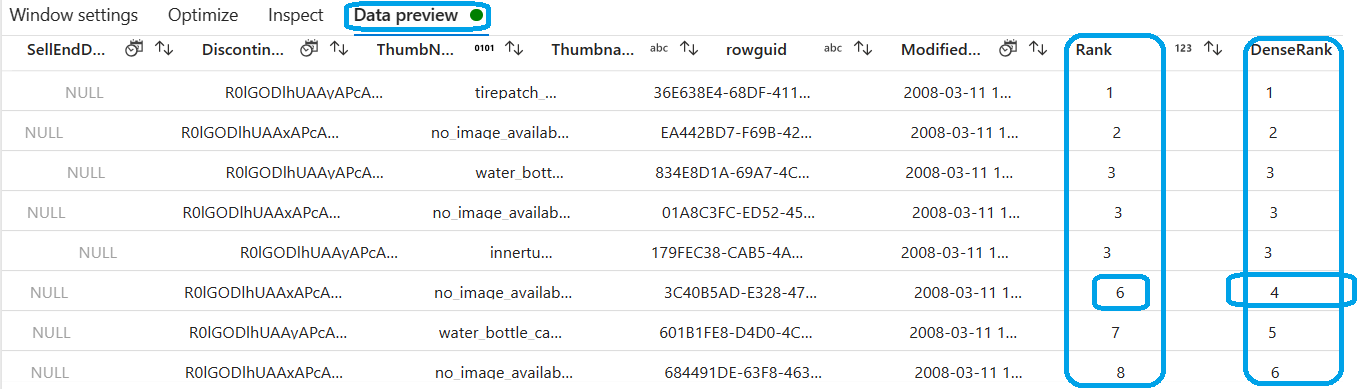
Click on Window transformation>>in Windows settings tab>>click on 4. Window columns>>click on +Add>>Add column(as shown below)>>type DenseRank for the newly launched column(on left as shown below)>>type denseRank()(on right as shown below) in expression box

A screenshot of a computer

Description automatically generated

**Step8:**

Click on Window transformation>>Data preview>>Refresh>>navigate to extreme right and here we can see Rank and DenseRank values and here in DenseRank if we see the next immediate ranks are not getting skipped(as shown below) as compare to Rank



**Step9:**

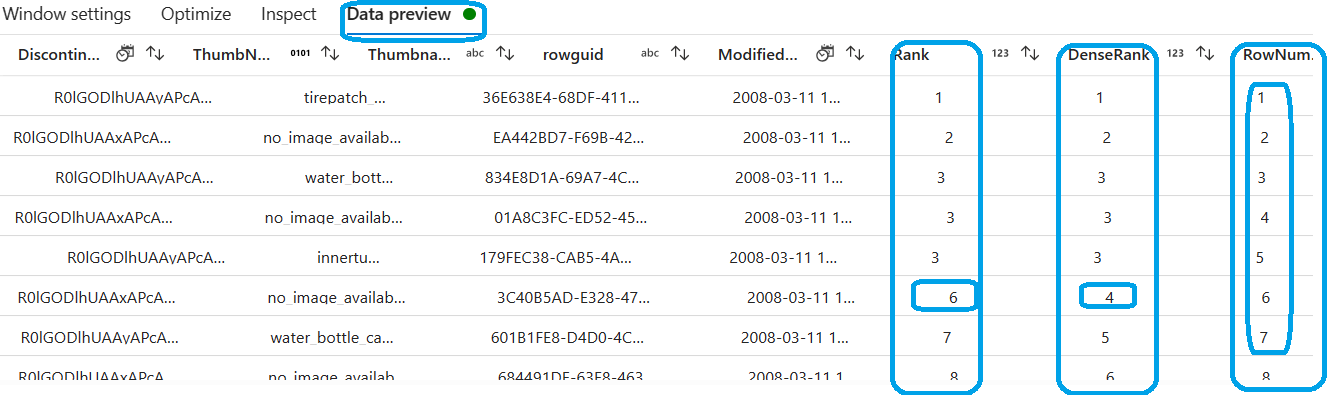
Click on Window transformation>>in Windows settings tab>> click on 4. Window columns>>click on +Add>>Add column (as shown below)>>type for the newly launched column (on left as shown below)>>type rowNumber()(on right as shown below) in expression box

A screenshot of a computer

Description automatically generated

**Step10:**

Click on Window transformation>>Data preview>>Refresh>>navigate to extreme right and here we can see Rank and DenseRank & RowNumber values as shown below



# 

# **What are the dis-advantages of using traditional frameworks:**

Big data or Hadoop carries huge volumes of data, here we try to process huge volume of data.

Spark is been used by everyone now a days for doing the transformation, and before Spark we have Hadoop and Hadoop is one of the solutions for Big Data(BD)and when we want to handle huge volume of data then we use Hadoop framework, earlier to handle the Big Data we use Hadoop and it is hdfs(Hadoop Distributed File System) plus map reduce, **we can use Hadoop for loading the data and map reduce for processing the data.**

Hadoop is one of the solutions for big data, we can use hdfs to load any kind of data(i.e: Big Data)and that can be generated from many different kind of sources and can generate any kind of data like below

(i)Structure data (ii)Semi structured data

This data will be generated from variety of different kind of sources and map reduce is the framework where we can do the transformations. we can use this map reduce for doing the transformations using Java language we should be having little knowledge of Java when we want to perform any kind of transformations and while doing transformations, actions like when we have mappers and reducers where we can perform the transformations and for each mapper and for each reducers we can able to load the data in a Hard Disk

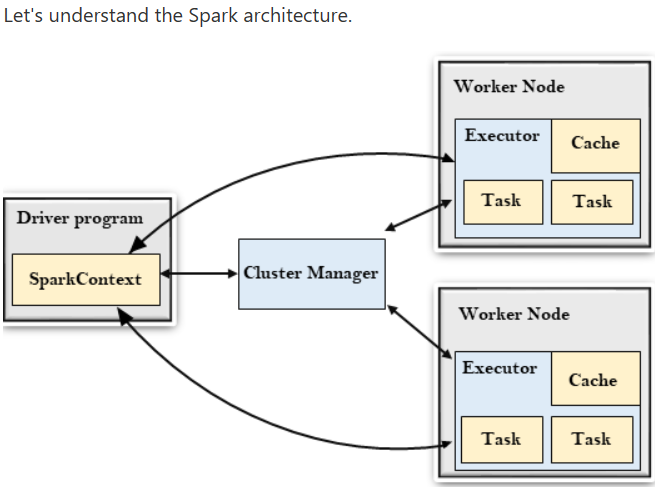
Loading the data in a Hard Disk is a costliest operation, it takes time to load each and every time and getting the data again from hard disk will take again a lot of time that’s the reason we started using Spark.

