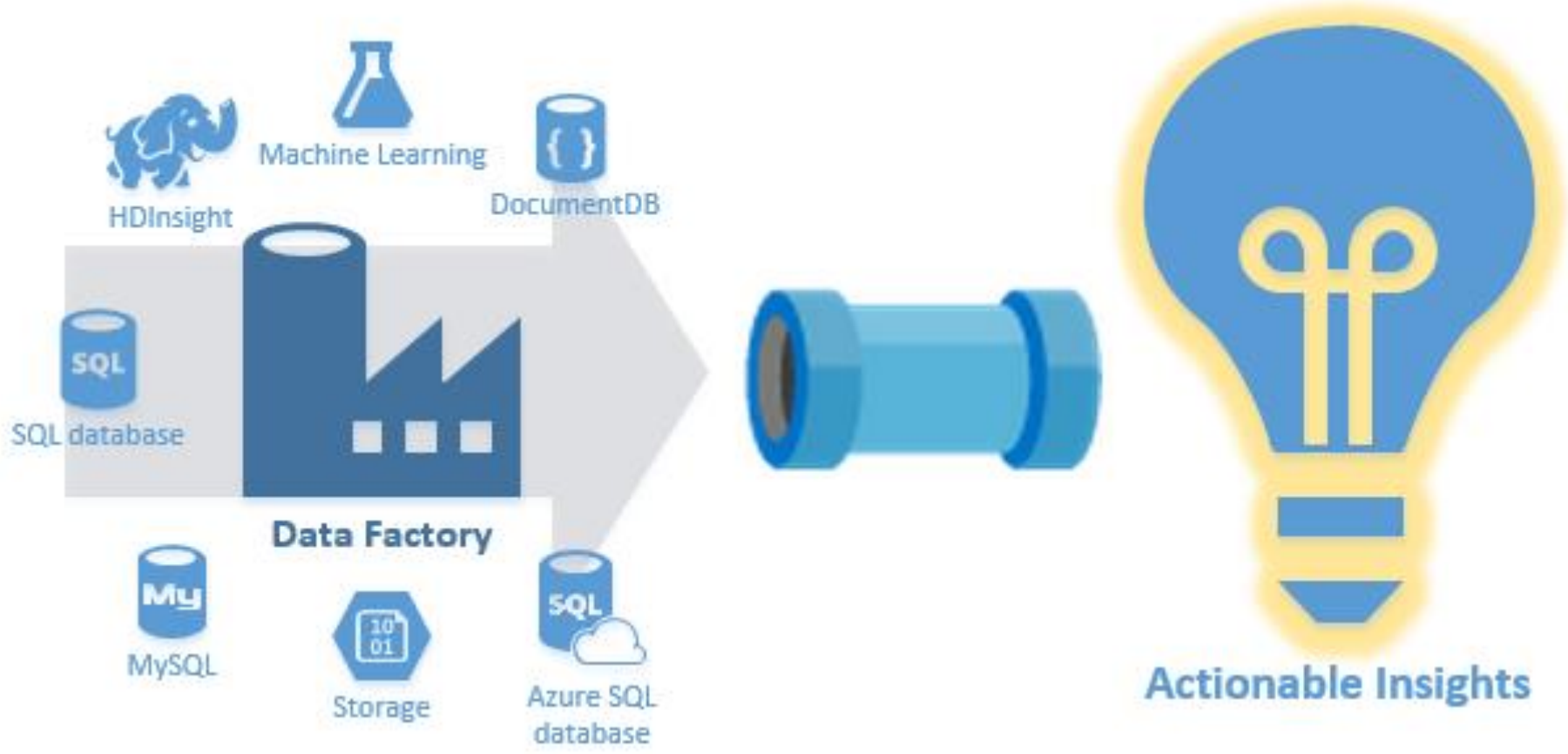
Azure Data Factory



Course Contents

- Introduction of Azure
- Introduction of Azure Data Factory
- Data Factory components
- Differences between v1 and v2
- Triggers
- Control Flow
- SSIS in ADFv2
- Demo

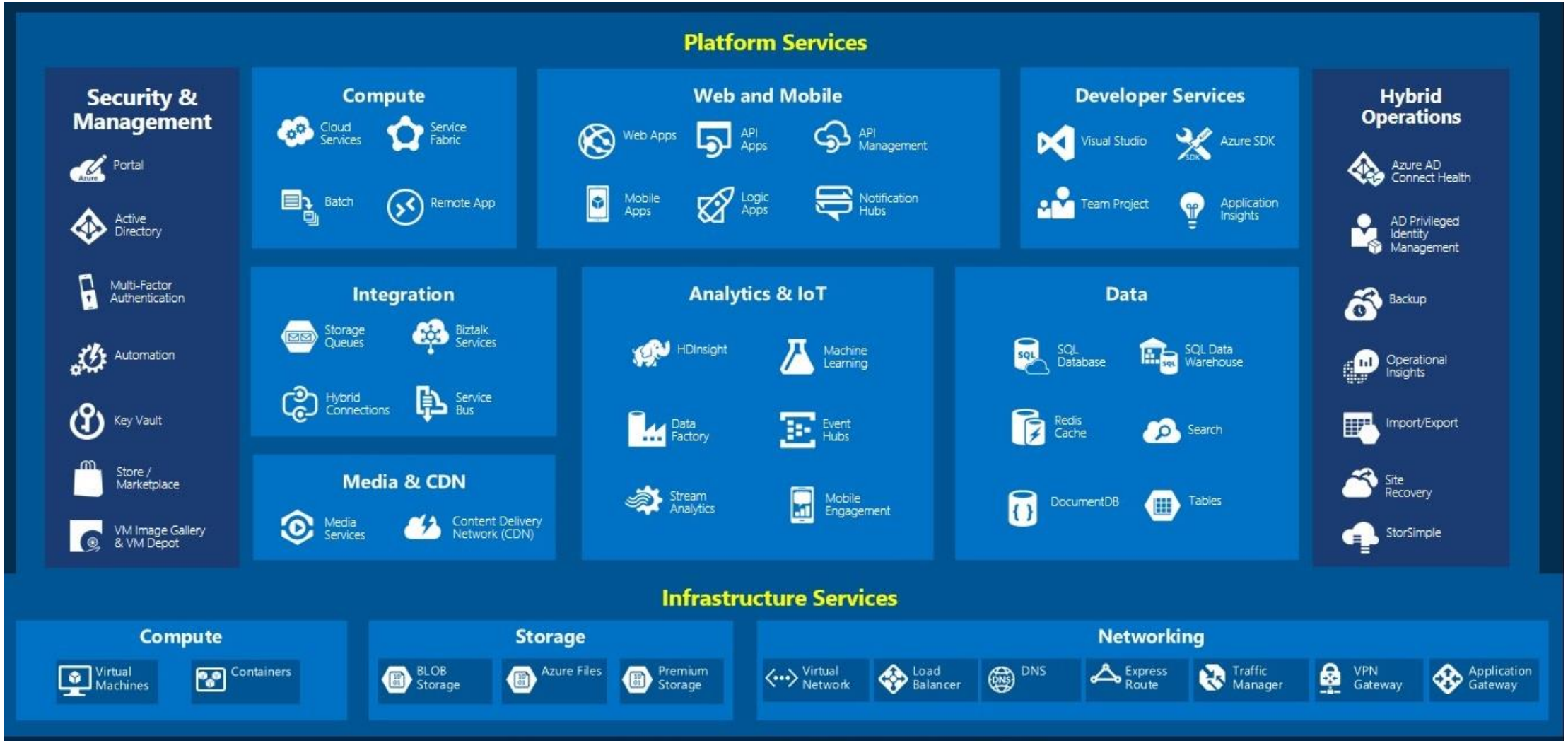
Introduction of Azure

- Azure is Microsoft's cloud computing platform, provides cloud services that gives you the freedom to build, manage, and deploy applications on a massive global network using your favorite tools and frameworks.

  A quick explanation on how Azure works

- Cloud computing is the delivery of computing services over the Internet using a **pay-as-you-go** pricing model. In other words it's a way to rent compute power and storage from someone's data center.
- Microsoft categorizes Azure cloud services into below product types:
  - Compute
  - Storage
  - Networking
  - Web
  - Databases
  - Analytics and IOT
  - Artificial Intelligence
  - DevOps

Introduction of Azure



Introduction of Azure Data Factory

- Azure Data Factory is a cloud-based data integration service to compose data storage, movement, and processing services into automated data pipelines.
- It compose of data processing, storage, and movement services to create and manage analytics pipelines, also provides orchestration, data movement and monitoring services.
- In the world of big data, raw, unorganized data is often stored in relational, non-relational, and other storage systems, big data requires service that can orchestrate and operationalize processes to refine these enormous stores of raw data into actionable business insights.
- Azure Data Factory is a managed cloud service that's built for these complex hybrid extract-transform-load (ETL), extract-load-transform (ELT), and data integration projects.
- Azure Data Factory is a data ingestion and transformation service that allows you to load raw data from over 70 different on-premises or cloud sources. The ingested data can be cleaned, transformed, restructured, and loaded back into a data warehouse.
- Currently, there are two versions of the service: version 1 (V1) and version 2 (V2).

Introduction of Azure Data Factory

- The pipelines (data-driven workflows) in Azure Data Factory typically perform the following four steps:

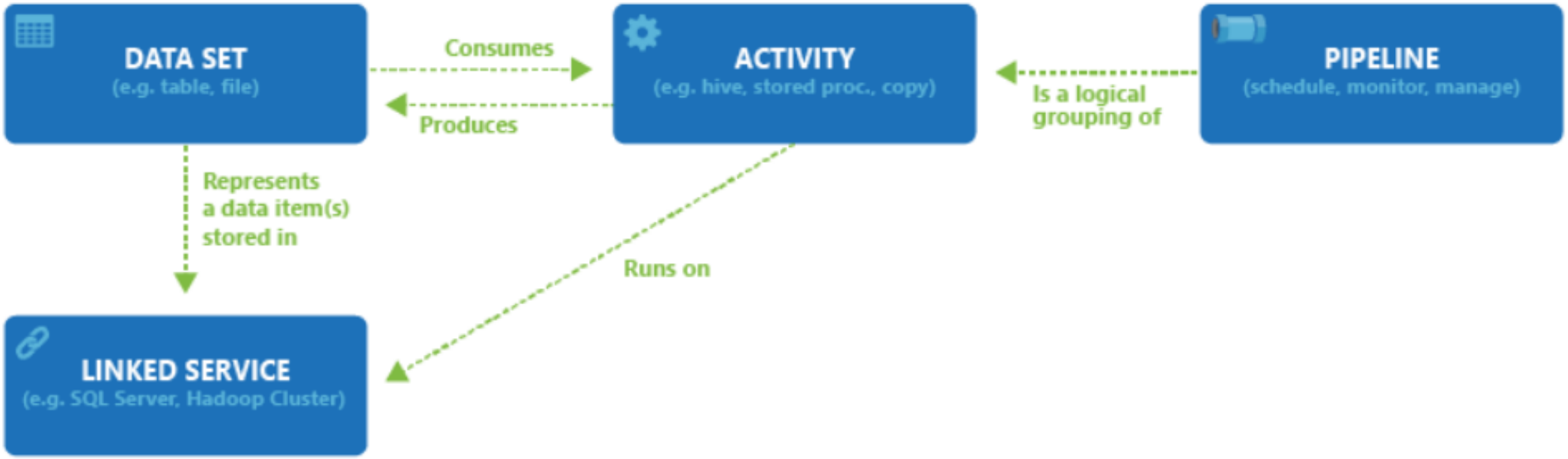
- **Connect and collect:** The first step in building an information production system is to connect to all the required sources of data and processing, such as software-as-a-service (SaaS) services, databases, file shares, and FTP web services. The next step is to move the data as needed to a centralized location for subsequent processing.
- **Transform and enrich:** After data is present in a centralized data store in the cloud, process or transform the collected data by using compute services such as HDInsight Hadoop, Spark, Data Lake Analytics, and Machine Learning.
- **Publish:** After the raw data has been refined into a business-ready consumable form, load the data into Azure Data Warehouse, Azure SQL Database, Azure Cosmos DB, or whichever analytics engine your business users can point to from their business intelligence tools.
- **Monitor:** After you have successfully built and deployed your data integration pipeline, providing business value from refined data, monitor the scheduled activities and pipelines for success and failure rates.

Data Factory Components

- Azure Data Factory is composed of four key components. These components work together to provide the platform on which you can compose data-driven workflows with steps to move and transform data.
- **Pipeline:** A data factory might have one or more pipelines. A pipeline is a logical grouping of activities that performs a unit of work. For example, a pipeline can contain a group of activities that ingests data from an Azure blob, and then runs a Hive query on an HDInsight cluster to partition the data.
- **Activity:** Activities represent a processing step in a pipeline. For example, you might use a copy activity to copy data from one data store to another data store. Data Factory supports three types of activities: data movement activities, data transformation activities, and control activities.
- **Datasets:** Datasets represent data structures within the data stores, which simply point to or reference the data you want to use in your activities as inputs or outputs.
- **Linked services:** Linked services are much like connection strings, which define the connection information that's needed for Data Factory to connect to external resources. For example, an Azure Storage-linked service specifies a connection string to connect to the Azure Storage account.
- Linked services are used for two purposes in Data Factory :
  - To represent a data store that includes, but isn't limited to, an on-premises SQL Server database, Oracle database, file share, or Azure blob storage account.
  - To represent a compute resource that can host the execution of an activity. For example, the HDInsight Hive activity runs on an HDInsight Hadoop cluster.

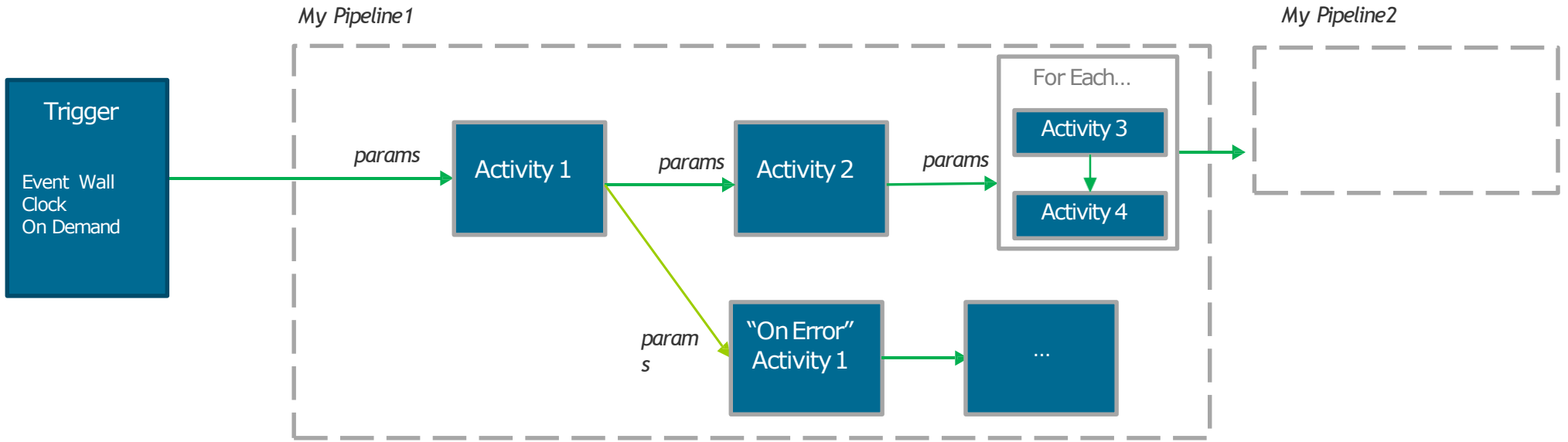
Data Factory Components

- Overview of Data Factory flow



Data Factory Components

- Overview of Data Factory flow



Data Factory Components

- Other components of Data Factory.

