Data engineering on Microsoft Azure: Data Factory

This course is designed to prepare us for Microsoft Data Engineering on Azure DP203 exam

Azure Data Factory – we need some mechanism to transform data into a usable format. ADF is a data integration service that is used to create automated data pipelines that can be used to copy & transform data.

ADF is Azure’s ETL & data integration services that allows users to create data-driven workflows for orchestrating data movement & transforming data at scale

Workflow: SSIS (SQL Server Integration Service)->ETL->Destination data->Power BI

Eg: A game company has a game which:

Logs – azure data lake storage: Bigdata

Customer info & market campaigns – On premises storage

You want to analyze the logs with customer data & generate a meaningful report to find which modes are frequently used & which ads are frequently played.

We use ADF to extract, transform & load data to say Azure SQL which is then used in Power BI.

Why ADF?

Easy for big data

Cloud based service for complex hybrid ETL & data integration

Some ADF Concepts

Pipeline – A logical group of activities that performs some work

Activity – an individual processing step in the pipeline

Linked Services – Define the connection information needed for ADF to connect to external resources. They are like connection strings.

Datasets – The data used in your activities. They represent the data structures within data stores that point to/reference the data you want to use.

ADF Triggers – Allows us to define when data pipelines run in ADF

Different tools to work with ADF:

Azure Portal

Azure PowerShell

.NET

Python

REST

Resource Manager Template (Azure PowerShell Az Module) – They are json files with pre-defined values that are used to create ADF

Data pipelines & pipeline runs

A trigger fires a pipeline run.

Pipeline run – a single execution of a pipeline. A single batch of values are loaded from the source, transformed as needed and then stored in a target/sink data source

Each pipeline run has a unique pipeline ID

A pipeline can be triggered in 2 ways:

On-demand execution – With a button click in the UI we can manually force the immediate creation & execution of a new pipeline. This is done either for one-off pipelines or for debugging purposes.

Triggered execution – a new pipeline run is created & executed each time a trigger is fired.

ADF Trigger types

Schedule – Set a calendar time trigger

Tumbling Window – For precise time frame triggers

Storage event – Triggering based on BLOB storage events

Custom event – Trigger based on any custom event or services or applications might generate

Schedule Trigger

Recurrence schedule. Eg: Trigger a run every 4 hrs

Fire & forget trigger – It doesn’t track pipeline run success or failure. It simply starts the run & then moves on.

Will only trigger pipeline run for times after it started running. It doesn’t run pipelines for past events. Eg:- if we create a schedule trigger for a pipeline to run every day, it’s only going to run from the day you activate the trigger & days moving forward. It will never run for the days preceding the activation time.

If the trigger is disabled for a week it won’t backfill that week even after activation.

They are unreliable. They are not precise & can’t be relied on for operations that require high accuracy in time.

Eg:- A scheduled trigger at 2AM will most probably run around 2. It could even be 2:10 when it runs.

It has a many-to-many relationship with the pipelines it can run. One schedule trigger can run multiple pipelines. Multiple pipelines can run 1 trigger.

Schedule triggers have no retry logic as they are fire & forget triggers.

Tumbling window trigger

Tumbling window is on a recurrence schedule. Eg: You can configure it to run every day at a given time

Schedule for a tumbling window trigger defines time windows. A pipeline run can accept the start & end time of time window as inputs

Time windows do not overlap

Tumbling windows maintains state. They can keep track of which time windows have been processed and their success or failure status.

It allows backfill of historical time windows.

Eg: If a tumbling window was disabled for a week & re-enabled, it can backfill the week that was missed by triggering for all of the time frames for that week.

It is a 100% reliable & accurate in how it manages time limits

It can automatically retry time windows in case of failure.

They are ideal for pipelines that are very time-sensitive & need reliability in runs.

Storage Event Triggers

Monitors a storage account container & the idea is that pipeline runs are triggered based on changes in the container.

It can fire whenever a blob is created/destroyed.

It can filter by blob path. So instead of monitoring all blobs in a container, it can monitor only a subset (i.e., given path) if needed.

It’s ideal in a situation where data that comes into the system is stored in BLOB storage. You don’t want to pull for changes in the BLOB storage. Instead, the trigger is fired automatically when BLOB is created or removed.

Custom event trigger

Triggers based on events

Not limited to BLOB events in a storage account.

It monitors/listens to an event grid which can be configured to handle any custom event that your services or applications may generate

Filters events by subject which allows us to make triggers listen to only a subset of events.

This trigger is great if they pipeline needs to run based on more sophisticated events.

Eg: A trigger can run a new pipeline run based on the status of another pipeline run.

Create a trigger in ADF:

Trigger>Add/Create> Once you make the trigger go back & click on the publish button to publish the trigger & activate it

Handling Schema drift in Data Flow

Schema drift – the ability of your data flow to automatically add changes to data schema on the fly

Eg: Adding a col to the dataset on the fly.

ADF allows schema drift by default but you can configure it to strictly adhere to the initial schema.

Optimizing pipelines in ADF pipelines

To specify which step runs first set the Work Order. Lower the work order higher the priority.

If you set logging level to Verbose it’s going to affect performance. Verbose is good for debugging. But in a production environment set the logging level to Basic/None.

Under sink properties if you check the run in parallel option, any grouped sinks would run in parallel.

To view the performance of an activity go to Monitor window > pipeline runs > Select pipeline > Select activity run’s view details button (glasses icon). This will give us all performance details of the activity.

Partition chart shows you how many rows were calculated in each partition.

Sink processing time – how long it took the sink to run.

Types of activities:

Data movement activities – moving data from one location to another

Data transformation activities – change nature/schema of data

Control flow activities – defines the sequence of execution of pipeline

Upsert with Azure SQL

Upsert – Upsert operation allows the insertion & updating of records in a SQL table at the same time. It allows both update operation as well as insert operation.

Troubleshooting in Azure Stream Analytics

When a stream analytics job suddenly stops resource logs give us a place to start.

Activity logs – tells us what’s happening with a stream analytics instance

To customize what kind of logs you need go to Diagnostic setting

Log analytics workspace – logs are collected & stored