



Azure Data Factory

Hybrid data integration, at global scale

Andrea Benedetti
Cloud Solution Architect

AZURE DATA FACTORY

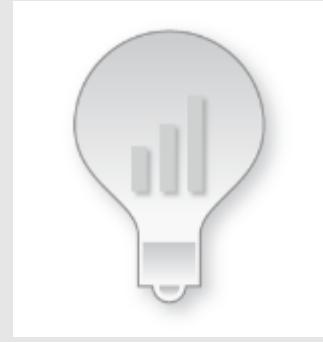
A fully-managed data integration service in the cloud



PRODUCTIVE



HYBRID



SCALABLE



TRUSTED

- ✓ Drag & Drop UI
- ✓ Codeless Data Movement

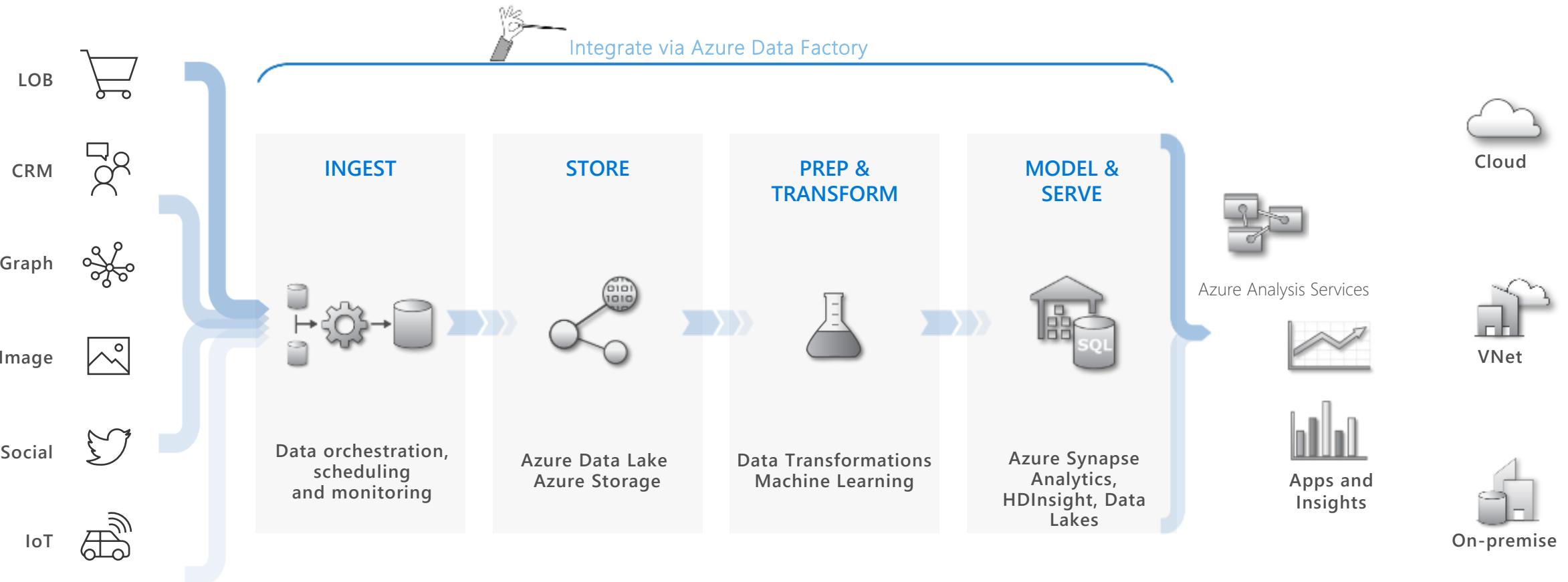
- ✓ Orchestrate where your data lives
- ✓ Lift SSIS packages to Azure

- ✓ Serverless scalability with no infrastructure to manage

- ✓ Certified compliant Data Movement

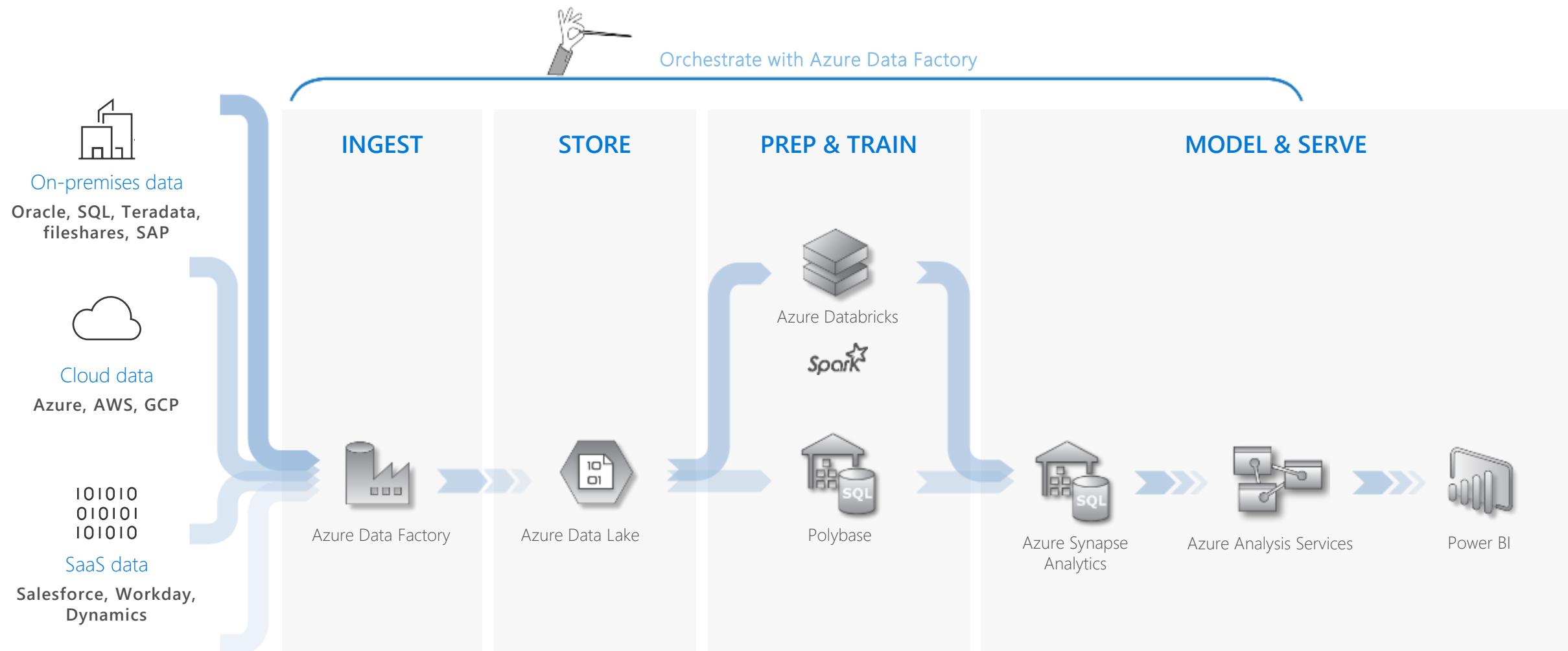
AZURE DATA FACTORY

Modernize your enterprise data warehouse at scale



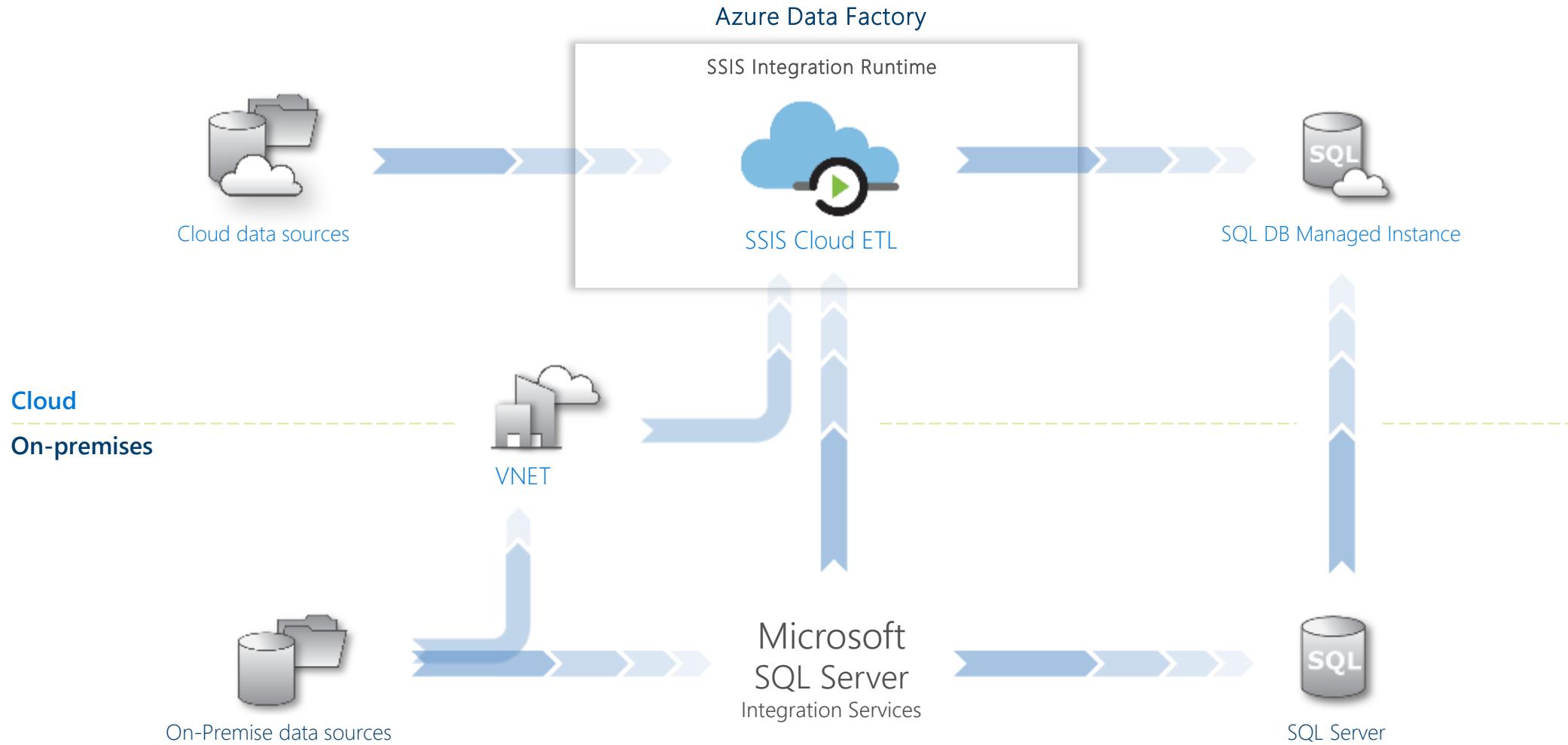
AZURE DATA FACTORY

Modernize your enterprise data warehouse at scale

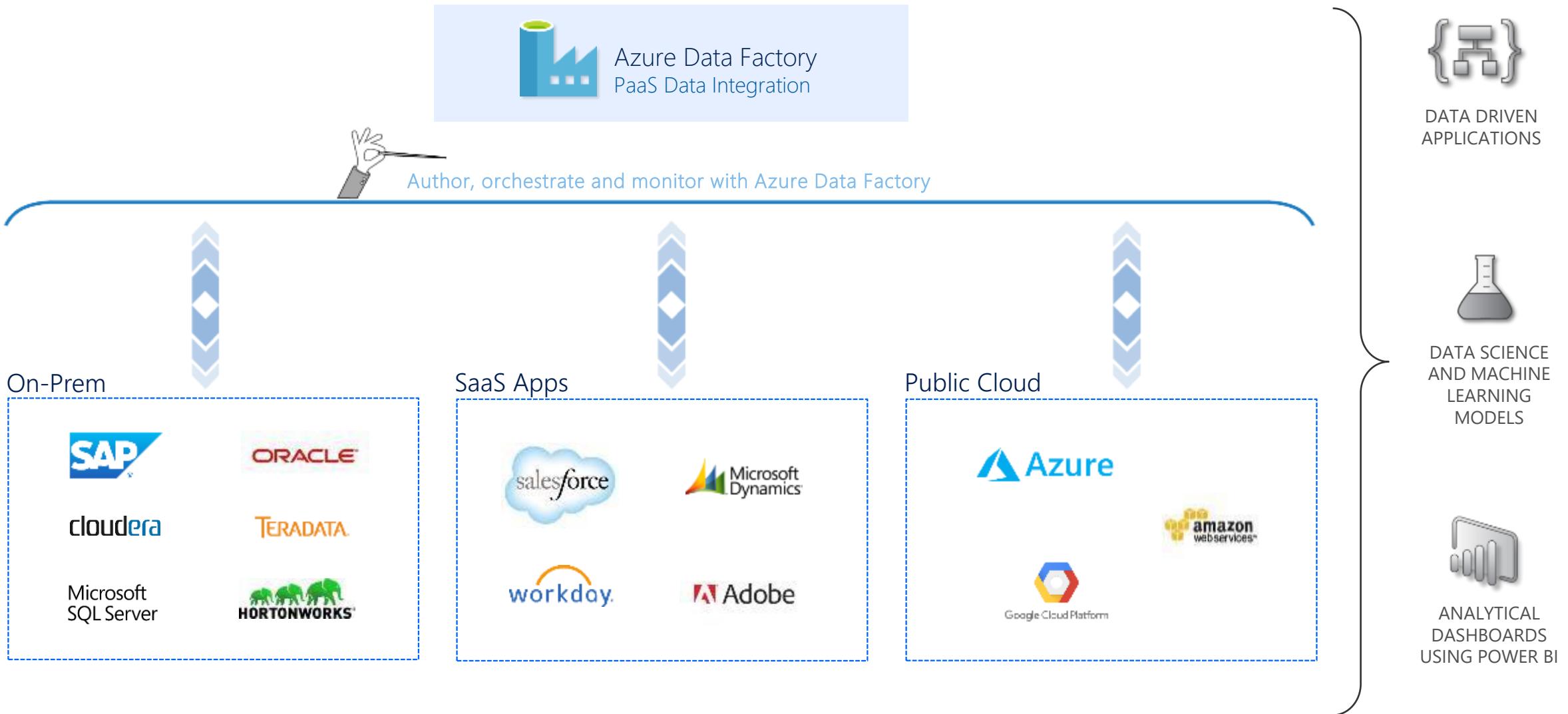


Microsoft Azure also supports other Big Data services like Azure HDInsight, Azure SQL Database and Azure Data Lake to allow customers to tailor the above architecture to meet their unique needs.

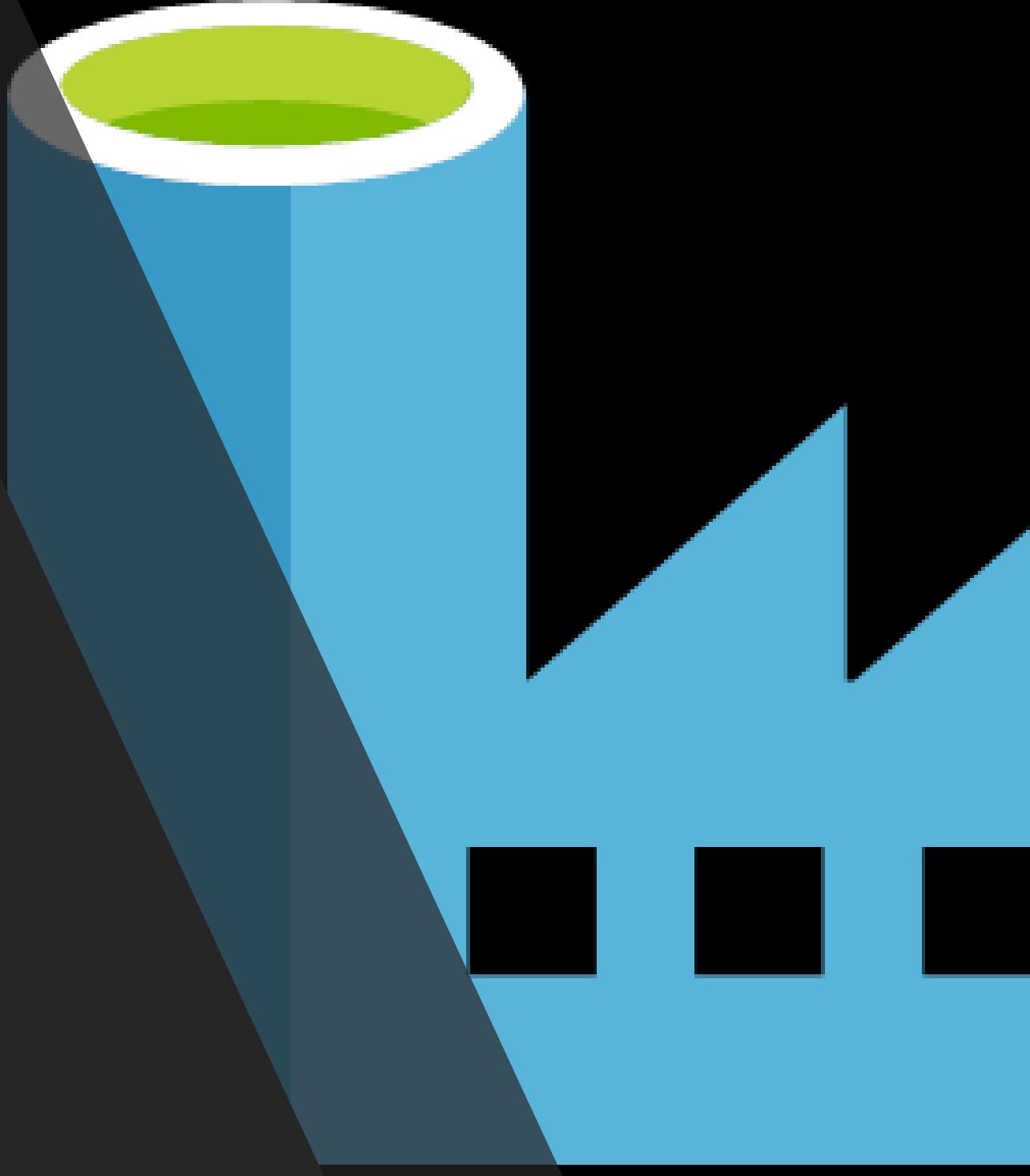
Lift your SQL Server Integration Services (SSIS) packages to Azure



Hybrid and Multi-Cloud Data Integration



Azure Data Factory



What is Azure Data Factory?

Azure Data Factory is a cloud service that *orchestrates*, *manages*, and *monitors* the integration and transformation of structured and unstructured data from on-premises and cloud sources at scale.

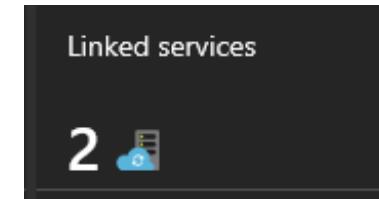
Where

Portal.azure.com

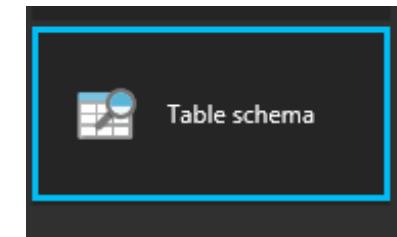
The screenshot shows the Microsoft Azure (Preview) interface for Data Factory. At the top, there's a navigation bar with a menu icon, the text "Microsoft Azure (Preview)", a "Report a bug" button, and a search bar. Below the navigation is a breadcrumb trail: "Home > New > Data Factory". The main title "Data Factory" is displayed with a Microsoft logo. To the left is a blue square icon featuring a white factory building. To the right of the icon, the text "Data Factory" is followed by a "Save for later" button with a heart icon. A "Create" button is also present. Below this section, there are two tabs: "Overview" (which is underlined, indicating it's the active view) and "Plans". A descriptive paragraph explains that Azure Data Factory is a cloud-based service for automating data movement and transformation. It lists several features: composing data storage, movement, and processing services; enhanced HDInsight integration; scheduling data pipelines; new data connectors; integration with Azure Machine Learning and Azure Batch; globally deployed data movement as a service; and creating, editing, and deploying pipelines with a Visual Studio plug-in. At the bottom, there's a "Useful Links" section with links to "Documentation", "Service overview", and "Pricing details".