- **Triggers**: Triggers represent the unit of processing that determines when a pipeline execution needs to be kicked off. There are different types of triggers for different types of events.
- **Pipeline runs:** A pipeline run is an instance of the pipeline execution. Pipeline runs are typically instantiated by passing the arguments to the parameters that are defined in pipelines. The arguments can be passed manually or within the trigger definition.
- **Parameters:** Parameters are key-value pairs of read-only configuration. Parameters are defined in the pipeline. Activities within the pipeline consume the parameter values.
- **Control flow:** Control flow is an orchestration of pipeline activities that includes chaining activities in a sequence, branching, defining parameters at the pipeline level, and passing arguments while invoking the pipeline on-demand or from a trigger. It also includes custom-state passing and looping containers, that is, For-each iterators.

Differences between v1 and v2

Feature	Version 1	Version 2
Datasets	A named view of data that references the data, can be utilized in activities as inputs and outputs. Datasets identify data within different data stores, such as tables, files, folders, and documents **Availability** defines the processing window slicing model for the dataset (for example, hourly, daily, and so on).	Datasets are the same in the current version. However, you do not need to define **availability** schedules for datasets.
Linked services	Linked services are much like connection strings, which define the connection information that's necessary for Data Factory to connect to external resources.	Linked services are the same as in Data Factory V1, but with a new **connectVia** property to utilize the Integration Runtime compute environment of the current version of Data Factory.

Differences between v1 and v2

Feature	Version 1	Version 2
Pipelines	A data factory can have one or more pipelines. A pipeline is a logical grouping of activities that together perform a task.	Pipelines are groups of activities that are performed on data. However, the scheduling of activities in the pipeline has been separated into new trigger resources. The Data Factory V1 concepts of startTime, endTime, and isPaused are no longer present in the current version of Data Factory.
Activities	Activities define actions to perform on your data within a pipeline. Data movement (copy activity) and data transformation activities (such as Hive, Pig, and MapReduce) are supported.	In this version of Data Factory, activities still are defined actions within a pipeline The current version of Data Factory introduces new control flow activities.

Differences between v1 and v2

Feature	Version 1	Version 2
Hybrid data movement and activity dispatch	Now called Integration Runtime, Data Management Gateway supported moving data between on-premises and cloud.	Data Management Gateway is now called Self-Hosted Integration Runtime. It provides the same capability as it did in V1. The Azure-SSIS Integration Runtime in the current version of Data Factory also supports deploying and running SQL Server Integration Services (SSIS) packages in the cloud.
Parameters	NA	Parameters are key-value pairs of read-only configuration settings that are defined in pipelines.

Differences between v1 and v2

Feature	Version 1	Version 2
Expressions	Data Factory V1 allows to use functions and system variables in data selection queries and activity/dataset properties.	In this version of Data Factory, one can use expressions anywhere in a JSON string value.
Pipeline runs	NA	A single instance of a pipeline execution. Each pipeline run has a unique pipeline run ID. The pipeline run ID is a GUID that uniquely defines that particular pipeline run.
Activity runs	NA	An instance of an activity execution within a pipeline.
Trigger runs	NA	An instance of a trigger execution. For more information.
Scheduling	Scheduling is based on pipeline start/end times and dataset availability.	Scheduler trigger or execution via external scheduler.

Differences between v1 and v2

Feature	Version 1	Version 2
Chaining activities	In V1, must configure the output of an activity as an input of another activity to chain them.	In this version of Data Factory, in the current version, one can chain activities in a sequence within a pipeline, by using the **dependsOn** property in an activity definition to chain it with an upstream activity.
Branching activities	NA	Can branch activities within a pipeline. The **If-condition** activity provides the same functionality that an if statement provides in programming languages.
Custom state passing	NA	Activity outputs including state can be consumed by a subsequent activity in the pipeline. By using this feature, we can build workflows where values can pass through activities.

Differences between v1 and v2

Feature	Version 1	Version 2
Looping containers	NA	The ForEach activity defines a repeating control flow in your pipeline. This activity iterates over a collection and runs specified activities in a loop.
Trigger-based flows	NA	Pipelines can be triggered by on-demand (event-based, i.e. blob post) or wall-clock time.
Invoking a pipeline from another pipeline	NA	The Execute Pipeline activity allows a Data Factory pipeline to invoke another pipeline.
Delta flows	NA	A key use case in ETL patterns is “delta loads”. New capabilities in this current version, such as lookup activity, flexible scheduling, and control flow, enable this use case.

Differences between v1 and v2

Feature	Version 1	Version 2
Other control flow activities	NA	ForEach activity, Web activity, Lookup activity, Get metadata activity, Wait activity.
Deploy SSIS packages to Azure	NA	We can Azure-SSIS if you want to move our SSIS workloads to the cloud, create a data factory by using the current version, and provision an Azure-SSIS Integration Runtime.
Custom activities	In V1, we implement (custom) DotNet activity code by creating a .NET class library project with a class that implements the Execute method of the IDotNetActivity interface. Therefore, you need to write your custom code in .NET Framework 4.5.2 and run it on Windows-based Azure Batch Pool nodes.	In a custom activity in this version, you don't have to implement a .NET interface. You can directly run commands, scripts, and your own custom code compiled as an executable.

Triggers

How do pipelines get started

1. on-demand
2. Wall-clock Schedule
3. Tumbling Window (aka time-slices in v1)
4. *Event*

Triggers

How do pipelines get started

1.Power Shell:
Invoke-AzureRmDataFactoryV2Pipeline +Parameters

2.Rest API: https://management.azure.com/subscriptions/mySubId/resourceGroups/myResourceGroup/providers/Microsoft.DataFactory/factories/{yourDataFactory}/pipelines/{yourPipeline}/createRun?api-version=2017-03-01-preview

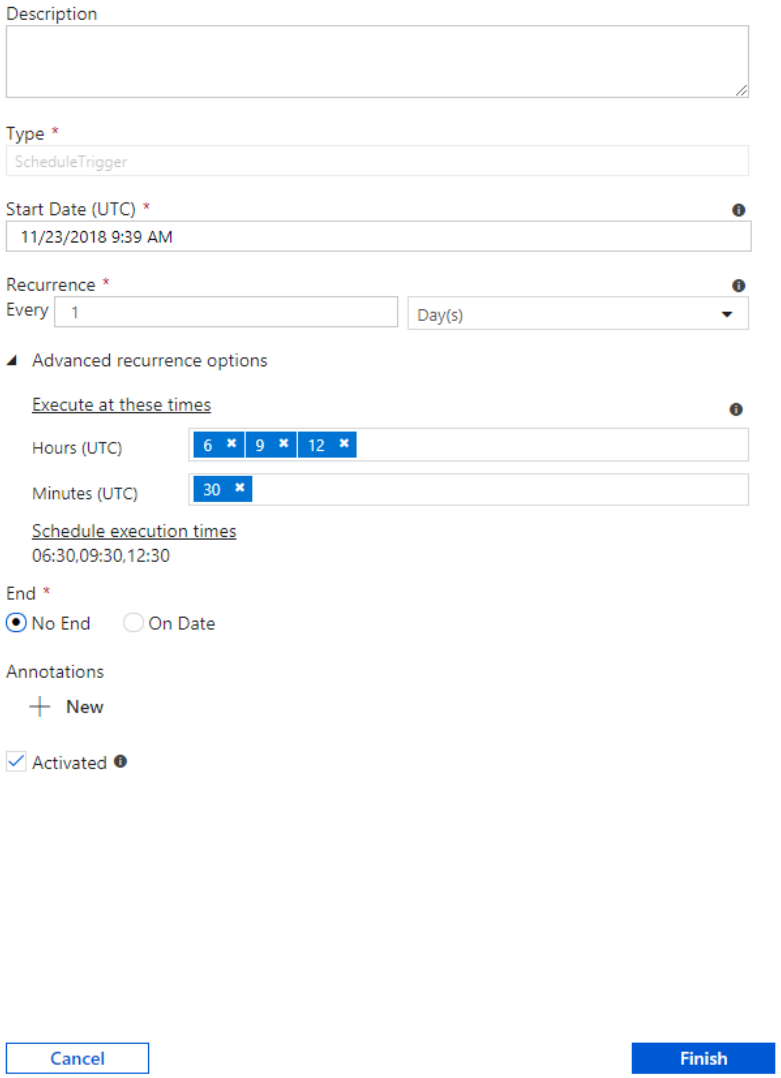
3..NET:
client.Pipelines.CreateRunWithHttpMessagesAsync(+ parameters)

4. Azure Portal
(Data factory -> <Author & Monitor> -> Pipeline runs)

Triggers

Run pipeline by schedule

```
{
  "properties": {
    "type": "ScheduleTrigger",
    "typeProperties": {
      "recurrence": {
        "frequency": <<Minute, Hour, Day, Week, Year>>,
        "interval": <<int>>,            // optional, how often to fire (default to 1)
        "startTime": <<datetime>>,
        "endTime": <<datetime>>,
        "timeZone": "UTC"
        "schedule": {                   // optional (advanced scheduling specifics)
          "hours": [<<0-24>>],
          "weekDays": ": [<<Monday-Sunday>>],
          "minutes": [<<0-60>>],
          "monthDays": [<<1-31>>],
          "monthlyOccurences": [
              {
                  "day": <<Monday-Sunday>>,
                  "occurrence": <<1-5>>
              }
```

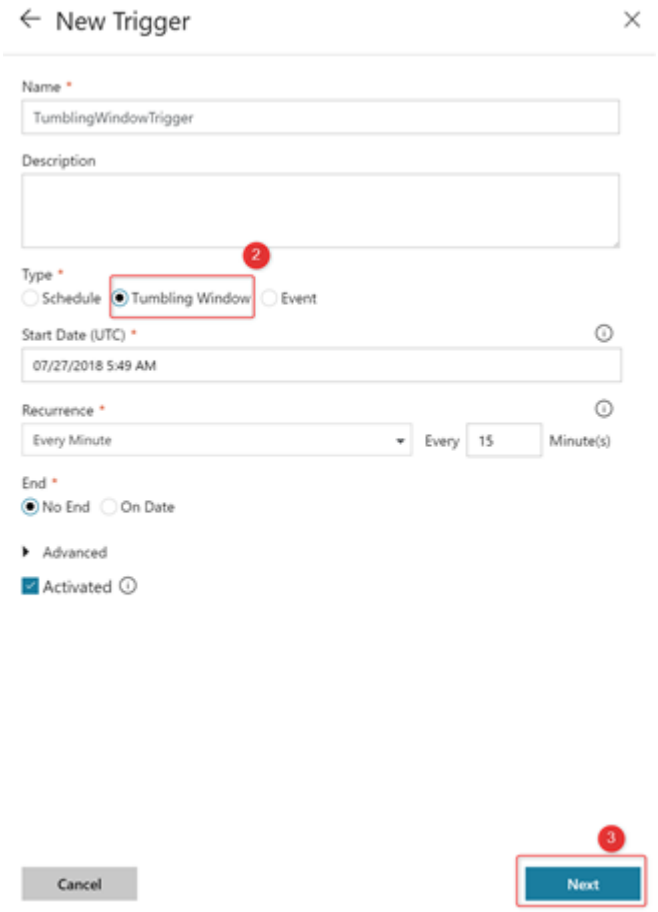


Triggers

Tumbling Window

Tumbling window triggers are a type of trigger that fires at a periodic time interval from a specified start time, while retaining state. Tumbling windows are a series of fixed-sized, non-overlapping, and contiguous time intervals.

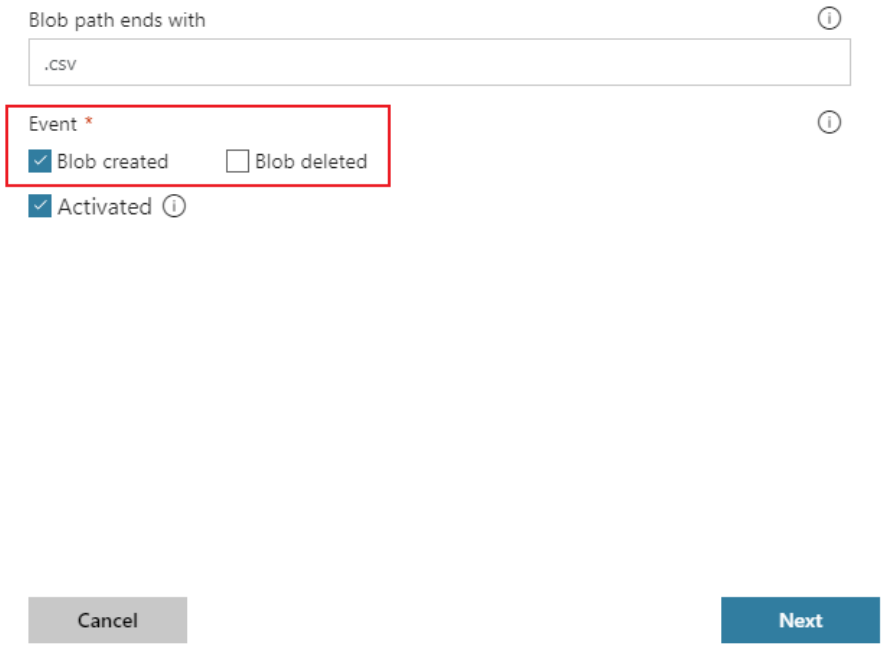
```
{
    "name": "MyTriggerName",
    "properties": {
        "type": "TumblingWindowTrigger",
        "runtimeState": "<<Started/Stopped/Disabled - readonly>>",
        "typeProperties": {
            "frequency": "<<Minute/Hour>>",
            "interval": <<int>>,
            "startTime": "<<datetime>>",
            "endTime: "<<datetime - optional>>"",
            "delay": "<<timespan - optional>>",
            "maxConcurrency": <<int>> (required, max allowed: 50),
            "retryPolicy": {
                "count": <<int - optional, default: 0>>,
                "intervalInSeconds": <<int>>,
            }
        },
        "pipeline": {
            "pipelineReference": {
                "type": "PipelineReference",
                "referenceName": "MyPipelineName"
            },
```