Spark frame work is basically a Hadoop eco system and spark is 100 times faster than Map reduce, Spark is not meant for loading the data, it meant for doing the transformations like map reduce, so we use HDFS and for underline storage we have many Hadoop based, or HDFS, we can use many things for Spark for loading the data and for doing the transformation we can use spark and is much much faster than Map reduce. Spark itself is an in-memory competing framework, if we are trying to load the data in the hard disk using spark then it is way much faster than Map reduce.

# [**Apache Spark**](https://spark.apache.org/)

Apache Spark is an open-source distributed general-purpose cluster-computing framework. We can use Spark if we are at a point where it does not makes sense to fit all our data on RAM and no longer makes sense to fit all our data on a local machine. On a high level, it is a unified analytics engine for Big Data processing, with built-in modules for streaming, SQL, machine learning, and graph processing. Spark is one of the latest technologies that is being used to quickly and easily handle Big Data and can interact with language shells like Scala, Python, R & SQL

# Apache Spark Architecture:

****

If we see above Spark Architecture we have a Driver Program and multiple worker nodes, we can have many worker nodes in Spark architecture, we can define this worker nodes while creating a cluster, here Driver Program will talk to worker nodes and once we submit the job to drivers program then driver program ill submit the job to all the worker nodes inside the cluster, in worker nodes we can see multiple components like Cache, Task, Executor. Etc and the actual task is performed at Executor level, we can see in Worker node we have an executor where we perform our tasks, once we submit the job to worker node then Driver will submit the job to all over worker nodes inside the cluster and cluster manager this cluster manager will do the resource handling, we can have multiple cluster managers in Apache Spark Architecture, this cluster managers will do the resource handling whatever the worker nodes needs all the resources then cluster manager will provide to worker nodes.

If we want to talk Hadoop related or If we want to use only Spark related framework or if we want to use spark standalone clusters and other cluster managers like Misos, Kubernetes…etc different cluster managers in the market where we need to handle the resource handling then this cluster managers can do the needful, the further information we can get it from below link

<https://www.javatpoint.com/apache-spark-architecture>

So, in Spark architecture we have Driver program/Driver node with multiple Worker nodes(as shown in above image) once we submit the job to driver program then driver will submit the job to all over worker nodes inside the cluster and actual task is performed at Executor level and once the task is done then the result will goes back to the driver program.

Spark basically carries Master & Slave Architecture when we have one master node and all metadata information, we have slave architecture, even in Hadoop also we have name node and different data node.

The Spark follows the master-slave architecture. Its cluster consists of a single master and multiple slaves.

The Spark architecture depends upon two abstractions:

* Resilient Distributed Dataset (RDD)
* Directed Acyclic Graph (DAG)

The Spark architecture consist of single master and multiple slaves, based upon the volume of data and work load we can configure data bricks cluster we can specify maximum 12 worker nodes based upon the work loads for you to do the transformation based upon the volume of data and based upon the workload. We have to choose all these things while creating a data bricks spark cluster and the spark architecture depends upon two abstractions one is (i)Resilient distributed dataset (ii)Directed as Acyclic graph.

In Spark Resilient Distributed Dataset(RDD) is mainly used for loading unstructured kind of data, when we have unstructured data and we want to convert to schema and we need to convert to the structure again.

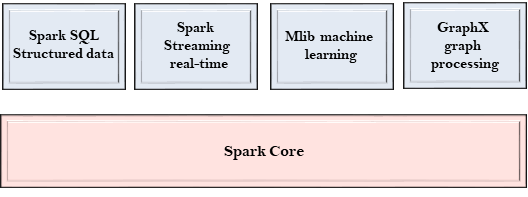
The Resilient Distributed Datasets are the group of data items that can be stored in-memory on worker nodes. Here,

* Resilient: Restore the data on failure.
* Distributed: Data is distributed among different nodes.
* Dataset: Groups of data.

# Spark Components:

The Spark project consists of different types of tightly integrated components. At its core, Spark is a computational engine that can schedule, distribute and monitor multiple applications.

Let's understand each Spark component in detail.



# Spark Core

* The Spark Core is the heart of Spark and performs the core functionality.
* It holds the components for task scheduling, fault recovery, interacting with storage systems and memory management.

# Spark SQL

* The Spark SQL is built on the top of Spark Core. It provides support for structured data.
* It allows to query the data via SQL (Structured Query Language) as well as the Apache Hive variant of SQL? called the HQL (Hive Query Language).
* It supports JDBC(one of the source providers) and ODBC(one of the source providers) connections that establish a relation between Java objects and existing databases, data warehouses and business intelligence tools.
* It also supports various sources of data like Hive tables, Parquet, and JSON.

# Spark Streaming

* Spark Streaming is a Spark component that supports scalable and fault-tolerant processing of streaming data.
* It uses Spark Core's fast scheduling capability to perform streaming analytics.
* It accepts data in mini-batches and performs RDD transformations on that data.
* Its design ensures that the applications written for streaming data can be reused to analyse batches of historical data with little modification.
* The log files generated by web servers can be considered as a real-time example of a data stream.

# MLlib

* The MLlib is a Machine Learning library that contains various machine learning algorithms.
* These include correlations and hypothesis testing, classification and regression, clustering, and principal component analysis.
* It is nine times faster than the disk-based implementation used by Apache Mahout.

# GraphX

* The GraphX is a library that is used to manipulate graphs and perform graph-parallel computations.
* It facilitates to create a directed graph with arbitrary properties attached to each vertex and edge.
* To manipulate graph, it supports various fundamental operators like subgraph, join Vertices, and aggregate Messages.

# Resilient Distributed Dataset(RDD):

The RDD (Resilient Distributed Dataset) is the Spark's core abstraction. It is a collection of elements, partitioned across the nodes of the cluster so that we can execute various parallel operations on it.

There are two ways to create RDDs:

* Parallelizing an existing data in the driver program
* Referencing a dataset in an external storage system, such as a shared filesystem, HDFS, HBase, or any data source offering a Hadoop Input Format.

RDD is a fundamental data structure of spark and it is the primary data abstraction in Apache spark and spark core, RDD’s are fault tolerant immutable distributed collections of objects, which means once we create an RDD we cannot change it, Each Dataset in RDD is divided into logical partitions which can be computed on different nodes of the cluster. The further information about RDD’s we can get from below link.