Three Main Elements

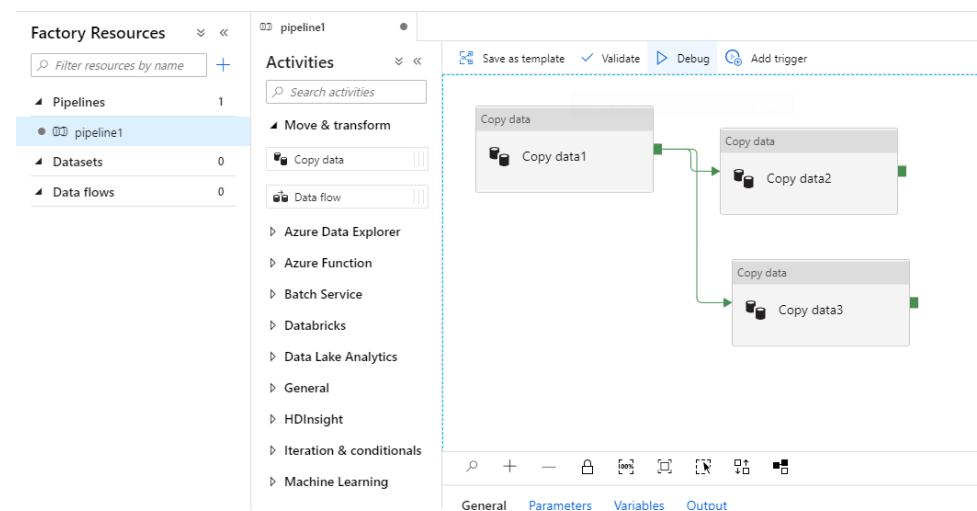
- Linked Services – Think Connection Managers



- Datasets—Schemas Think mapping of Data Flows

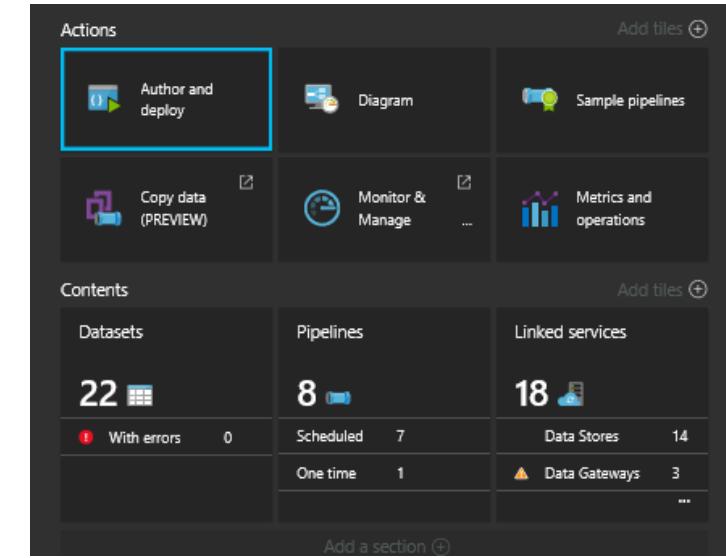


- Pipeline –Think Data Flows
 - Activities –Types of Data Flows

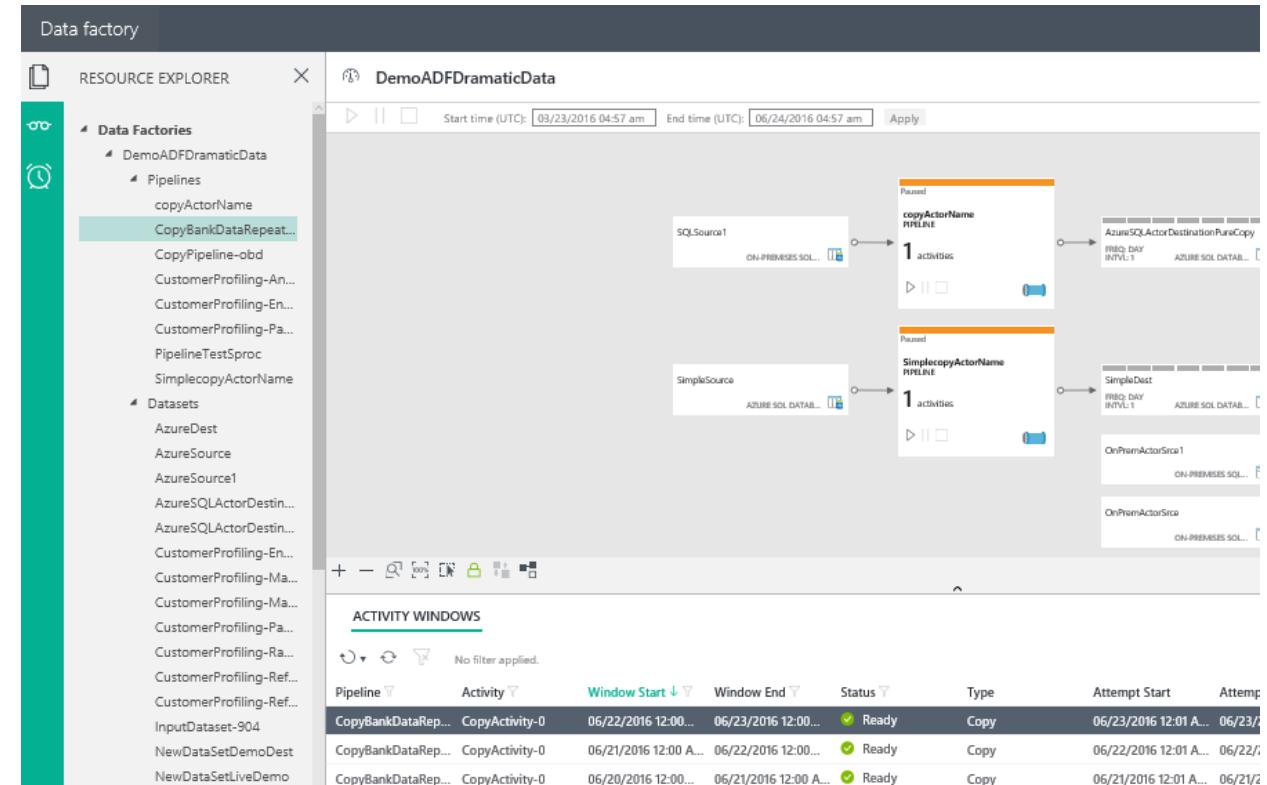
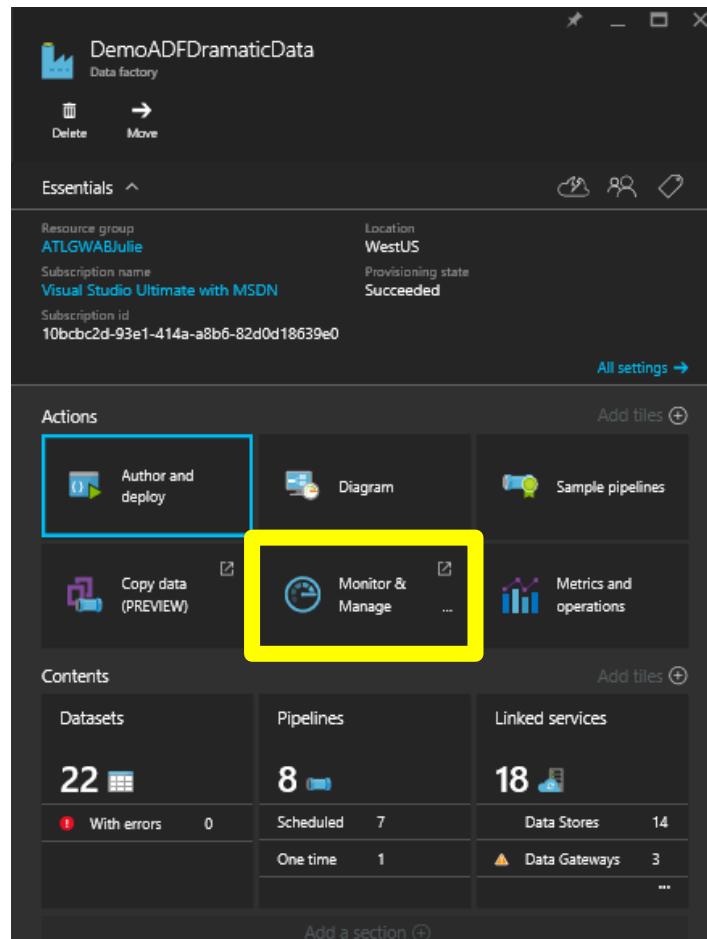


Main Dev Environments

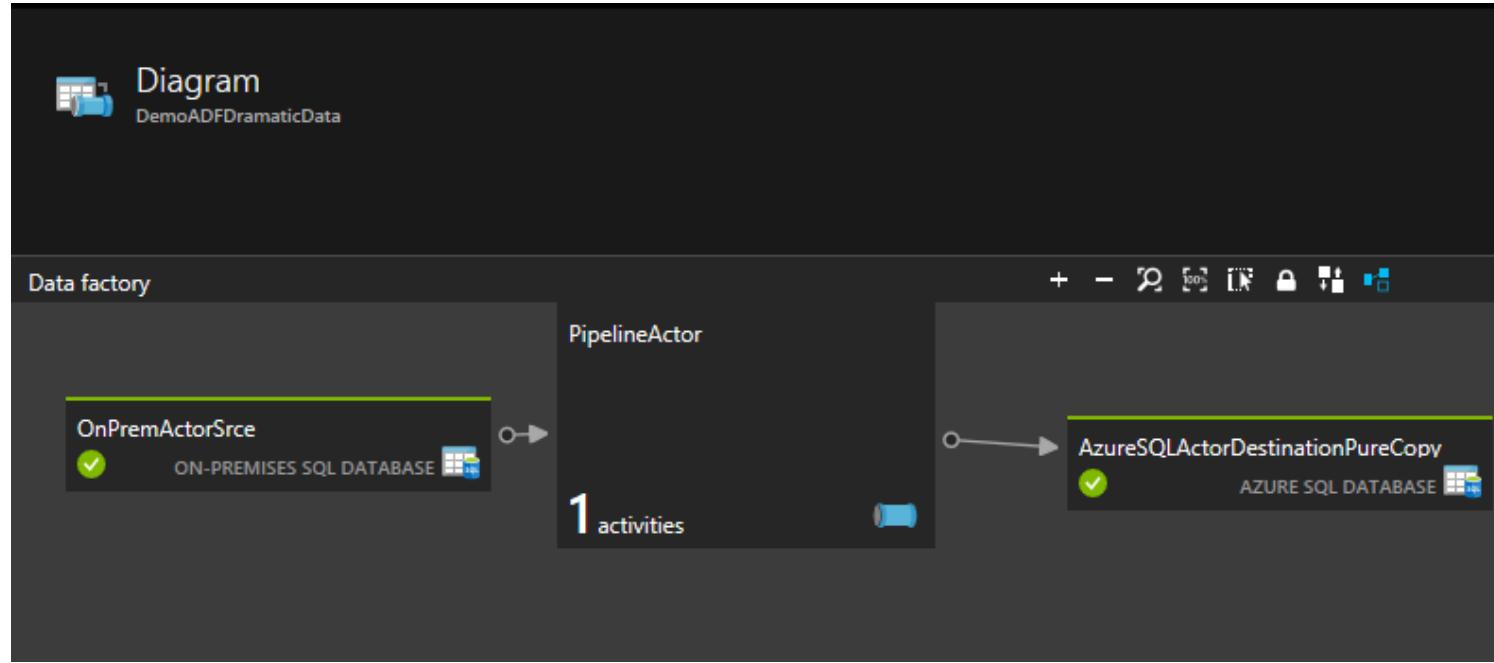
- Portal
- Monitor and Manage
- Visual Studio



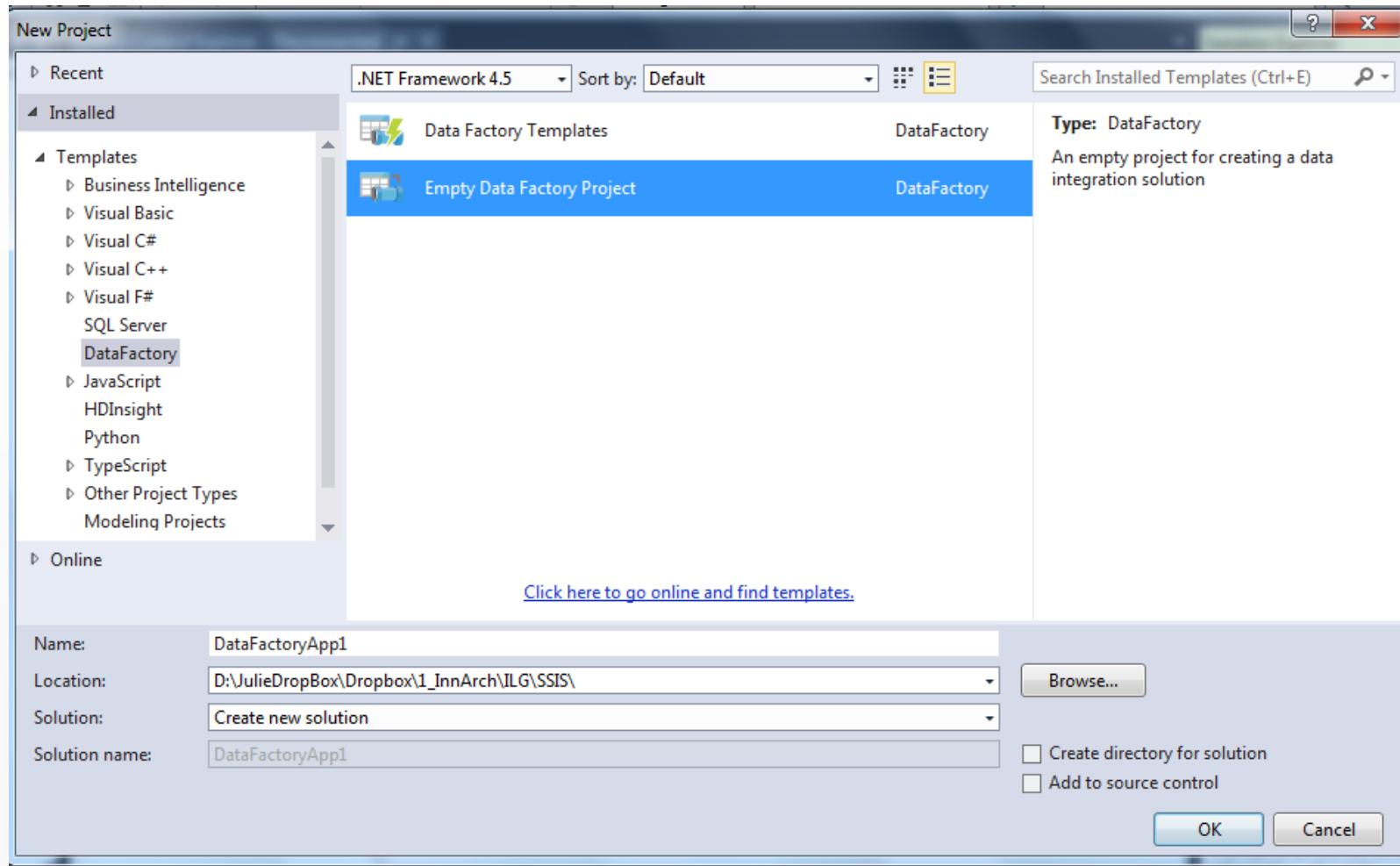
Monitor and Manage



Diagram



Visual Studio Extension



JSON

JSON is built on two structures:

- A collection of **name/value** pairs. In various languages, this is realized as an *object*, record, struct, dictionary, hash table, keyed list, or associative array. { }
- An ordered list of values. In most languages, this is realized as an *array*, vector, list, or sequence. []

JavaScript Object Notation

<http://json.org>

JSON in ADF, Dataset Example

```
{  
    "name": "OnPremActorSrce",  
    "properties": {  
        "published": false,  
        "type": "SqlServerTable",  
        "linkedServiceName": "NorthWindStg",  
        "typeProperties": {  
            "tableName": "Actor"  
        },  
        "availability": {  
            "frequency": "Day",  
            "interval": 1  
        },  
        "policy": {  
            "externalData": {  
                "retryInterval": "00:01:00",  
                "retryTimeout": "00:10:00",  
                "maximumRetry": 3  
            }  
        }  
    }  
}
```

Access all your data

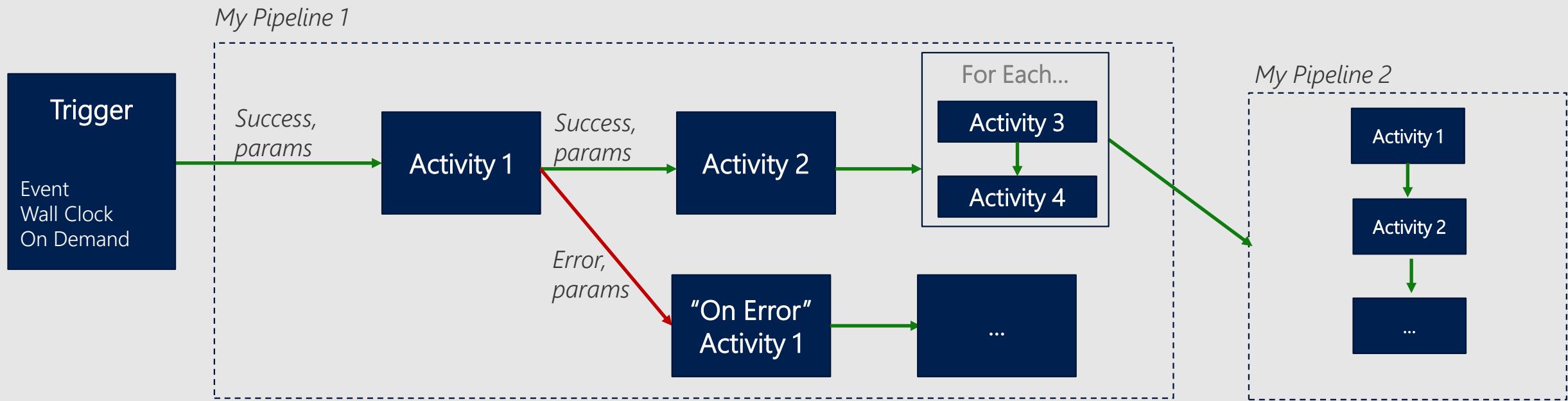
- 90+ connectors & growing
- Azure IR available in 20 regions
- Hybrid connectivity using self-hosted IR: on-prem & VNet

Azure	Database		File Storage	NoSQL	Services and Apps		Generic
Azure Blob Storage	Amazon Redshift	SQL Server	Amazon S3	Couchbase	Dynamics 365	Salesforce	HTTP
Azure Data Lake Store	Oracle	MySQL	File System	Cassandra	Dynamics CRM	Salesforce Service Cloud	OData
Azure SQL DB	Netezza	PostgreSQL	FTP	MongoDB	SAP C4C	ServiceNow	ODBC
Azure SQL DW	SAP BW	SAP HANA	SFTP		Oracle CRM	Hubspot	
Azure Cosmos DB	Google BigQuery	Informix	HDFS		Oracle Service Cloud	Marketo	
Azure DB for MySQL	Sybase	DB2			SAP ECC	Oracle Responsys	
Azure DB for PostgreSQL	Greenplum	MariaDB			Zendesk	Oracle Eloqua	
Azure Search	Microsoft Access	Drill			Zoho CRM	Salesforce ExactTarget	
Azure Table Storage	Hive	Phoenix			Amazon Marketplace	Atlassian Jira	
Azure File Storage	Hbase	Presto			Magento	Concur	
	Impala	Spark			PayPal	QuickBooks Online	
	Vertica				Shopify	Xero	
					GE Historian	Square	

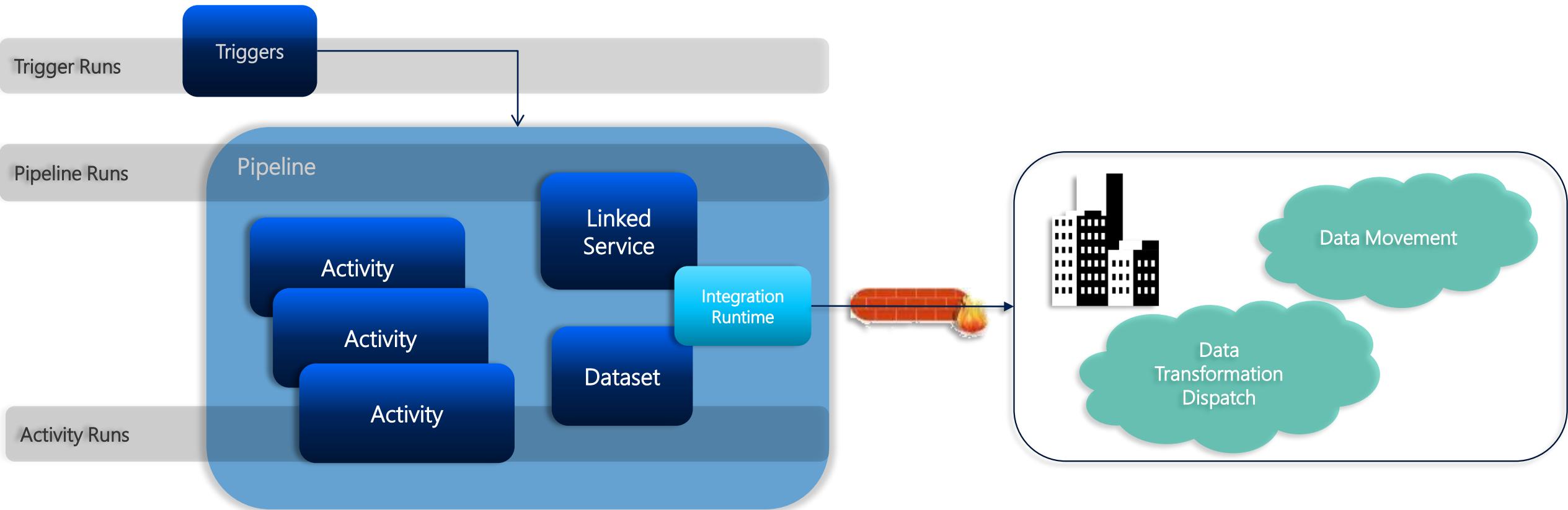
* Supported file formats: CSV, AVRO, ORC, Parquet, JSON

Control Flow Introduced in Azure Data Factory

Coordinate pipeline activities into finite execution steps to enable looping, conditionals and chaining while separating data transformations into individual data flows



Azure Data Factory Updated Flexible Application Model



ADF: Cloud-First Data Integration Objectives

- Consume hybrid disparate data
 - On-prem + Cloud
 - Grow ADF ecosystem of structured, un-structured, semi-structured data connectors
- Calculate and format data for analytics
 - Transform, aggregate, join, normalize
 - Separate data flow (transformation) from control flow (orchestration)
- Address large-scale Big Data requirements
 - Scale-up or Scale-out data movement and transformation
 - Support multiple processing engines
- Operationalize
 - Support flexible scheduling and triggering mechanism for broad range of use cases
 - Manage & monitor multiple pipelines (via Azure Monitor & OMS)
 - Support secure VNET environments
- Lift and Shift SSIS to the Cloud
 - Execute SSIS packages in ADF Integration Runtime

