Azure Data Factory is a ETL tool-Extract Transform, and Load tool used for creating, scheduling and orchestrating data pipelines to move data from various source systems to different target systems We can use this tool for batch processing needs. Batch Processing is useful for scenarios where data is generated periodically or in large volumes and we need to process it in chunks or batches rather than in real time. Its commonly used for data ware housing, data analytics and reporting applications.

Data Processing is of 2 types: Batch Processing & Streaming Processing.

**Batch Processing:**

In batch processing, data is collected over a period (a batch), and processing is performed on that entire batch of data. Batch data is collected over a period of time and processed as a group or batch. Processing occurs after data has been collected and is often performed at specific intervals, such as daily, hourly, or weekly. It is suitable for scenarios where data can be processed and analyzed in retrospect.

**Use cases of Batch Processing:**

DWH, ETL, Daily or Periodic Reporting, Historical data analysis.

**Tools for Batch Processing:** Hadoop MapReduce, Apache Spark (for batch processing), and SQL-based data warehouses.

Maximum batch count in foreach loop in ADF is 50. i.e., up to 50 loop iterations can be executed at once in parallel when is Sequential property is set to false.

**Streaming Processing:** In streaming processing, data is processed in real-time as it is generated or ingested. Data is processed as individual data events or micro-batches, typically without waiting for an entire batch to be collected. It is suitable for scenarios that require immediate analysis and quick responses to events.

**Use cases of Stream Processing:**

Real time monitoring and alerting, Iot(Internet of things) data processing, Fraud detection, Clickstream analysis, Social media sentiment analysis

**Tools for Stream Processing:** Tools like Apache Kafka, Apache Flink, Apache Storm, and Azure Stream Analytics used for streaming processing.

In some cases, a hybrid approach that combines both batch and streaming processing is used to leverage the benefits of both paradigms. For example, data can be ingested in real-time and stored for batch processing and historical analysis. The choice between batch and streaming processing depends on the specific requirements of your data processing tasks and the nature of your data.

**Difference between Azure Synapse Analytics, Azure Data Factory & Azure Data Bricks:**

Azure Synapse Analytics is primarily for data warehousing and structured data analytics. We can transfer data on to a data warehouse within azure synapse.

Azure Data Factory is for data integration and ETL tasks, and Azure Databricks is for big data analytics, real-time processing, and machine learning. The choice of which service to use depends on your specific data processing and analytics needs. Additionally, these services can be used together in a comprehensive data processing and analytics pipeline when required.

**EXTRACT, TRANSFORM AND LOAD:** Azure data factory service is a data integration service (means extracting, transforming, and loading) and can be used for ETL Process. We can store raw data in ADLS Gen2 account. If we wanna have structured data when it comes on to analysis, we can create a data warehouse within a dedicated SQL pool within the azure synapse service. ADF will create a pipeline that can collect data from various sources. You can then transform the data accordingly, and then the transformed data can be loaded in the target data store.

**Difference between ETL and ELT:**

**ETL:** After extracting data from various sources, data is transformed accordingly and transformed data is then load onto a target data store.

**ELT:** Here after data is extracted from various sources, transformation happens after data is loaded in target store. This transformation happens in the target data store.

**AZURE DATA FACTORY:** ADF is a cloud based ETL (extract, transform and load) service, and data integration service. Here we can create our data-driven workflows that can be used for orchestrating our data movement and it can be used to transform your data at scale. Here we can actually connect onto a variety of data stores as the source and the destination. Using ADF also we can copy data from source and transfer on to Azure Synapse DWH via pipelines. Pipeline is a set of activities. we can have 1 activity for copying data , cleaning data , another activity for transforming data .

* **Linked Service:** This is a link to data store. If we want to get data from ADLS Gen2 account at that point of time we will create a linked service. Similarly, onto a destination if we wanna connect onto a table in Azure Synapse we create another linked service for another resource /another instance.
* **Datasets:** We will have 2 datasets, source data set and destination dataset. This represents data.
* **Integration runtime:** underlying compute infrastructure that's required for taking our data, transforming our data, and loading it onto a destination data store. This is known as the integration runtime. Here the underlying compute infrastructure is managed for us by ADF only . Compute power we choose depends on datasets size. More file size more compute power is required.

**CREATION OF AZURE DATA FACTORY SERVICE:**

Home 🡪 All Resources 🡪 +🡪 search for Data Factory 🡪 create.

We will enter subscription, Resource Group.

Instance details:

Name: unique data factory name

Region: Location Name

Version: V2

Click on next

Check box enable Configure GIT later.

ADF allows us to configure a GIT repository with either azure devops or GitHub. Git is a version control system that allows for easier change tracking and collaboration.

