Microsoft

Solution Guide: Canvas API Provisiong Reports Data extract and Upload to SDS + Teams Classes LTI Integration

Introduction 1

1) Solution Summary 1

1.1) Azure resources setup 1

1.2) Data Factory setup 2

2) Details of the pipelines and data flows 4

3) Operational Guide 5

3.1) Scheduling via triggers 5

3.2) Monitoring and Alerts 7

3.3) Tracking cost 7

4) Debugging 7

5) Reference Materials 9

Introduction

Microsoft School Data Sync is a free service in Office 365 for Education that reads the school and roster data from a school's Student Information System (SIS). It creates Office 365 Groups for Exchange Online and SharePoint Online class teams for Microsoft Teams and OneNote Class notebooks, school groups for Intune for Education, and rostering and SSO integration for many other third-party applications.

This solution leverages Azure Data Factory (ADF) to handle the daily orchestration of activities necessary to:

- Extract data from Canvas API Provisioning Reports and convert to SDS CSV v2.1 format.
- Run validation on the CSV's and remove records that don't have required fields.
- Upload the CSV's to SDS and start the sync. **Note: The ADF integration will use an existing inbound flow or create a new one for you in SDS. Turn on SDS by going to https://sds.microsoft.com click 'Get Started' then 'Continue' before running the ADF integration if no inbound flow exists.**
- Send an execution notice to designated admins via email.

This document provides a summary of the solution setup and operational info for Canvas.

1) Solution Summary

1.1) Azure resources setup

The Azure resources for this solution consist of the following:

Resource Name	Resource	Description
rg-CanvasAPItoSDS	Resource group	Serves as a container for the resources in this solution
adf-CanvastoSds	Data Factory (V2)	The data factory instance, containing the scheduled integration pipelines.
kv-canvas-sds	Key vault	Contains 2 keys: 1) ClientSecretForSdsApiKeyVaultUrl — The client secret created via the App Registration for the ADF instance (details further down) 2) CanvasToken– The token to access the Canvas REST API for your instance. OAuth2 - Canvas LMS REST API Documentation (instructure.com)
stcanvassds	Storage account	V2 storage account, Read-access geo-redundant storage, Encryption type: Microsoft-managed keys

The setup within the Azure subscription consists of provisioning and configuring the above resources with the following steps:

1) In the Azure portal, type "Deploy a custom template" in the search box
2) On the Custom Deployment page, click "Build your own template in the editor" to open the template editor.
3) Load the Canvas_to_SDS_ADF_template.json file by clicking "Load file" in the template editor and selecting the appropriate file to upload. Create a resource group where to place the resources if not already done. (Suggested name rg-CanvasAPItoSDS)
4) Add the mandatory parameters value to create resources (Key vault name, Storage account name, Data Factory name).
5) Remaining parameter values can be added after deployment (refer to Data Factory setup/Global parameters).
6) Modify key vault access to enable managed identity adf-CanvastoSds (ADF instance) to retrieve secrets from the key vault (Assign "Key Vault Secrets User" role). Ensure that the data factory is created first.

   Grant permission to applications to access an Azure key vault using Azure RBAC | Microsoft Learn

   Alt: https://learn.microsoft.com/en-us/azure/data-factory/store-credentials-in-key-vault#steps

7) Modify the key vault to provide access to users who need to update the secret values. (At least "Key Vault Secrets Officer" role for creating). Also, the user's IP address should only be temporarily added to the firewall in the networking tab before updating the key vault secrets. This must be done even if the user has access control privileges

8) Do the same for authorized users who need to modify data in storage. Also, the user's IP address should only be temporarily added in the firewall in the networking tab before updating the storage contents. This must be done even if the user has access control privileges

9) Create an app registration in Entra to allow the ADF resource to call the Graph API's needed then create a secret for the app registration. Quickstart: Register an app in the Microsoft identity platform - Microsoft identity platform | Microsoft Learn

10) Add the key vault secret values needed from the above table (Existing values were created as dummies and can be disabled). Azure Quickstart - Set and retrieve a secret from Key Vault using Azure portal | Microsoft Learn