ADF: Cloud-First Data Integration Scenarios

Lift and Shift to the Cloud

- Migrate on-prem DW to Azure
- Lift and shift existing on-prem SSIS packages to cloud
- No changes needed to migrate SSIS packages to Cloud service

DW Modernization

- Modernizing DW arch to reduce cost & scale to needs of big data (volume, variety, etc)
- Flexible wall-clock and triggered event scheduling
- Incremental Data Load

Build Data-Driven, Intelligent SaaS Application

- C#, Python, PowerShell, ARM support

Big Data Analytics

- Customer profiling, Product recommendations, Sentiment Analysis, Churn Analysis, Customized offers, customer usage tracking, customized marketing
- On-demand Spark cluster support

Load your Data Lake

- Separate control-flow to orchestrate complex patterns with branching, looping, conditional processing

ADF V2 Improvements

- Integration Runtimes (IR) replace DMG, provide data movement and activity dispatch on-prem or in the cloud
- Supports resources within virtual networks
- Integration Runtime includes SSIS option to lift & shift SSIS packages to the Cloud
- Separation of “control flow” & “data flow” capabilities for more flexible pipeline management
- Looping, conditionals, dependencies, parameters
- Python SDK
- Built-in Source Control Support
- On-Demand Spark support
- Transform data in Azure Databricks
- Flexible pipeline scheduling with wall-clock, tumbling windows and triggered executions
- Expanded use cases: From primarily time window-oriented pipelines, to trigger-based on-demand for more flexible ETL and data integration orchestrations
- Graphical UI pipeline builder for a code-free experience

New ADF V2 Concepts

Concept	Description	Sample
Control Flow	Orchestration of pipeline activities that includes chaining activities in a sequence, branching, conditional branching based on an expression, parameters that can be defined at the pipeline level and arguments passed while invoking the pipeline on demand or from a trigger. Also includes custom state passing and looping containers, i.e. For-each, Do-Until iterators.	{ "name": "MyForEachActivityName", "type": "ForEach", "typeProperties": { "isSequential": "true", "items": "@pipeline().parameters.mySinkDatasetFolderPathCollection", "activities": [{ "name": "MyCopyActivity", "type": "Copy", "typeProperties": ...] } }
Runs	A Run is an instance of the pipeline execution. Pipeline Runs are typically instantiated by passing the arguments to the parameters defined in the Pipelines. The arguments can be passed manually or properties created by the Triggers.	POST <a href="https://management.azure.com/subscriptions/<subId>/resourceGroups/<resourceGroupName>/providers/Microsoft.DataFactory/factories/<dataFactoryName>/pipelines/<pipelineName>/createRun?api-version=2017-03-01-preview">https://management.azure.com/subscriptions/<subId>/resourceGroups/<resourceGroupName>/providers/Microsoft.DataFactory/factories/<dataFactoryName>/pipelines/<pipelineName>/createRun?api-version=2017-03-01-preview
Activity Logs	Every activity execution in a pipeline generates activity start and activity end logs event	
Integration Runtime	Replaces DMG as a way to move & process data in Azure PaaS Services, self-hosted or on prem or IaaS Works with VNets Enables SSIS package execution	Set-AzureRmDataFactoryV2IntegrationRuntime -Name \$integrationRuntimeName -Type SelfHosted
Scheduling	Flexible Scheduling Wall-clock scheduling Event-based triggers	"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": "<<default UTC>>" }, "schedule": { // optional (advanced scheduling specifics) "hours": "[<<0-24>>]", "weekDays": "[<<Monday-Sunday>>]", "minutes": "[<<0-60>>]", "monthDays": "[<<1-31>>]", "monthlyOccurrences": [{ "day": "<<Monday-Sunday>>", "occurrence": "<<1-5>>" }] } }

New ADF V2 Concepts

Concept	Description	Sample
On-Demand Execution	Instantiate a pipeline by passing arguments as parameters defined in a pipeline and execute from script / REST / API.	Invoke-AzureRmDataFactoryV2PipelineRun -DataFactory \$df -PipelineName "Adfv2QuickStartPipeline" -ParameterFile .\PipelineParameters.json
Parameters	<p>Name-value pairs defined in the pipeline. Arguments for the defined parameters are passed during execution from the run context created by a Trigger or pipeline executed manually. Activities within the pipeline consume the parameter values.</p> <p>A Dataset is a strongly typed parameter and a reusable/referenceable entity. An activity can reference datasets and can consume the properties defined in the Dataset definition</p> <p>A Linked Service is also a strongly typed parameter containing the connection information to either a data store or a compute environment. It is also a reusable/referenceable entity.</p>	Accessing parameters of other activities Using expressions @parameters("{name of parameter}") @activity("{Name of Activity").output.RowsCopied
Incremental Data Loading	Leverage parameters and define your high-water mark for delta copy while moving dimension or reference tables from a relational store either on premises or in the cloud to load the data into the lake	<pre>name": "LookupWaterMarkActivity", "type": "Lookup", "typeProperties": { "source": { "type": "SqlSource", "sqlReaderQuery": "select * from watermarktable" } }</pre>
On-Demand Spark	Support for on-demand HDI Spark clusters, similar to on-demand Hadoop activities in V1	<pre>"type": "HDInsightOnDemand", "typeProperties": { "clusterSize": 2, "clusterType": "spark", "timeToLive": "00:15:00",</pre>
SSIS Runtime	Lift & shift, deploy, manage, monitor SSIS packages in the cloud with SSIS Azure IR Service in Azure Data Factory	Start-AzureRmDataFactoryV2IntegrationRuntime -DataFactoryName \$DataFactoryName -Name
Code-free UI	Build end-to-end data pipeline solutions for ADF without writing code or JSON	

Azure Data Factory

Let's get started



Create pipeline



Create data flow



Create pipeline
from template



Copy data



Configure SSIS
Integration



Set up code
repository

Videos

[View all videos](#)



Microsoft

Microsoft

CustomerChurnFactory

Pipeline 1 X

Activities

Save Run

Search Resources +

Search Activities

AzureML

Custom.NET

Data Prep

Hive

Map Reduce

Pig

Stored Procedure

Spark

Data Flow

Pipelines 4

CustomerChurnPipeline

MarketingCampaignPipeline

ProductUsagePipeline

ProductUsagePipeline2

Data Flows 3

CustomerChurned

CustomerInfo

CallDataRecords

Datasets

Linked Services

Integration Runtimes

Triggers

Repository Settings

Enter Git repository information to be associated with your Data Factory: makadfv2c

Repository type:

Visual Studio Team Services Git

Visual Studio Team Services Account:

markkromer

Project Name:

MarkADFV2

Git repository name:

Create new Use existing

makadfv2c

Built-in source control support

```
graph LR; A[CallDataRecords] --> C[CustomerChurned]; B[ProcessCallLogs] --> C; D[CustomerInfo] --> C; C --> E[SendEmail]
```

Output Log

Run Details

Errors

Configuration

7/12/2017 8am UTC: Started CallDataRecords & CustomerInfo to CustomerChurned activity.

Search output list

CustomerChurnPipeline < CallDataRecords

Save ✓ Validate

Source

- File
- Azure Blob Storage
- Amazon S3

Sink

- File
- Azure Blob Storage
- Amazon S3

Settings

General Mapping

Mapping Options

- Automatic
- Auto Map

Source fields: 25 / 25 mapped

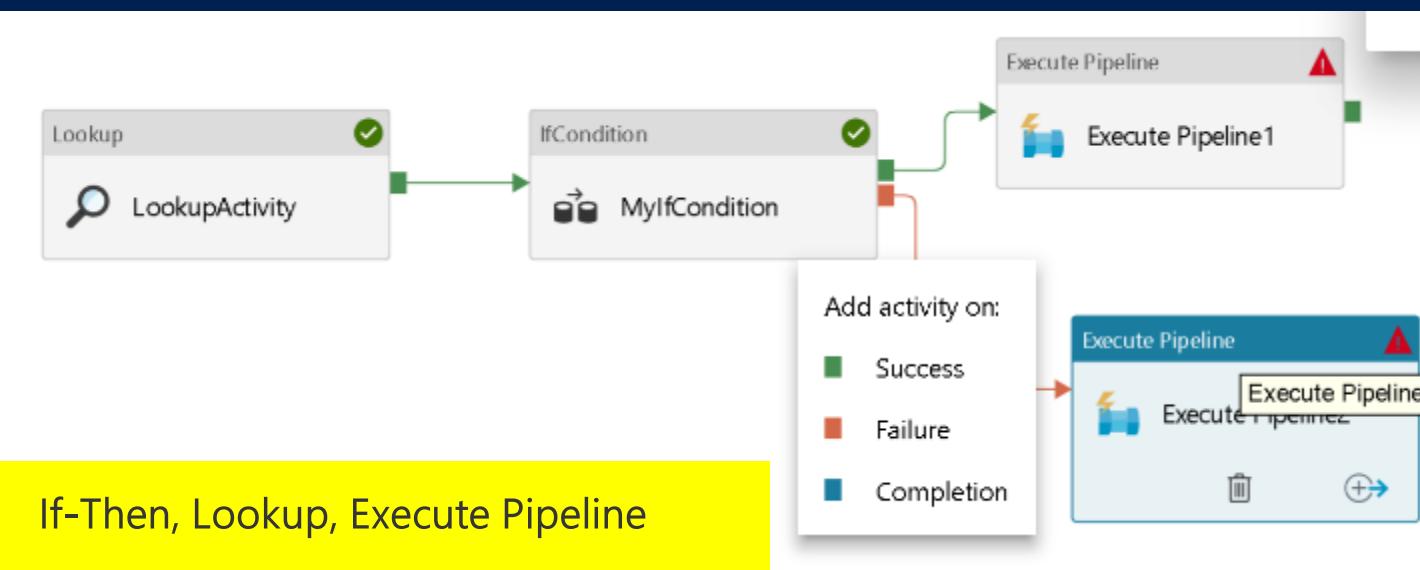
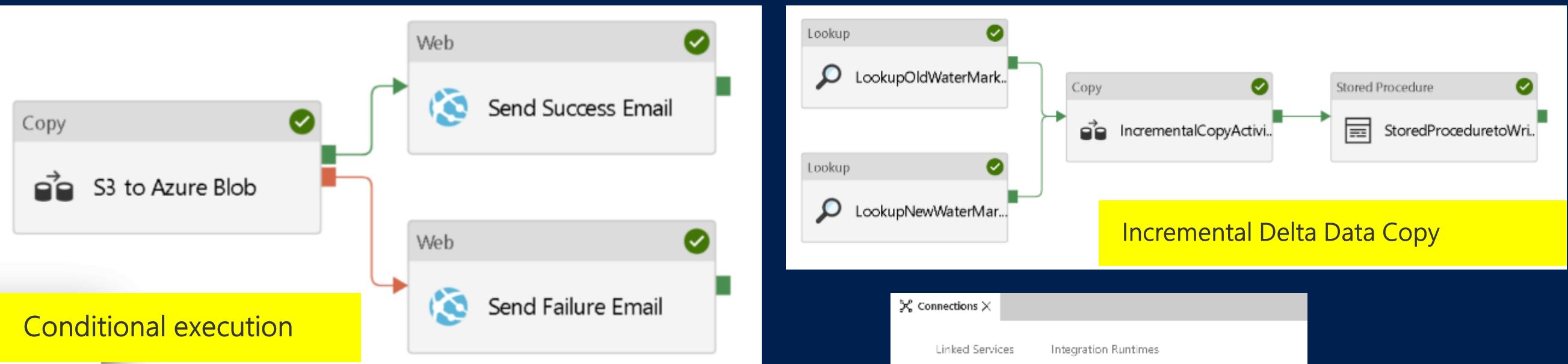
Sink fields: 25 / 25 mapped

FIELD	TYPE	FIELD	TYPE
Age	int	Age	int
AnnualIncome	BigInt	AnnualIncome	BigInt
CallDropRate	Double	CallDropRate	Double
CallFailureRate	Double	CallFailureRate	Double
CallingNum	String	CallingNum	String
CustomerID	Int	CustomerID	Int
CustomerSuspended	String	CustomerSuspended	String
Education	String	Education	String
Gender	String	Gender	String
HomeOwner	String	HomeOwner	String
MaritalStatus	String	MaritalStatus	String
MonthlyBilledAmount	Int	MonthlyBilledAmount	Int
NoAdditionalLines	Int	NoAdditionalLines	Int

Diagram:

```
graph LR; A[Amazon S3] --> B[Azure Blob Storage];
```

The screenshot shows a data pipeline configuration in a software interface. On the left, there are two sections: 'Source' and 'Sink'. The 'Source' section lists 'File', 'Azure Blob Storage', and 'Amazon S3'. The 'Sink' section lists 'File', 'Azure Blob Storage', and 'Amazon S3'. In the center, there is a mapping interface for a pipeline named 'CallDataRecords'. It shows a flow from 'Amazon S3' to 'Azure Blob Storage'. The 'Mapping' tab is selected, showing an 'Automatic' mapping option and an 'Auto Map' button. Below this, a table maps source fields to sink fields, with 25 fields mapped on both sides. The table has columns for 'FIELD' and 'TYPE' on the source side, and 'FIELD' and 'TYPE' on the sink side. The mapping is mostly one-to-one, with some field names being identical between source and sink.



Connections X		
Linked Services		Integration Runtimes
+ New		
Name	Actions	Type
AzureSQLDatabaseLinkedService	Edit Delete	Azure SQL Database
AzureSqlLinkedService	Edit Delete	Azure SQL Database
AzureStorageLinkedService	Edit Delete	Azure Storage
AzureBatchLinkedService	Edit Delete	Azure Batch
AzureStorage1	Edit Delete	Azure Storage
129bb8d5-5f6-4847-be67-a49b4f438771	Edit Delete	Amazon S3
Sa680b8c-40b0-46d9-b5e4-6b4359ae3b	Edit Delete	Azure Storage
SQLDBLS	Edit Delete	Azure SQL Database

Connection Managers



Refresh

Operationalize – Monitor your data pipelines

Custom Range 11/01/2017 9:00 AM - 12/23/2017 9:00 AM ▾

Time Zone (UTC-08:00) Los Angeles ▾

All Succeeded In Progress Failed

Pipeline Name	Actions	Run Start	Duration	Triggered By	Status	Parameters	Error	RunID
LookupPipeline		12/04/2017, 4:59:33 PM	00:00:49	Manual trigger	Succeeded...			8fd7c2e1-440c-45d7-aff0-21dc8552c207
LookupPipeline		12/04/2017, 4:56:24 PM	00:00:53	Manual trigger	Succeeded...			ecd6bec4-b7b8-47b0-aaac-c32ba199a5ff
LookupPipeline		12/04/2017, 4:53:34 PM	00:00:33	Manual trigger	Failed			c272ebf7-f784-4d8c-9b82-c5e10f06250b
LookupPipeline		12/04/2017, 4:20:25 PM	00:00:29	Manual trigger	Failed			6018a772-81c8-4ec0-ab18-24424c25195c
LookupPipeline		12/04/2017, 4:10:50 PM	00:00:33	Manual trigger	Failed			06c7db30-d77b-47d2-917a-935244f1c2c5
pipeline4_7e0990af-c...		11/27/2017, 11:12:27 AM	00:00:05	Manual trigger	Failed			c3aa1144-ebdc-448b-a1b8-9f1b5d65cb40
MyWebActivityPipeline		11/26/2017, 9:37:02 PM	00:00:10	Manual trigger	Failed			23c5e44c-a191-4a1f-ac21-ff276b7da43b
batchpipe		11/17/2017, 3:24:19 PM	00:00:38	Manual trigger	Succeeded...			b2ef549a-b5cf-4786-9ffd-f9f71948c6d9
batchpipe		11/17/2017, 3:20:12 PM	00:00:00	Manual trigger	Failed			a3dec17f-a370-4e8b-9a3e-285483680fde
ifconditionpipeline2		11/16/2017, 6:00:20 PM	00:00:04	Manual trigger	Failed			07b7812d-0af0-4f67-a0b8-ec64ddd38fc9
ifconditionpipeline		11/16/2017, 6:00:11 PM	00:00:05	Manual trigger	Failed			8ac7565d-eefd-4831-92c5-33bfebd9c260
ifconditionpipeline		11/15/2017, 4:58:45 PM	00:00:07	Manual trigger	Succeeded...			dcff3e04-6158-40e7-b21d-70d417ae646f
ifconditionpipeline		11/15/2017, 4:52:36 PM	00:00:06	Manual trigger	Failed			f1d615ca-f4d9-47bf-930b-0bc47dbb3430
pipeline3_9a1f3c55-e...		11/10/2017, 2:52:13 PM	00:00:05	Manual trigger	Failed			052056da-9cd6-48c8-8441-4d11feb911a4
IncrementalCopyPipeli...		11/01/2017, 2:02:16 PM	00:01:36	Manual trigger	Succeeded...			f176d4e0-1535-4aec-8eca-25dc7a4b0e80
IncrementalCopyPipeli...		11/01/2017, 1:56:06 PM	00:01:13	Manual trigger	Succeeded...			1f3d9bc2-9b30-4245-9489-786ca77796ca
IncrementalCopyPipeli...		11/01/2017, 1:49:30 PM	00:00:36	Manual trigger	Failed			7824bd16-9e72-4409-ae80-238faf861a5c

1 Properties
One time copy**2 Source**

- Connection
- Dataset

3 Destination**4 Settings**
Fault tolerance**5 Summary****6 Deployment**

Source data store

Specify the source data store for the copy task. You can use an existing data store connection or specify a new data store. Click [HERE](#) to suggest new copy sources or give comments.

Easy-to-use Wizard for Copying Data at Scale

FROM EXISTING CONNECTIONS

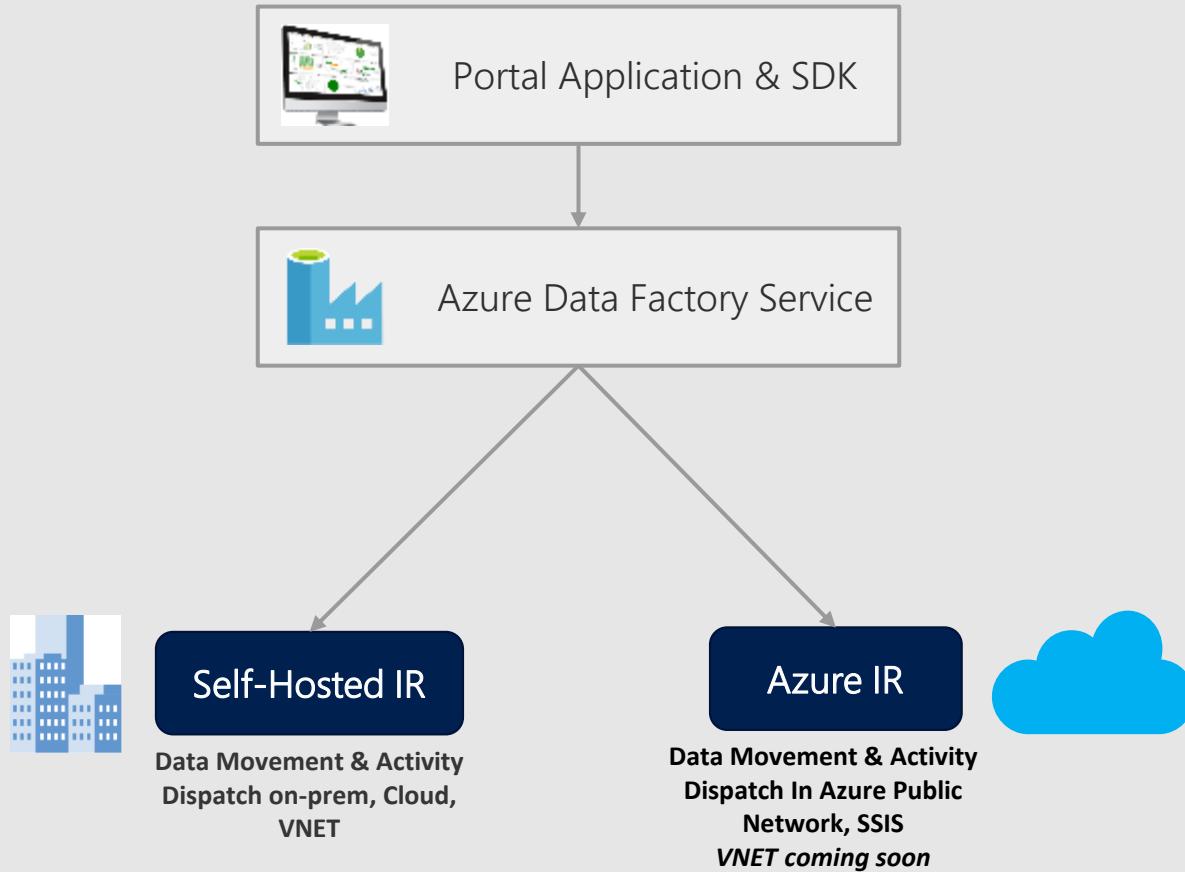
CONNECT TO A DATA STORE

					
Amazon Redshift	Amazon S3	Azure Blob Storage	Azure Cosmos DB	Azure Data Lake Store	Azure Database for MySQL
					 cassandra
Azure Database for PostgreSQL	Azure File Storage	Azure SQL Data Warehouse	Azure SQL Database	Azure Table Storage	Cassandra
DB2					

Previous

Next

ADF Integration Runtime (IR)



- ADF compute environment with multiple capabilities:
 - Activity dispatch & monitoring
 - Data movement
 - SSIS package execution
- To integrate data flow and control flow across the enterprises' hybrid cloud, customer can instantiate multiple IR instances for different network environments:
 - On premises (similar to DMG in ADF V1)
 - In public cloud
 - Inside VNet
- Bring a consistent provision and monitoring experience across the network environments