# **Azure Databricks:**

It is an industry-leading, cloud-based data engineering tool used for processing, exploring, and transforming Big Data and using the data with machine learning models. It is a tool that provides a fast and simple way to set up and use a cluster to analyse and model off of Big data. In a nutshell, it is the platform that will allow us to use PySpark (The collaboration of Apache Spark and Python) to work with Big Data.

Azure Databricks is a data analytics platform optimized for the Microsoft Azure cloud services platform.

# **Features of Azure Databricks:**

* It can process large amounts of data with Databricks and since it is part of Azure; the data is cloud native.
* The clusters are easy to set up and configure.
* It has an [Azure Synapse Analytics](https://intellipaat.com/blog/azure-synapse-analytics/) connector as well as the ability to connect to Azure DB.
* It is integrated with Active Directory.
* It supports multiple languages. Scala is the main language, but it also works well with Python, SQL, and R.
* Azure Data bricks is also an ETL tool where we extract the data from source and loads it into target

**Note:**

Always implement Azure Databricks and cluster in **Azure paid subscription** bcoz in free trials the Databricks cluster are not supported.

# **Implementations steps for Azure Databricks:**

**Step1:**

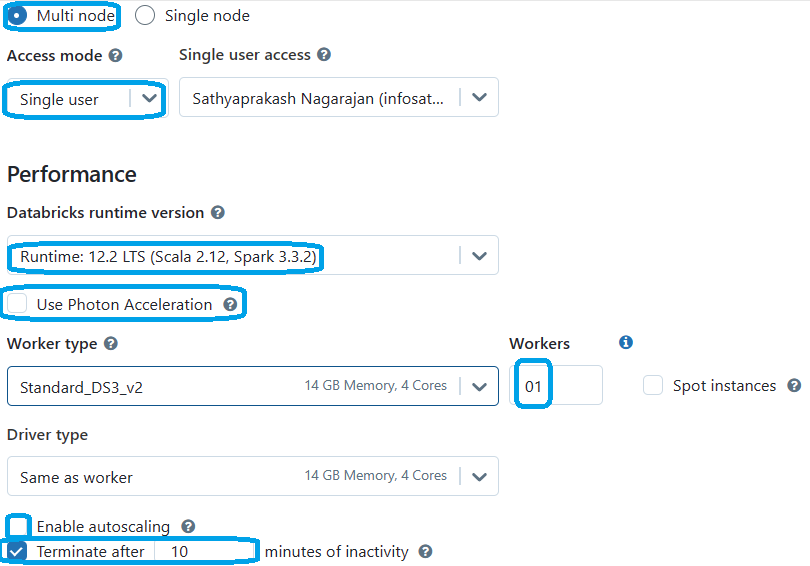
Search for Azure Databricks>>fill the details accordingly as explained in the class>>Pricing Tier: Trial (Premium – 14-Days Free DBU’s)>>Networking>>Encryption>>Tags>>Review+Create>>Create.

**Step2:**

Wait for some time till the Databricks workspace gets Deployed>>click on Launch Workspace.

**Step3:**

Mouse over on left and a window will get open and in that click on compute>>Create a compute>>Create a cluster>> and follow the diagram as shown below



**Step4:**

Click on workspace (on left side)>>workspace>>Add>>Notebook>> a new Notebook will get created>>change the title(Ex:Python NoteBook1) of the notebook by clicking on top.

Hence, we have created an Azure Databrick workspace, a cluster and Python Notebook in it.

Whatever the transformations we have done till now like moving the data from source to target and even with Dataflows then we can use ADF, and if we want to do the complex transformations and if we want to do any user defined functions then we use the Azure Databricks where we can process huge volume of data, if we are having Peta bytes of data or Giga bytes of data that we are receiving from source where we need to do complex kind of transformations then we can use Azure Databricks service.

In Azure Databricks we can do the transformations sufficiently using Spark based API’s, most of the people are familiar with SQL, or Python or Scala or R language, since Spark supports all these 4 API’s we can choose any of these languages and we can write the code in Azure Databricks service, we can write any complex code, or a user defined functions we can still do that in Azure Data Bricks Service.

Azure Databricks is fully managed by Microsoft we can use of any language as per our familiarity, we have to create the Azure Databricks service before we start writing the code as shown above.

In Azure Databricks we can connect to any kind of data source, we can connect to On-premises, we can connect to Azure Blob Storage or DataLake Gen2 Storage…etc. and we can move the data from any source to any destination using Azure Data Bricks

When we are defining clusters in Azure Databricks then we have 2 nodes…

i.e:

**(i)Multi node>>**Here multiple users/nodes can connect to the cluster and cluster Notebooks that has been created...

**(ii)Single node>>**Here single users/nodes can connect to the cluster and cluster Notebooks that has been created…

We are having multiple Long term support (LTS) versions for Azure Databricks workspace provisioning as shown below…

A screenshot of a computer

Description automatically generated

Use Photon Acceleration:

This accelerates modern Apache Spark workloads, which reduces total cost per workload.

**Implementations steps for Azure Databricks and DataBricks cluster:**

**Note:**

Always implement Azure Databricks and cluster in Azure paid subscription bcoz in free trials the Databricks cluster are not supported.

**Step1:**

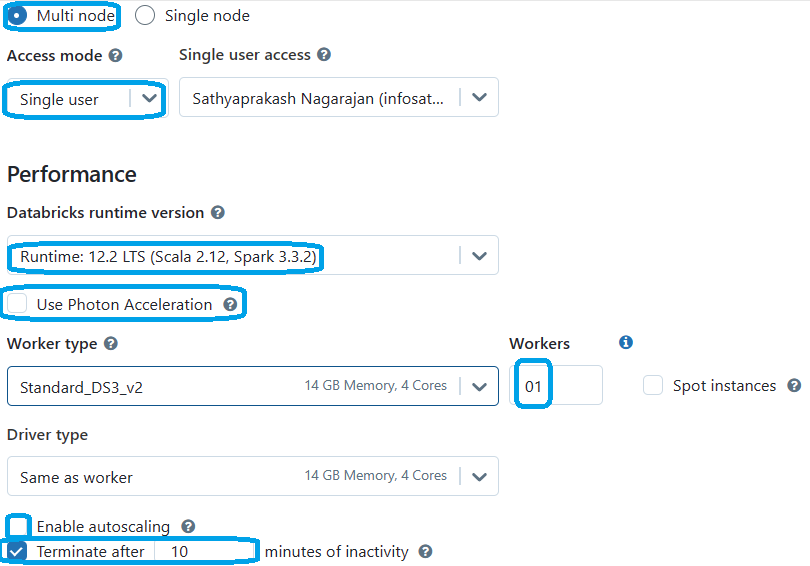
Search for Databricks>>fill the details accordingly as explained in the class>>Pricing Tier: Standard (Apache Spark, Secure with Azure AD)>>Networking>>Encryption>>Tags>>Review+Create>>Create>>Wait for some time till the Data bricks gets launched>>click on Launch Workspace.