11) Add the Graph API application permissions from the table below to the app registration. Remember to grant admin consent for the added permissions. Quickstart: Configure an app to access a web API - Microsoft identity platform | Microsoft Learn

Permission	Purpose
IndustryData-SourceSystem.ReadWrite.All	Create a source system for SDS file upload if none exists
IndustryData-TimePeriod.ReadWrite.All	Create academic year for sync if none exists
IndustryData-InboundFlow.ReadWrite.All	Upload files to SDS
IndustryData-DataConnector.ReadWrite.All	Upload files to SDS
IndustryData-DataConnector.Upload	Upload files to SDS
Mail.Send	Email file validation run report
Group.ReadWrite.All	Updating groups with LTI data

1.2) Data Factory setup

1) Go to "Private endpoint connections" in the networking tab for both the key vault and storage account and approve each. (Also verify that public access is disabled and there are no exceptions in "Firewalls and virtual networks")

2) Go to the Data Factory in Azure Portal and click 'Launch studio' to make changes. Once inside, go to the Manage tab on the left menu.

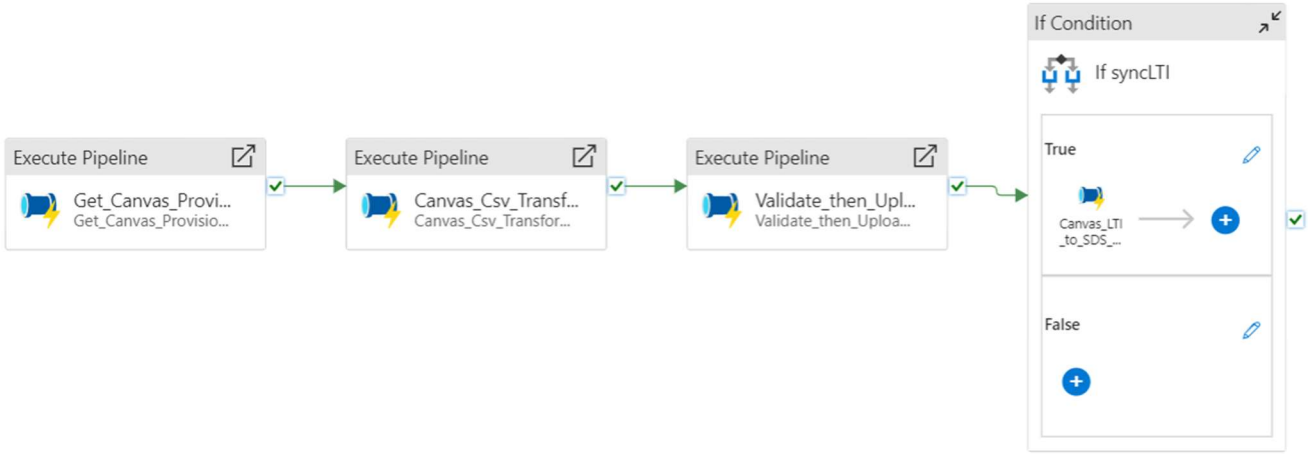
The final step in the ADF setup is to configure the global parameters in the Manage menu as shown below, and further described in the table following.