←→ Command & Control

↔ Data Flow



UX & SDK

Authoring | Monitoring/Mgmt

Azure Data Factory Service

Scheduling | Orchestration | Monitoring

On Premises Apps & Data



cloudera



ORACLE®

Cloud Apps, Svcs & Data



Microsoft Azure



workday



Adobe

↔ Command & Control

↔ Data Flow



UX & SDK

Authoring | Monitoring/Mgmt

Azure Data Factory Service

Scheduling | Orchestration | Monitoring

Azure Cloud

PaaS Cloud Host

Integration Runtime

On Premises Apps & Data



Cloud Apps, Svcs & Data



Command & Control

Data Flow



UX & SDK

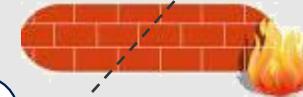
Authoring | Monitoring/Mgmt

Azure Data Factory Service

Scheduling | Orchestration | Monitoring

Installable Agent

Integration Runtime



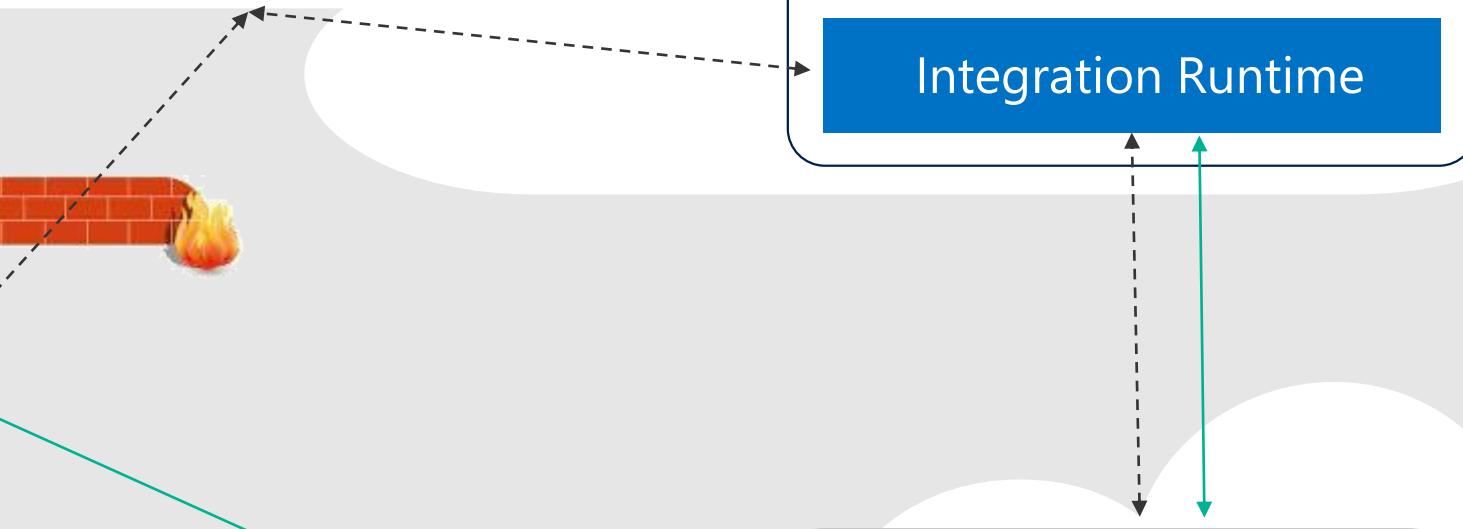
On Premises Apps & Data



Azure Cloud

PaaS Cloud Host

Integration Runtime



Cloud Apps, Svcs & Data



Command & Control

Data Flow



UX & SDK

Authoring | Monitoring/Mgmt

Azure Data Factory Service

Scheduling | Orchestration | Monitoring

Azure Cloud

PaaS Cloud Host

Runtime

Integration Runtime

- Activity Dispatch/Monitor (spark, copy, ml, etc)
- Data Movement
- SSIS Package Execution

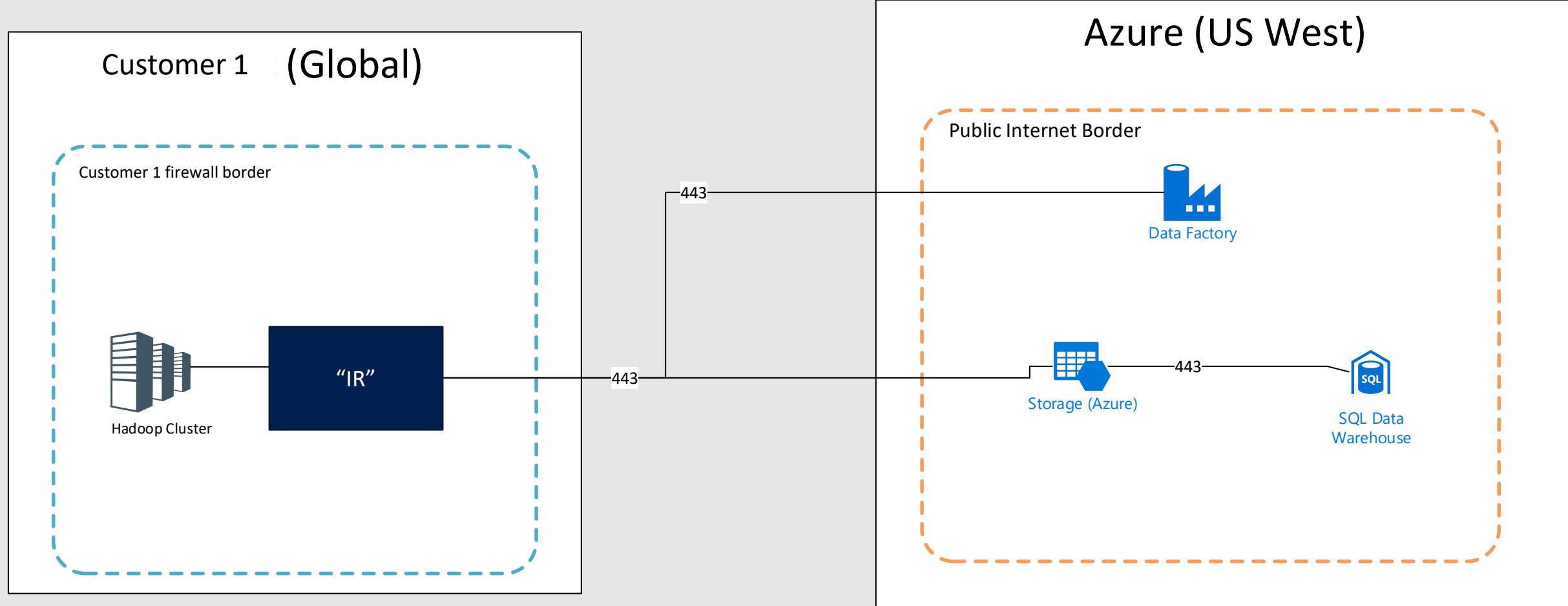
On Premises Apps & Data



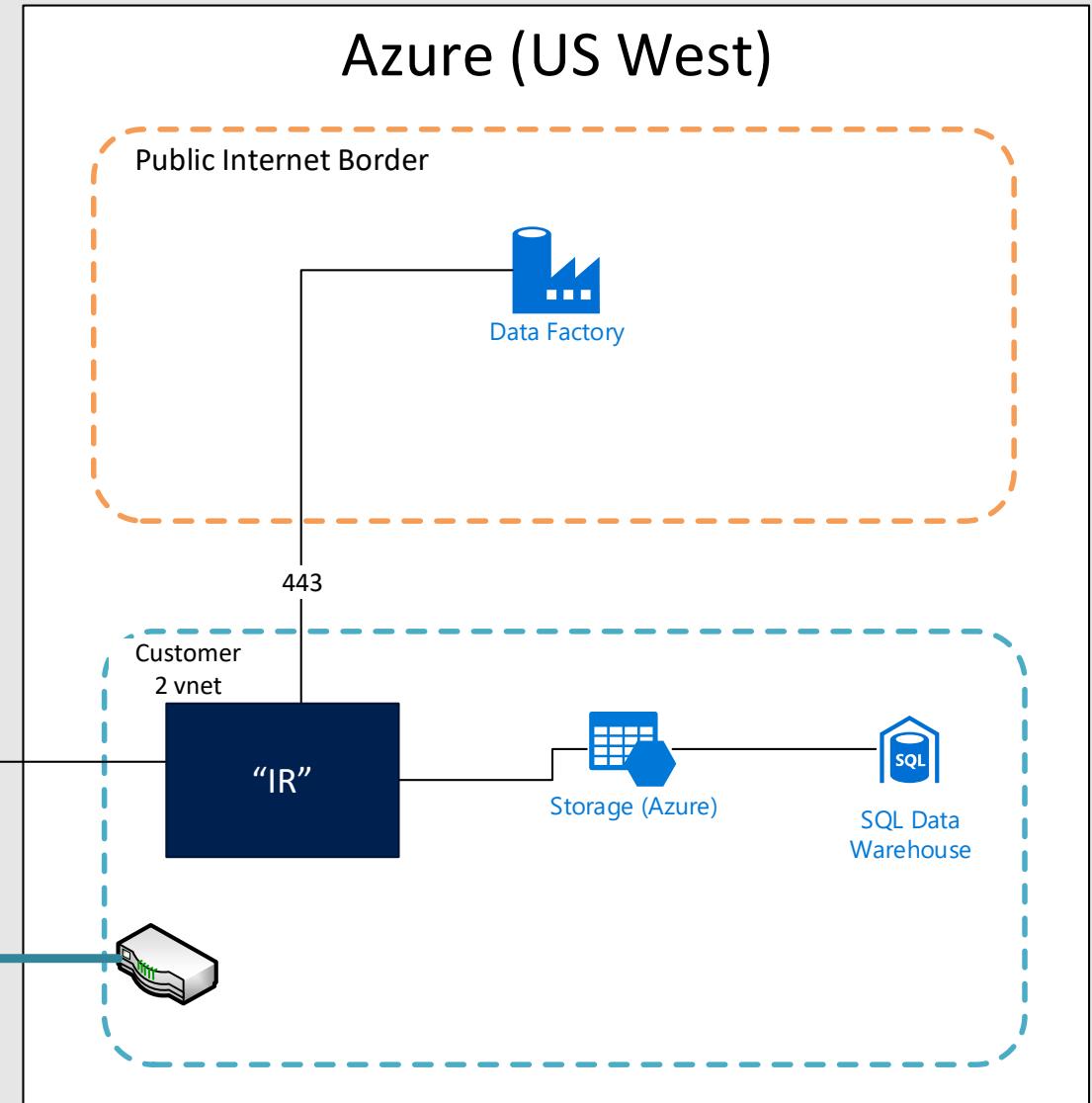
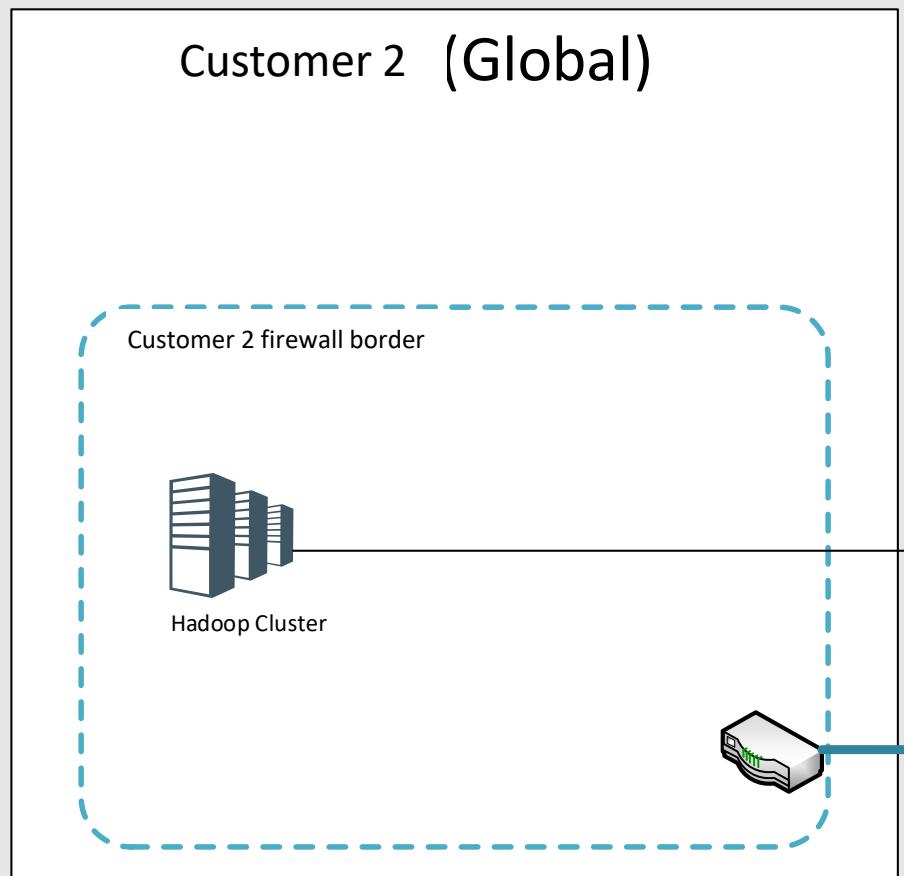
Cloud Apps, Svcs & Data



Azure Data Factory “Integration Runtime” deployed on premises for transformation and then moved to cloud



Azure Data Factory “Integration Runtime” deployed inside VNet



Integration Runtime Setup

X

Integration Runtime is the native compute used by ADF to execute or dispatch activities. Choose what integration runtime to create based on required capabilities.



Perform data movement and dispatch activities to external computes.



Lift-and-shift existing SSIS packages to execute in Azure.

Integration Runtime Setup

Choose the network environment of the data source/destination or external compute the integration runtime will connect to for data movement or dispatch activities:



Public Network ⓘ



Private Network ⓘ

Integration Runtime Setup

X

ADF manages the integration runtime in Azure to connect to required data source/destination or external compute in public network. The compute resource is elastic allocated based on performance requirement of activities.

Name *

ⓘ

integrationRuntime3

Description

Enter description here...

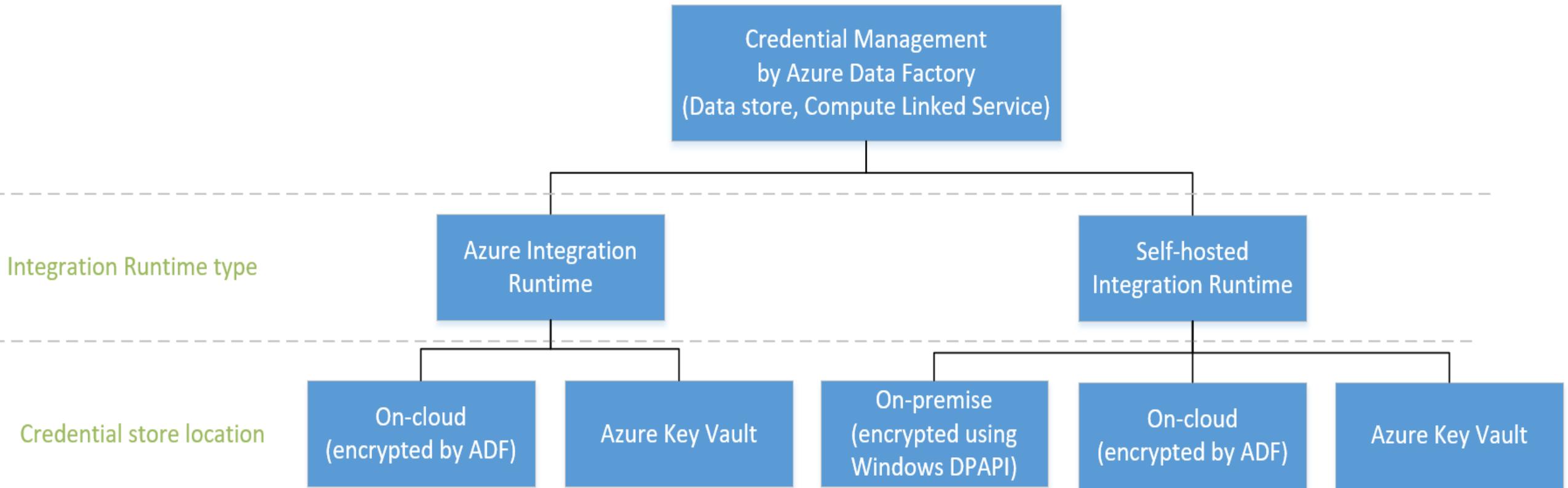
Type

Azure

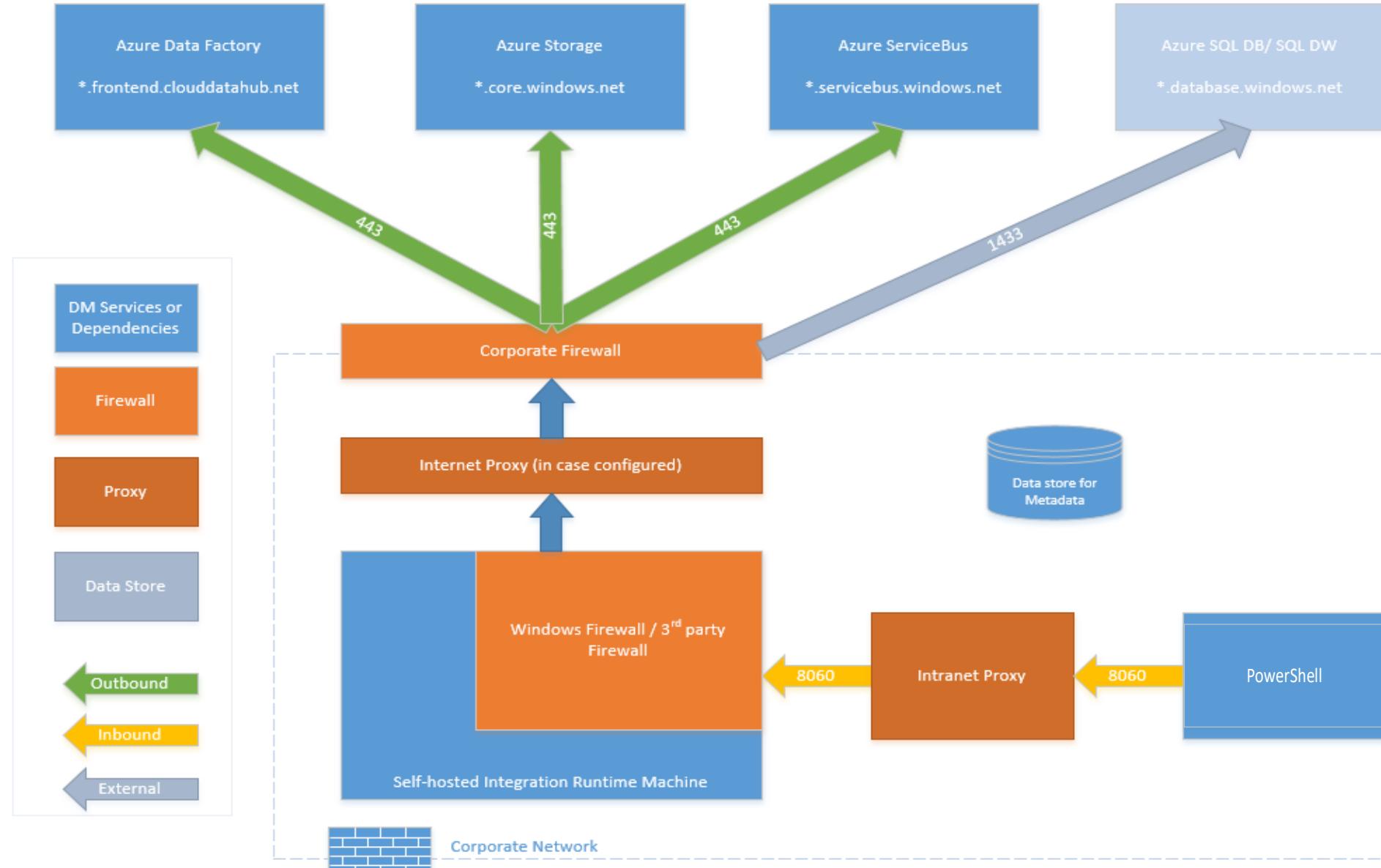
Region *

Central US

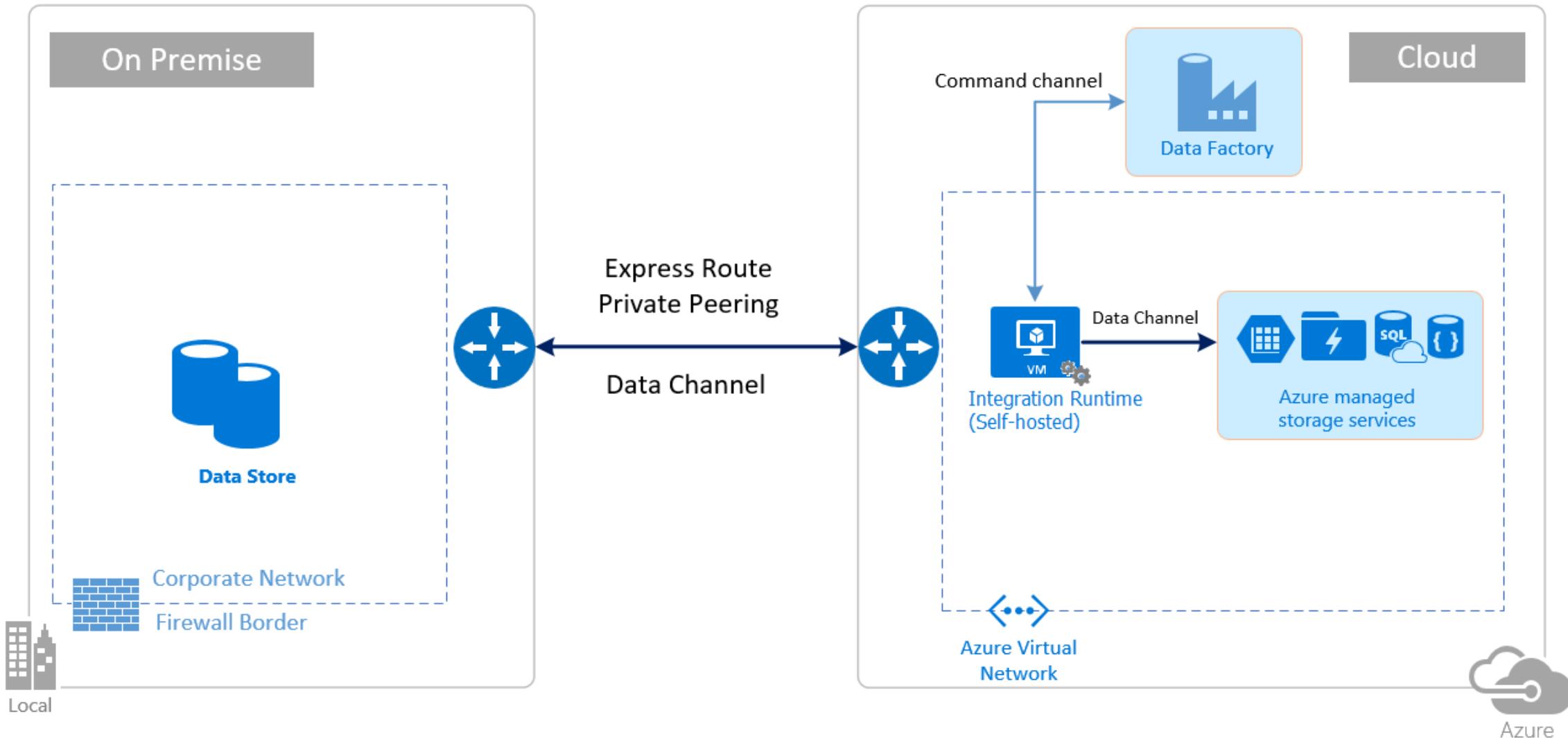
Credential Management (Linked service)