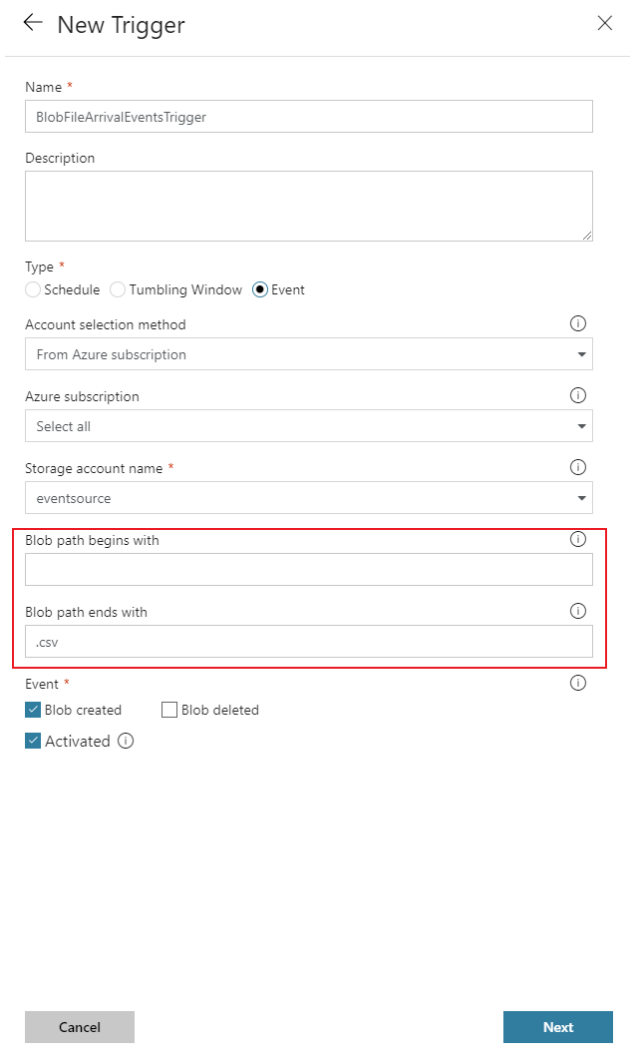


Triggers

Event Based Trigger

Data integration scenarios often require Data Factory customers to trigger pipelines based on events. Data Factory is now integrated with Azure Event Grid, which lets you trigger pipelines on an event.





Control Flow

Activities Known from v1 - Data Transformation Activities

Data transformation activity	Compute environment
Hive	HDInsight [Hadoop]
Pig	HDInsight [Hadoop]
MapReduce	HDInsight [Hadoop]
Hadoop Streaming	HDInsight [Hadoop]
Spark	HDInsight [Hadoop]
Machine Learning activities: Batch Execution and Update Resource	Azure VM
Stored Procedure	Azure SQL, Azure SQL Data Warehouse, or SQL Server
U-SQL	Azure Data Lake Analytics

Control Flow

New! Control Flow Activities in v2

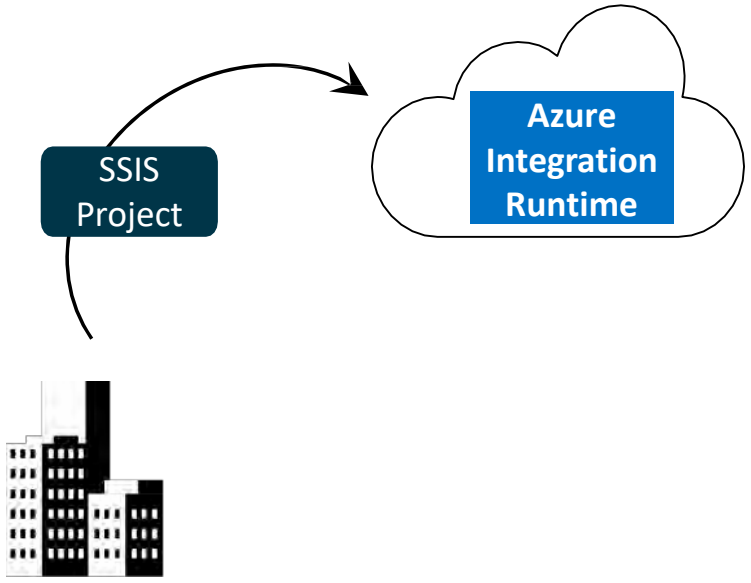
Control activity	Description
Execute Pipeline Activity	allows a Data Factory pipeline to invoke another pipeline.
ForEach Activity	used to iterate over a collection and executes specified activities in a loop.
Web Activity	call a custom REST endpoint and pass datasets and linked services
Lookup Activity	look up a record/ table name/ value from any external source to be referenced by succeeding activities. Could be used for incremental loads!
Get Metadata Activity	retrieve metadata of any data in Azure Data Factory e.g. did another pipeline finish
Do Until Activity	similar to Do-Until looping structure in programming languages.
If Condition Activity	do something based on condition that evaluates to true or false.

Control Flow

New! Control Flow Activities in v2

Control activity	Description
Append Variable Activity	to add a value to an existing array variable defined in a Data Factory pipeline.
Filter activity	to apply a filter expression to an input array.
Set Variable Activity	to set the value of an existing variable of type String, Bool, or Array defined in a Data Factory pipeline.
Validation activity	to ensure the pipeline only continues execution once it has validated the attached dataset reference exists
Wait activity	the pipeline waits for the specified period of time before continuing with execution of subsequent activities.
Webhook activity	to control the execution of pipelines through your custom code.
Data flow activity	to run your ADF data flow in pipeline debug (sandbox) runs and in pipeline triggered runs. (This Activity is in public preview)

SSIS in ADFv2



Managed Cloud Environment

Pick# nodes & node size

Resizable

SQL Standard Edition, Enterprise coming soon

Compatible

Same SSIS runtime across Windows, Linux, Azure Cloud

SSIS + SQL Server

SQL Managed instance + SSIS (in ADFv2) Access on premises data via VNet

Get Started

Hourly pricing (no SQL Server license

SSIS in ADFv2

Integration runtime - Different capabilities

1. **Data Movement**

Move data between data stores, built-in connectors, format conversion, column mapping, and performantand scalable data transfer

2. **Activity Dispatch**

Dispatch and monitor transformation activities (e.g. Stored Proc on SQL Server, Hive on HD Insight..)

3. **SSIS package execution**

Execute SSIS packages

SSIS in ADFv2

Combinations of IR types, networks and capabilities

IR type	Public network	Private network
Azure	Data movement Activity dispatch	
Self-hosted	Data movement Activity dispatch	Data movement Activity dispatch
Azure-SSIS	SSIS package execution	SSIS package execution

SSIS in ADFv2

Integration runtimes

1. Azure Integration Runtime

- move data between cloud data stores
- fully managed
- serverless compute service (PaaS) on Azure
- cost will occur only for time of duration
- user could define data movement units
- compute size auto scaled for copy jobs

SSIS in ADFv2

Integration runtimes

2. Self-hosted Integration Runtime

- perform data integration securely in a private network environment w/o direct line-of-sight from the public cloud environment
- Installed on-premises in your environment
- Supports copy activity between a cloud data stores and a data store in private network
- Supports dispatching the transform activities
- Works in your corporate network or virtual private network
- Only Outbound http based connections to open internet
- Scale out supported

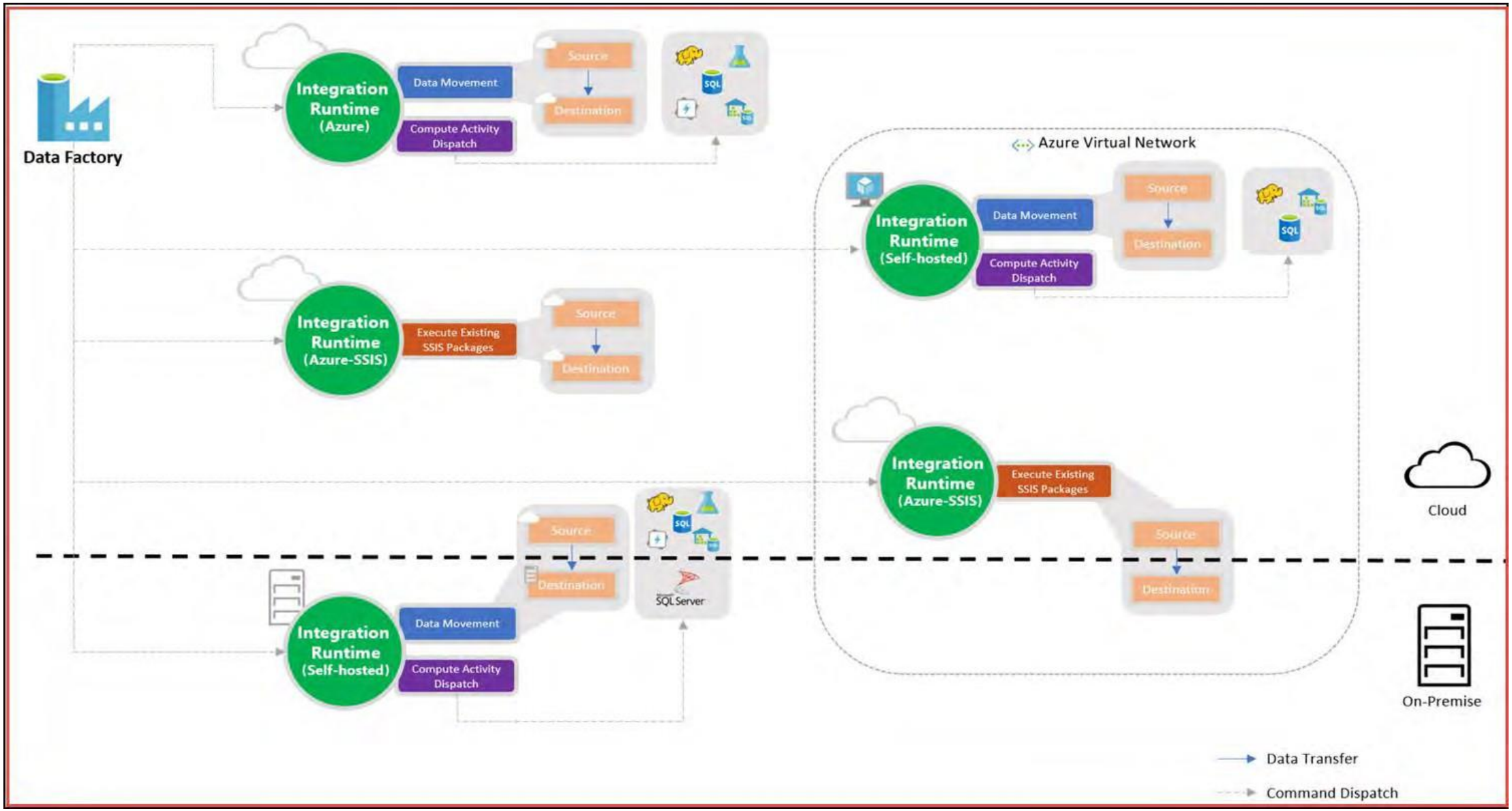
SSIS in ADFv2

Integration runtimes

3. Azure-SSIS Integration Runtime

- fully managed cluster of Azure VMs for native execution of SSIS packages.
- Access to on-premises data access using Vnet (classic in preview)
- SSIS Catalog on Azure SQL DB or SQL Managed Instance
- scale up: set node size
- scale out: number of nodes
- reduce costs by start/stop of service

SSIS in ADFv2



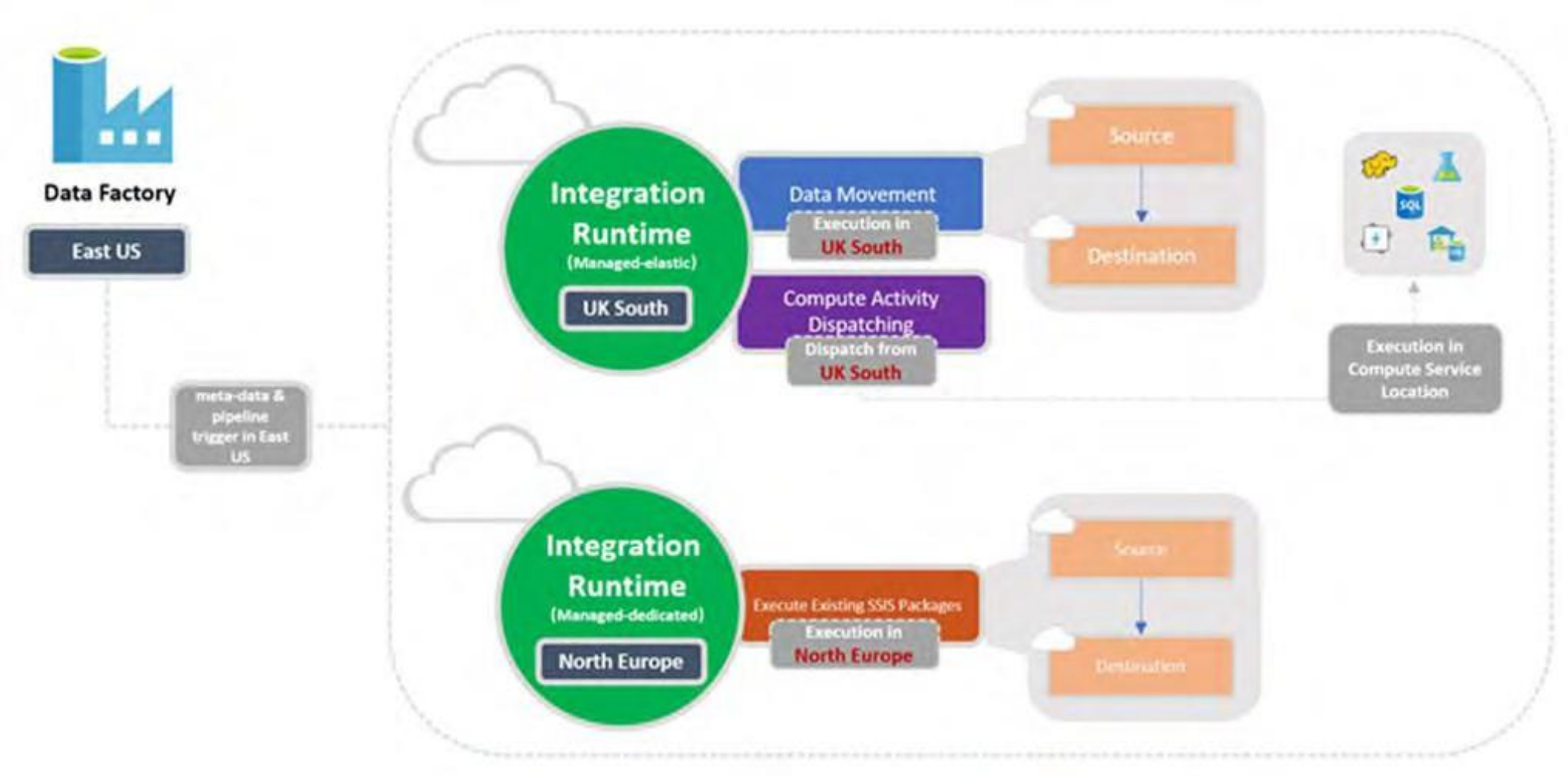
SSIS in ADFv2

Determining which IR to use

- IR is referenced as linked service in the data factory
- Transformation Activity: target compute needslinked service
- Copy Activity: source and sink need linked service, the computation is determined automatically (see detailson msdn)
- Integration runtime locations can differ from its Data Factory location which usesit