Global parameter name	Type	Description
canvasAPIBaseUrl	string	Canvas API Base URL needed for syncing LTI data with Class Teams.
reportRecipientEmails	string	The list of email addresses to send email reports to. This needs to be entered as a json array in the following format: [{"emailAddress":{"address":"admin@contosoisd3.onmicrosoft.com"}},{"emailAddress":{"address": "joe@contosoisd3.onmicrosoft.com"}}]
clientSecretForSdsApiKeyVaultUrl	string	The key vault access URL for the client secret for SDS API access. Found in the key vault resource under Secrets -> ClientSecretForSdsApiKeyVaultUrl > Current Version - > Secret Identifier
canvasApiKeyVaultUrl	string	Store the customer canvasToken in key vault secrets and use that secret URL for this parameter.
canvasAccountId	string	Canvas Customer Account Id
entraAppClientId	string	The clientId for the app registration. Found at Azure Active Directory -> App registrations -> <AppName> -> Overview -> Application (client) ID
validationErrorThreshold	Int	If the number of total validation errors for a single profile exceeds this value, the data will not be sent to SDS for that profile. [Note that the record count function has a max value of 5000, so entering a value of 5000 or greater for this setting will effectively turn off the validation threshold entirely]
checkForEmptyFilesBeforeSending	Bool	This validation protects against empty files being inadvertently sent to SDS as the result of a data issue (such as an invalid header in the inbound data file). If set to true, this validation will cause the pipeline to stop if any of the required files or All files to be sent to SDS have no records
tenantid	String	The tenant id
subscriptionId	String	The subscription where the ADF assets are located.
reportSendAsUPN	String	School Data Sync Admin userPrincipalName
sdsInboundFlowId	String	Existing SDS inbound flow id found in Sync > Configuration screen on the SDS left menu pane. Note: Parameter value can be left as it is if no sync exists
useFamilyAndGivenNames	Bool	Will send familyName and givenName as required for users if the option to 'Create unmatched users' is chosen in SDS (otherwise optional). A value of false will not send familyName and givenName.
syncTeamsClassesLTI	Bool	This is a pipeline parameter. Set the value to "true" to add Schoology section LTI data to Class Teams. Set the value to "false" if LTI syncing is not wanted. **Note: Class Groups and Teams must be created by SDS first before running the LTI pipeline.**
staffSourceIdentifier	String	Staff attribute based on data that is coming from your SIS / SMS. Valid values are username, email, or activeDirectory. This can be left as it is if using a sync that has already been created in SDS.
staffMatchTarget	String	Property in Microsoft Entra ID to match to staff. Valid values are userPrincipalName or mail. This can be left as it is if using a sync that has already been created in SDS.

staffDomainAddon	String	A valid tenant domain to append to the source value if your staff data doesn't include the @domain value. (Example: contoso.onmicrosoft.com) This can be left as it is if not using appending a domain or using a sync that has already been created in SDS.
studentSourceIdentifier	String	Student attribute data based on that is coming from your SIS / SMS. Valid values are username, email, or activeDirectory. This can be left as it is if using a sync that has already been created in SDS.
studentMatchTarget	String	Property in Microsoft Entra ID to match to students. Valid values are userPrincipalName or mail. This can be left as is if using a sync that has already been created in SDS.
studentDomainAddon	String	A valid tenant domain to append to the source value if your student data doesn't include the @domain value. (Example: contoso.onmicrosoft.com) This can be left as is if not using appending a domain or using a sync that has already been created in SDS.
yearStartDate	String	Start of the academic year in yyyy-mm-dd format. This can be left as is if using a sync that has already been created in SDS.
yearEndDate	String	End of the academic year in yyyy-mm-dd format. This can be left as is if using a sync that has already been created in SDS.
expirationDateTime	String	The date when SDS should stop syncing data based on the defined academic year for this source. It must not exceed the year end date. This can be left as is if using a sync that has already been created in SDS.
optionalFilesRequired	Bool	This is a pipeline parameter, if customer want to use all the files for the sync process, they must pass the value as “true.” If customer want to use only Required Files for the sync process, then have to set value as “false.”

2) Details of the pipelines and data flows

The main integration pipeline is Master_Canvas_to_SDS_Vnext



This pipeline utilizes sub pipelines to execute the following steps:

1) Get data from Canvas API Data and store to Azure storage.
2) Transform Canvas Data to SDS CSV v2.1 format and validate the data.
3) Upload to SDS:

a. Validate the data and write validated data to “valid” folder and write invalid data to “invalid” folder.

b. Check the number of validation errors against the validationErrorThreshold, and check that each of the csv files have data (must have at least one record in addition to the header, otherwise the assumption is that there has been an error).

c. If validation is passed, upload to SDS, start the sync, and send an email report.

d. If is not passed, send a validation email report.

The validation logic is as follows:

1) Validate each of the csv files for required fields.

2) Write validation errors to validation.log, and store invalid records within the folder named “invalid.”