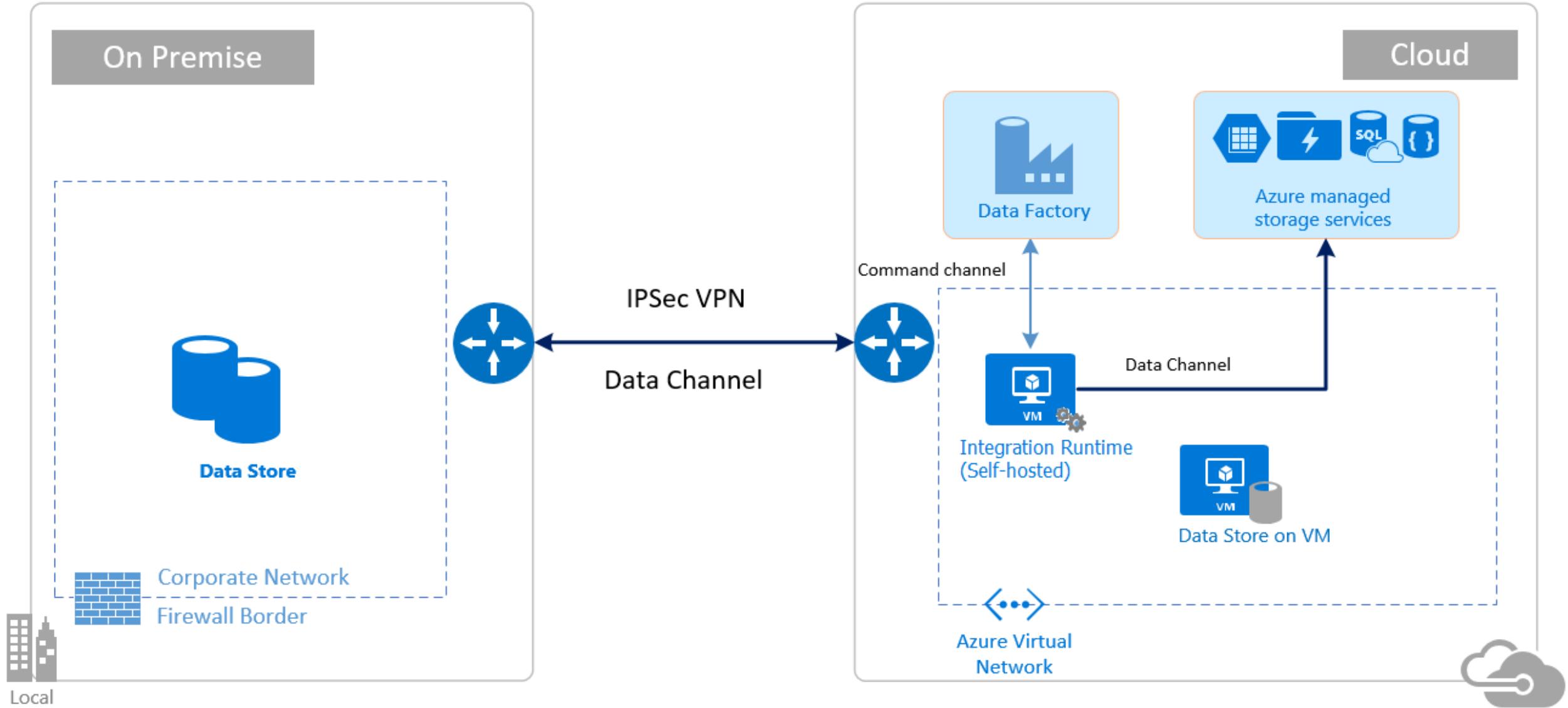
Self-hosted IR – Firewall Requirements



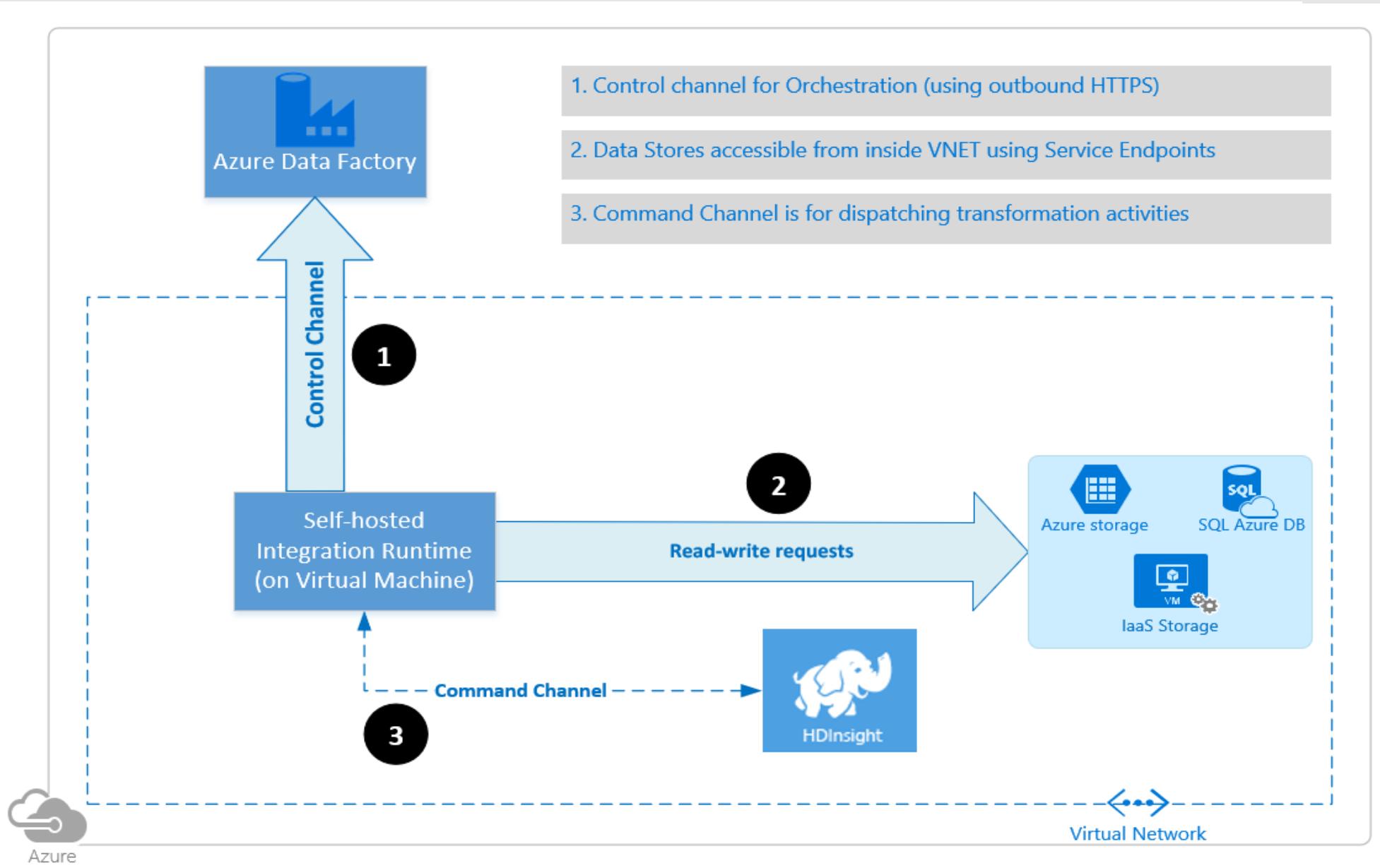
Network Topology (with ExpressRoute)



Network Topology (with VPN)



VNet



ADF Certifications

[HIPAA/HITECH](#)

[ISO/IEC 27001](#)

[ISO/IEC 27018](#)

[CSA STAR](#)

Customer Insights

- SSIS is a traditional ETL tool that comes bundled with SQL Server on-premises
 - Has been around for more than 10 years
 - Some customers have started to lift & shift their ETL workloads to the cloud to reduce their on-prem infra, but found managing Infrastructure as a Service (IaaS)/VMs challenging

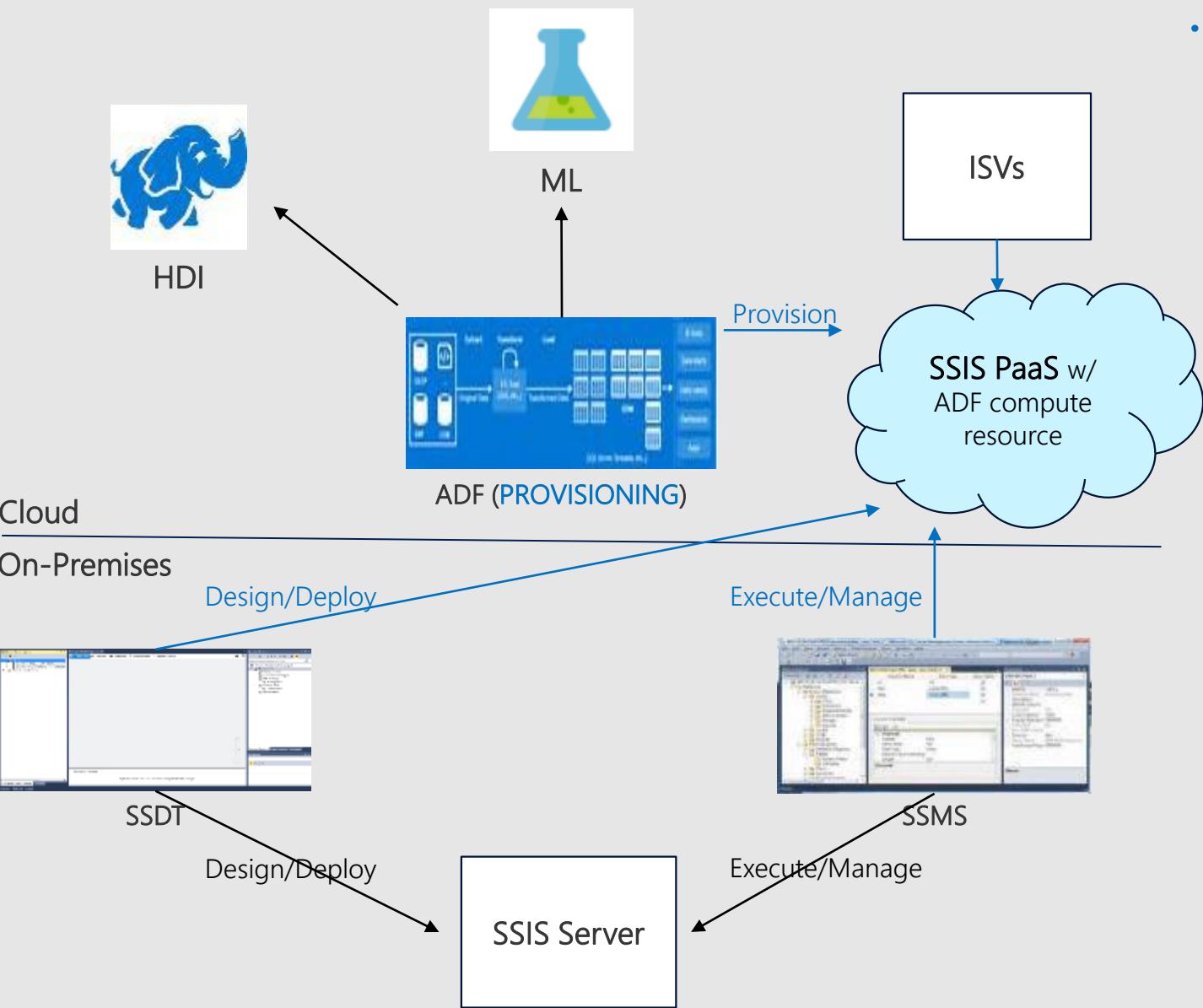
Customer Insights

- SSIS is a traditional ETL tool that comes bundled with SQL Server on premises
 - Has been around for more than 10 years
 - Some customers have started to lift & shift their ETL workloads to the cloud to reduce their on-prem infra, but found managing Infrastructure as a Service (IaaS)/VMs challenging
- Azure Data Factory (ADF) is a modern ELT tool that moves/copies data and dispatches transformations for Big Data Analytics in the cloud
 - Some gaps in ELT workflows can be filled w/ code-free authoring of transformations/built-in tasks from SSIS
 - Some customers have started to combine ADF with SSIS on IaaS/VMs, but found managing IaaS/VMs challenging

Customer Insights

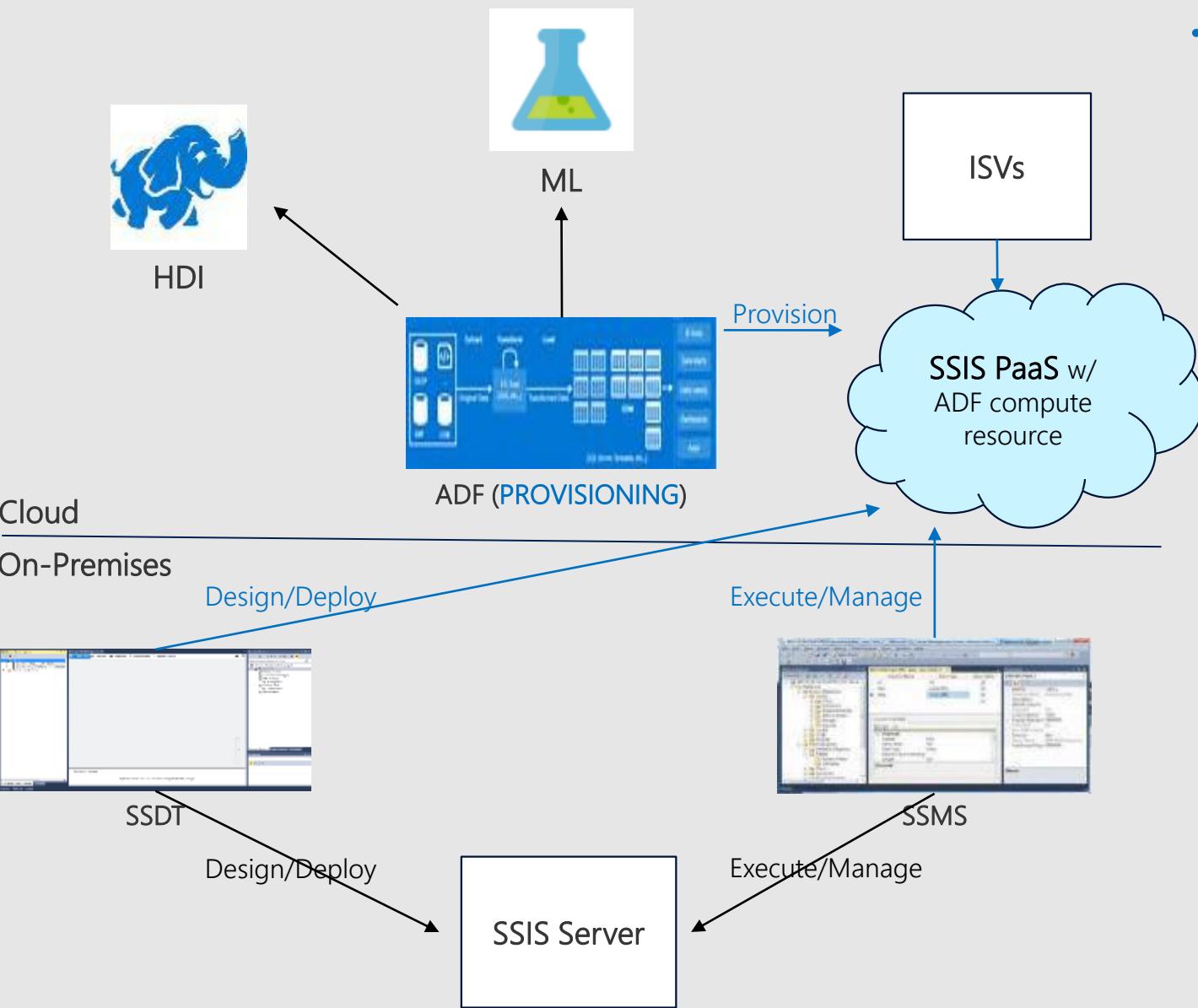
- SSIS is a traditional ETL tool that comes bundled with SQL Server on premises
 - Has been around for more than 10 years
 - Some customers have started to lift & shift their ETL workloads to the cloud to reduce their on-prem infra, but found managing Infrastructure as a Service (IaaS)/VMs challenging
- Azure Data Factory (ADF) is a modern ELT tool that moves/copies data and dispatches transformations for Big Data Analytics in the cloud
 - Some gaps in ELT workflows can be filled w/ code-free authoring of transformations/built-in tasks from SSIS
 - Some customers have started to combine ADF with SSIS on IaaS/VMs, but found managing IaaS/VMs challenging
- Evolution of a cloud-first product: SSIS on premises -> IaaS -> PaaS
 - The stage is set for SSIS PaaS...

Microsoft ETL/ELT Services in Azure



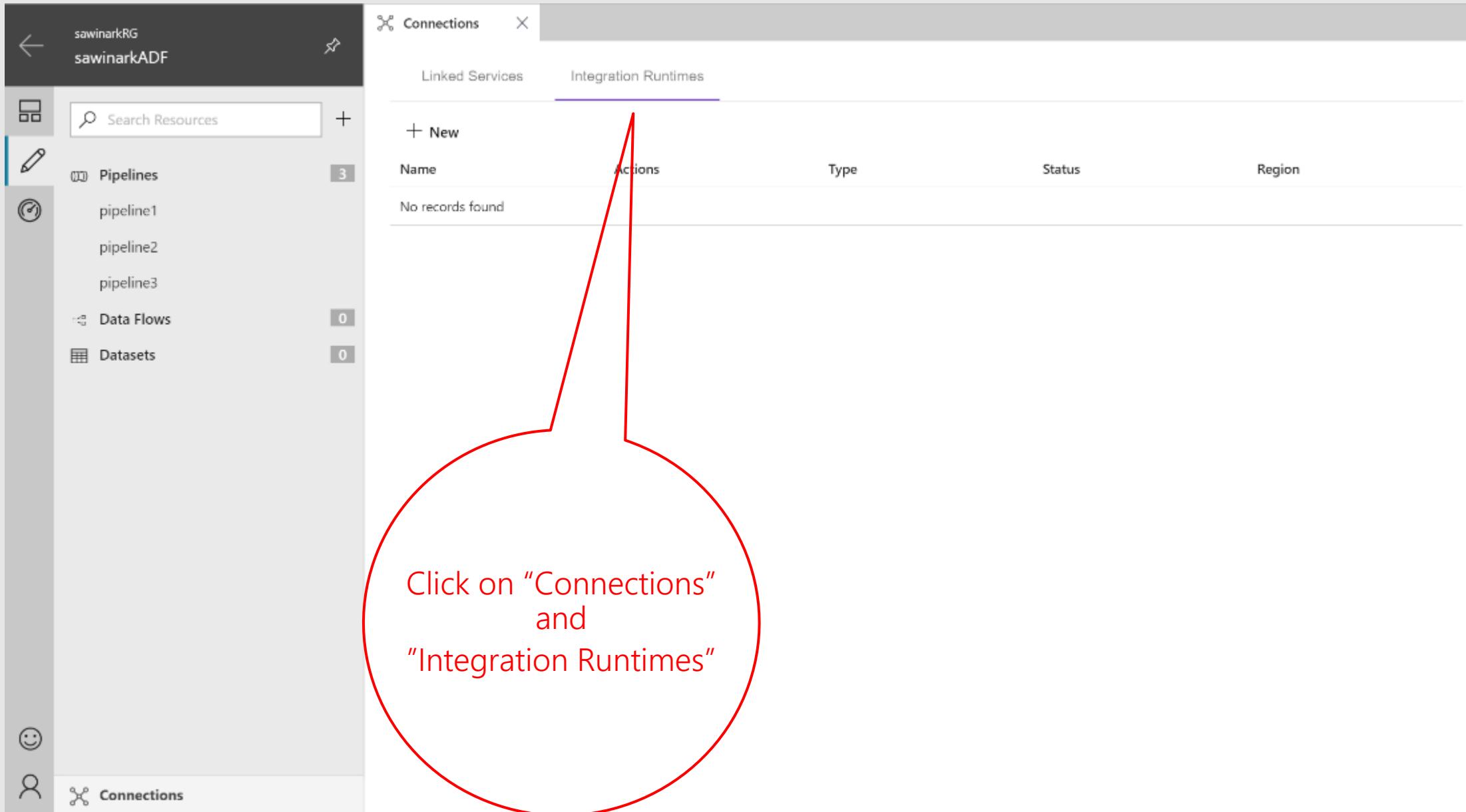
- Introducing Azure-SSIS IR: Managed cluster of Azure VMs (nodes) dedicated to run your SSIS packages and no other activities
 - You can scale it up/out by specifying the node size /number of nodes in the cluster
 - You can bring your own Azure SQL Database (DB)/Managed Instance (MI) server to host the catalog of SSIS projects/packages (**SSISDB**) that will be attached to it
 - You can join it to a Virtual Network (VNet) that is connected to your on-prem network to enable on-prem data access
 - Once provisioned, you can enter your Azure SQL DB/MI server endpoint on SSDT/SSMS to deploy SSIS projects/packages and configure/execute them just like using SSIS on premises

Microsoft ETL/ELT Services in Azure



- Customer cohorts for Phase 1:
 1. **"SQL Migrators"**
These are SSIS customers who want to retire their on-prem SQL Servers and migrate all apps + data ("complete/full lift & shift") into Azure SQL MI – For them, SSISDB can be hosted by Azure SQL MI inside VNet
 2. **"ETL Cost Cutters"**
These are SSIS customers who want to lower their operational costs and gain High Availability (HA)/scalability for just their ETL workloads w/o managing their own infra ("partial lift & shift") – For them, SSISDB can be hosted by Azure SQL DB in the public network

Provision SSIS IR in ADF



Provision SSIS IR in ADF

Click on "+ New" and "Lift-and-Shift..."