Go on next and click on create.

**In CSV comma separated file, every column will be treated as strings.**

**COPYING DATA FROM ADLS TO AZURE SYNAPSE VIA AZURE SYNAPSE:**

Home 🡪

Open ADF Studio, Home 🡪click on Ingest, copy data tool wizard will be open (same wizard as azure synapse)

Properties: select task type: built-in copy task and task schedule: run once now.

Source: Source Linked service connection will be established here. we will create connection on to our source select source as ADLS and then give Linked Service name, select ADLS storage account name, container/file path and click on next, we will check source file format settings column delimiter (, separated). Row delimiter as (new line feed), and first \_row as header enable it and click on next.

Destination: Destination linked service will be created here. We will create connection on to destination. Enter synapse workspace name for Name field, enter server name (Dedicated SQL pool end point), db name, username and password, test connection and create connection. We will choose now target table where data should be uploaded. click on next. In destination settings, column mapping will be done.

For time column in source csv file, it is of string type. For Destination column(time) data type is of datetime.

Settings: Enter pipeline name, select loading method (Bulk Insert), click on next and pipeline will be created.

Monitor 🡪 Pipeline runs🡪 we can see pipeline run succeeded or not.

Author 🡪 pipeline 🡪 we will have created pipelines here. If we click on pipeline name (Copy-To-logdata), we can see activities there, we will see different categories when it comes onto activities that we can actually drag on to canvas to build our pipelines. If I choose activity in the canvas, we will see settings at the bottom.

Author🡪 Datasets 🡪 we will see 2 datasets here source datasets and destination datasets which points out to source data and destination data.

Manage 🡪 Linked services 🡪 we can see source and destination linked services here.

**COPYING DATA FROM CSV TO PARQUET:**

We can also create pipeline in another way, go to Author 🡪 Pipeline 🡪 click on + 🡪 Create Pipeline 🡪 enter pipeline name (Copy-to-Parquet) at RHS and hide properties by clicking on properties icon. Now select our created pipeline (Copy-to-Parquet), In Move & Transform Activity 🡪 drag copy data activity to canvas. Now in

General Tab:

Name: Give name to copy data activity (Copydatatoparquet)

Source: we will mention source dataset here from where we want to pick up the data.

Destination: we will mention destination data set here. We will select ADLS Gen2 in parquet file format here. And we will give name for destination dataset. choose linked service. Browse for the container in which I want to place my file and click on ok.

**Mapping**:

Settings:

User Properties:

Now click on validate all and after validated, click on publish all

Now click on Add trigger 🡪 Trigger now 🡪 click on ok.

Go to Monitor section 🡪 Pipeline runs 🡪 we can see status of pipeline here. If pipelines fail means, there is spaces in column names in parquet-based file format. We need to remove spaces between column names. For that we need to go to mapping tab as shown above mapping highlighted in green colour.

Go to Mapping: Import schemas, now remove spaces in destination table columns wherever there is spaces.

Now validate all and publish all again. Now Add trigger 🡪 trigger now, pipeline will be triggered.

Go to Monitor section 🡪 Pipeline runs 🡪 Pipeline will be succeeded now.

Now go to ADLS Gen2 account 🡪 Parquet container 🡪 Parquet file will be uploaded now.

Copy method (BulkInsert/PolyBase/Copy Command) will not available in sink tab if we are sending the destination file to container in ADLS. If destination store is in Azure synapse, we will get copy method in sink.

**TWO STEPS PROCESS FOR THE PIPELINE:** If I want to do 2 activities in a single pipeline, we can do it. Let’s example in 1 copy activity (click on created pipeline, in move and transform drag copy Data activity to canvas as 1st copy activity. in 1st copy activity data from csv (ADLS Gen2) will be converted to parquet in ADLS gen2 account. Click on validate all and publish all. Now drag another copy activity in the same canvas, 2nd copy activity is used to move data from parquet file in ADLS Gen2 container to Azure Synapse Analytics in one table. We need to link 2 copy activities by clicking on tick mark which is there at 1st copy activity and drag that to 2nd copy activity .and validate all and publish all and add trigger and trigger run. now pipeline will be run. Go to monitor 🡪 Pipeline runs 🡪 check pipeline succeeded or not.

In single pipeline we can do multiple activities like converting csv to parquet one activity and copying parquet to Azure synapse analytics as another activity.

**HOW TO CHANGE COLUMN DATA TYPE IN ADF:**

Note: If I want to change source data type in source columns, in pipeline activity, click on copy activity which is there in canvas, in mapping tab we can change data types here. Also, in destination table I dropped table and created table again by changing data type to DATETIME in Time column. now in copy activity, go to sink, click on open, data set will be opened, there go to schemas and click on import schema again in order to get new data type of DATETIME in time column in destination.