Refer the below documentation for Cluster implementation in Azure Data Bricks

[Azure Databricks Hands-on. This tutorial will explain what is… | by Jean-Christophe Baey | Medium](https://medium.com/@jcbaey/azure-databricks-hands-on-6ed8bed125c7)

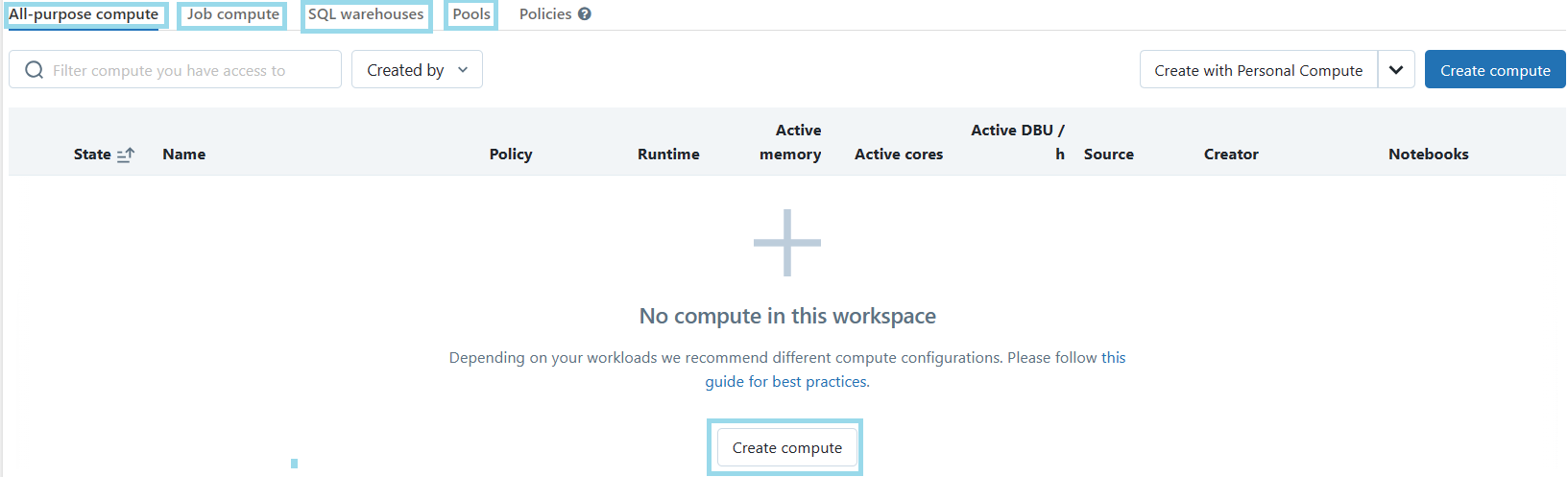
**Step2:**

Mouse over to left>>click on Compute>>Create compute(center)>>create a cluster>>Fill the details accordingly as shown in image below and finally click on Create compute(@ below)



**Steps to see the clusters in Azure Data Bricks:**

* Search for Databricks>>Create>>fill the details and wait till the Databricks gets deployed>>Launch workspace>>Compute (left side)>>then on top will see as shown below.



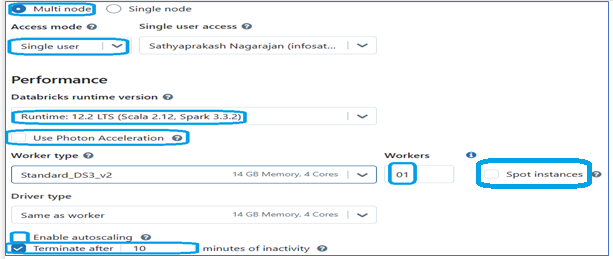
# We can create 2 types of clusters in Azure Databricks as mentioned below.

(i)All-purpose clusters/Interactive based cluster: When we are working interactively with notebook or Number of notebooks then we use this interactive based clusters

(ii)Job clusters/Instant clusters:  We use job clusters to run fast and robust automated jobs using UI & API.

**Pools:** When we want to make a list of resources and that we want to keep in all those pools then we can make use of all these pools.

**Azure Databricks makes** a distinction between all-purpose clusters and Job clusters. We use all-purpose clusters to analyze data collaboratively using interactive notebooks and we use Job clusters to run fast and robust automated jobs, we can create an All-purpose cluster using UI, CLI & Rest API.

* Click on create compute to create a cluster and while creating cluster in Azure Databricks we can set the worker type and driver type configurations (ex: 14GB Memory, 4 Cores.Etc…) based upon the volume of data that we are processing with this Azure cluster Databricks and while creating the cluster we can follow as below and don’t check the Spot instance check box as shown below
* Once we have setup the above details then click on create compute/create cluster and wait for some 15-20 minutes till the cluster gets created.
* After the cluster gets deployed>>click on More (top right)>>Permissions>>click on the knob>>All Users>>click on the knob(beside)>>choose Can Manage>>+Add>>Save>>close (top cross)
* To create a Notebook in Databricks cluster>>click on +New(top left)>>Notebook>>change the title of the notebook and select the language/API we want like Python, SQL, R programming…etc and in body of the notebook we can write the script, the default script will get Python.