SSIS in ADFv2

Samples ADF and IR locations



SSIS in ADFv2

Scaleable Integration Services

How to scale up/out using 3 Settings on Azure SSIS IR

1. Configurable number of nodes on which SSIS is executed

```
$AzureSSISNodeSize = "Standard_A4_v2" # minimum size, others avail.*
```

2. Configurable size of nodes

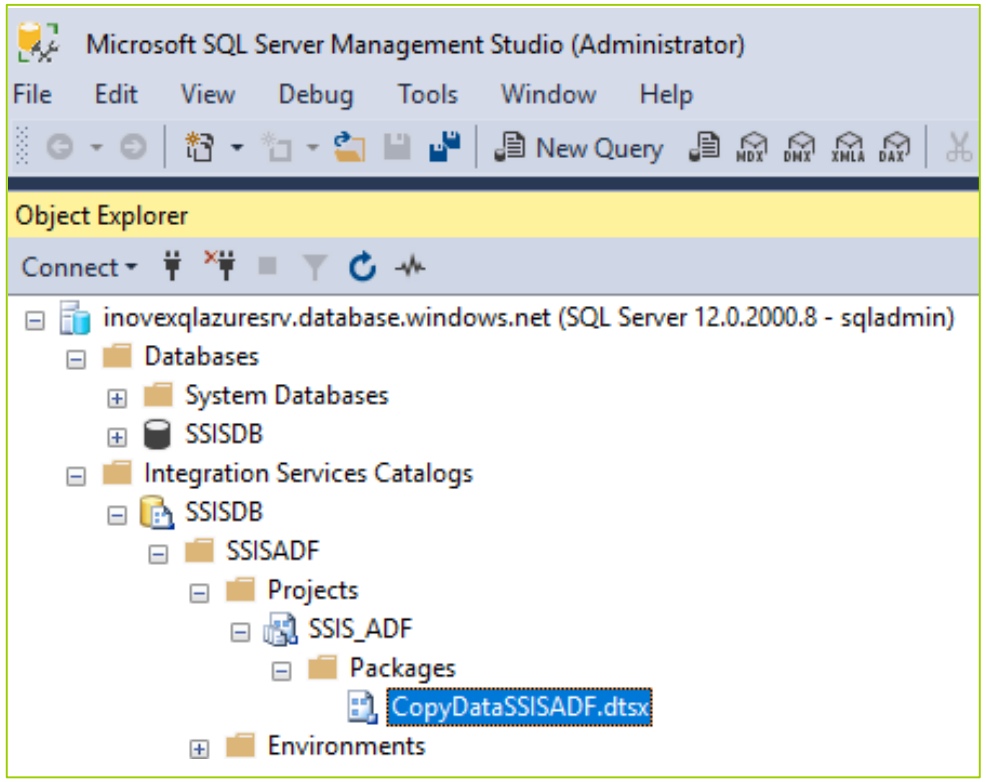
```
$AzureSSISNodeNumber = 2 #between 1 and 10 nodes*
```

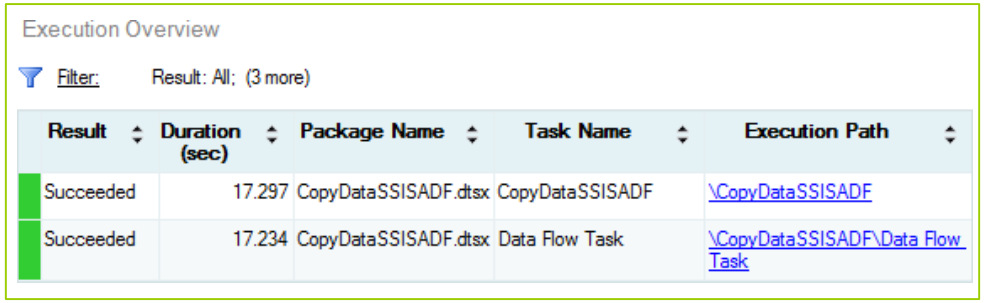
3. Configurable maximum parallel executions per node

```
$AzureSSISMaxParallelExecutionsPerNode = 2 # between 1-8*
```

In Powershell CommandLet: Set-AzureRmDataFactoryV2IntegrationRuntime

SSIS in ADFv2





SSIS in ADFv2

Notes from the field

1. Connect in SSMS directly to the DB SSISDB to see SSIS Catalog
2. Deploy from Visual Studio only in Project Deployment Mode, workaround SSMS Import
3. In Preview no SSIS 3rd Party components supported (e.g. Theobald for SAP, cozyroc..)
4. V2 IR supports only V2 pipelines and V1 IR supports only V1. Both cannot be used interchangeably. Even though it is the same installer
5. Use same location for Azure-SSIS IR and the used (SQL Azure) DB for SSIS Catalog

SSIS in ADFv2

Execution Methods

1. SSIS packages can be executed via SSMS

2. SSIS packages can be executed via CLI
   - Run dtexec.exe from the command prompt

3. SSIS packages can be executed via custom code/PSH using SSIS MOM .NET SDK/API
   - Microsoft.SqlServer.Management.IntegrationServices.dll is installed in .NET GAC with SQL Server/SSMS installation

4. SSIS packages can be executed via T-SQL scripts executing SSISDB sprocs
   - Execute SSISDB sprocs [catalog].[create_execution] + [catalog].[set_execution_parameter_value] + [catalog].[start_execution]

SSIS in ADFv2

Scheduling Methods

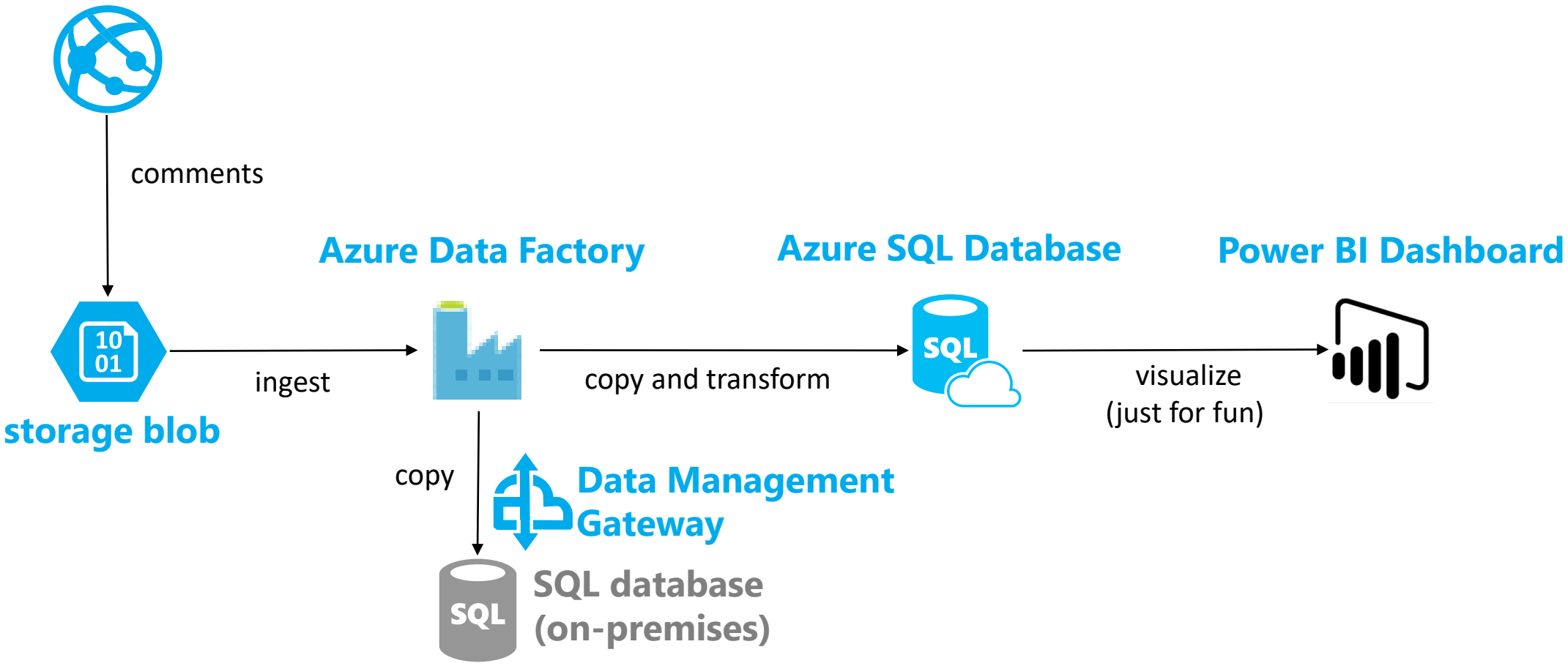
1. SSIS package executions can be directly/explicitly scheduled via ADFv2 App (Work in Progress)
   - For now, SSIS package executions can be indirectly/implicitly scheduled via ADFv1/v2 Sproc Activity

2. If you use Azure SQL MI server to host SSISDB
   - SSIS package executions can also be scheduled via Azure SQL MI Agent (Extended Private Preview)

3. If you use Azure SQL DB server to host SSISDB
   - SSIS package executions can also be scheduled via Elastic Jobs (Private Preview)

4. If you keep on-prem SQL Server
   - SSIS package executions can also be scheduled via on-prem SQL Server Agent

Demo


comments
Azure Data Factory
Azure SQL Database
Power BI Dashboard
10
01
ingest
copy and transform
SQL
visualize
(just for fun)
storage blob
copy
Data Management
Gateway
SQL
SQL database
(on-premises)

What is Azure Data Factory?

- Data Factory is a fully managed, cloud-based, data-integration service that automates the movement and transformation of data. Like a factory that runs equipment to transform raw materials into finished goods, Azure Data Factory orchestrates existing services that collect raw data and transform it into ready-to-use information.
- By using Azure Data Factory, you can create data-driven workflows to move data between on-premises and cloud data stores. And you can process and transform data by using compute services such as Azure HDInsight, Azure Data Lake Analytics, and the SQL Server Integration Services (SSIS) integration runtime.
- With Data Factory, you can execute your data processing either on an Azure-based cloud service or in your own self-hosted compute environment, such as SSIS, SQL Server, or Oracle. After you create a pipeline that performs the action you need, you can schedule it to run periodically (hourly, daily, or weekly, for example), time window scheduling, or trigger the pipeline from an event occurrence.

Introduction to Azure Data Factory

- The data landscape is more varied than ever with unstructured and structured data originating from many cloud and on-premises sources.
- Data Factory enables you to process on-premises data like SQL Server, together with cloud data like Azure SQL Database, Blobs, and Tables.
- These data sources can be composed, processed, and monitored through simple, highly available, fault-tolerant data pipelines.
- Combining and shaping complex data can take more than one try to get it right, and changing data models can be costly and time consuming.
- Using Data Factory you can focus on transformative analytics while the service 'takes care of the plumbing'.

What are the top-level concepts of Azure Data Factory?

- An Azure subscription can have one or more Azure Data Factory instances (or data factories).
- Azure Data Factory contains four key components that work together as a platform on which you can compose data-driven workflows with steps to move and transform data.

Pipelines

- A data factory can have one or more pipelines. A pipeline is a logical grouping of activities to perform a unit of work. Together, the activities in a pipeline perform a task. For example, a pipeline can contain a group of activities that ingest data from an Azure blob and then run a Hive query on an HDInsight cluster to partition the data. The benefit is that you can use a pipeline to manage the activities as a set instead of having to manage each activity individually. You can chain together the activities in a pipeline to operate them sequentially, or you can operate them independently, in parallel.

Activities

- Activities represent a processing step in a pipeline. For example, you can use a Copy activity to copy data from one data store to another data store. Similarly, you can use a Hive activity, which runs a Hive query on an Azure HDInsight cluster to transform or analyze your data. Data Factory supports three types of activities: data movement activities, data transformation activities, and control activities.

Datasets

- Datasets represent data structures within the data stores, which simply point to or reference the data you want to use in your activities as inputs or outputs.

Linked services

- Linked services are much like connection strings, which define the connection information needed for Data Factory to connect to external resources. Think of it this way: A linked service defines the connection to the data source, and a dataset represents the structure of the data. For example, an Azure Storage linked service specifies the connection string to connect to the Azure Storage account. And an Azure blob dataset specifies the blob container and the folder that contains the data.

Linked services have two purposes in Data Factory:

- To represent a *data store* that includes, but is not limited to, an on-premises SQL Server instance, an Oracle database instance, a file share, or an Azure Blob storage account..
- To represent a *compute resource* that can host the execution of an activity. For example, the HDInsight Hive activity runs on an HDInsight Hadoop cluster.

Triggers

- Triggers represent units of processing that determine when a pipeline execution is kicked off. There are different types of triggers for different types of events.

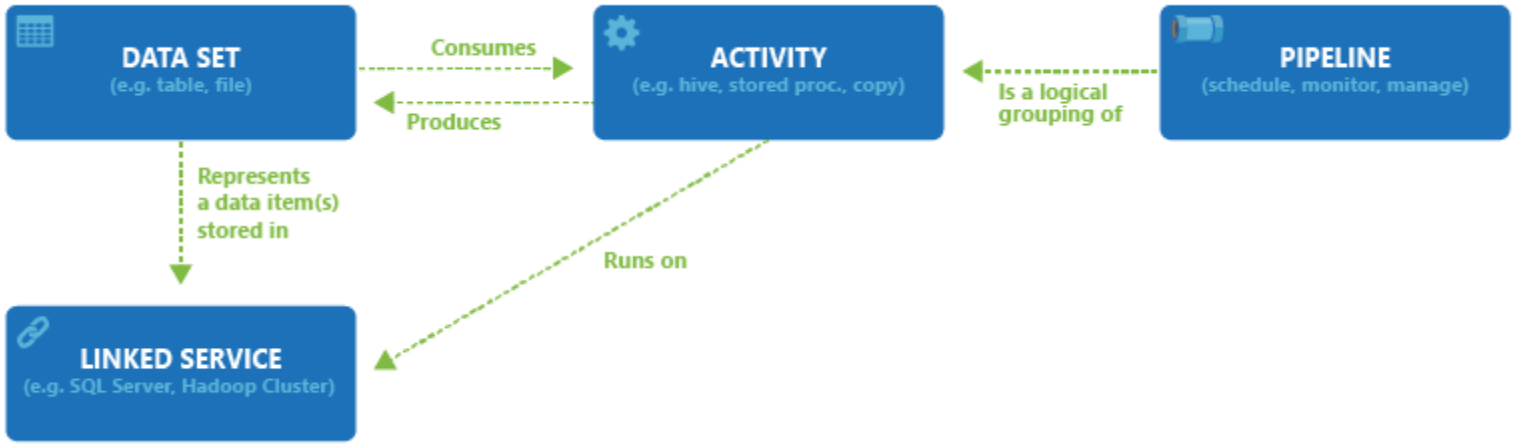
Pipeline runs

- A pipeline run is an instance of a pipeline execution. You usually instantiate a pipeline run by passing arguments to the parameters that are defined in the pipeline. You can pass the arguments manually or within the trigger definition.