3) Write out the valid records for orgs.csv, users.csv, roles.csv, classes.csv, courses.csv, enrollments.csv, relationships.csv and demographics.csv in the folder named “valid.”

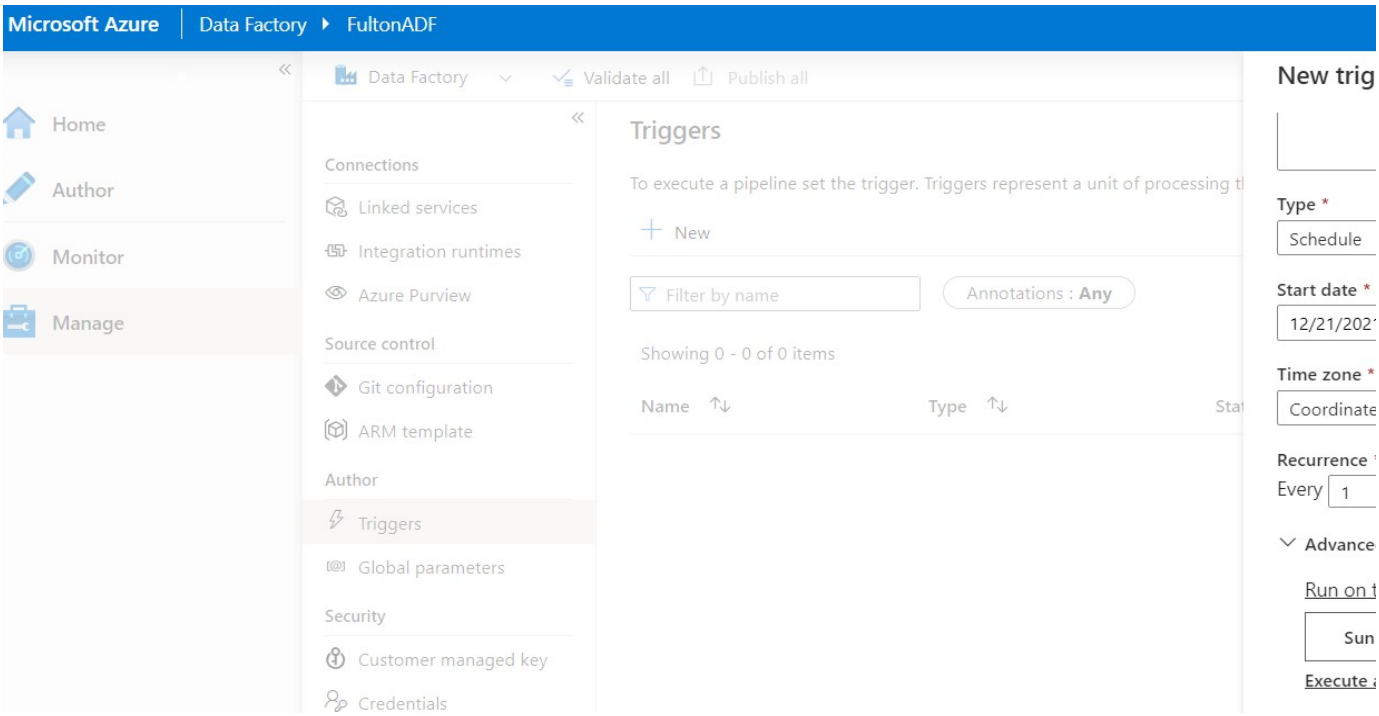
Note that the records in the “invalid” folder represent those records that were rejected because of missing required fields.

To get a specific list of which records were excluded due to validation, perform a diff operation between the file in the outgoing folder and the file in the “valid” folder.