A screenshot of a computer

Description automatically generated

**The other way around to create Azure Databricks Workspace and Databricks cluster using community Edition:**

**Step1:** Go to the below lilnk

https://www.databricks.com/learn/training/home

**Step2:** Click Try Databricks and fil the details like

First name:

Last name:

Email:

Company:

Title:

Country:

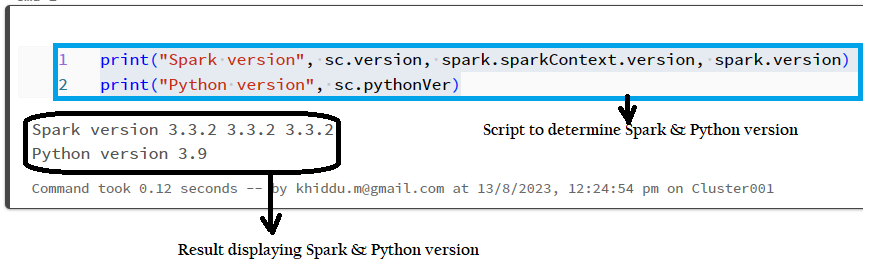
click on Continue>>Click on Microsoft Azure>>click on Continue>>click on Get Started with Community Edition>>Start Puzzle>>goto the mail which we gave and here will find one mail landed in our inbox from Databricks>>go inside the email and click on this link(hyperlink)>>Pass Password & Confirm Password(any password we can give)>>Click on Reset password

**Step3:** First will create a cluster by clicking on compute>>create compute>>then will launch a Pyspark notebook in data bricks cluster.

* Copy the below Python script and paste it in Python notebook(as shown below) and hit shift+enter to run the script in Python notebook

print("Spark version", sc.version, spark.sparkContext.version, spark.version)

print("Python version", sc.pythonVer)



* In Azure Data bricks cluster Spark supports four (4) different types of languages/APIs…i.e: (i)Python, (ii)Scala, (iii)SQL & (iv)R-programming…
* If we want to see the version history of a notebook, then click on File(@ the top) in the notebook>>scroll down>>Version history.
* Sometimes If we are getting errors while executing the python scripts in Cluster notes books then click on Run on top in notebook>>click on Restart compute resource or go to compute/cluster and restart the cluster.
* Refer the link below for Azure Databricks hands-on!

[Azure Databricks Hands-on. This tutorial will explain what is… | by Jean-Christophe Baey | Medium](https://medium.com/@jcbaey/azure-databricks-hands-on-6ed8bed125c7) (From this link we can get all the python, Scala codes…etc)

* Generate a new cell in the notebook by click on top right in the cell as shown in image below

A screen shot of a computer

Description automatically generated

* After the cell has generated in notebook paste the Scala code(the Scala code we can get from above link) in the cell body as shown below



* In a single Notebook we can use Python, Scala, SQL, R-programming based upon the requirements only we must choose at the top right(like Scala, Python…etc.)
* Python libraries we are using, and dealing with spark within Python notebooks that’s why we named it as Pyspark.

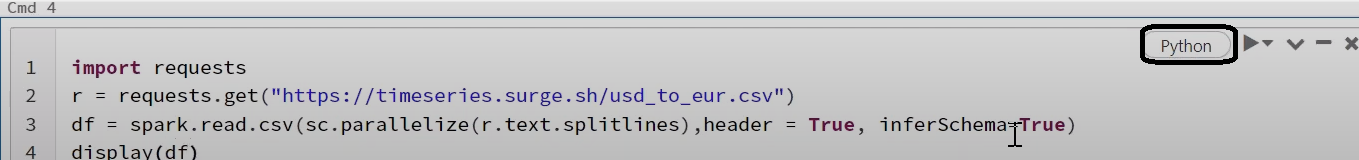
# **Implementations steps to read .csv file from Python notebook in Azure Databricks cluster:**

**Step1:** Create Azure Data bricks.

**Step2:** Launch the workspace.

**Step3:** Create a cluster (with minimal requirements of configurations).

**Step4:** After the cluster got created>>click on +New>>Notebook and ensure Python selected at the top>>Paste the below code shown in image (this code we can get from above link) and click on run cell(on extreme right in the cell body by clicking on the nob)



**Explanation of above Python script notebook:**

**import requests>>**Importing a library.

**r = requests.get("https://timeseries.surge.sh/usd\_to\_eur.csv")>>**inthis command we are defining a variable(r) and getting .csv file from some portal/website(timeseries.surge.sh)

**df = spark.read.csv(sc.parallelize(r.text.splitlines()), header=True, inferSchema=True)>>**command to read the .csv file

**display(df)>>**method to display the file.

**Note:** For all the labs the python scripts we are considering is from the link below.

[Azure Databricks Hands-on. This tutorial will explain what is… | by Jean-Christophe Baey | Medium](https://medium.com/@jcbaey/azure-databricks-hands-on-6ed8bed125c7)

# **Connecting to Blb SA from Azure Databricks cluster for mounting the directory:**

**Step1:** Create a Blob Storage Account & container inside the storage account.

**Step2:** Create an Azure Databricks>>Launch the workspace>>and create a cluster (by clicking on compute left side).

**Step3:** Get the Storage account; Access keys; & container name inside the storage account as shown below as an example

**Storage Account Name:** 1961mysa

**Access Keys:** I/4UBW2dm+Cl1XX2i2N9Y5LA3d1VCQB6WbX64p+fRpXxQPcfDG/DLKbcwAgPbigoE0cufB+4TIH6+ASt9xzTnA==

**Container name:** mycon

**Step4:** Now from the above link copy the entire code (To set up the file access, you need to do this:) and paste it in Pyspark notebook cell and make the changes accordingly as per the SA, Container name & Access keys.

**Step5:** click on extreme right top nob in the cell body and run the python code inside the cell by clicking on **Run cell** and we see output as below.

Mounting: /mnt/mycon

=> Directory /mnt/mycon already mounted

# **Reading .csv file from Blob SA with Windows Azure Storage Blob Service(WASBS) METHOD from Azure Databricks cluster:**