Now again go to mapping tab, click on CLEAR button to clear mapping and click import schemas again to map data from source to destination, why we are clearing and importing again is we changed data type in destination table for one column, so in order to get updated data type for time column, we cleared and imported schema again.

ADDITION OF NEW COLUMNS: When we go to Author 🡪 Pipelines 🡪 Click on created pipeline 🡪 in canvas copy activity will be appeared. If we click on copy activity, in source tab, there is Additional columns field. click on +, now in **value**, drop down appear with Custom, $$FILEPATH, $$COLUMN , and name is blank.

|  |  |
| --- | --- |
| Name | Value |
|  | Custom |
|  | $$FILEPATH |
|  | $$COLUMN |

If we want to add a column with static value, we will select Custom.

$$FILEPATH is used to specify the file path when it comes on to dataset.

$$COLUMN is to duplicate an existing source column.

If we select filepath from drop down in value. In name we will type FilePath as shown below.

|  |  |
| --- | --- |
| Name | Value |
| FilePath | $$FILEPATH |

We have added new column of FilePath for each row in source tab of Additional column field.

Now in sink db (sql pool), we dropped table and created table again by including FilePath extra column as shown below.

DROP TABLE [logdata\_parquet]

CREATE TABLE [logdata\_parquet]

(

[Correlationid] [varchar](200) NULL,

[Operationname] [varchar](200) NULL,

[Status] [varchar](100) NULL,

[Eventcategory] [varchar](100) NULL,

[Level] [varchar](100) NULL,

[Time] [datetime] NULL,

[Subscription] [varchar](200) NULL,

[Eventinitiatedby] [varchar](1000) NULL,

[Resourcetype] [varchar](1000) NULL,

[Resourcegroup] [varchar](1000) NULL,

[Resource] [varchar](2000) NULL,

[FilePath] [varchar](400) NULL

)

Now click on sink tab of copy activity, click on open, data set will be opened. Go to schemas and import schemas again in order to get FilePath column.

Now go to mapping tab, click on clear 🡪 click on import schemas again.

Click on publish all . click on Add trigger 🡪 Trigger now.

Now run select query , select \* from [logdata\_parquet]

In FilePath column, we will see name of the file(Log.parquet).

**COPY DATA USING COPY COMMAND:** Specifying additional columns created is not allowed when using copy command in copy method.

**COPY DATA USING POLYBASE COMMAND:** PolyBase copy can only be run directly with source wild card file name setting \*.\* or \*,Please enable stagging or fix wild card file name.

i.e; click on source tab of copy activity ,

|  |
| --- |
| Wild card folder path |

Wildcard Paths : Parquet/

|  |
| --- |
| \*.\* |

/

|  |
| --- |
|  |

Instead of \*. parquet, we need to keep \*.\* as above.

Or else

We can keep \*. parquet only above in order to get only parquet-based files from container. And go to settings of copy activity and enable stagging. For staging It needs to have a storage account in place. so this gonna be temporary place for entire transfer activity. so go to ADLS storage account, go to container, and create container of staging. Now in settings after enabling staging, select ADLS storage account and browse container directory or container folder (stagging folder) there and validate all, publish all and trigger now.

When we are looking at an effective way of copying data into Azure Synapse, always consider using PolyBase. So, PolyBase can actually make use of a parallel mechanism, wherein data can be shifted in parallel onto our, or the use of multiple compute nodes if we do have them configured for your dedicated SQL pool. So, it makes it much more efficient when it comes onto the data transfer.

**MAPPING DATA FLOWS:** When you wanna perform more complex transformations

on your data, you can actually make use of mapping data flows Here you can visualize your data transformations within Azure Data Factory. You can actually write the required transformation logic without the need of actually writing any code.

When it comes to copy data tool wizard, we just copied data from source and moved to destination. If you wanna make use of more complex transformations, you can make use of mapping data flows. Now, the reason that you can actually perform more complex transformations using mapping data flows, is because instead of using that Azure integration runtime that's normally used for your copy data activity, here the actual data flow is going to run on Apache Spark clusters, and this is charged separately. So, you won't be charged for the Azure integration runtime, but you will be charged for the runtime when it comes onto the Apache Spark clusters. So automatically in the background, Azure Data Factory will spin up the compute infrastructure for your clusters, run your mapping data flows on the clusters itself, and you are only charged for the, at least from a compute perspective, for the uptime of those clusters, and Azure Data Factory will get rid of those clusters once the data flow is complete.