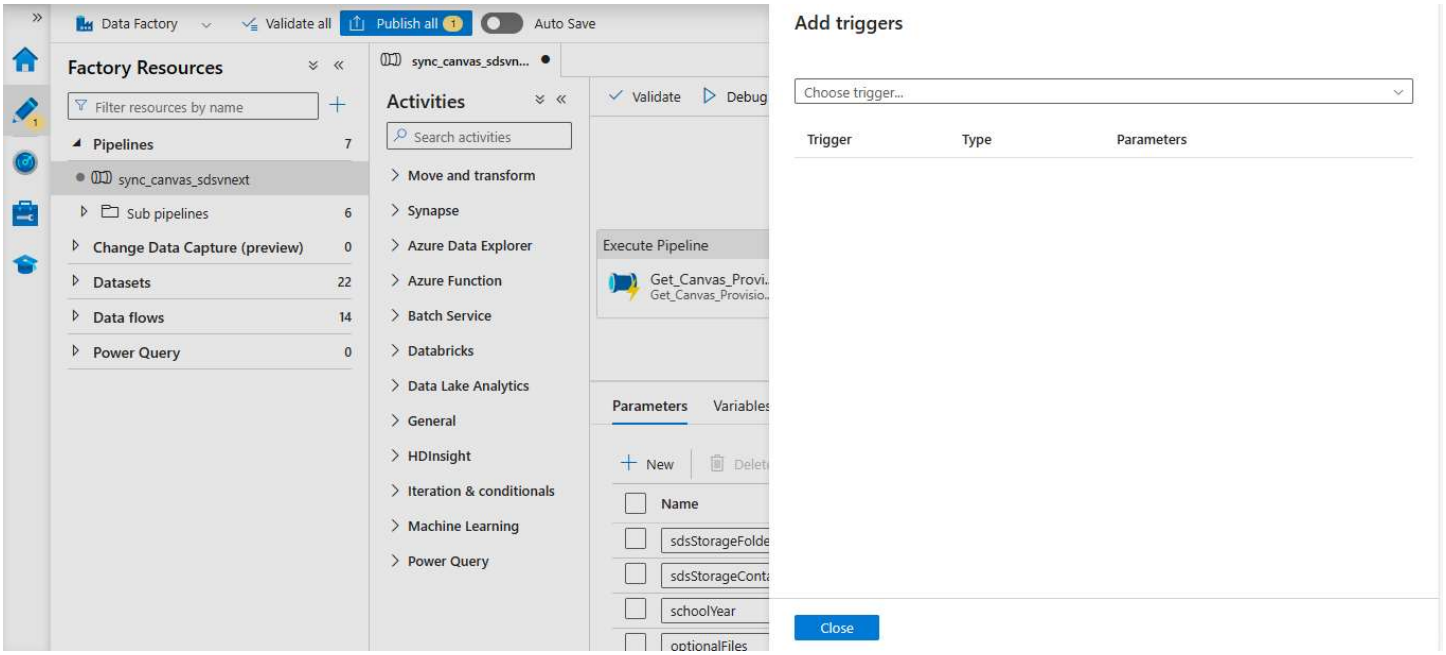
3) Operational Guide

3.1) Scheduling via triggers

Pipelines in ADF can be triggered manually as needed or scheduled to execute based on a configured trigger (more info on triggers here). Triggers are managed in Manage -> Triggers and can be associated with multiple pipelines.