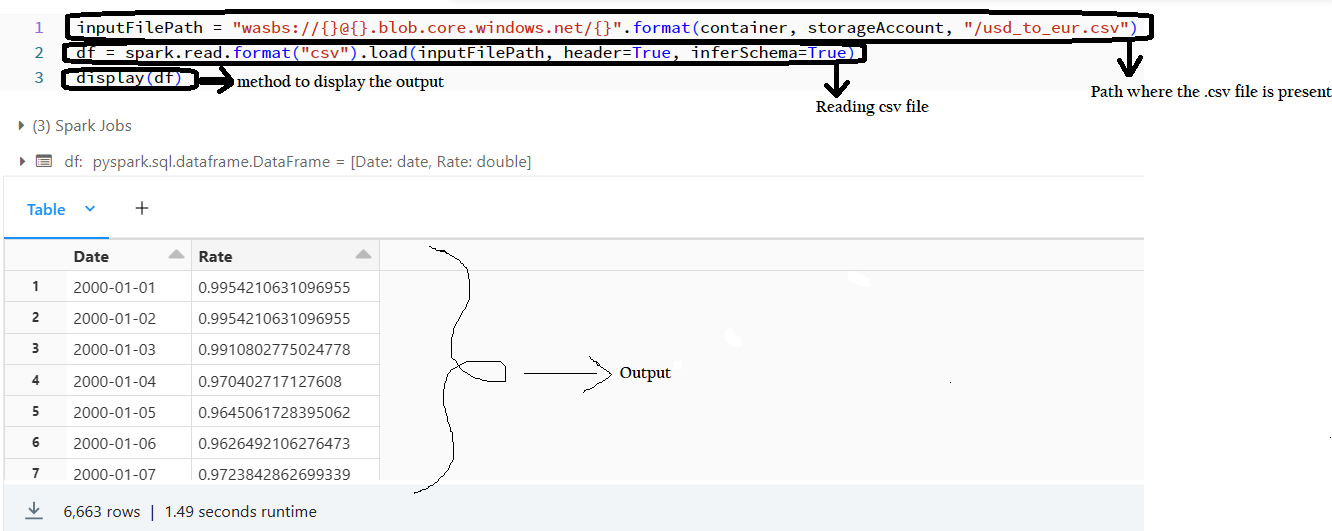
**Step1:** Create a Blob Storage Account & container inside the storage account and upload the .csv file inside the SA container.



**Step2:** Create an Azure Databricks>>Launch the workspace>>and create a cluster (by clicking on compute left side).

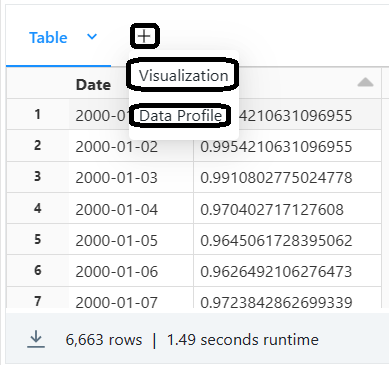
**Step3:** Create a notebook and paste the code (Get the code from above link and also as shown below) inside the cell and run the code.

**Step4:** Then will see the output as shown below



**Note:** These commands are absolute case sensitive, so while typing these commands in cluster notebook we have to give **100% attentions with upper case and lower case.**

**Step5:** click on + as shown below in image and here we can do the data visualization and multiple types of charts(Line chart, Bar chart, Area chart, Pie chart, Scatter chart, bubble chart…etc. etc.)and can also apply various filters on it.



**Step6:** create a new cell and type the below commands which gives different results

**df.printSchema()>>**this command shows the schema of csv file

**df.describe().show()>>**this command shows the aggregates values of the file records(like count, mean, min, max, stddev…etc)

**df.head(5)>>**this command shows only top 5(whatever the No we pass here that many records will be displayed)records.

**Step7:** create a new cell and type the below commands which helps us to create the temporary view and to convert or replace the code from Python to SQL

df.createOrReplaceTempView("xrate")>>hold the result in temp view and to convert from Python to SQL

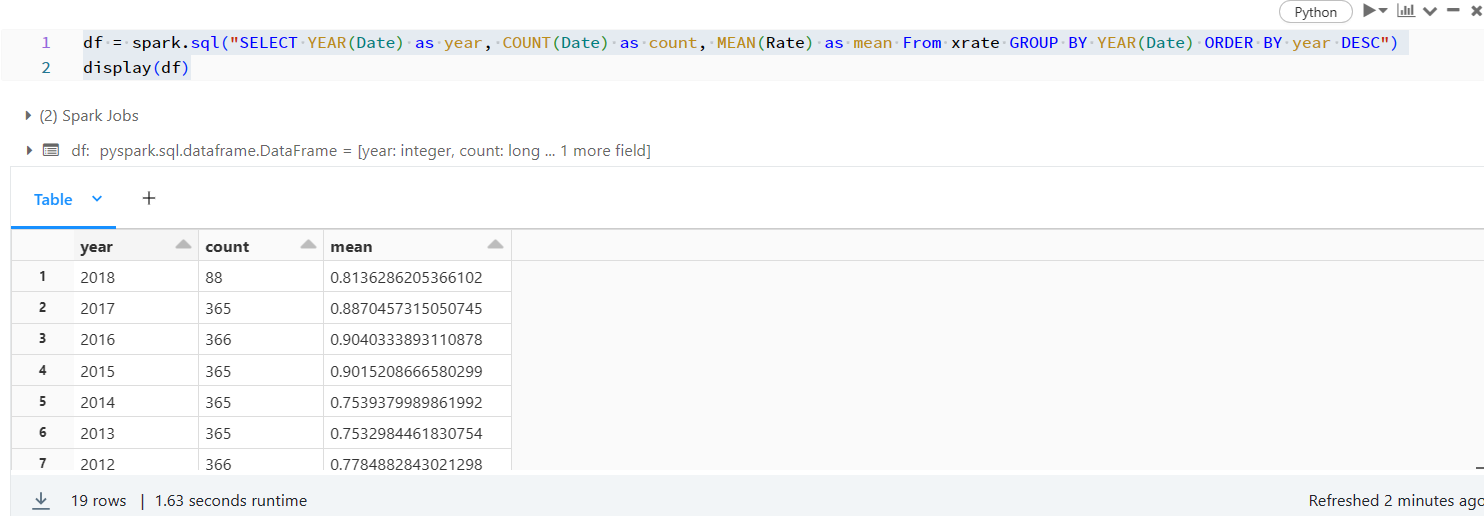
df = spark.sql("select \* from xrate")>>Typing SQL query with spark.sql method

display(df)>>method to display the output.

**Step8:** Create a new cell and paste the below command to get the output displayed as Group by year and order by year Desc from xrate(temp view)

df = spark.sql("SELECT YEAR(Date) as year, COUNT(Date) as count, MEAN(Rate) as mean From xrate GROUP BY YEAR(Date) ORDER BY year DESC")>>command to get the data from xrate

display(df)**>>**command to display the output. And the result is shown below



**Step9:**

Create a new cell and if we want to write the SQL query directly then first select SQL on to right inside the cell and then directly, we can write the SQL Queries as shown below.

%sql

SELECT YEAR(Date) as year, COUNT(Date) as count, MEAN(Rate) as mean From xrate GROUP BY YEAR(Date) ORDER BY year DESC

A screenshot of a computer

Description automatically generated

Here on top if we see **%sql>>this is called as Magic command**, when we select SQL on top right then we can see this magic command will automatically be printed in our cell body as shown above.

# Pyspark:

When we are integrating with Python library and with this spark we can able to call it in our coding. Hence wit this we can Pyspark.