Cancel Next →

Connections

Linked Services Integration Runtimes

+ New

Name	Actions	Type
No records found		

Integration Runtime Setup

IR is the native compute used by ADF to execute or dispatch activities. Choose what IR to create based on required capabilities.

 Perform data movement and dispatch activities to external computers.

 Lift-and-shift existing SSIS packages to execute in Azure.

Provision SSIS IR in ADF

The screenshot shows the Azure Data Factory (ADF) interface. On the left, the 'Connections' blade is open, displaying 'Linked Services' and 'Integration Runtimes'. The 'Integration Runtimes' tab is selected, showing a table with columns 'Name', 'Actions', and 'Type'. A message 'No records found' is displayed below the table. On the right, a modal dialog titled 'Integration Runtime Setup' is open, containing the 'General Settings' section. The 'Name' field is set to 'DemoAzureSSIS'. The 'Description' field contains 'Azure-SSIS IR for demo'. Under 'Region', 'East US' is selected. Under 'Node Size', 'Standard_D3_v2 (4 Core(s), 14336 MB)' is chosen. The 'Node Number' slider is set to 4, with a yellow callout bubble highlighting the value. Under 'SKU', 'Standard' is selected. At the bottom of the dialog are 'Cancel', 'Previous', and 'Next' buttons.

Connections

Integration Runtimes

+ New

Name	Actions	Type
No records found		

Integration Runtime Setup

General Settings

Name *

DemoAzureSSIS

Description

Azure-SSIS IR for demo

Region *

East US

Node Size *

Standard_D3_v2 (4 Core(s), 14336 MB)

Node Number *

4

SKU *

Standard

Cancel

← Previous

Next →

Provision SSIS IR in ADF

The screenshot shows the Azure Data Factory (ADF) interface. On the left, there's a navigation sidebar with icons for Home, Pipelines, Data Flows, Datasets, and Connections. The main area is titled "Connections" and shows two tabs: "Linked Services" and "Integration Runtimes". The "Integration Runtimes" tab is selected, displaying a table with one row:

Name	Actions	Type	Status	Region
DemoAzureSSIS	More Edit Delete	Managed Dedicated	Starting	East US

The "Pipelines" section on the left lists three pipelines: pipeline1, pipeline2, and pipeline3.

Provision SSIS IR in ADF

The screenshot shows the Azure Data Factory (ADF) portal interface. On the left, there's a sidebar with icons for Pipelines (3), Data Flows (0), and Datasets (0). The main area is titled 'Connections' and has tabs for 'Linked Services' and 'Integration Runtimes'. Under 'Integration Runtimes', there's a table with one row:

Name	Actions	Type	Status	Region
DemoAzureSSIS	⋮ ⎯ ⏚	Managed Dedicated	Started	East US

A large red circle highlights the 'Started' status of the 'DemoAzureSSIS' runtime. A red callout bubble points to this circle with the text: 'Once your Azure-SSIS IR is started, you can deploy SSIS packages to execute on it and you can stop it as you see fit'.

Provisioning via PSH

```
##### SSIS in ADFv2 specifications (please refer to SSIS in ADFv2 Public Preview documentation for field descriptions) #####
# If your inputs contain PSH special characters, e.g. "$", please precede it with the escape character "`" like `$`.
# ADFv2 info
$SubscriptionName = "[your Azure subscription name]"
$ResourceGroupName = "[your Azure resource group name]"
$DataFactoryName = "[your ADFv2 name]"
$DataFactoryLocation = "EastUS" # In Public Preview, only EastUS|EastUS2 are supported for now

# Azure-SSIS Integration Runtime info - This is ADFv2 compute resource for running SSIS packages
$AzureSSISName = "[your Azure-SSIS Integration Runtime name]"
$AzureSSISDescription = "This is my Azure-SSIS Integration Runtime"
$AzureSSISLocation = "EastUS" # In Public Preview, only EastUS|NorthEurope are supported for now
$AzureSSISNodeSize = "Standard_A4_v2" # In Public Preview, only Standard_A4_v2|Standard_A8_v2|Standard_D1_v2|Standard_D2_v2|Standard_D3_v2|Standard_D4_v2 are supported for now
$AzureSSISNodeNumber = 2 # In Public Preview, only 1-10 nodes are supported for now
$AzureSSISMaxParallelExecutionsPerNode = 2 # In Public Preview, only 1-8 parallel executions per node are supported for now
$VnetId = "[your VNet resource ID or leave it empty]" # OPTIONAL: In Public Preview, only Classic VNet is supported for now
$SubnetName = "[your subnet name or leave it empty]" # OPTIONAL: In Public Preview, only Classic VNet is supported for now

# SSISDB info
$SSISDBServerEndpoint = "[your Azure SQL Database server name.database.windows.net or your Azure SQL Managed Instance server endpoint]"
$SSISDBServerAdminUserName = "[your server admin username]"
$SSISDBServerAdminPassword = "[your server admin password]"
$SSISDBPricingTier = "[your Azure SQL Database pricing tier, e.g. S3, or leave it empty for Azure SQL Managed Instance]" # Not applicable for Azure SQL Managed Instance
##### End of SSIS in ADFv2 specifications #####
```

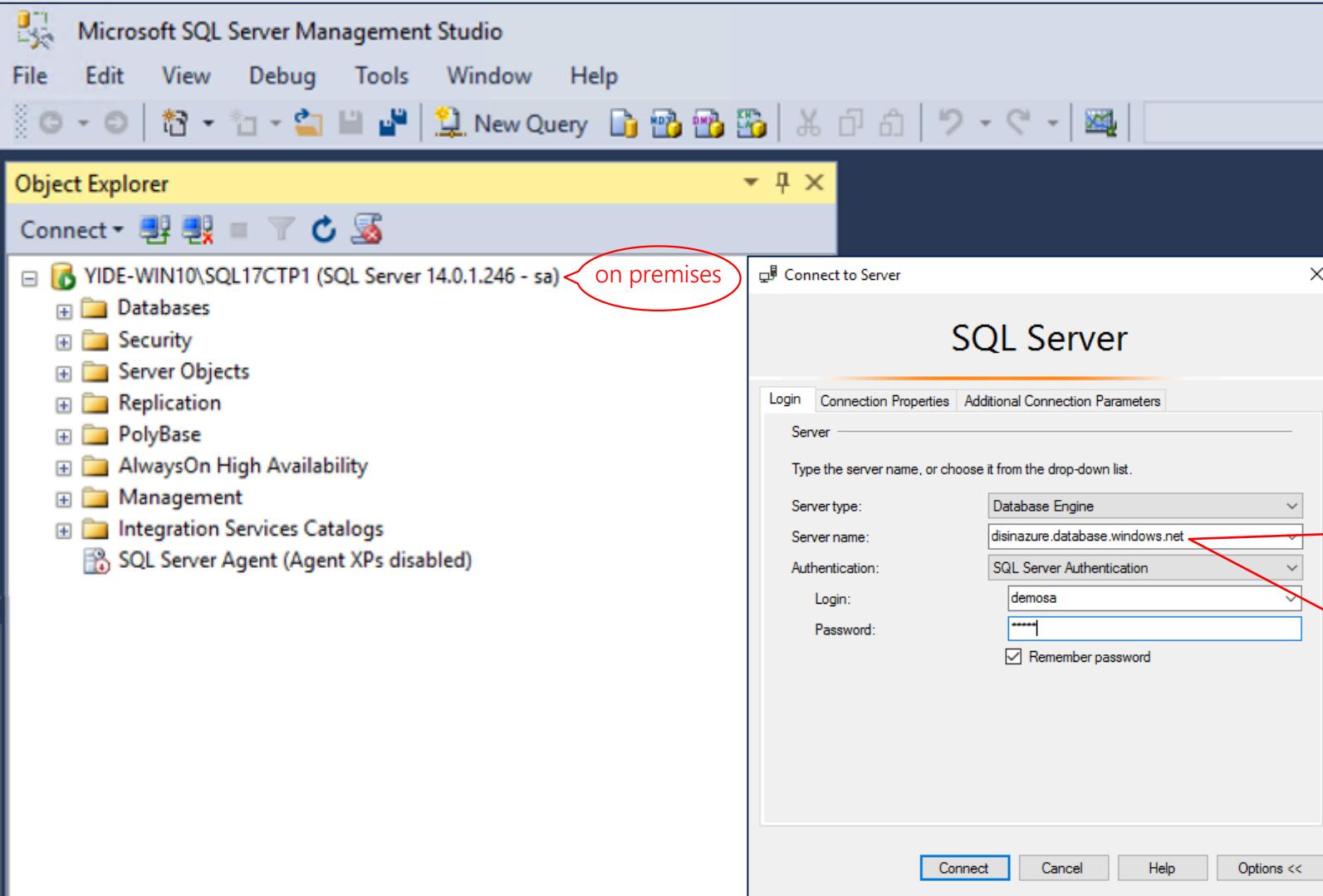
Deployment Methods

- SSIS PaaS supports the project deployment model used in SSIS 2012/later versions
 - Projects built in the legacy package deployment model used in SSIS 2008/earlier versions can be converted into this model via SSDT/SSMS using Integration Services Project Conversion Wizard
 - Packages built in SSIS 2008/earlier versions can be upgraded to the latest version supported by SSIS PaaS via SSDT/SSMS using SSIS Package Upgrade Wizard
 - In this model, the whole project needs to be deployed after any package changes – An incremental package deployment feature will be provided in the near future
 - Projects containing environment references/run-time parameters can be saved into project deployment files (.ispac extension)
 - Projects are deployed into SSISDB hosted by Azure SQL DB/MI server, packages are run by creating/starting jobs via SSISDB sprocs that will be executed on Azure-SSIS IR, and execution logs are written back into SSISDB

Deployment Methods

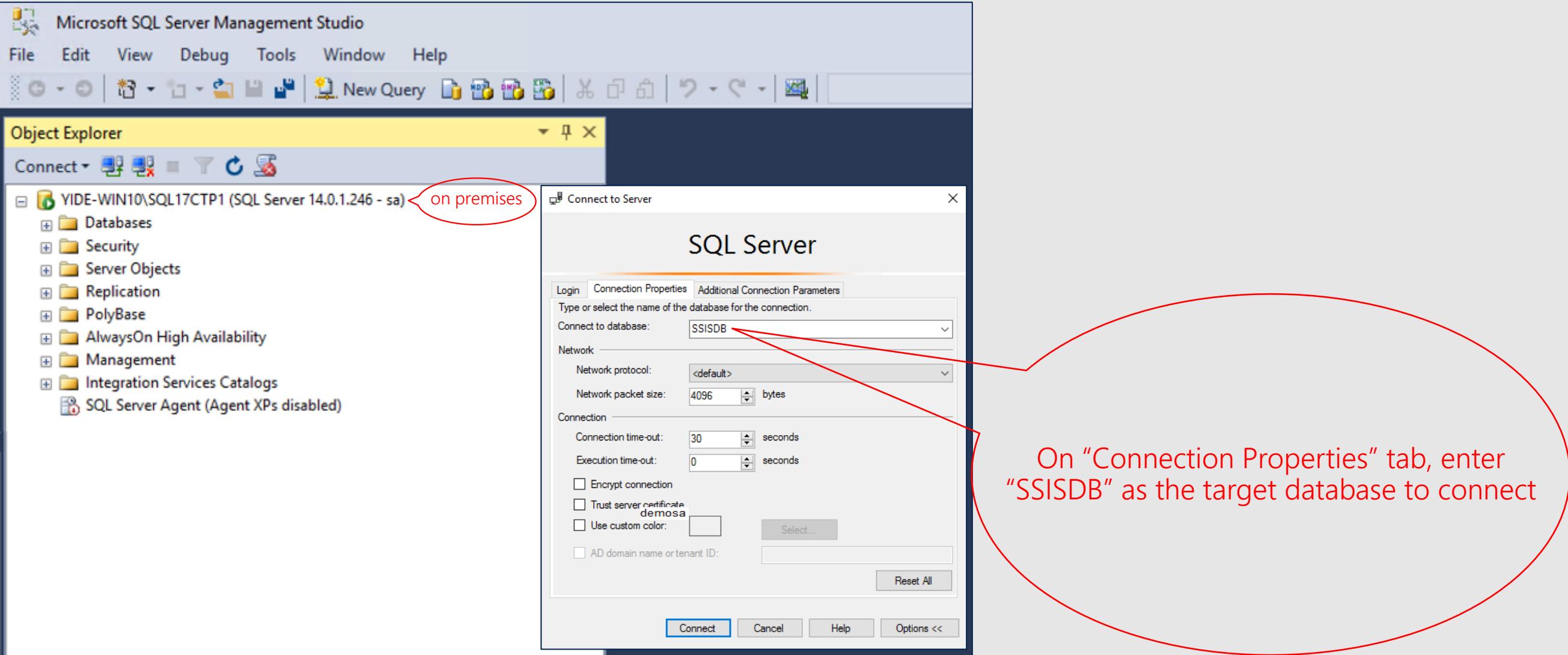
- SSIS projects can be deployed via SSDT/SSMS using Integration Services Deployment Wizard
- SSIS projects can be deployed via Command Line Interface (CLI)
 - Run isdeploymentwizard.exe from the command prompt (TBD)
- SSIS projects can be deployed via custom code/PSH using SSIS Managed Object Model (MOM) .NET SDK/API
 - Microsoft.SqlServer.Management.IntegrationServices.dll is installed in .NET Global Assembly Cache (GAC) with SQL Server/SSMS installation
- SSIS projects can be deployed via T-SQL scripts executing SSISDB sprocs
 - Execute SSISDB sproc [catalog].[deploy project]

Deployment via SSMS

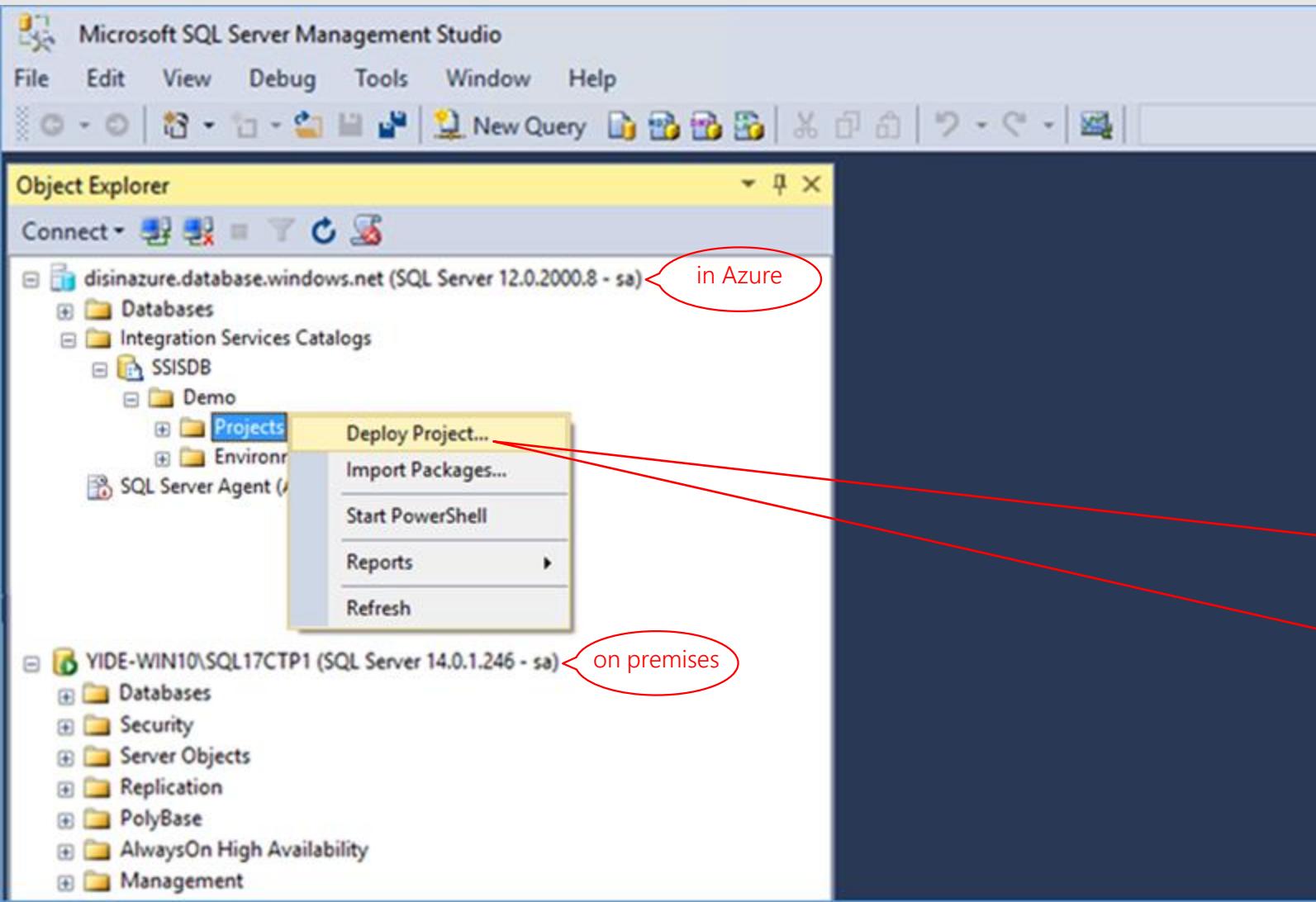


On SSMS, you can connect to SSIS PaaS using its connection info and SQL/AAD authentication credentials