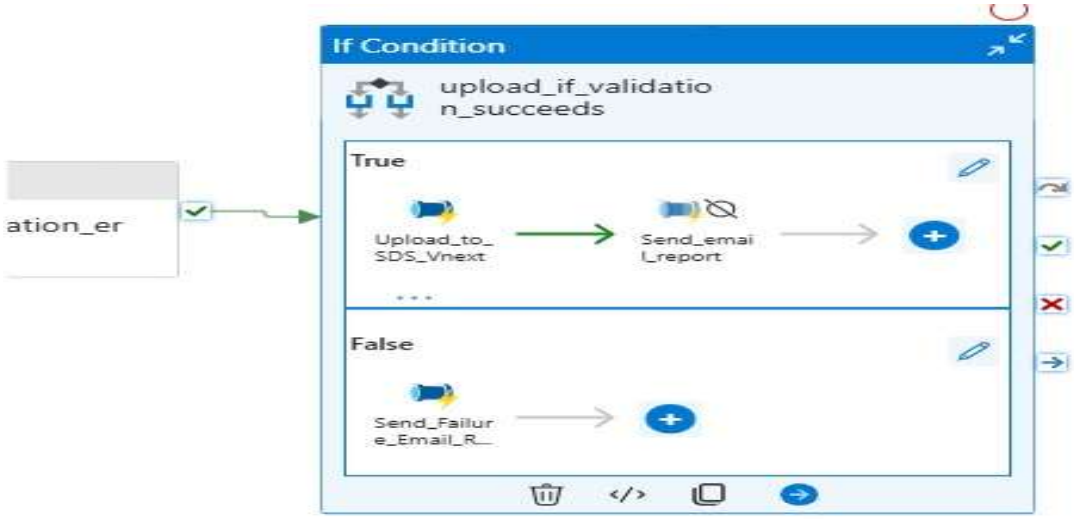


To assign a pipeline to a trigger, open the pipeline and click Add trigger -> New/Edit



Note that when adding a trigger to a pipeline, you must publish the pipeline for the change to take effect.

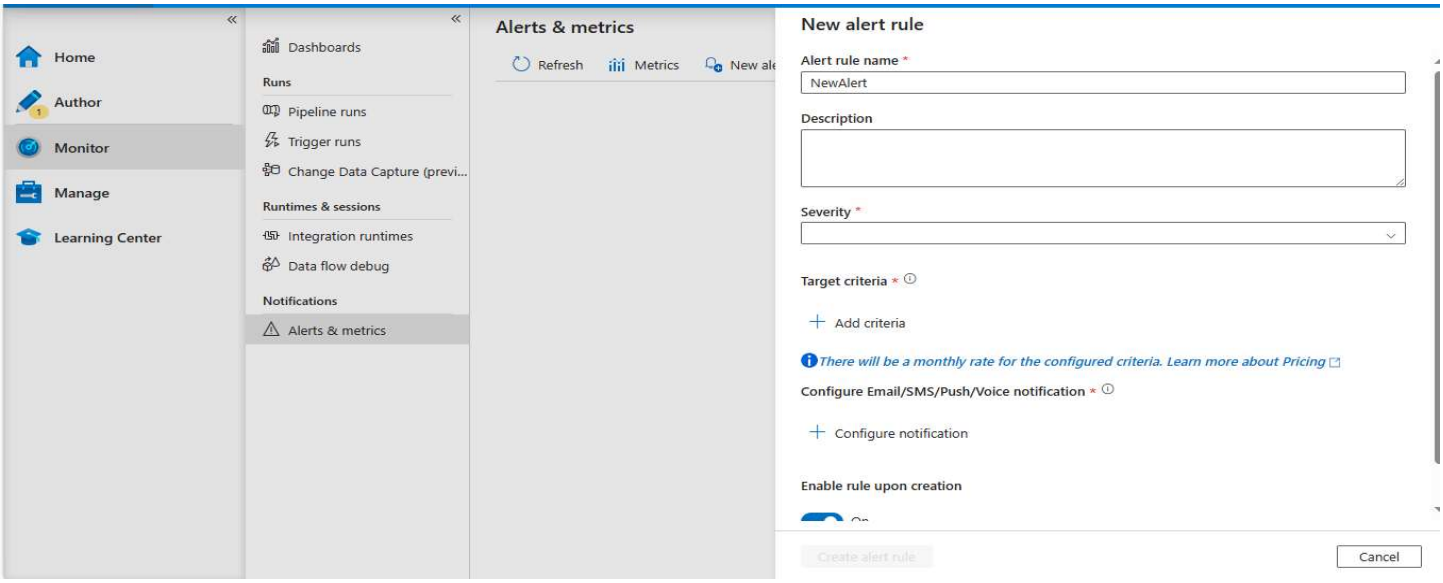
Activities (i.e. Send_email_report) can be disabled by clicking on them and selecting Deactivated then publishing.



3.2) Monitoring and Alerts

Monitoring within ADF is done on the Monitor tab. More info can be found here: Visually monitor Azure Data Factory

Alerts can be configured with Alerts & metrics -> New alert rule.