Here in the above methods, groupBy, agg, sort..etc.. these are all methods and we are applying these methods on top of dataframe(df).

# Importing Apache Spark libraries & writing the code in Databricks cluster in %Scala:

Create a new cell in same cluster Notebook and paste the below code

|  |
| --- |
| %scala |
|  |  |
|  | import org.apache.spark.sql.functions.\_ |
|  | var df = spark.table("xrate") |
|  | // or |
|  | // df = spark.sql("select \* from xrate") |
|  | var Row(minValue, maxValue) = df.select(min("Rate"), max("Rate")).head |
|  |  |
|  | println(s"Min: ${minValue}, Max: ${maxValue}") |

Here in above code, we are importing Apache Spark function to get the Min & Max values for Rate column

Hence, like this we can write the code in Azure Data bricks cluster notebooks on either Python, Sql or Scala by mentioning the magic command in cell body.

# **Azure AZ copy:**

AzCopy is a command-line utility/tool that we can use to copy blobs or files to or from a storage account; we have to download AZ Copy software/tool, connect to our storage account, and then can transfer/migrate the files.

# **(i)Migration From Private cloud to Public cloud(Forward Migration):**

* Download the AzCopy dll from below link in our laptop

<https://docs.microsoft.com/en-us/azure/storage/common/storage-use-azcopy-v10>

* After its gets downloaded>>extract the zip file>>go inside the folders>>copy(ctrl+c) the azcopy.exe and paste it in below path in our laptop>>

**C:\Windows\System32**>>

* Go to Azure portal and create a storage account and create a container(blob) storage service
* come inside the blob/container storage>>click on properties (left side inside)>>copy the URL and paste it in a separate notepad
* Now go to Shared access signature (inside the storage account)>>select all the options>>click radio button HTTPS and HTTP(for sure) >>click on Generate SAS and connection string>>copy the SAS token(carefully)>>concatenate this SAS token with container blob storage service URL(as example shown below)

https://mysa1972.blob.core.windows.net/mycontainer >> Blob storage Container URL

?sv=2020-08-04&ss=b&srt=sco&sp=rwdlacitfx&se=2021-12-24T20:09:55Z&st=2021-12-24T12:09:55Z&spr=https,http&sig=HmtQmRiO0C%2BablXp8%2B961rT6GtcYZSuJxakd8josccs%3D>> SAS generated token

# **Doing the Concatenation (as shown below):**

https://mysa1972.blob.core.windows.net/mycontainer/?sv=2020-08-04&ss=b&srt=sco&sp=rwdlacitfx&se=2021-12-24T20:09:55Z&st=2021-12-24T12:09:55Z&spr=https,http&sig=HmtQmRiO0C%2BablXp8%2B961rT6GtcYZSuJxakd8josccs%3D

* Now search for command prompt/Azure CLI in our laptop and open with run as administrator>>type azcopy.exe copy "here give the source path where our files are present in our local laptop to copy to our Azure container storage service" "here give the container storage service URL along with SAS token" --recursive>>and then finally hit enter
* Now come to our container storage service and we could be able to see all the files/data that we have uploaded using Azcopy from our local laptop to Azure cloud storage services.
* Hence we have migrated the Data from On-prem(Private Cloud) to Public cloud computing

# **Migration from Public cloud to Private cloud (Reverse Migration):**

* Create an empty folder in any drive in your laptop (ex: F drive)
* Create a Storage Account and create a blob container and let’s keep some data init (ex: files)
* Open the cmd prompt with Administrator access and pass the below commands using AzCopy as shown below.

azcopy.exe copy “Source Storage Account URL (container\_url\_followed\_with SAS token)” “Destination Path this is our F drive local path from our local laptop” --recursive

# **Migration of Data from One storage Account to Another (Platform or Cloud to Cloud Migration)**

**Note:** Firstly, ensure the 2nd storage account (or) destination storage account should be empty and must be having a container/Blob storage service created inside the storage account and not having the same files/data which we are going to migrate using azure AZ Copy and if the same files/data already present in the destination storage account and if we run the command AZ Copy again then it **will ignore** if already files are present.

* Follow all the steps same as above and now in command prompt
* azcopy.exe copy “Source Storage Account URL (container\_url\_followed\_with SAS token)” “Destination Storage Account URL(container\_url\_followed\_with SAS token)” --recursive

# **DataBase Migration Assistant (DMA):**

Data migration Assistant helps us to migrate the Databases often from on-premises locations to a cloud platform. The Data Migration Assistant (DMA) also helps us to upgrade to a modern data platform by detecting compatibility issues that can impact database functionality when we upgrade to a new version of SQL Server or migrate to [Azure SQL Database](https://learn.microsoft.com/en-us/azure/azure-sql/database/sql-database-paas-overview), it recommends performance and reliability improvements for our target environment and allows us to move our schema, data, and DB objects from our source servers(on-prem) to our target server(Public cloud) or vice versa.

# **Assess on-premises SQL Server Instances migrating to Azure:**

Assess on-premises SQL Server instance(s) migrating to Azure SQL Database or Azure SQL Managed Instance. The assessment workflow helps us to detect the following issues, which may affect our Azure SQL migration and provides detailed guidance on how to resolve them.

* Migration blocking issues: Discovers the compatibility issues that block migrating on-premises SQL Server database to Azure SQL Database. DMA provides recommendations to help you address those issues.
* DMA provides a comprehensive set of recommendations, alternative approaches available in Azure, and mitigating steps so that we can incorporate them into our migration projects.

# **Use the below script to change the collation(if required) of the DB in our local Server/SSMS:**

USE MASTER

GO

ALTER DATABASE DB Name

COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS

GO

# **To change the collation of DB first convert it to to Single User Operations with below script:**

USE MASTER

ALTER DATABASE DBName SET SINGLE\_USER WITH ROLLBACK IMMEDIATE

# **Again convert it back to Multi User Operations with below script:**

ALTER DATABASE DBName(give the DB name here)

SET MULTI\_USER WITH ROLLBACK IMMEDIATE

# **Migration Process/Procedure of SQL DB/DB Objects from On-prem(Private cloud) to Azure (Public cloud) using Database Migration Assistant(DMA)**

**Note:**

To do this migration ensure that we have installed SQL Server 2019 & SSMS and Microsoft Data Migration Assistant in our local laptop & also collation of source and target DB should be same.