Parameters

- Parameters are key-value pairs in a read-only configuration. You define parameters in a pipeline, and you pass the arguments for the defined parameters during execution from a run context. The run context is created by a trigger or from a pipeline that you execute manually. Activities within the pipeline consume the parameter values.
- A dataset is a strongly typed parameter and an entity that you can reuse or reference. An activity can reference datasets, and it can consume the properties that are defined in the dataset definition.
- A linked service is also a strongly typed parameter that contains connection information to either a data store or a compute environment. It's also an entity that you can reuse or reference.

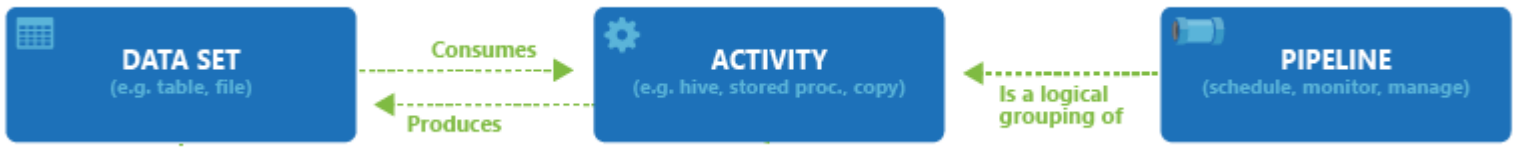
- **Control flows**
- Control flows orchestrate pipeline activities that include chaining activities in a sequence, branching, parameters that you define at the pipeline level, and arguments that you pass as you invoke the pipeline on demand or from a trigger. Control flows also include custom state passing and looping containers (that is, foreach iterators).



JSON

```
{
    "name": "AzureBlobInput",
    "properties": {
        "type": "AzureBlob",
        "linkedServiceName": {
                "referenceName": "MyAzureStorageLinkedService",
                "type": "LinkedServiceReference",
        },

        "typeProperties": {
            "fileName": "input.log",
            "folderPath": "adfgetstarted/inputdata",
            "format": {
                "type": "TextFormat",
                "columnDelimiter": ","
            }
        }
    }
}
```


DATA SET
(e.g. table, file)
Consumes
Produces
ACTIVITY
(e.g. hive, stored proc., copy)
Is a logical grouping of
PIPELINE
(schedule, monitor, manage)

Data transformation activities

Azure Data Factory supports the following transformation activities that can be added to pipelines either individually or chained with another activity.

Data transformation activity	Compute environment
Hive	HDInsight [Hadoop]
Pig	HDInsight [Hadoop]
MapReduce	HDInsight [Hadoop]
Hadoop Streaming	HDInsight [Hadoop]
Spark	HDInsight [Hadoop]
Machine Learning activities: Batch Execution and Update Resource	Azure VM
Stored Procedure	Azure SQL, Azure SQL Data Warehouse, or SQL Server
U-SQL	Azure Data Lake Analytics
Custom Code	Azure Batch
Databricks Notebook	Azure Databricks

Control activities

The following control flow activities are supported:

Control activity	Description
Execute Pipeline Activity	Execute Pipeline activity allows a Data Factory pipeline to invoke another pipeline.
ForEachActivity	ForEach Activity defines a repeating control flow in your pipeline. This activity is used to iterate over a collection and executes specified activities in a loop. The loop implementation of this activity is similar to Foreach looping structure in programming languages.
WebActivity	Web Activity can be used to call a custom REST endpoint from a Data Factory pipeline. You can pass datasets and linked services to be consumed and accessed by the activity.
Lookup Activity	Lookup Activity can be used to read or look up a record/ table name/ value from any external source. This output can further be referenced by succeeding activities.
Get Metadata Activity	GetMetadata activity can be used to retrieve metadata of any data in Azure Data Factory.
Until Activity	Implements Do-Until loop that is similar to Do-Until looping structure in programming languages. It executes a set of activities in a loop until the condition associated with the activity evaluates to true. You can specify a timeout value for the until activity in Data Factory.
If Condition Activity	The If Condition can be used to branch based on condition that evaluates to true or false. The If Condition activity provides the same functionality that an if statement provides in programming languages. It evaluates a set of activities when the condition evaluates to `true` and another set of activities when the condition evaluates to `false`.
Wait Activity	When you use a Wait activity in a pipeline, the pipeline waits for the specified period of time before continuing with execution of subsequent activities.

JSON

```
{
    "name": "PipelineName",
    "properties":
    {
        "description": "pipel
        "activities": [
         {
            "name": "MyFirstA
            "type": "Copy",
            "typeProperties":
            },
            "linkedServiceNam
            }
        },
        {
            "name": "MySecond
            "type": "Copy",
            "typeProperties":
            },
            "linkedServiceNam
            },
            "dependsOn": [
            {
                "activity": "
                "dependencyCo
                    "Succeede
                ]
            }
          ]
        }
      ],
      "parameters": {
       }
    }
}
```

JSON

Copy

```
{
    "name": "TransformPipeline",
    "properties": {
        "description": "My first Azure Data Factory pipeline",
        "activities": [
            {
                "type": "HDInsightHive",
                "typeProperties": {
                    "scriptPath": "adfgetstarted/script/partitionweblogs.hql",
                    "scriptLinkedService": "AzureStorageLinkedService",
                    "defines": {
                        "inputtable": "wasb://adfgetstarted@<storageaccountname>.blob.core.windows.net/inputdata
                        "partitionedtable": "wasb://adfgetstarted@<storageaccountname>.blob.core.windows.net/par
                    }
                },
                "inputs": [
                    {
                        "name": "AzureBlobInput"
                    }
                ],
                "outputs": [
                    {
                        "name": "AzureBlobOutput"
                    }
                ],
                "policy": {
                    "retry": 3
                },
                "name": "RunSampleHiveActivity",
                "linkedServiceName": "HDInsightOnDemandLinkedService"
            }
        ]
    }
}
```

JSON

```
Azure SQL table",
```

Data Factory Overview

Congratulations on continuing the path to learn Azure Courses!

Welcome to Azure Data Factory.

Hope you have a good foundational understanding of Azure cloud services. If not, you should go through the Azure Databases and Azure Storage courses before going any further.

In this course, you will be introduced to fundamental concepts of data factory, creating data factory, building pipelines, copying data, transforming data.

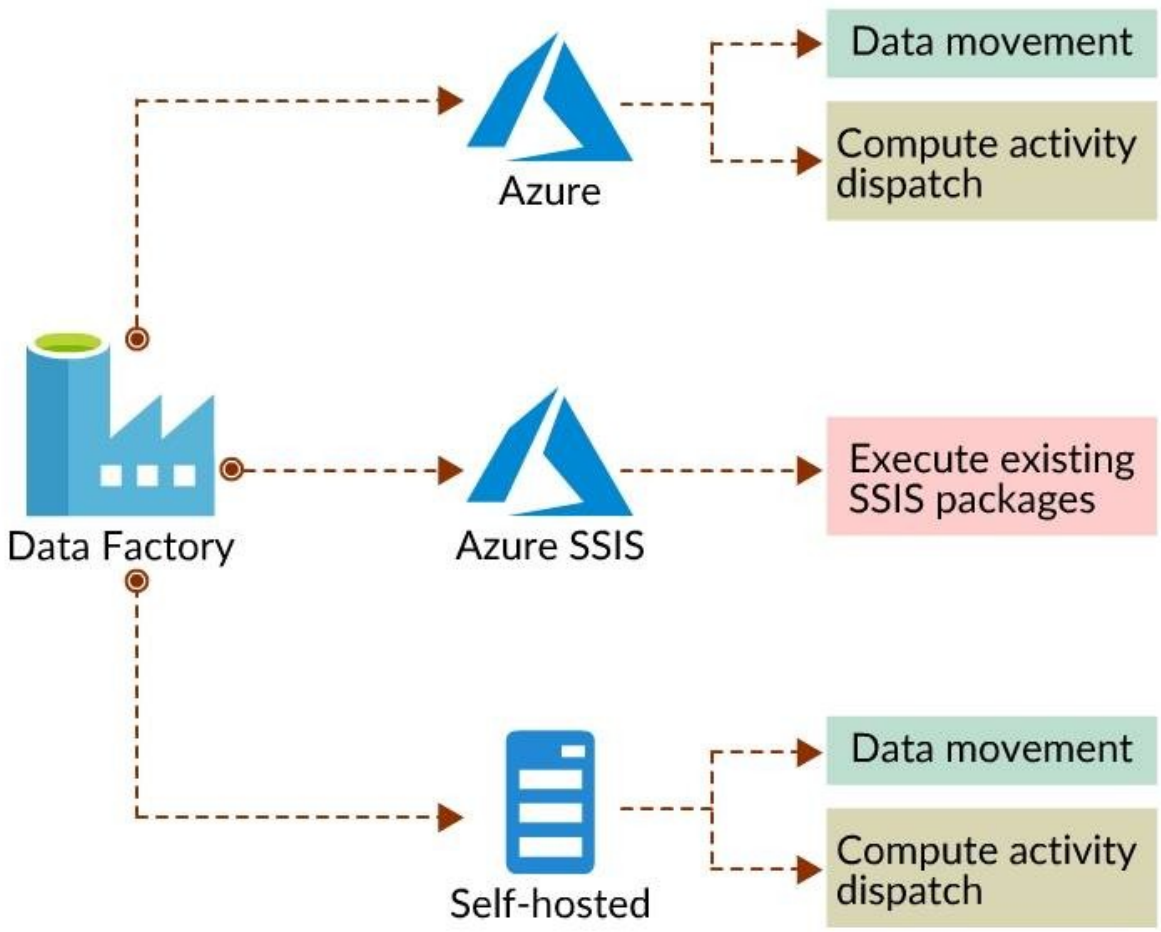
Happy Learning!

Prelude

In the world of Big data, the raw and unorganized data is stored in relational, non-relational, and other storage systems. The raw data doesn't have the proper context or meaning to provide meaningful insights to data scientists, analysts, or business decision makers. It requires some service to orchestrate and refine these raw data into actionable business insights.

Azure Data Factory is a managed cloud integration service that is built for these complex hybrid extract-load-transform (ELT), extract-transform-load (ETL), and data integration projects.

Integration Runtime



The image shows how the different integration runtimes can be used in combination to offer rich data integration capabilities and network support.

Azure Data Factory

- ***It is a cloud-based data integration service that supports to create data-driven workflows (pipelines) in the cloud for data transformation, data orchestration, and automating data movement***.

- **Azure Data Factory (ADF)** allows to create and schedule pipelines that ingest data from different data stores.

- It can process the transformation of data by using compute services such as Azure HDInsight (Hadoop), Azure Data Lake Analytics, and Azure Machine Learning.

Key Components in ADF

There are a set of top-level concepts to be familiar with them before diving into the Azure Data Factory.

- **Pipeline**:

  It is the logical grouping of activities that perform a task, where each can operate individually or independently in parallel. A data factory can contain one or more pipelines. The major benefit of these pipelines is, it allows us to manage a set of operations instead of managing each operation individually.

- **Activity**:

  An activity is represented as a processing step or task in a pipeline such as you can have a copy activity for copying data between data stores. It performs three kinds of activities such as data transformation, data movement, and control activities.

Key Components in ADF

- **Datasets**:

  Datasets represent data structures with a data store that points to data that needs to use in your activities as inputs and outputs.

- **Linked Services**:

  Linked services are used to represent connection objects for sources, destinations and compute resources that contains the connection strings (connection information needed for data factory to connect with external resources).

These four components comprise the ADF that works together to compose pipelines with steps to move and transform data.

Note: An Azure subscription can have more than one data factory instances.

Triggers and Pipeline Runs

- **Triggers**:

  Triggers represent the processing unit that determines when a pipeline execution needs to be kicked off. These are of different types for different events.

- **Pipeline Runs**:

  It is an instance of the pipeline execution that is instantiated by passing arguments to the parameters, which are defined in pipelines. The arguments can be passed manually or within the trigger definition.

ADF Pricing

The pricing is broken down into four ways that you are paying for this service.

1. **Azure activity runs vs Self-hosted activity runs**:

There are different pricing models for these. For the **Azure activity runs**, it is about copying activity. So you are moving data from an Azure Blob to an Azure SQL database or Hive activity running the high script on an Azure HDInsight cluster. With **self-hosted activity runs**, you can copy activity moving from an on-premises SQL Server to an Azure Blob Storage, a stored procedure to an Azure Blob Storage, or a stored procedure activity running a stored procedure on an on-premises SQL Server.

2. **Volume of data moved**:

It is measured in data movement units (DMUs). You should be aware of it, as this will change from default to auto, by using all the DMUs it can handle. This is paid on an hourly basis. Let's say you specify and use two DMUs. It takes an hour to move that data. The other option is that you could use eight DMUs and it takes 15 minutes. This price is going to end up the same. You're using 4X the DMUs, but it's happening in a quarter of the time.

ADF Pricing

3. **SSIS integration run times**:

Here, you're using A-series and D-series compute levels. When you go through these, you will understand that it depends on the compute requirements to invoke the process (how much CPU, how much RAM, how much attempt storage you need).

4. **The inactive pipeline**:

You're paying a small account for pipelines (about 40 cents currently). A pipeline is considered inactive if it's not associated with a trigger and hasn't been run for over a week. Yes, it's a minimal charge, but they do add up. When you start to wonder where some of those charges come from, it's good to keep this in mind.

Supported Regions

The regions currently supporting for provisioning the data factory are West Europe, East US, and East US 2.

However, the data factory can access data stores and compute resources from other regions to move data between data stores or process data using compute services, the service that powers data movement in data factory is available globally in many areas.