3.3) Tracking cost

The overall cost for this integration solution is the sum of the costs for each resource within the resource group.

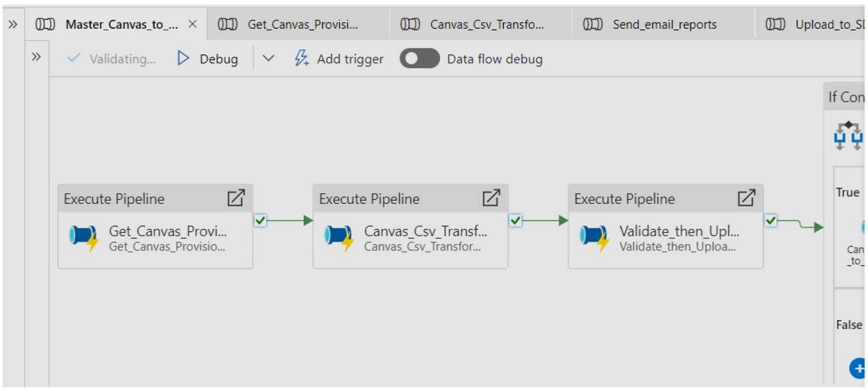
Within the resource group, under Cost Analysis, in the View dropdown, select Daily ADF Cost for a breakdown of the daily cost. You can click Download for an excel report with this daily breakdown. There are several other reports that can be configured and saved from this screen.

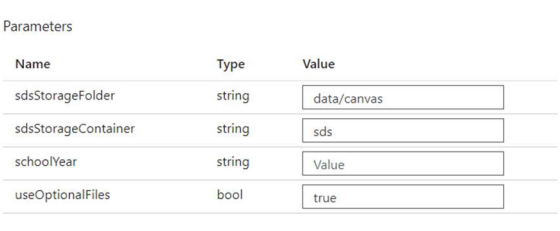
Note that the Data Factory resource is a "pay only for what you use" type of resource, with pricing based on pipeline orchestration and execution, data flow execution and debugging, and number of data factory operations (eg, creating/deploying pipelines, pipeline monitoring). More info on pricing can be found here: Azure Data Factory V2 pricing

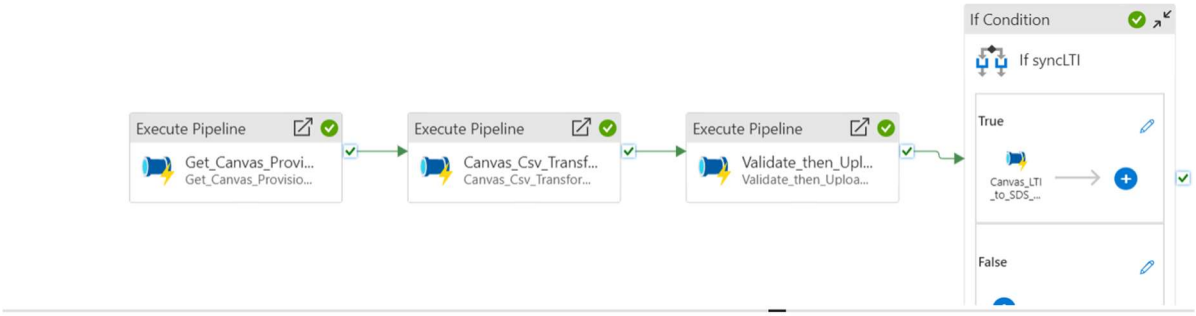
4) Debugging

Debugging within ADF is done on two levels – the pipeline level and the data flows level.

For pipelines which do not include a data flow activity, such as the Send_email_report, you can simply click on the Debug link within the pipeline view. This allows you to execute the pipeline in its current state (without publishing changes) and review the input and output of each activity within the pipeline.