**Step1:** Create a DB in our local server (On-prem)

**Step2:** Create tables in our local DB with some data init.

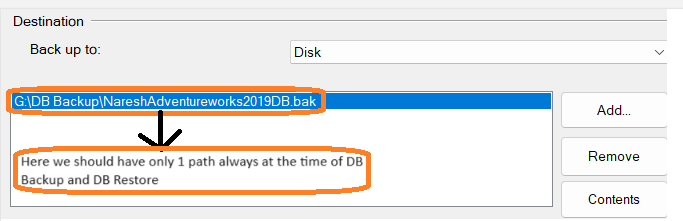
**Step3:** Create a Sql Server and SQL DB in Azure cloud platform

**Step4:** Now open local Database Migration Assistant in our laptop and follow the steps as shown in the class.

**Step5:** After the migration has been completed in DMA tools connect to Azure cloud computing SQL server and SQL DB via SSMS and check whether the data has been migrated successfully or not with all the data and entries.

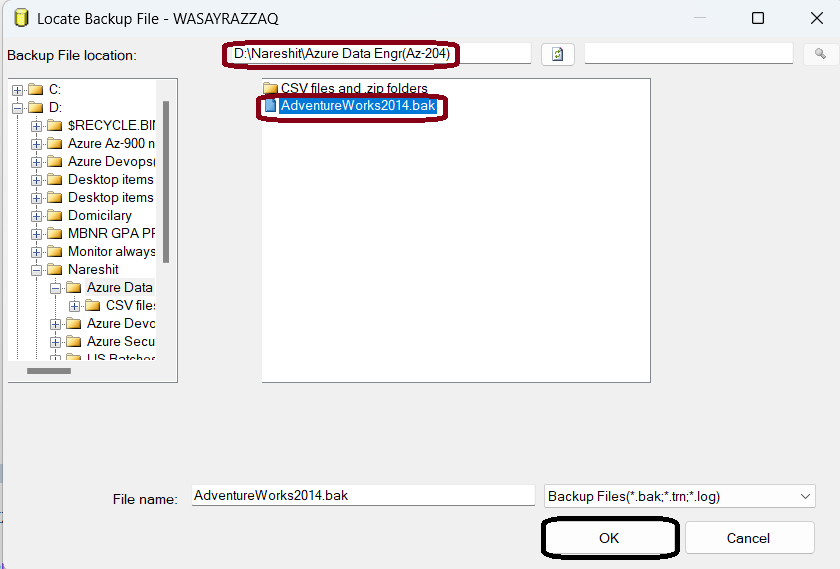
# **Steps for taking the DB Backup:**

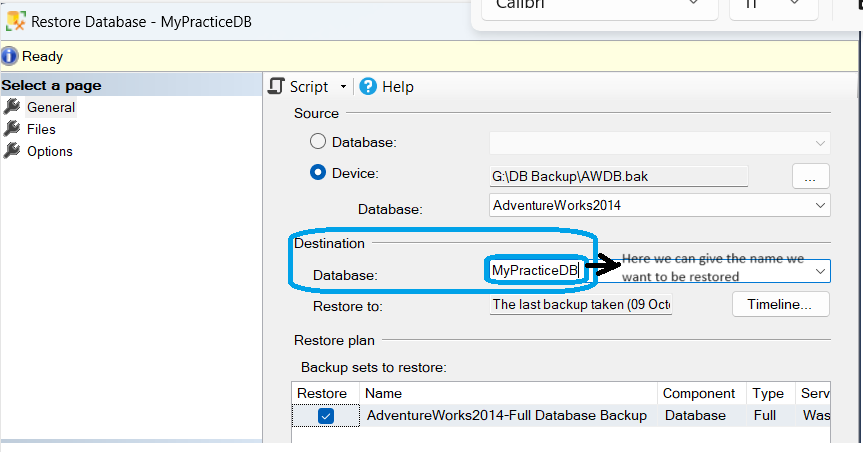
Login/connect to local server SSMS>>right click DB>>Tasks>>Backup>>Add>>click on 3 dots>>browse the path (ex: D:\Nareshit\Azure Data Engr (Az-204)\AdventureWorks DBs) where we want to keep the DB backup>>File name: PracticeNareshDB.bak (any name)>>Ok>>Ok>>Ok>>Ok



# **Restoring Adventure Works DB in Private Cloud(local SSMS):**

**Step1:** Open SSMS>>expand the Databases folder>>Right click on Databases folder>>Restore Database…>>click on Device radio button>>click on 3 dots beside Device radio button>>Add>>Navigate to the path in our local laptop correctly where we kept our **.bak file of Adventure Works (as shown in below image)>>click on the file and say ok**>>again ok>>Now here in Destination give the Database Name whatever we want to be restored(as shown in image below)>>click Ok>>then will get a popup saying the DB file has been restored successfully>>say ok again.

****

****

**Step2:** Refresh the Database Folder and here will see AdventureWorks2014 DB in our SSMS

**Structured Query Language (SQL) Queries:**

Structured query language (SQL) is a programming language for storing and processing information in a relational database. A relational database stores information in tabular form, with rows and columns representing different data attributes and the various relationships between the data values. We can use SQL statements to store, update, remove, search, and retrieve information from the database. We can also use SQL to maintain and optimize database performance.

Structured query language (SQL) is a popular query language that is frequently used in all types of applications. Data analysts and developers learn and use SQL because it integrates well with different programming languages. For example, they can embed SQL queries with the Java programming language to build high-performing data processing applications with major SQL database systems such as Oracle or MS SQL Server. SQL is also fairly easy to learn as it uses common English keywords in its statements

SQL was invented in the 1970s based on the relational data model. It was initially known as the structured English query language (SEQUEL). The term was later shortened to SQL. Oracle, formerly known as Relational Software, became the first vendor to offer a commercial SQL relational database management system.

# What are the components of a SQL system?

Relational database management systems use structured query language (SQL) to store and manage data. The system stores multiple database tables that relate to each other. MS SQL Server, MySQL, or MS Access are examples of relational database management systems. The following are the components of such a system.

SQL table is the basic element of a relational database. The SQL database table consists of rows and columns. Database engineers create relationships between multiple database tables to optimize data storage space.

# Refer the link below to practice SQL queries:

(i) [shivaniNKi8/SQL-Server-Adventure-Works: Basic to Complex SQL Server queries with Adventure Works database including window functions, triggers, string manipulation, user-defined functions, CTE, recursive CTE, DML and DDL (github.com)](https://github.com/shivaniNK8/SQL-Server-Adventure-Works)

(ii) [SQL HAVING Clause (w3schools.com)](https://www.w3schools.com/sql/sql_having.asp)