Deployment via SSMS

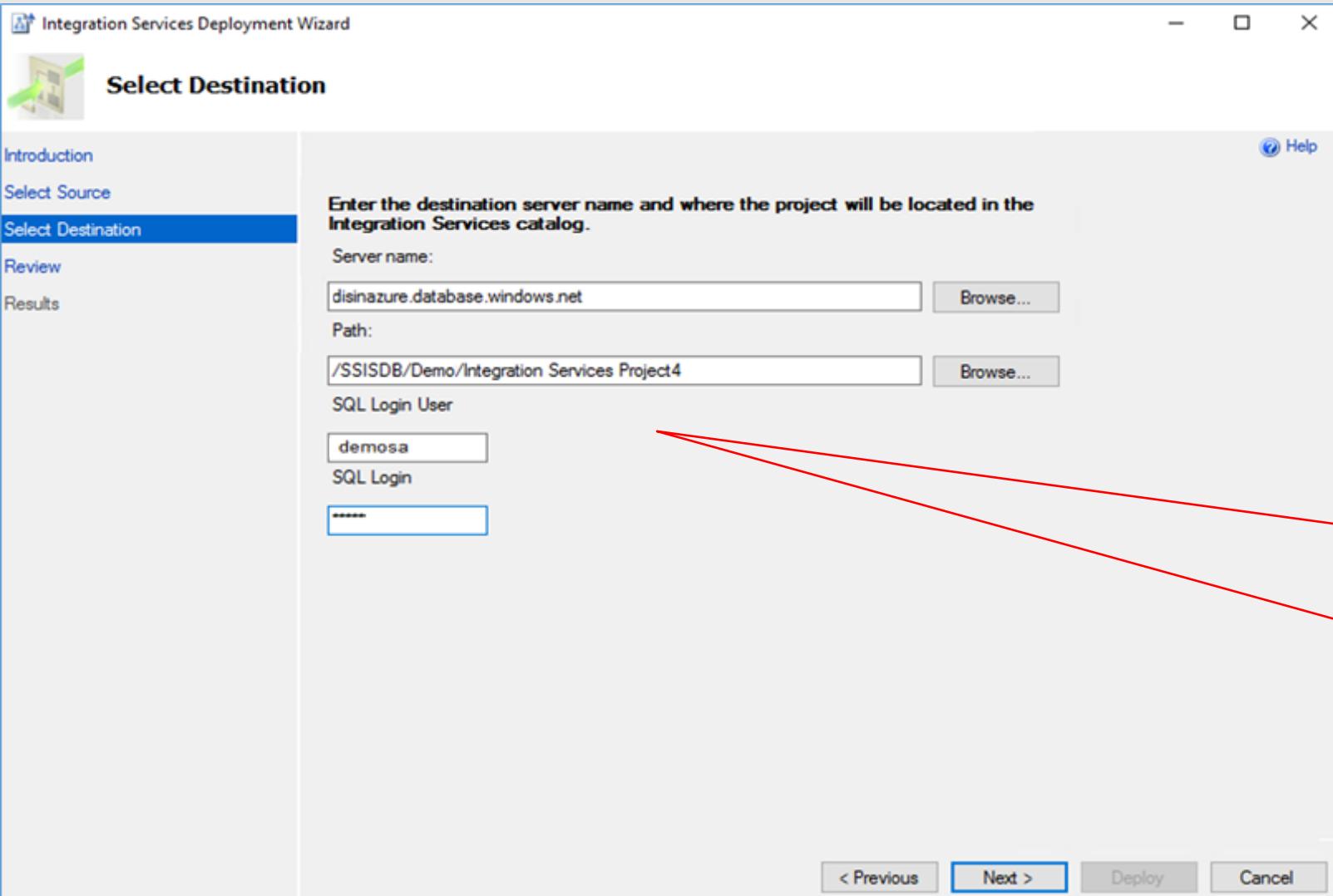


Deployment via SSMS



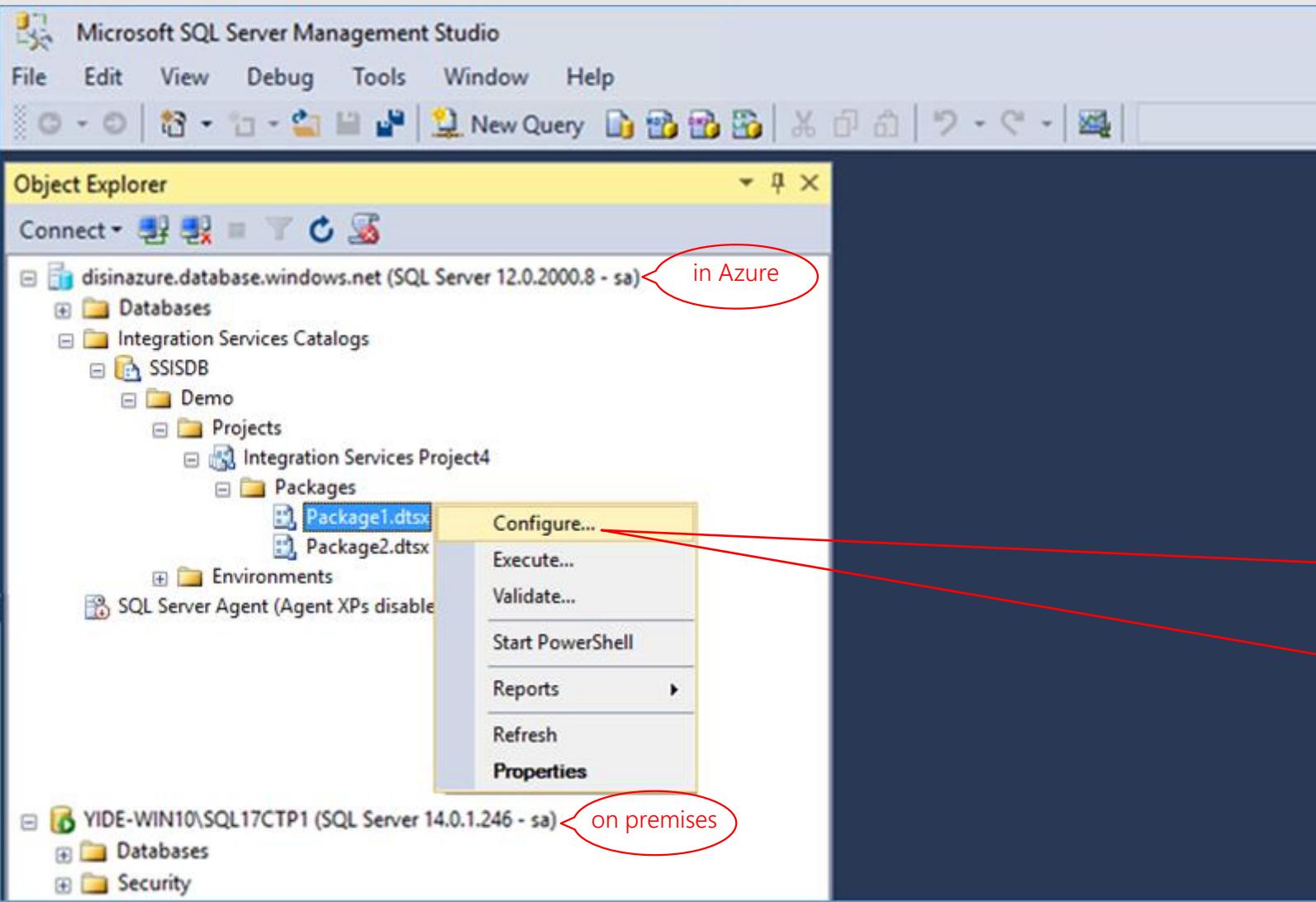
Once connected, you can deploy projects/packages to SSIS PaaS from your local file system/SSIS on premises

Deployment via SSMS



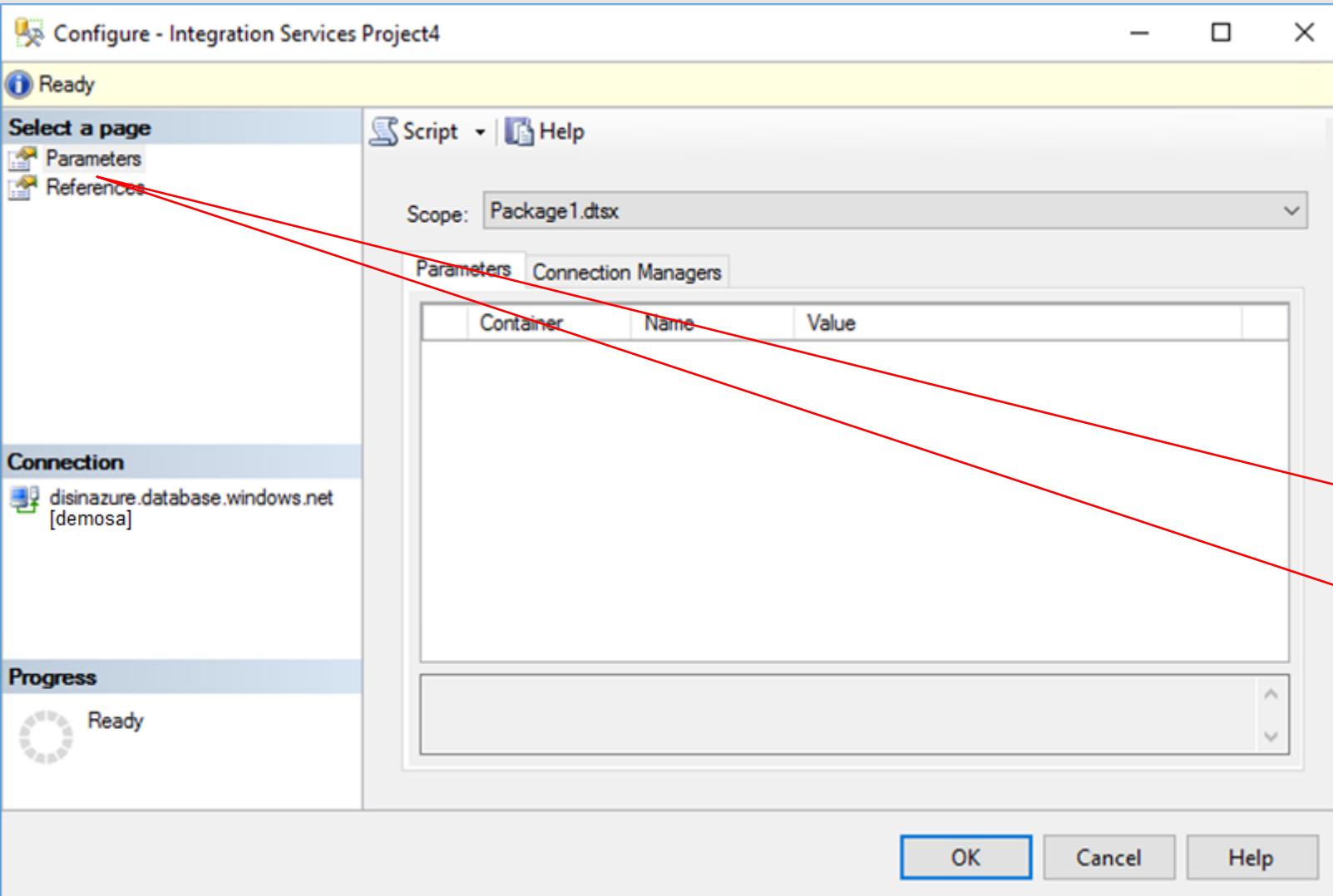
On Integration Services Deployment Wizard,
enter SSIS PaaS connection info and SQL
authentication credentials

Execution via SSMS



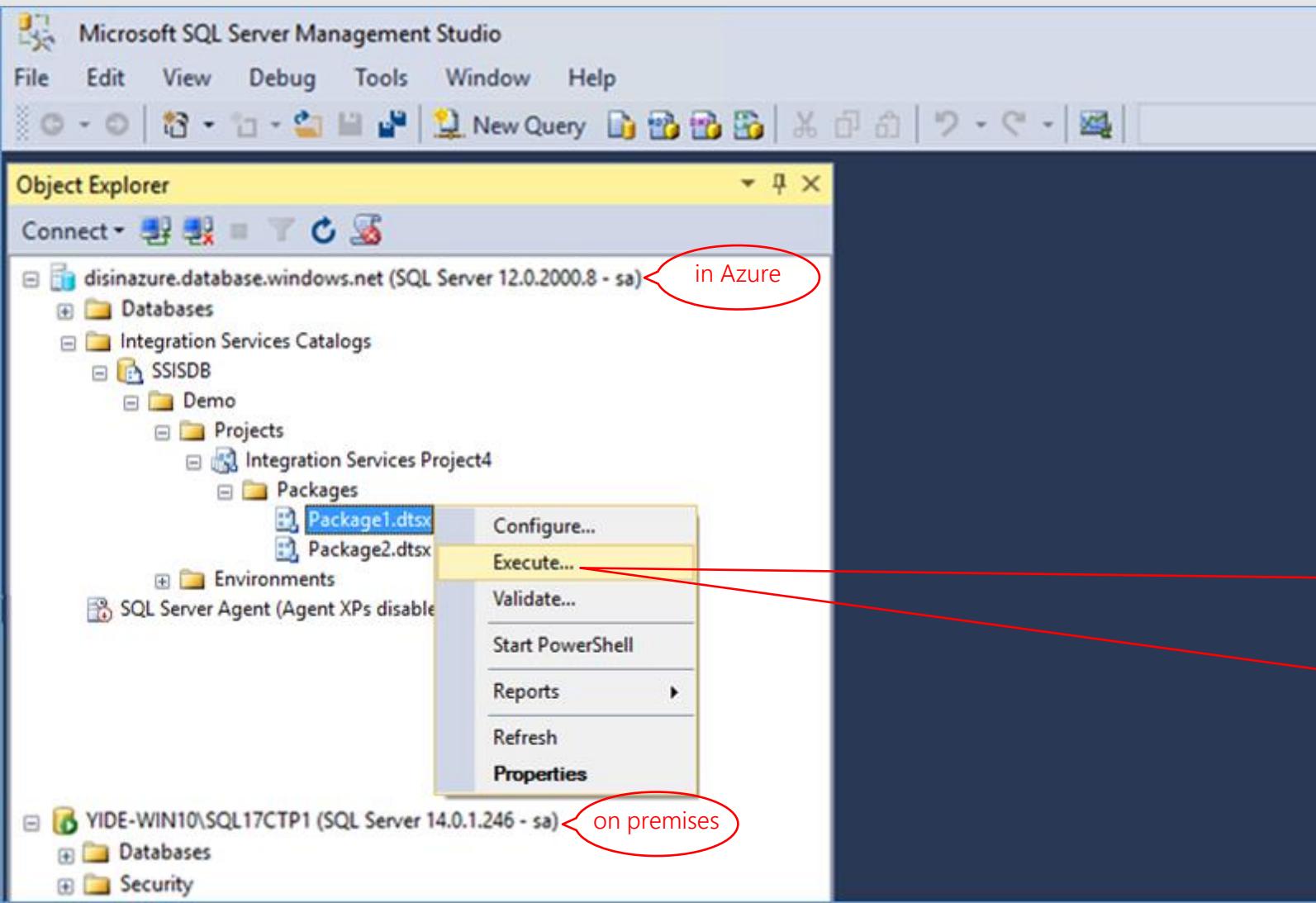
Once deployed, you can configure packages
for execution on SSIS PaaS

Execution via SSMS



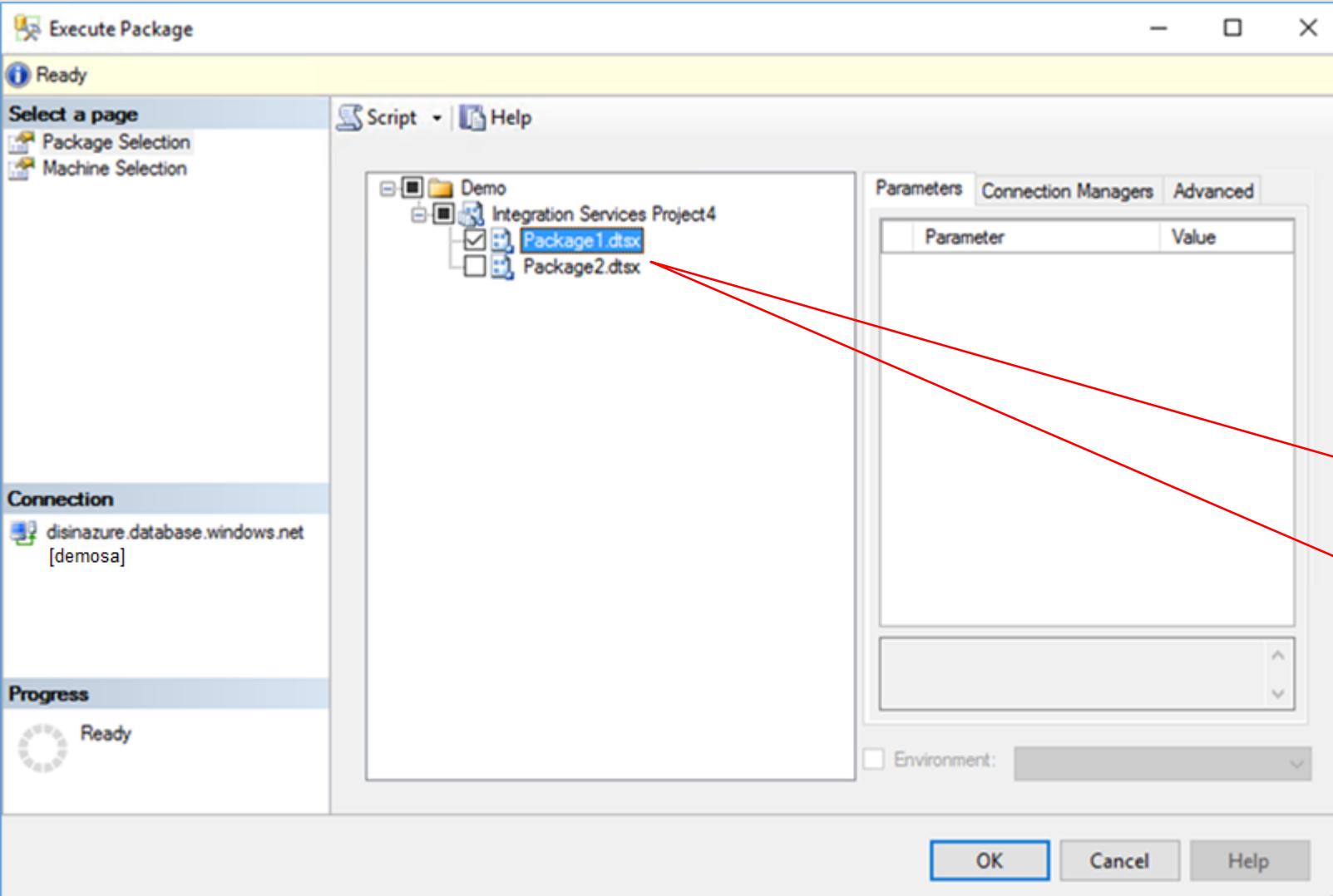
You can set package run-time
parameters/environment references

Execution via SSMS



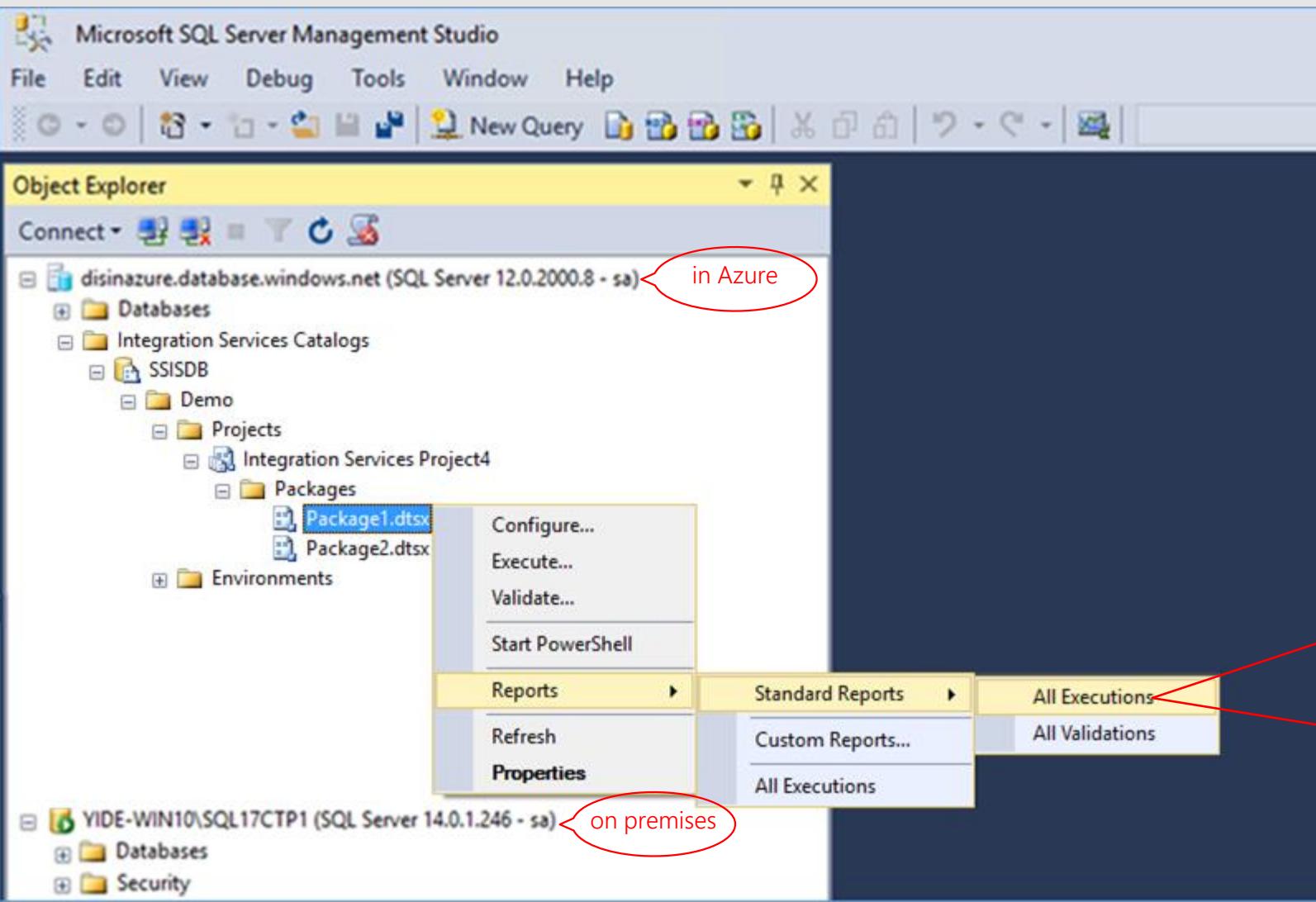
Once configured, you can execute packages
on SSIS PaaS

Execution via SSMS



You can select some packages to execute on
SSIS PaaS

Monitoring via SSMS



You can see reports of all package executions
on SSIS PaaS

Monitoring via SSMS

The screenshot shows the Microsoft SQL Server Management Studio interface. On the left, the Object Explorer pane displays two server connections:

- disinazure.database.windows.net (SQL Server 12.0.2000.8 - sa)** (circled in red with the annotation "in Azure")
- YIDE-WIN10\SQL17CTP1 (SQL Server 14.0.1.246 - sa)** (circled in red with the annotation "on premises")

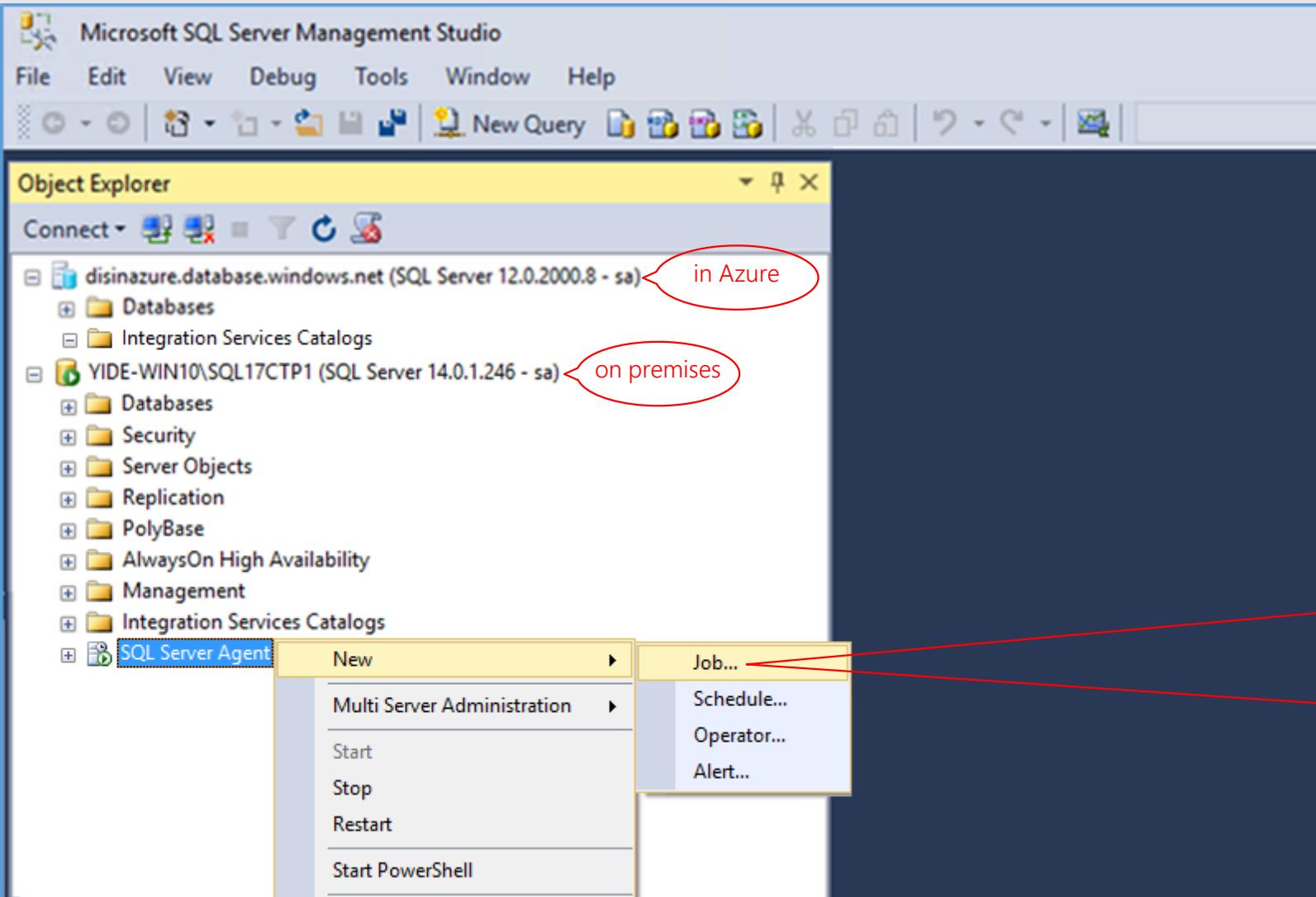
The main window shows the "All Executions" report for the Azure connection. The report title is "All Executions - 1...- CN-SAWINARK-DES". The content includes:

- All Executions**: A summary message: "on disinazure.database.windows.net at 12/9/2016 11:22:45 AM".
- Execution Information**: A summary of execution counts:
 - Failed: 0
 - Running: 0
 - Succeeded: 1
 - Others: 0
- ID**, **Status**, **Report**, **Folder Name**, **Project Name**, **Package Name** columns for the execution details.