Parameters Variables Settings Output

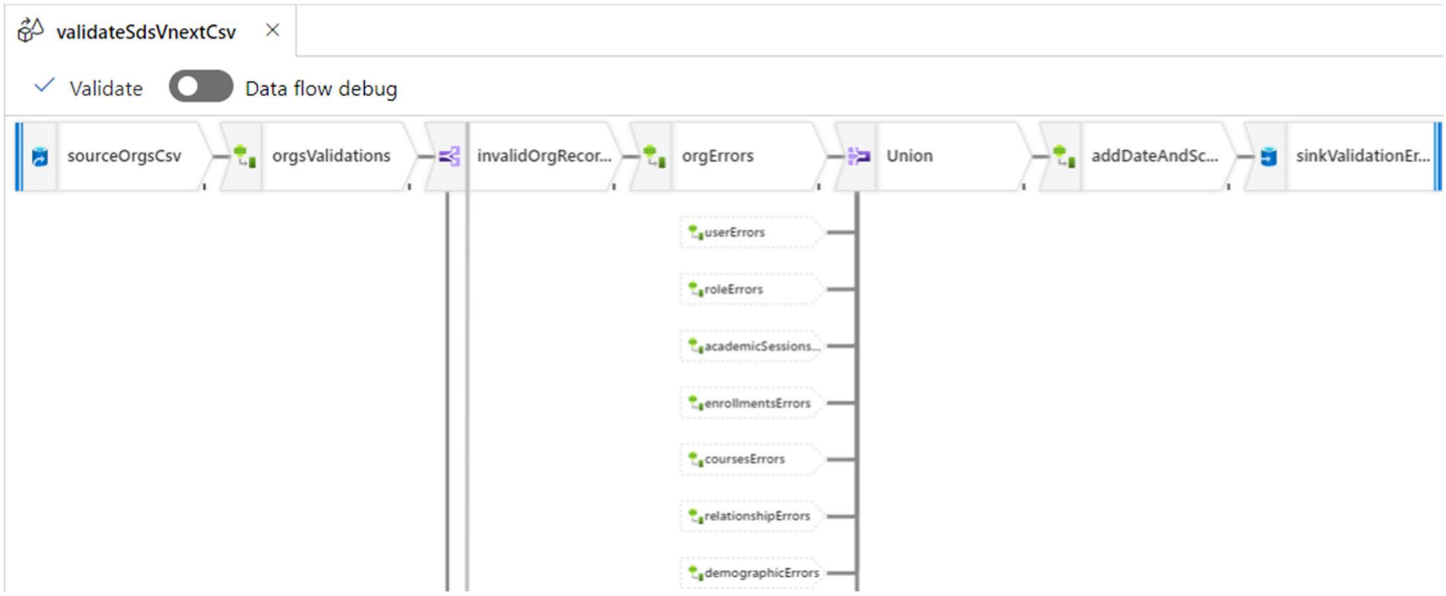
Pipeline run ID: 0e1cc2f7-0492-4d7d-901f-fb3bfb35547d **Pipeline status** In progress

All status List

Showing 1 - 4 of 4 items

Activity name	Activity status	Activity type	Run start	Duration	Integration runtime	User properties	A
If syncLTI	Succeeded	If Condition	4/12/2024, 4:30:42 PM	3s			7
Validate_then_Upload_SDS_V...	Succeeded	Execute Pipeline	4/12/2024, 4:26:05 PM	4m 37s			8
Canvas_Csv_Transform_to_SD...	Succeeded	Execute Pipeline	4/12/2024, 4:20:27 PM	5m 38s			8
Get_Canvas_Provisioning_CSV...	Succeeded	Execute Pipeline	4/12/2024, 4:19:39 PM	49s			el

To debug a data flow such as the validate_SDS_Vnext_All_CSVs or Validate_Required_SDS_Vnext_CSV data flows, you have to first turn on the "Data flow debug" setting.



More info is available here: [Mapping data flow debug mode]

5) Reference Materials

Reference	Description
Homepage of Data Factory app	The homepage within the app provides links to documentation, videos, and tutorials.
Azure Data Factory documentation	Technical reference, with links to tutorials, samples, how-to guides
Stack overflow for ADF	Developer community questions and answers on ADF
MSDN forum for ADF	Developer community questions and answers on ADF
ADF videos on Channel 9	Short videos from Microsoft about features of ADF
SDS management API	The documentation for the management API for SDS