Along with mapping data flows, you have a feature known as "Debug Mode". You can actually see the results of the data flow while designing the flow itself. Again, in the debug mode session, the data flow will run interactively on a Spark cluster and again, in the debug mode, you are charged on an hourly basis for the active cluster itself.

**MAPPING DATA FLOWS-FACT TABLE:**

SELECT dt.[ProductID],dt.[SalesOrderID],dt.[OrderQty],dt.[UnitPrice],hd.[OrderDate],hd.[CustomerID],hd.[TaxAmt]

FROM [SalesLT].[SalesOrderDetail] dt

LEFT JOIN [SalesLT].[SalesOrderHeader] hd

ON dt.[SalesOrderID]=hd.[SalesOrderID]

We need to deploy this data to Azure Synapse Analytics. we will do this via mapping data flows.

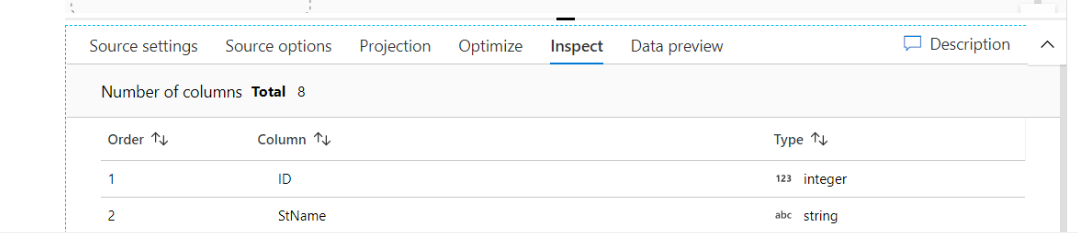
In mapping data flow, we are building visual representation of the data flow itself.

Author 🡪 Data Flows 🡪 click on context menu (…) 🡪 New data flow

Now Data flow wizard will be opened. In RHS properties, we will give data flow name.

In Canvas Add Source click on downward arrow, click on Add Source.

Below few tabs will be available as shown below.



**Source Settings:** Click on Source settings, we will enter below fields.

* **Output Stream Name:** we will enter stream name here (SalesOrderDetailStream), no underscores should keep. This Output Stream Name we mentioned in source settings will appear in source stream flow name in canvas.
* **Source Type:** We will select dataset
* **Dataset:** Click on +, we will connect on to Azure SQL Database and then continue. We will enter

1. **Name:** Dataset Name (SalesOrderDetail\_Table)
2. **Linked Service:** We will select linked service of Azure SQL database(appdb\_service)
3. **Table Name:** SalesLT.SalesOrderDetail
4. **Import Schema:** from connection/store

Click On Ok.

**Overview of Summary Settings Tab:** We have built something known as stream of data. This current stream of data is gonna be taking the information from our SalesOrderDetail table based on importing the schema from our source table. It understands that there are n columns of data and there is dataset also in place, click on open and can check schema.

**Source Options:**

* **Input:** Table/Query/Stored Procedure. Will choose table.
* **Isolation level:** Read Uncommitted.

Next step is as per our query we are doing left join with SalesOrderHeader table. So, we will add another source, down to SalesOrderDetailstream.

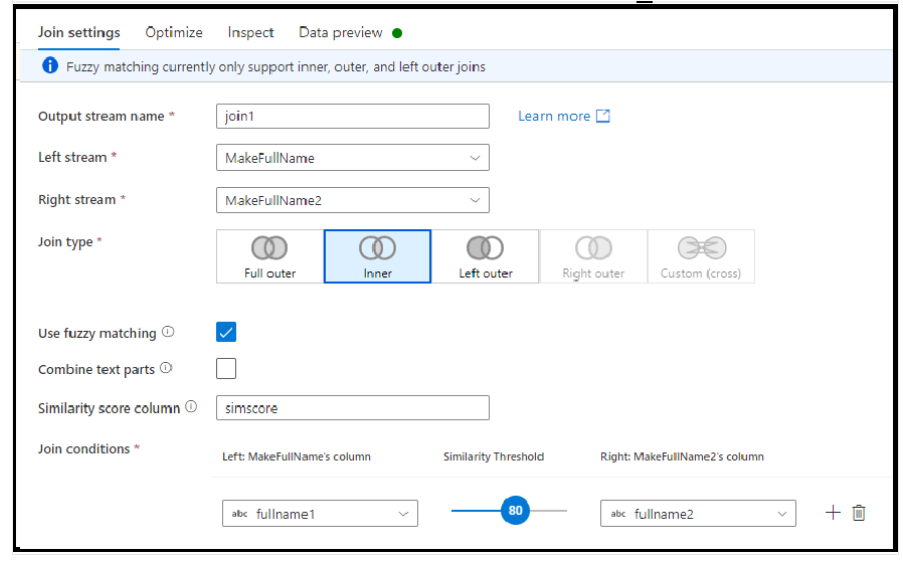
We will follow same steps again.