Provisioning Data Factory

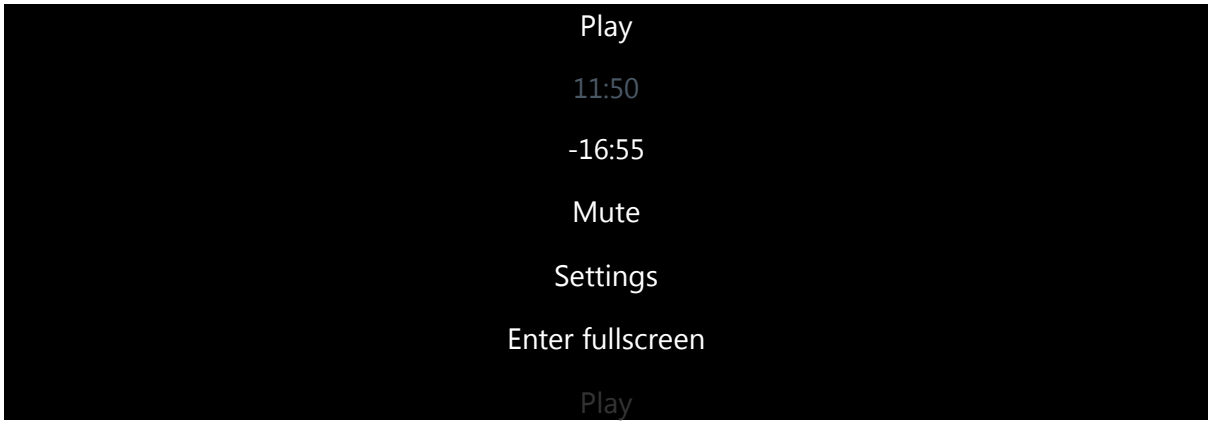
Azure subscription is the prerequisite for the data factory instance provision. The video shows how to provision the ADF from the Azure portal.

If you have trouble playing this video, please click here for help.

No transcript is available for this video.

Copy Data Using Copy Data Tool

Copy Data option in ADF allows copying data between data sources. The video shows a demo of copying data from Azure blob to SQL database using Azure portal and Visual Studio Cloud explorer.



If you have trouble playing this video, please click here for help.

No transcript is available for this video.

On-Premises Data Sources, Data Gateway

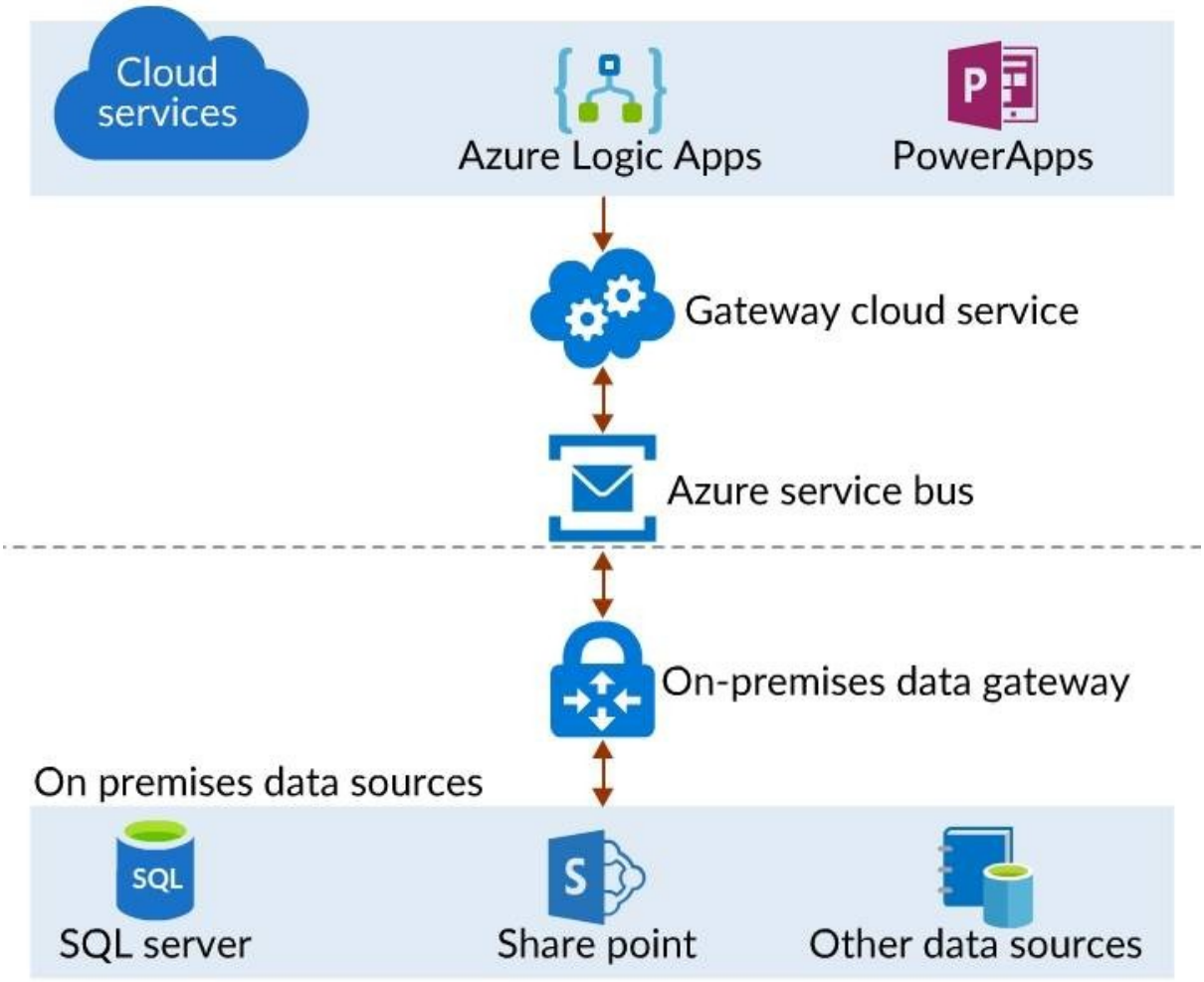
As you know, Azure lets you connect with the on-premises data sources such as SQL server, and SQL server analysis services. for queries or processing by provisioning the data gateway.

The data gateway provides secure data transfer between on-premises data sources and your Azure services in the cloud. It requires the following steps:

1. Download and run setup in an on-premise computer.
2. Registering your gateway by specifying a name and recovery key.
3. Creating a gateway resource in Azure in a subscription.
4. Connecting data sources to the gateway resource.

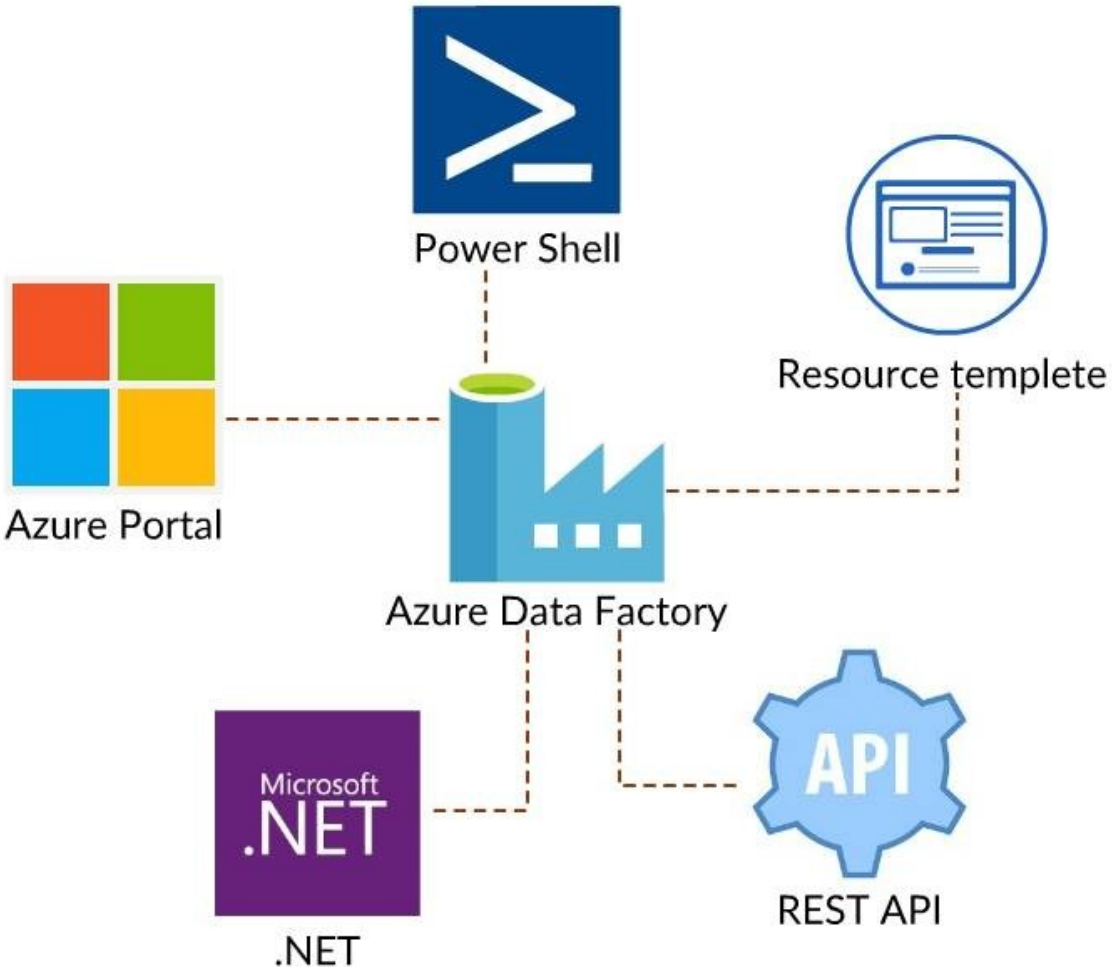
Note: You can register your gateway resource in any region, but it recommended to be in the same region of your data factory.

On-Premises Data Gateway



The picture shows how the data gateway works between on-premises data sources and Azure services.

ADF Management Tools



Azure Data Factory can be managed such as creating ADF, creating pipelines, monitoring pipelines through various ways such as:

1. Azure Portal
2. Azure PowerShell
3. Azure Resource Manager Templates
4. Using REST API
5. Using .NET SDK

Datasets

Datasets identify data such as files, folders, tables, documents within different data stores. For example, an Azure SQL dataset specifies the schema and table in the SQL database from which the activity should read the data.

Before creating a dataset, you have to create a linked service to link your data store to the data factory.

Both linked service and datasets are defined in JSON format in ADF.

Linked Service Structure

Linked Service structure is defined in a JSON format as below for an **AzureStorage** linked service.

```
{
    "name": "AzureStorageLinkedService",
    "properties": {
        "type": "AzureStorage",
        "typeProperties": {
            "connectionString": {
                "type": "SecureString",
                "value":
"DefaultEndpointsProtocol=https;AccountName=<accountname>;AccountKey=<accountkey>"
            }
        },
        "connectVia": {
            "referenceName": "<name of Integration Runtime>",
            "type": "IntegrationRuntimeReference"
        }
    }
}
```

Dataset Structure

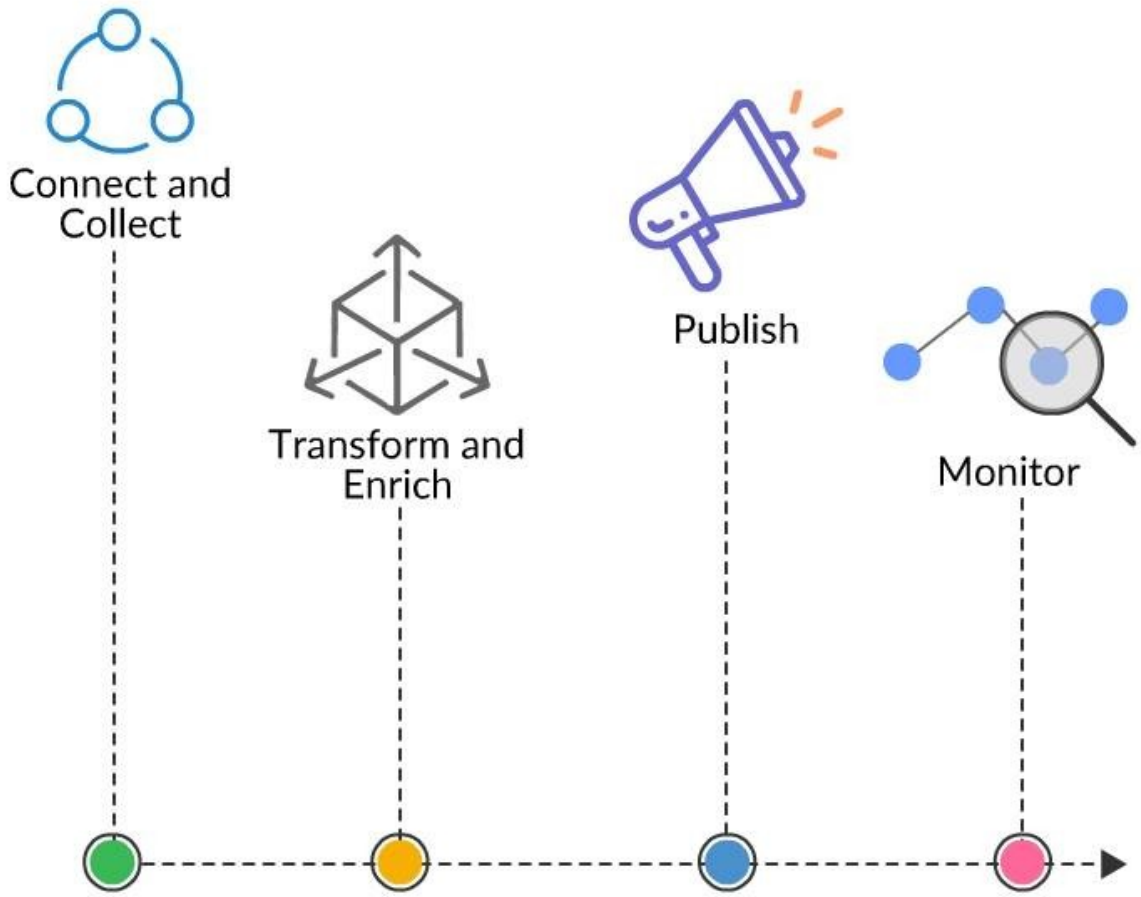
Dataset structure is defined in JSON format for an **AzureBlob** dataset as shown below.

```
{
    "name": "AzureBlobInput",
    "properties": {
```

```
        "type": "AzureBlob",
        "linkedServiceName": {
                "referenceName": "MyAzureStorageLinkedService",
                "type": "LinkedServiceReference",
        },

        "typeProperties": {
            "fileName": "input.log",
            "folderPath": "adfgetstarted/inputdata",
            "format": {
                "type": "TextFormat",
                "columnDelimiter": ","
            }
        }
    }
}
```

Pipeline Overview



A typical pipeline in an Azure data factory performs the above four activities represented in the picture.