A red callout points to the "Succeeded" count with the text "You can see package execution error messages". Another red arrow points to the "All Messages" link in the report header.

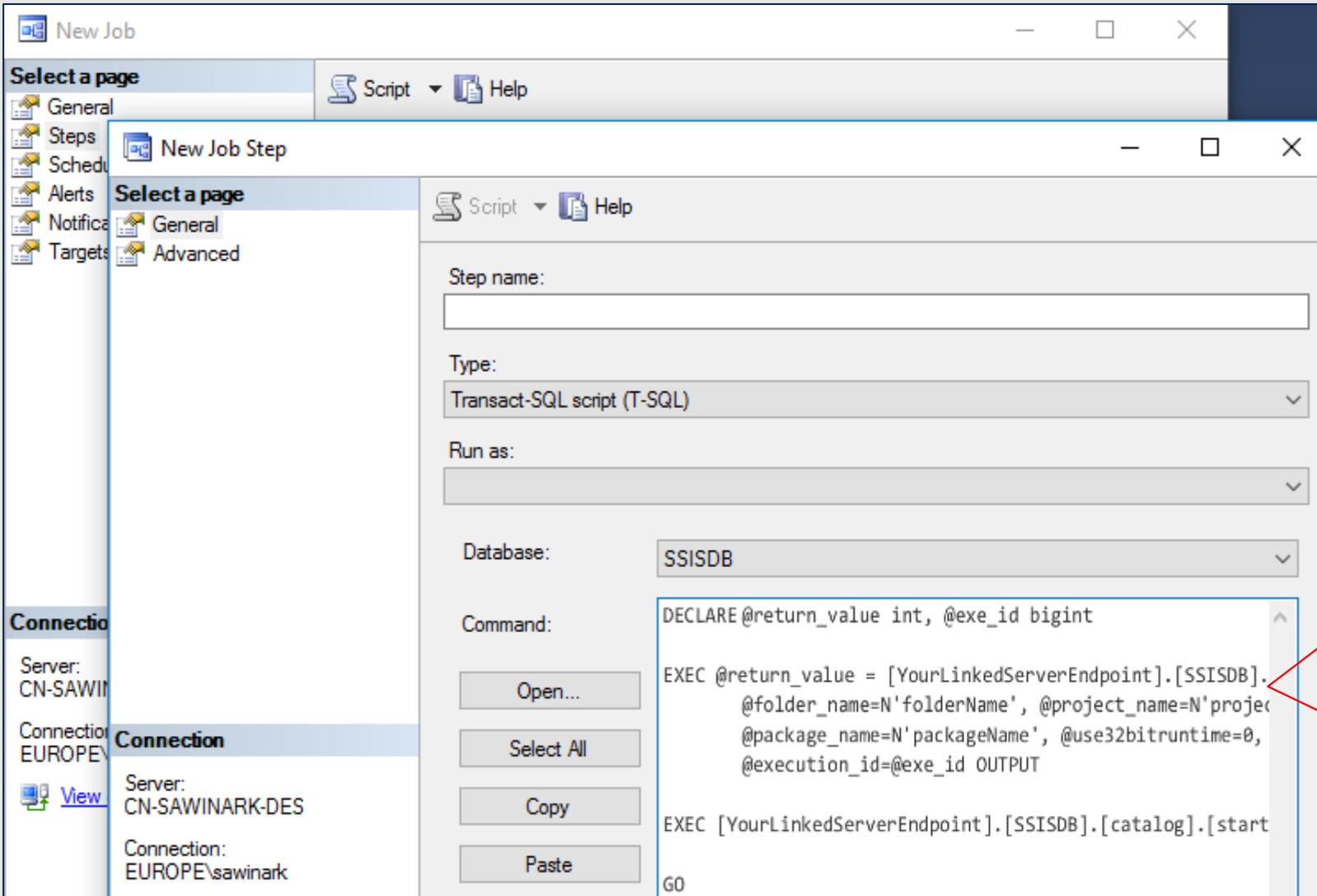
ID	Status	Report	Folder Name	Project Name	Package Name		
5	Succeeded	Overview	All Messages	Execution Performance	Demo	Integration Services Project4	Package1.dtsx

Scheduling via On-Prem SQL Server Agent



You can schedule T-SQL jobs to execute packages on SSIS PaaS via on-prem SQL Server Agent

Scheduling via On-Prem SQL Server Agent



T-SQL jobs can execute SSISDB sprocs in Azure SQL DB/MI server that has been added as a linked server to on-prem SQL Server

Execution via ADF

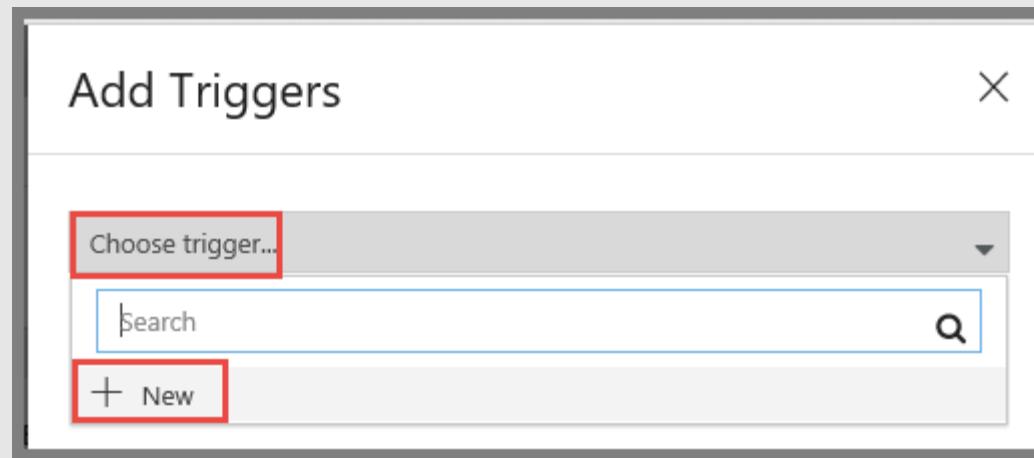
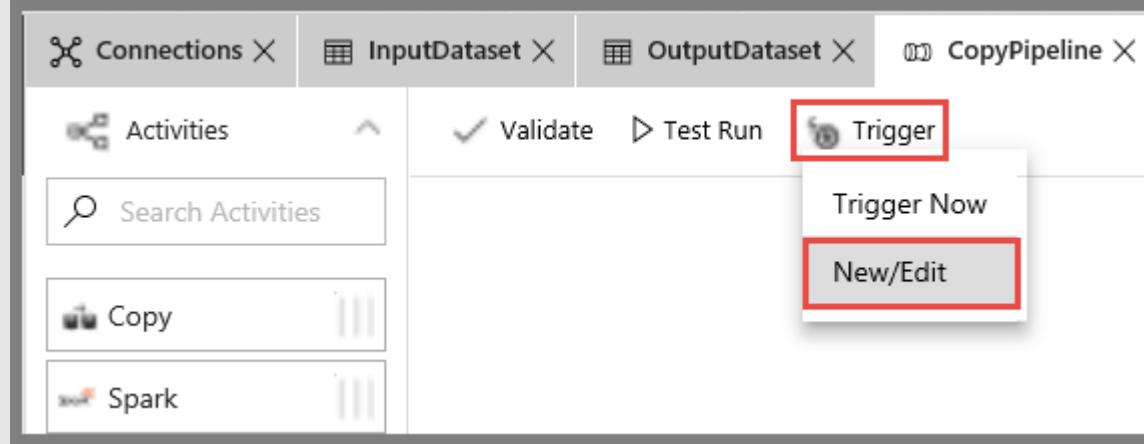
The screenshot shows the Azure Data Factory pipeline editor. On the left, the navigation pane includes 'Factory Resources', 'Pipelines' (selected), and 'Datasets'. In the center, a pipeline named 'pipeline1' is displayed with an 'Execute SSIS Package' activity selected. The pipeline run history table at the bottom shows columns: Name, Type, Run Start, Duration, Status, and Actions.

The settings dialog for the 'Execute SSIS Package' activity. It has tabs for General, Settings (selected), Parameters, and Advanced. The 'Azure-SSIS IR' dropdown is set to 'Loading...'. Under '32-Bit runtime', there is a checkbox which is unchecked. The 'Logging level' dropdown is set to 'Basic'. There are also 'Customized' and 'Environment path' options. A blue callout points from the 'Execute SSIS Package' activity in the pipeline to this dialog.

Execute SSIS package from SSISDB
hosted in the Cloud and choose
environments for parameters

Execute SSIS Package Activity

Scheduling via ADF Wall-Clock Triggers



A screenshot of the trigger configuration dialog for 'trigger1'. The dialog includes fields for Description, Type (Schedule selected), Start Date (01/03/2018, 10:54 PM), Recurrence (Every Minute), and End On (01/03/2018, 11:10 PM). The 'On Date' radio button is selected under End. The dialog also features a calendar for selecting the end date and time inputs for the end date. Three numbered callouts point to specific elements: 1 points to the 'On Date' radio button; 2 points to the hour input field in the end time selector; and 3 points to the minute input field in the end time selector.

Monitoring via ADF

The screenshot shows the Azure Data Factory Monitor Pipeline Runs interface. It displays a table of pipeline runs for 'pipeline1'. The first row shows a run started on 01/18/2018 at 5:27:18 PM, which is currently 'In Progress...'. The 'Actions' column contains a refresh button and a status link. The 'Status' column shows a green checkmark for success. The 'Integration Runtime' is listed as 'DefaultIntegrationRuntime (East US)'. The 'Pipeline Name' column has a dropdown arrow.

The screenshot shows the Azure Data Factory Monitor Activity Runs interface. It displays a table of activity runs for 'Execute SSIS Package1'. The first row shows a run started on 01/18/2018 at 5:27:21 PM, which is 'Succeeded'. The 'Actions' column contains a refresh button and a status link. The 'Status' column shows a green checkmark for success. The 'Integration Runtime' is listed as 'DefaultIntegrationRuntime (East US)'. The 'Activity Name' column has a dropdown arrow.

The screenshot shows the Azure Data Factory Pipeline Runs interface. It displays a table of pipeline runs for 'dbb501e'. The first row shows a run started on 05/11/2018 at 6:05:07 PM, which is 'Succeeded'. The 'Status' column shows a green checkmark for success. The 'Integration Runtime' is listed as 'myAzureSSISIntegrationRuntime (East US)'. The 'RunID' column shows the ID of the run. The 'Output' pane shows the JSON output of the activity run, with the 'ExecutionId' field highlighted in red. The 'Activity Name' column has a dropdown arrow.

SSISDB execution ID from the output of the pipeline activity run can be used to check more comprehensive execution logs and error messages in SSMS

The screenshot shows Microsoft SQL Server Management Studio with two windows. The left window is 'Object Explorer' showing the database structure, including 'SSISDB' and its contents like 'demo', 'Projects', 'ScaleOutProject', 'Packages', and 'Environments'. The right window is 'All Executions' report for May 23, 2018, at 2:29 PM. It shows 5 successful executions. The bottom part of the right window is a table of execution details:

ID	Status	Report	Folder Name	Project Name
88	Succeeded	Overview All Messages Execution Performance	demo	ScaleOutProject
87	Succeeded	Overview All Messages Execution Performance	demo	ScaleOutProject
86	Succeeded	Overview All Messages Execution Performance	demo	ScaleOutProject
85	Succeeded	Overview All Messages Execution Performance	demo	ScaleOutProject
84	Succeeded	Overview All Messages Execution Performance	demo	ScaleOutProject



Automatic Data warehouse generating firm cuts development time by a third

Challenge

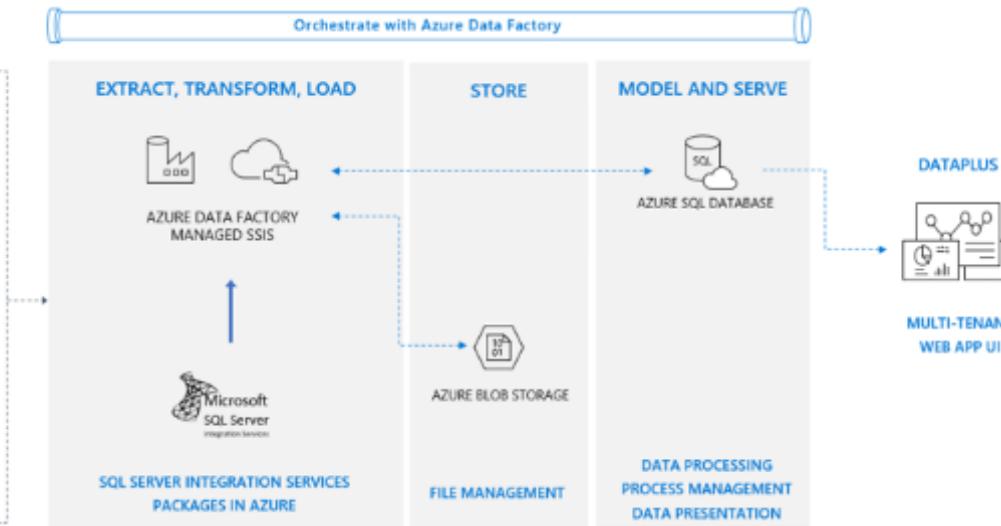
Most of Concentra customers were challenged by disparate data sources from hybrid and multi-cloud environments. As the customers varied in size, it was difficult to maintain scale without adding infrastructure.

As the firm's data warehouse practice grew, maintenance of existing data warehouses increasingly required process updates to accommodate changes to data feeds, especially those of healthcare and government customers.

Solution

Build Software-as-a-service on Azure to automate data-driven workflows with SQL Server Integration Services in Azure Data Factory, creating an easy-to-use, cloud-based version of their data warehouse generator.

Customers can redeploy their existing SSIS packages from on premises environments to Azure with minimal modification in managed SSIS environment in Azure Data Factory.



- Time to production – 3 weeks
- 100 GB data moved weekly
- Saved 3 months of development effort by moving to cloud

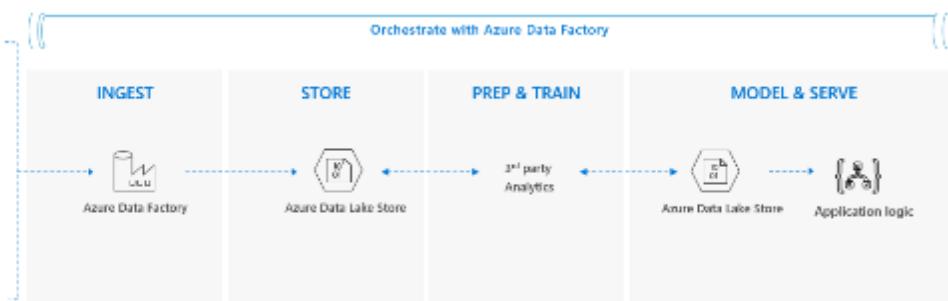


Adobe uses global cloud to create, deliver, and manage better digital experiences

Challenge

Developers building out the next generation of consumer experiences need a semantic platform with curated data, APIs and machine learning services on top, designed explicitly for that purpose.

Bringing massive volume of behavioral and experience data across on-premises, cloud and SaaS applications that needs to be ingested at scale, prepared and transformed into experience data models.



Solution

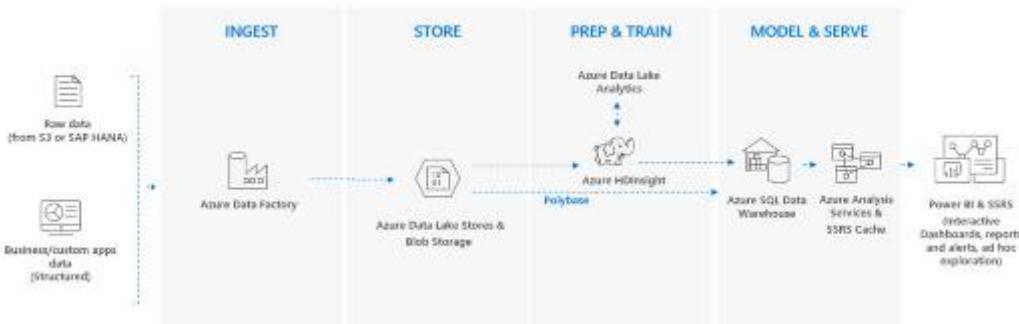
With Azure Data Factory, Adobe can connect to hybrid data sources and ingest data at scale into the lake powering the Adobe cloud platform.

Data may consist of CRM data, e-commerce transactions, offline transactions, loyalty program data, behavioral data from mobile, web or emails, and social interaction data.

- 2M activity runs per month
- 10 TB data moved per month
- Time to production – 6 months
- Expedited go to market by 6 months
- Pulling data from Salesforce, Dynamics, AWS S3, MySQL and FileShare



HEALTH • HYGIENE • HOME



- 7.1 TB data moved weekly
- 4.7K activity runs per week
- Time to production – 2.5 months

Consumer goods firm empowers employees to work smarter using big data

Challenge

Data. To make good decisions, salespeople, marketers, product developers, and executives all need data to understand what's going on with the business and the team realized that to gain those kinds of crystal-ball insights, RB needed more than just internal operational data, they needed third-party data.

Need to do a better job of helping employees make sense of proliferating data.

Solution

Use Microsoft Azure to build a modern data warehouse solution to replace AWS data warehouse and Hadoop components and save money in the process.

Bring together a variety of data from different sources using Azure Data Factory in Azure HDInsight and Azure SQL Data Warehouse and make it easily accessible in one view across the organization.



Improved Health Outcomes by integrating healthcare data

Challenge

Integration of Healthcare data from multiple hospitals and heterogenous sources

Transformation and orchestration of dataflow at scale

Increasing costs of scaling on-premise data pipelines

Solution

Built end-to-end data integration with Azure Data Factory, leveraging big data and compute offerings in Azure

Enabled data ingestion from hybrid sources of data

Orchestrated complex data workflows at scale



Industrial automation firm cuts up to 90 percent of costs

Challenge

Manage data in excess of the existing data warehouse and systems capacity

Integrate data on-premises with cloud and batch processing with live streaming

Monitor a complex supply chain – undersea wells, refining facilities, gas stations

Solution

Integrated complex data systems with Azure Data Factory

Composed, orchestrated, and managed highly available data pipelines

Reduced up to 90 percent of costs



Actuarial firm unlocks the potential of human capital to drive business

Challenge

Increasing complexity of insurance products and regulations

Diversion from firm's core purpose

Rise in on-premises hardware investments to handle high-performance grid computing

Solution

Built software-as-a-service on Azure to address modeling and reporting of large, compute-intensive workloads and used Azure Data Factory to automate extract, transform, and load (ETL) processes

Gained real-time insight into financial data