Enter Source Settings: enter here SalesOrderHeaderstream

Source Type: Dataset

Dataset: we will select Azure SQL Database, continue, now in set properties, we will enter Dataset name (SalesOrderHeader\_Table. Linked Service, Table name, import schema, click ok.

Now between 2 sources we need to do a join. So, click on + button for the first source, choose JOIN. we can now join multiple streams of data. The below wizard appeared at canvas bottom after selecting join.



**JOIN SETTINGS:**

Output stream name: SalesOrderDetailjoinSalesOrderHeader

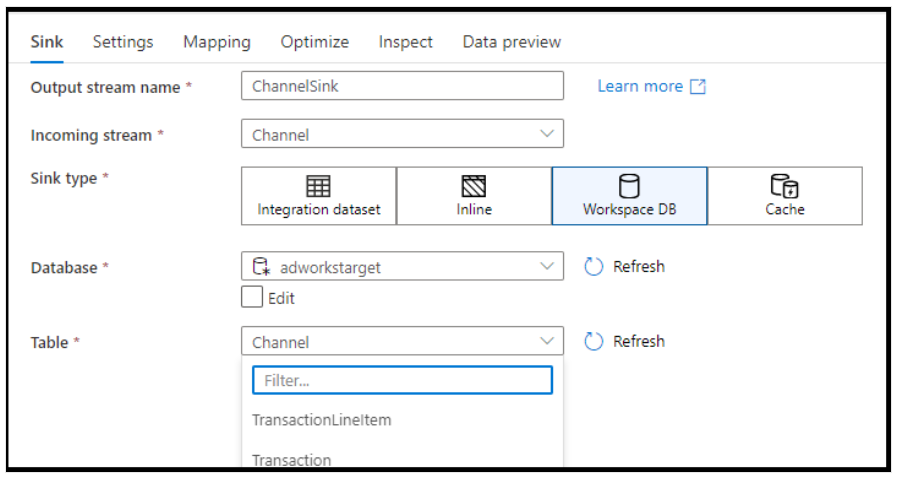
Left stream: we will select first table here (SalesOrderDetailstream)

Right stream: we will select second table here (SalesOrderHeaderstream)

Join Type: Left Join

Join Conditions: Based on what column ON condition is there, that we will specify here. It’s based on SalesOrderID column in both tables.so from drop down we will select SalesOrderID in both drop downs.

Now click on + after join stream and select sink from + button. The below wizard will be appeared at bottom after selecting sink from + button of join stream.



**Sink :**

**Output stream name:** Now we want to copy result of join data to my FactSalesStream table so output stream name we kept as FactSalesStream.

**Select incoming stream from drop down:** here it is SalesOrderDetailjoinSalesOrderHeader stream.

**Sink type:** Dataset

**Dataset:** click on +, in set properties we will enter Name: FactSales\_Table. select Linked Service. Choose table, import schemas click on ok.

**Mapping:** In Mapping tab, instead of doing automapping, uncheck automapping, columns and its data types will appear. I don’t need all columns whatever there in source , we need only few columns, if I click auto mapping, all columns from source is pointing to destination, so unchecked automapping, then only few columns whatever there in destination table that only will be mapped.

Now click on validate all and click on publish button.

**HOW TO RUN PIPELINE FOR MAPPING DATA FLOWS:**

Author 🡪 Pipeline 🡪 click on + 🡪 Create a pipeline 🡪 now enter pipeline name (04-Mappingdataflow-FactSales) at RHS properties.

From Move and Transform Activity: expand, drag Data Flow activity on to the canvas.

In bottom Wizard, 4 tabs will be seen.

General, Settings, Parameters, User Properties.

In **General** tab:

Name Field: enter here data flow activity name (dtflowFactSales)

**Settings:**

**Data Flow:** select here mapping data flow that we need to run.

**Logging level:** selected verbose here (among verbose/basic/none)

**Staging:** We need to select staging here because we are moving data onto Azure Synapse.

We need to select **Staging linked service:** we will select ADLS Gen2 storage account service here.

**Staging Storage Folder:** we will select staging container here.

Click on validate all and publish all. And trigger pipeline.

**WHEN TO USE COPY DATA ACTIVITY AND DATA FLOW ACTIVITY:**

When we are only copying data from source to destination, we can go with copy activity.