Workflow of Pipelines

Connect and Collect:

The first step in building a pipeline is connecting to all the required sources of data and processing the movement of data as needed to a centralized location for subsequent processing. Without data factory, it requires to build custom data movement components or write services to move to integrate these data sources.

Transform and Enrich

The collected data that is presented in the centralized data store is transformed or processed by using compute services such as HDInsight Hadoop, Spark, Data Lake Analytics, and Machine Learning that is produced as feed to production environments.

Workflow of Pipelines

Publish:

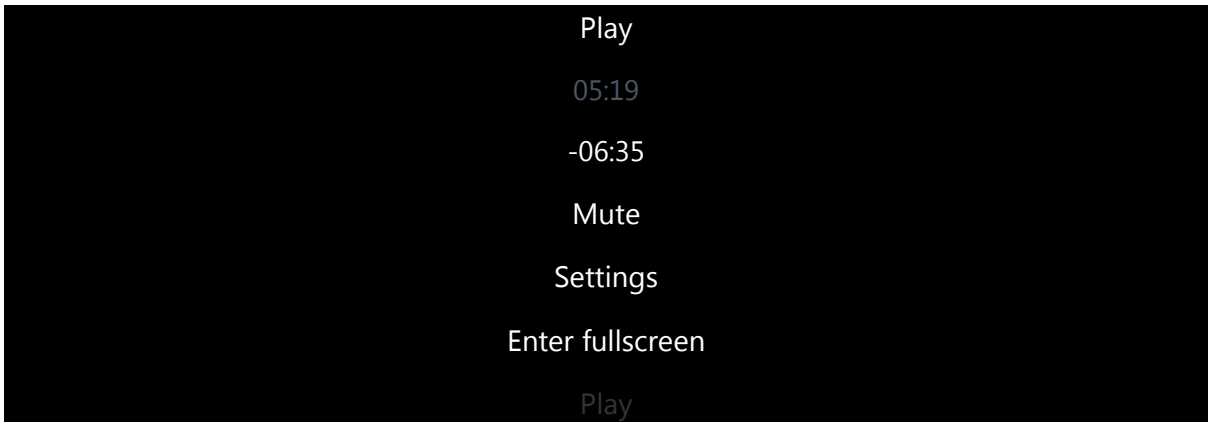
After the raw data was refined it is loaded into Azure Data Warehouse, Azure SQL Database, Azure CosmosDB, or whichever analytics engine your business users can point to from their business intelligence tools.

Monitor:

After the successful build and deployment of your data integration pipeline, you can monitor the scheduled activities and pipelines for success and failure rates. ADF has built-in support for monitoring pipeline Azure Monitor, API, PowerShell, Log Analytics, and health panels on the Azure portal.

Creating a Pipeline

The video shows creating a pipeline from scratch in the Azure portal.

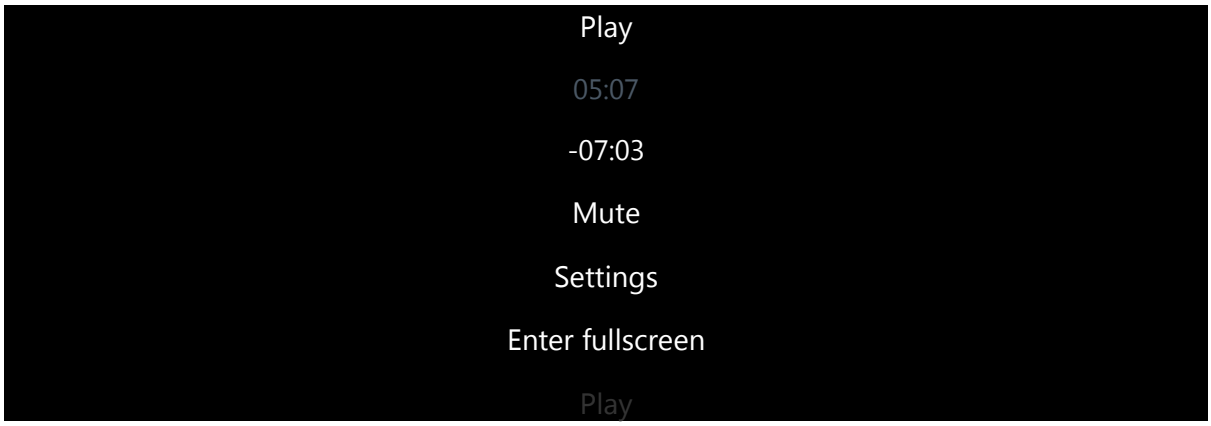


If you have trouble playing this video, please click here for help.

No transcript is available for this video.

Monitoring Pipelines

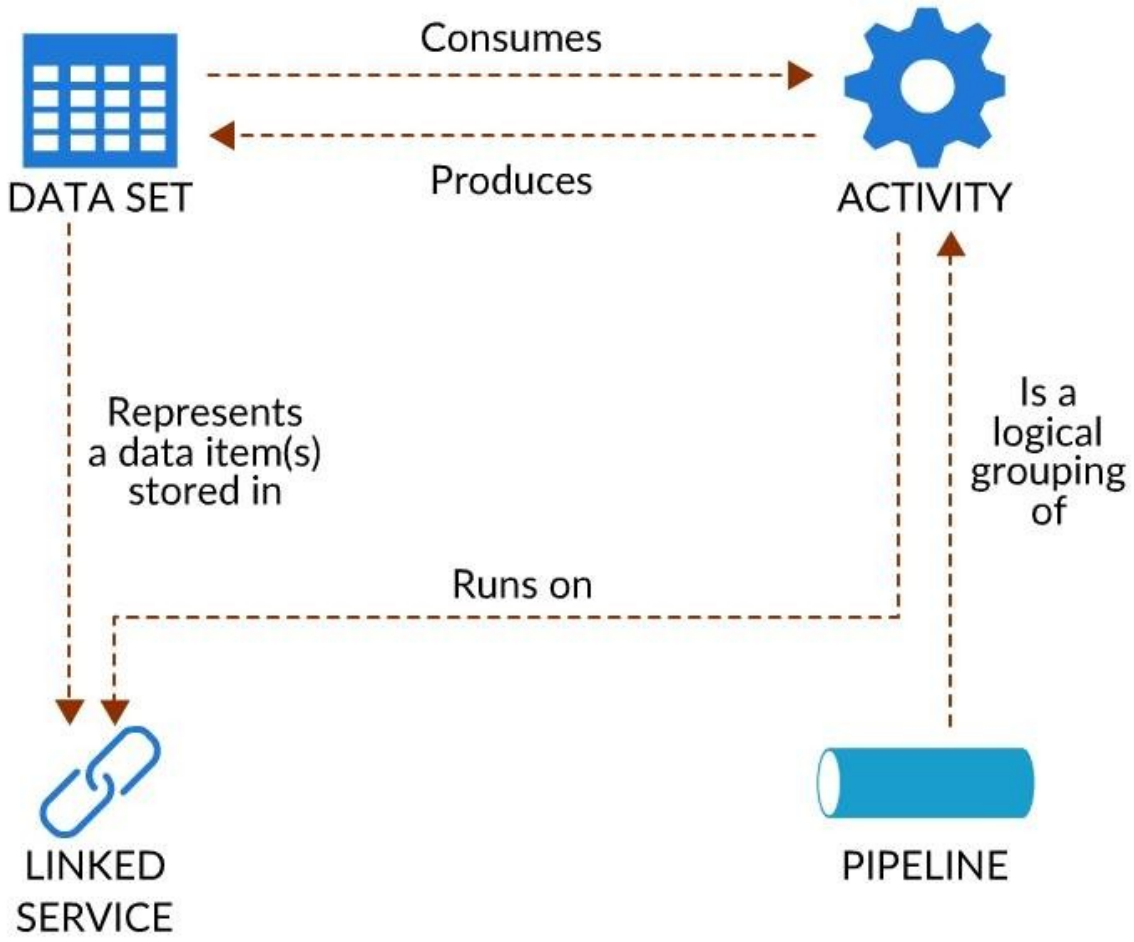
The video shows monitoring an up and running pipeline from ADF resource explorer in Azure Portal.



If you have trouble playing this video, please click here for help.

No transcript is available for this video.

Relationship Between Four Components



The picture shows the relationship between the four key components in ADF.

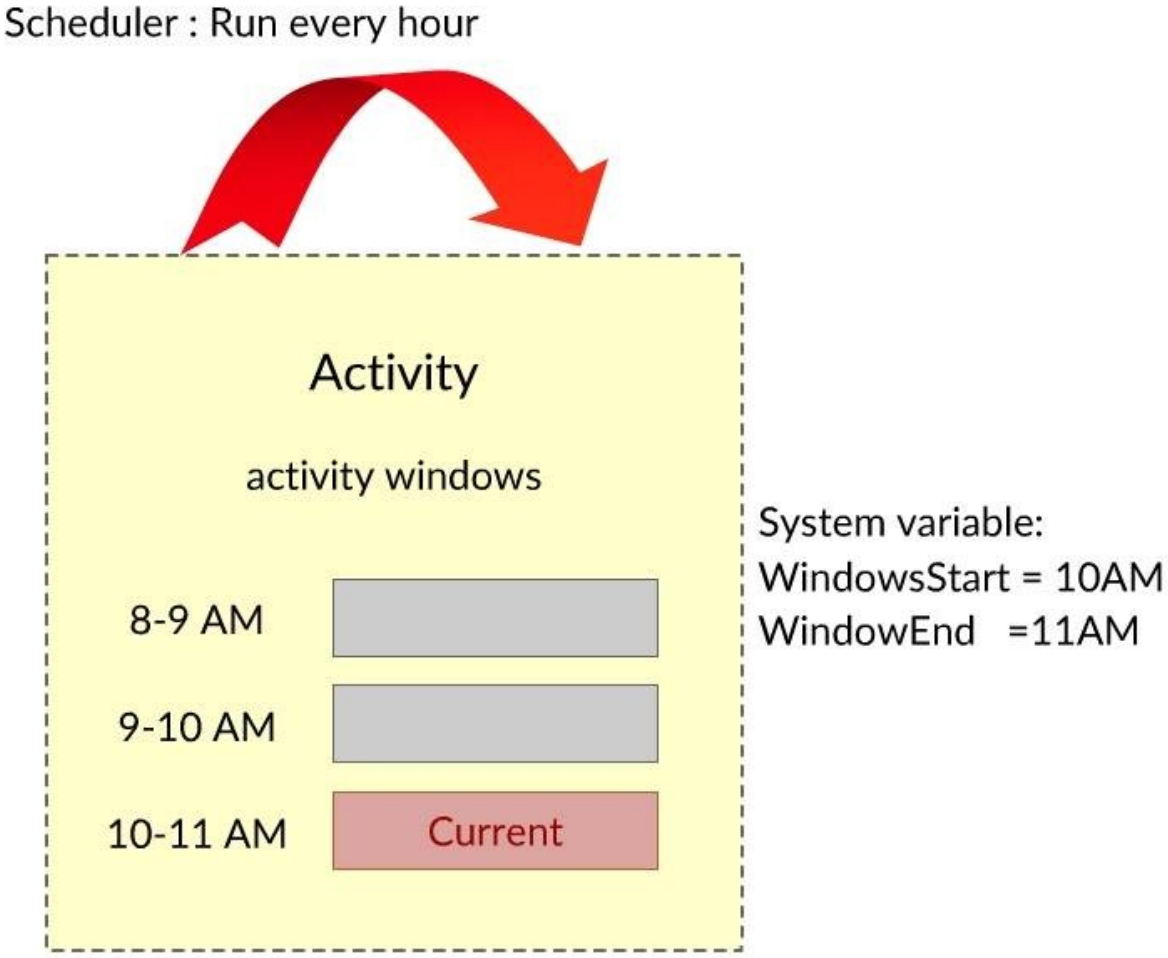
Scheduling Pipelines

A pipeline is active only between its start time and end time that does not execute before, or after the start and end times, respectively. If it is paused, it does not get executed irrespective of its start and end time. For a pipeline to run, it should not be paused.

You can define the pipeline start and end times in the pipeline definition in JSON format as below.

```
"start": "2017-04-01T08:00:00Z",
"end": "2017-04-01T11:00:00Z"
"isPaused": false
```

Specifying Schedule for an Activity



You must specify schedulers for the activities that are executing in the pipeline to run the pipeline.

Example of an activity that is scheduled to run hourly in the JSON format.

```
"scheduler": {
    "frequency": "Hour",
    "interval": 1
},
```

Specify Schedule to Dataset

Similar to scheduling activity in a pipeline, you can specify a schedule to the dataset, such as, you can specify a schedule that takes input datasets and produce output data in a frequency (hourly, daily, weekly, monthly).

For example, the below JSON format file defines the input and output data available in an hourly frequency:

```
{
    "name": "AzureSqlInput",
    "properties": {
        "published": false,
        "type": "AzureSqlTable",
        "linkedServiceName": "AzureSqlLinkedService",
        "typeProperties": {
            "tableName": "MyTable"
        },
        "availability": {
            "frequency": "Hour",
            "interval": 1
        },
        "external": true,
        "policy": {}
    }
}
```

Note: You must specify schedule frequency and intervals in the **availability** section.

Scheduling Pipelines: Demo

The video shows scheduling the pipelines at regular intervals in visual studio, you can try this in the portal as well.

If you have trouble playing this video, please click here for help.

No transcript is available for this video.

Troubleshooting a Scheduled Pipeline

The video shows how to troubleshoot a scheduled pipeline by monitoring it and fixing the failed execution activity.

If you have trouble playing this video, please click here for help.

No transcript is available for this video.

Visual Authoring

The Azure Data Factory user interface experience (UX) lets you visually author and deploy resources for your data factory without having to write any code.

You can drag activities to a pipeline canvas, perform test runs, debug iteratively, and deploy and monitor your pipeline runs. There are two approaches for using the UX to perform visual authoring:

- Author directly with the Data Factory service.
- Author with Visual Studio Team Services (VSTS) Git integration for collaboration, source control, or versioning.

Visual authoring with the Data Factory service differs from visual authoring with VSTS in two ways:

- The Data Factory service doesn't include a repository for storing the JSON entities for your changes.
- The Data Factory service isn't optimized for collaboration or version control.

Integration JSON in ADF

The Integration Runtime (IR) is the compute infrastructure used by Azure Data Factory to provide data integration capabilities across different network environments such as data movement, activity dispatch, and SSIS package execution.

While moving data from data stores in public and private networks, it provides support for built-in connectors, format conversion, column mapping, and scalable data transfer.

The IR (Integration runtime) provides the bridge between the activity and linked Services.

Types of Integration Runtime

There are three different types of IR which you have to choose the type that best serve the data integration capabilities you are looking for:

- Azure
- Self-hosted
- Azure-SSIS

Azure IR type is recommended to choose for **data movement** and **activity dispatch** activities over a public network whereas the self-hosted IR type is recommended over both public and private networks for the same activities.