If we are going to do complex transformations, then we can go with Data Flow Activity.

**Note: In Single Mapping Data Flow activity I mean (Author 🡪 Dataflows 🡪 click on + 🡪 Create Data Flow 🡪 there we can work on different activities which means creation of 2 tables data can be done. In single data flow one table can be constructed(dim\_customer) using 1 query, and in same data flow another table(dim\_product) can be constructed using another query. Once 2 activities were done in same data flow, validate and publish all. Now go to pipeline and drag data flow and run trigger the pipeline.**

**Now in both tables (dim\_customer and dim\_product) data will be loaded in azure synapse analytics.**

**CACHE SINK AND LOOKUP:**

Cache sink is used to write data into Spark cache and not the data store. Cache lookup can then be used to reference the data in the cache sink. Let’s say you want to continue the ProductSK key from where it left off. So we can get the maximum value first in the ProductSK column and then store it in the cache. And then reference that value using cache lookup feature via expression in expression builder by selecting derived column after selecting SK.

**DATA FLOW DEBUG FEATURE:** used to check whether we are getting desired results or not by enabling data flow debug option. By clicking on data preview and refresh we can check the data is coming correct or not. if there is issue in data and If we run pipeline multiple times it will increases cost and waste of time by running pipeline multiple times.so we will enable data flow debug in data flow .it will tell us whether we are receiving data correct or not.

From source to destination, we will do multiple transformations, so in order to see we are getting desired result of data in each transformation, it is good to enable data flow debug option.

HANDLING DUPLICATE ROWS: Duplicate rows can be handled by exists transformation.

FILTER TRANSFORMATION usage: In my source table, one of my columns is having null values, would transfer data to the sink only not null values for that column. We will select filter transformation after source, and we will add this expression in expression builder.

!(isNull({Resourcegroup}))

**PROCESSING JSON ARRAYS:**

This is an array of documents eg as shown in below json file

[

{"customerid":1,"customername":"UserA","registered":true,"courses":["AZ-900","AZ-500","AZ-303"]},

{"customerid":2,"customername":"UserB","registered":true,"courses":["AZ-104","AZ-500","DP-200"]}

]

If I want to load this data in to sink in ASA in dedicated sql pool we need to use flatten feature to unroll the array of day here it is courses.

**PROCESSING JSON OBJECTS: OBJECTS WITH IN OBJECTS IN OUR JSON DOCUMENTS.**

We will also do this by flatten feature we will unroll Courses, along with that we will add mapping to get mobile and city details and in sink mapping also reset the mapping and select mobile and city from drop down.

The below e.g., is json object within another json object. Here we are having 2 objects in place, one is details object and other is overall object.

[

{

"customerid":1,

"customername":"UserA",

"registered":true,

"courses":["AZ-900","AZ-500","AZ-303"],

"details" :

{

"mobile":"111-1112",

"city":"CityA"

}

},

{

"customerid":2,

"customername":"UserB",

"registered":true,

"courses":["AZ-104","AZ-500","DP-200"],

"details" :

{

"mobile":"333-1112",

"city":"CityB"

}

}]

In Sink dedicated SQL pool, we will create table like below.

CREATE TABLE [Customercourse]

(

[CustomerID] int,

[customer Name] varchar(200),

[Registered] BIT,

[Courses] varchar(200),

[Mobile] varchar(200),

[City] varchar(200)

)

**SELF HOSTED INTEGRATION RUN TIME:**  In the copy activity, a simple copy activity, we were using the Azure integration runtime. And that runtime would provide the compute infrastructure let’s say, transferring data from Azure Data Lake onto tables in Azure Synapse🡪 Azure Integration run time. But we might be having data that could be hosted on a machine. When, let's say, that you have data that's being hosted on a virtual machine, this machine could be in your on-premises infrastructure. It could be on another cloud platform. Maybe you might want to transfer data files or even data in a SQL database hosted on that machine while the use of Azure Data Factory on, let's say, tables in the Azure Synapse service.

I have a data that could be hosted on a machine for this, you first need to register this machine with Azure Data Factory by installing something known as the self-hosted integration runtime. if you have data in your own server, instead of hosting it on an Azure based service, you're to make use of the self-hosted integration runtime.

**SELF HOSTED INTEGRATION RUN TIME-BUILDING an MACHINE:**

If we have files hosted on a machine. To pick up that files as they are from machine, we need to use self-hosted integration run time.

First, we need to create a virtual machine. Home 🡪 Virtual machine 🡪 click on + 🡪 Create Azure Virtual machine. This service allows us to host machines on the cloud.

After virtual machine is created, click on connect, choose RDP, download RDP file.

Open downloaded RDP file, enter credentials of username and password (we gave this during creation of VM), click on yes and

Now we will be connecting on to windows server. We will install nginx web server on this machine. In terms of taking data from our machine and processing it in Azure Data Factory, we want to generate some log base bars from web server and hence I want to install nginx web server.

In Server Manager 🡪 go to Local Server 🡪 GO to IE enhanced security configuration 🡪 click off there.

Now install nginx server.

Now we need to copy files from Virtual machine from server to Azure data factory, that we can do only by installing self-hosted integration run time.

Go to Manage🡪 Integration Runtimes 🡪 click on + 🡪 select Azure self-hosted

Linked Service is connection details to our source or sync.

Dataset is pointer to our data .

**Difference between linked Service and Dataset:**

For example, i want to run ADB notebook in ADF, here linked service is ADB--> ADB name and particular notebook is dataset.

Author --> pipeline --> create new pipeline--> Activities tab will come.

Integration run time is compute service, it is bridge between activity and linked service objects.

There are 3 different types of integration run times in ADF, 1 is Azure, 2 Self Hosted 3rd Azure SSIS .

1) **Auto Resolve Integration Run Time** : Auto Resolve Integration Run Times comes default by azure. Any service when we are trying to integrate with azure that can be handled by azure itself. If my source account is azure, we can easily go ahead and use AutoResolveIntegrationRuntime.If we want to do cloud to cloud migration within azure we can go for azure/auto resolve integration run time.

**Azure SSIS**: If we want to lift and shift any SSIS package, we will establish connection through integration run time.

**Self-Hosted Integration Run Time**: When we are going to define integration run time , then that’s called as Self-Hosted Integration Run Time. When to use self-hosted? Whenever we connected to on premise to azure, we go for self-hosted .

**different types of triggers in ADF?**

If we don’t want to run pipeline manually, we will use triggers.

**Event Trigger**: Whenever there is some kind of event happening , it will automatically trigger. let’s say file comes inside blob storage or deleting file in blob storage ,we can trigger a pipeline. Coming of file into blob storage is an event . Based on event pipeline will trigger.

**Tumbling Window Tigger**: We can execute pipeline even for historical data . we can specify window period ; we can execute pipelines in that window period .

**Schedule Trigger**: Mostly on-premises technology use schedule trigger. If we want to execute pipeline at 1 am every day, then we can go for schedule trigger.

**What are different ways to execute pipelines in ADF?**

Under debug mode

Can execute pipeline manually using Trigger now

Using Triggers(schedule, event, and Tumbling window trigger)

**What are ARM templates in ADF and What are they used for ?**

ARM Template is JSON file that defines infrastructure and configuration details of our pipeline activities , linked services and datasets etc.

**Use of ARM Template:** If we want to migrate ADF code from dev to QA/dev to prod , we will not create pipelines manually in higher environments (PROD). All this code gets converted in the form of ARM templates . so, we can use ARM template files to deploy into higher environments . we just create/automate CI/CD pipelines through which we will deploy ARM templates into UAT/PROD .

**How do you deploy data factory code to higher environments?**

1. Set up code repository and Create feature branch. Feature branch means we will create our own code , it will not affect development /UAT/PROD
2. Create Pull Request to merge code from feature branch to Dev branch
3. Once code is present in Dev branch , come back to data factory and click on publish option to publish the code from dev to generate ARM Templates. Once we click on publish all, ARM Templates will be generated in the repository .
4. This can trigger an automated CI-CD Devops pipeline to pass code to higher environments.

**What are Difficulties that you have faced while copying data from on premises to cloud? How did you resolve it?**

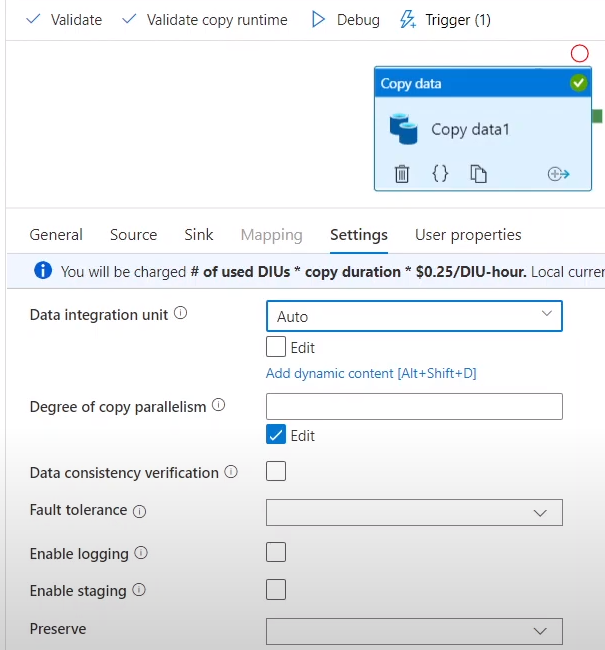
Let’s say I want to copy data from SQL Server to ADLS Gen2 , source is SQL Server and Sink is Data Lake Gen2 , copying data from source to sink is always very slow. Copying from on premise to cloud is always slow. How to solve this issue?