The Azure SSIS IR type is recommended to choose for **SSIS package execution** over both public and private networks.

Azure Integration Runtime

An Azure integration runtime is capable of:

Running copy activity and data movement activities between cloud data stores.

Activity dispatching the following data transform activities in public network: HDInsight Hadoop, Machine Learning Batch Execution and update resource activities, Data Lake Analytics U-SQL activity and other custom activities.

Azure IR supports connecting data stores over a public network with public accessible endpoints.

Self-Hosted Integration Runtime

Self-hosted IR is to perform data integration securely in private network. You can install a self-hosted IR on-premises environment behind your corporate firewall, or inside a virtual private network.

It is capable of:

Running copy activity and data movement activity between data stores.

Activity dispatching the following transform activities against compute resources in On-Premise or Azure Virtual Network: HDInsight Hadoop, Machine Learning Batch Execution and update resource activities, Data Lake Analytics U-SQL activity, and other custom activities.

Lift and Shift SSIS

It is easy to move your SQL Server Integration Services (SSIS) workloads, projects, and packages to the Azure cloud, as it deploys, runs and manages SSIS projects and packages in the SSIS Catalog on Azure SQL Database or with familiar tools such as SQL Server Management Studio (SSMS).

Moving your on-premises SSIS workload to Azure will reduce your operational costs and provides maximum scalability.

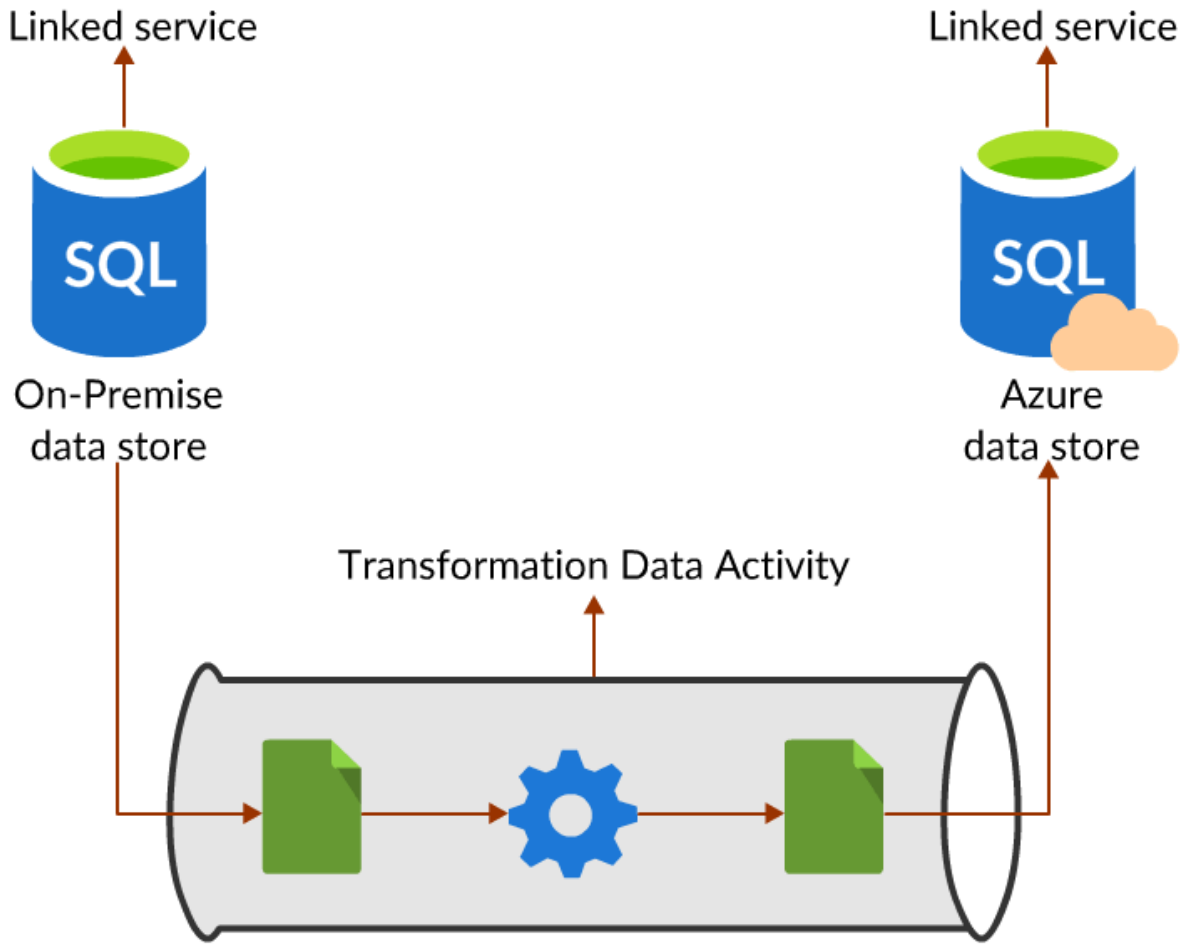
Azure SSIS Integration Runtime

To lift and shift existing SSIS workload, you can create an Azure-SSIS IR. which is dedicated to run or execute SSIS packages. It can be provisioned in either public network or private network.

Integration Runtime Location:

The IR Location defines the location of its back-end compute, and essentially the location where the data movement, activity dispatching, and SSIS package execution are performed. The IR location can be different from the location of the data factory it belongs to.

Azure Data Factory Working



The above GIF represents the working of ADF, it requires to install three IRs in the whole scenario of data movement and data transformation from on-premise to Azure.

1. Installs **self-hosted** IR type in the on-premise that provides integration capabilities overs private network.
2. Installs **Azure** IR type at the transformation part.
3. Installs **Azure** IR type at the Azure data store that provides integration capabilities over the public network.

Scenarios for Copy Data Using ADF

There are three types of copy data scenarios performed across different environments between data stores such as:

On-premise to Azure

Azure cloud data store instance to another Azure cloud data store instance

SaaS Application to Azure

There are two ways to create a pipeline for copy activity:

From ADF editor and monitor tile, choosing editor option to manually create key components (dataset, linked services) and perform copy activity.

Using copy data tool in ADF.

Scenario 1: On-Premises to Azure

Let's assume a scenario copying data from on-premise SQL server to Azure Data Lake Store using Copy data tool in ADF. Follow the below steps to perform this kind of activity:

Choose Copy data option from the ADF

It has six steps to be evaluated in a sequence such as Properties, Source, Destination, Settings, Summary, and Deployment.

Properties:

Provide the task name, task description, and task schedule.

Source Settings

Source:

1. Choose the SQL server data store and provide details of it such as **connection name, Server name, Database name, Authentication type, username, and password**.
2. It requires to run Integration runtime to provide capabilities for data movement. For this, you have to choose **create integration runtime** and **launch express setup on this computer**. This will install **self-hosted IR** on your computer.
3. Choose the existing tables from your SQL server that needs to be copied. It also allows copying a table by filtering a column.

Destination and Settings

Destination:

Choose the data store data lake store and provide the details of it such as connection name, network environment, azure subscription, data lake store account name, authentication type, tenant.

Choose the output file or folder (where to copy data).

Choose file format settings.

Settings:

Choose actions that need to be performed such as:

Abort copy on the first incompatible row.

Skip all incompatible rows.

Skip and log all incompatible rows.

Summary and Deployment

Summary:

Shows you the properties and source settings, click next for deployment.

Deployment:

Shows the deployment status and other options that allow to edit pipeline and monitor.

Watch this video in open network (not using tcs LAN) from 9th minute to have a better understanding about how to perform copy activity from on-premise to Azure.

Scenario 2: Azure to Azure

Copying data from one Azure data store to another data store using copy tool is explained in the first topic (Introduction to Azure data factory), please refer to card no: 11.

Note: Check this video on open network, it explains performing copy activity manually by creating linked service, datasets for data stores using Azure portal. This is a similar process for any type of scenario to create pipeline.

Scenario 3: SaaS Application to Azure

The video shows copying data from a dynamic CRM tool to Azure data lake store using copy data tool.

If you have trouble playing this video, please click here for help.

No transcript is available for this video.

Transform Data in ADF

Data transformation is executed as a transformation activity that executes in a computing environment such as an Azure HDInsight cluster or an Azure Batch.

Data factory supports various transformation activities that can be added to pipelines either individually or chained with another activity. Which are:

1. **HDInsight (Hadoop)** compute environment is used for **Hive, Pig, MapReduce, Hadoop Streaming, Spark** data transformation activities.
2. **Azure VM** compute environment is used for **Machine learning batch execution and update resource** data transformation activities.
3. Azure SQL, Azure SQL data warehouse compute environment is used for **Stored procedure** data transformation activity.
4. **Azure Data Lake Analytics** compute environment is used for **U-SQL** data transformation activity.

U-SQL Transformations

Data Lake Analytics U-SQL Activity that runs a U-SQL script on an Azure Data Lake Analytics compute linked service.

1. Create an Azure Data Lake Analytics account before creating a pipeline with a Data Lake Analytics U-SQL Activity.
2. The Azure Data Lake Analytics linked service requires a service principal authentication to connect to the Azure Data Lake Analytics service.
3. To use service principal authentication, register an application entity in Azure Active Directory (Azure AD) and grant it the access to both the Data Lake Analytics and the Data Lake Store it uses.

U-SQL Transformations Activity: Demo

The video shows creating a pipeline for data transformation activity that uses the Azure data lake compute environment.

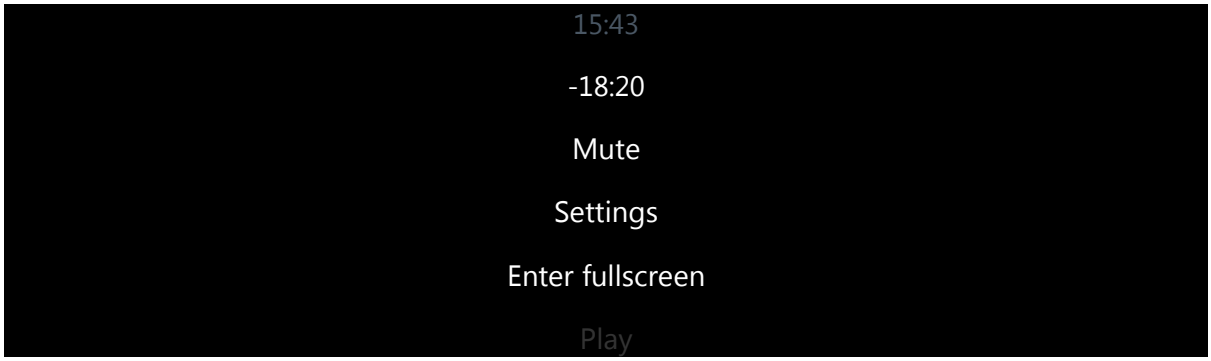
If you have trouble playing this video, please click here for help.

No transcript is available for this video.

Hive Transformations: Demo

Similar to the Data lake activity, HDHadoop Hive activity executes Hive queries.

Play



If you have trouble playing this video, please click here for help.

No transcript is available for this video.

Custom Query and Parameters

You can parameterize a linked service and pass dynamic values at run time. Linked service parameterization is supported in the Data Factory UI in the Azure portal for the following data stores. For all other data stores, you can parameterize the linked service by selecting the Code icon on the Connections tab and using the JSON editor.

- Azure SQL Database
- Azure SQL Data Warehouse
- SQL Server
- Oracle
- Cosmos DB
- Amazon Redshift
- MySQL
- Azure Database for MySQL

Custom Activity

If you need to transform data in a way that is not supported by Data Factory, you can create a custom activity with your own data processing logic and use the activity in the pipeline. You can configure the custom .NET activity to run using either an Azure Batch service or an Azure HDInsight cluster.

You can create a Custom activity with your data movement or transformation logic and use the activity in a pipeline.

The parameters required to define for a custom activity in JSON format are: **name, activities, typeproperties**.

Refer to this link for azure batch basics as the custom activity runs your customized code logic on an Azure Batch pool of virtual machines.

Custom Activity Example

The following JSON snippet defines a pipeline with a simple Custom Activity. The activity definition has a reference to the Azure Batch linked service.

```
{
  "name": "MyCustomActivityPipeline",
  "properties": {
    "description": "Custom activity sample",
    "activities": [{
      "type": "Custom",
      "name": "MyCustomActivity",
      "linkedServiceName": {
        "referenceName": "AzureBatchLinkedService",
        "type": "LinkedServiceReference"
      },
      "typeProperties": {
        "command": "helloworld.exe",
        "folderPath": "customactv2/helloworld",
        "resourceLinkedService": {
          "referenceName": "StorageLinkedService",
          "type": "LinkedServiceReference"
        }
      }
    }]
  }
}
```

Hands-on scenario

Your company has a requirement to maintain employee details in the SQL database. You are given a text file containing employee details that has to be migrated. As a cloud computing professional, you plan to simplify the task by using Azure Data Factory. i) Create a storage account: Location: (US) East US 2, Performance: Standard, Account Kind: Storage V2 (general-purpose v2), Replicapagetion: Locally-redundant storage (LRS). ii) Create a Blob storage and upload the file containing employee data (refer to the sample employee data provided in the following table). iii) Create SQL database: Server: Create new, Location: (US) East US 2, Compute + storage: Basic, Network Connectivity: Public endpoint, Allow Azure services and resources to access the server: Yes, Add current client IP address: Yes. iv) Set a firewall rule for the SQL database to allow IP addresses from 0.0.0.0 to 255.255.255.255 and write an SQL query to create a table database (refer to the SQL query provided in the following

table). v) Create Data Factory: Region: East US 2, Configure Git later: mark as later. vi) Create a Copy data tool in the data factory to move data from blob storage to the SQL database..

Employeedata.txt

```
FirstName|LastName


John|Brito


Alwin|Chacko
```

SQL Query

```
CREATE TABLE dbo.emp
(
    ID int IDENTITY(1,1) NOT NULL,
    FirstName varchar(50),
    LastName varchar(50)
)
GO
CREATE CLUSTERED INDEX IX_emp_ID ON dbo.emp (ID);
```

Note:

Use the credentials given in the hands-on to log in to the Azure Portal, create a new resource group and use the same resource group for all resources. The Username/Password/Services Name can be as per your choice, after completing the hands-on, delete all the resources created.

Course Summary

So far in this course, you have learned the following topics:

Provisioning an Azure Data Factory

Creating Pipelines for various activities

Transforming data through different compute environments

Monitoring ADF

Hope you had a better understanding of the data factory concept. Hope you had hands-on practice in creating pipelines and activities.

Azure Updates

Like many other cloud infrastructure platforms today,

Azure is continuously developing updates to its services and components.

Every effort has been taken to update course content where there are significant changes to product capability. However, there will be occasions where the course content does not exactly match the latest version of the product.

Hence, we encourage you to check Azure updates as a starting point for the latest information about updates in Azure.