We need to enable compression and enable stagging in settings tab of copy activity canvas. When data is copying from source to sink, file will be decompressed, and data will be loaded to storage layer ADLS Gen2 so that speed/throughput will be more if we compress the source file.

DIU-Data Integration Unit means number of CPU’s

Enable compression , enable stagging and Degree of copy parallelism are necessary to improve the issues.

Enable stagging use is to store temporary files of compressed files and decompress it and store the data in ADLS Gen2.



**What are different activities used in Azure Data factory?**

**Copy Data Activity:** used to copy data from source to destination.

**ForEach Activity:** Something activity I want to execute inside for loop, then we can go with this activity.

**Metadata Activity:** I have source dataset and I want to derive some information about source dataset, then we will use this activity.

**Set Variable Activity:** here we actually define a variable and value to that variable.

**Lookup Activity:**  we can get output through a query /singleton output from a table as well

**Wait Activity:** Allows pipeline to wait for some time. The time depends on what we specify inside wait activity.

**Validation Activity:** performs data checks here. It just validates presence of our files. True or not

**Web and Web hook Activity:**  used to trigger http requests

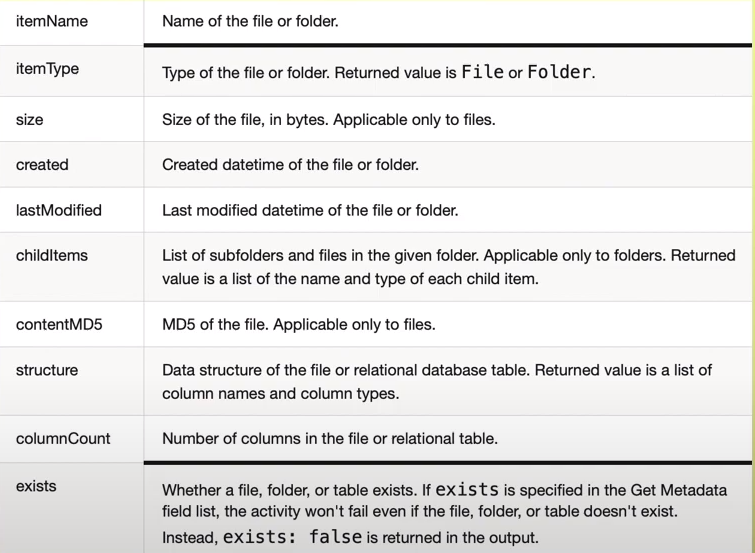
**Until Activity**

**If Condition**

**Filter Activities.**

**Can you execute For-Each inside another For-Each activity?** No

**What are different metadata options that get in Metadata activity?**

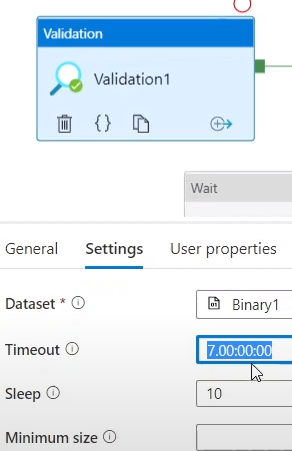


**If we want to use output by executing a query, which activity shall you use?**

Look Up Activity

**How do you verify presence of file in a storage?**

Validation Activity

Time out specifies number of days to check whether file is present or not, by default it is 7 days.

**Have you used Execute Notebook Activity ? How do you pass parameters to our notebooks to execute notebook activity?**

Using baseParameters property we can pass parameters to notebook ,in case parameters are not specified in activity, default values from notebook will be executed.

How do you handle incremental data in Data Factory?

Let’s say my source is Azure SQL Database and table is source and let’s say we has a watermark value or Date column and these rows are inserted today and tomorrow data comes in that particular table , we have another set of rows with tomorrows date , incremental data is the date column is the watermark value, in this case we take highest watermark value from that table and load it and store that data value in another table which we left and refer again. Again, tomorrow if number of rows are coming for that table, now we will have different value for that date, so will have different value for watermark. So, we can compare watermark value which comes this time with last time water mark value and we can only load incremental data.

