

Foreword

This book is not the Great American Novel. This book will not change your life.

It might, **however**, help you address your more challenging Azure Data Solution use cases.

In these pages you will find step-by-step instructions for solutions to a variety of use cases that I have experienced as a Microsoft Cloud Solution Architect.

My role title falls short of self-explanation... when asked, I explain that I exist in the grey area between various roles:

- **I sell, but I am not a sales person...** I prefer Microsoft solutions, but if a customer desires a non-Microsoft Product X, I am happy to incorporate it in the architecture
- **I serve, but I am not a service consultant...** I love firsthand development, but there is a role-defined limit to how deep I can dig (and customer resources must take over)
- **I support, but I am not a support engineer...** I will dig into any issue to help with a challenge, but Support Engineers have the knowledge and resources to go far deeper than I

My north star is Customer Success (where “success” means whatever aligns best with customer goals). I collaborate with customers to navigate the diverse landscape of resources necessary to architect a solution that they can implement and own.



This is not a book that you will read from start to finish... rather, I encourage you to **browse directly to those topics that align with your focus and interest**.

Table of Contents

Foreword.....	1
Preparation	3
Naming Convention	3
Resource Instantiation.....	4
Acquisition	54
Migration	54
No-Load.....	59
Simple Load.....	74
Incremental Load.....	100
API Load	154
Deployment	191
Some Basics.....	191
Solution Deployment.....	214
Protect Resources.....	234
Presentation.....	236
Query from On-Prem.....	236
Detect Anomalies.....	240
Application+ AI.....	247
Audit Usage.....	268
Governance.....	271
Discover Data.....	271
Classify Data.....	286
Understand Lineage.....	292

Preparation

This book will not go deep into the many philosophical discussions {e.g., Naming Convention} that arise when implementing Azure; for that sort of content, I recommend the Well-Architected Framework documentation at <https://docs.microsoft.com/en-us/azure/architecture/framework/>

That said... I might quickly delve into brief discussions of topics like naming conventions when it will directly impact implementation activities.

Naming Convention

When deciding on a naming convention, remember that Azure inconsistently applies naming rules. Depending on the resource that you are trying to instantiate, you might see rules that:

- Prevent names with a dash (“-”) or other special characters
- Requires names with only lowercase characters
- Limit total length to 128 characters

In the examples in this book, I choose names based on lowest common denominator {i.e., naming that considers all the “you can’t do this or that” rules}, consistently applied to all resources.

I do not adhere religiously to the guidance in the Cloud Adoption Framework; rather, I typically use the combination of values that my customer feels is appropriate to their implementation.

For example:

- Workload / Application: “rchapler”
- Environment: “dev,” “test,” “prod”
- Resource Type: “dl” (as in Data Lake)

...which results in resource naming like “rchapler-test-dl”

Note: Because I have authored this book over time and in relation to various projects, you might see different naming used in different exercises. I have tried to parameterize these naming choices with “<UseCase>” so you can insert a value appropriate to your work

Resource Instantiation

The instructions in this section are mostly about the use of Azure Portal to manually instantiate the resources necessary to complete the objectives in this book. In some cases, there are also instructions for PowerShell and Bash.

I have tried to keep this section free of repetitive instructions. For example:

- When the interface says **Subscription**, use the dropdown to select the appropriate subscription
- When the interface says **Resource Group**, select the appropriate resource group
- When the interface says **Region** (or something similar like Location), select the same region used by the resource group (or a different value, if appropriate)
- When the interface asks for a **Name**, enter a meaningful name aligned with standards
- When I mention that you need to use a resource {e.g., a Key Vault}, this should imply that you will need to have instantiated a Key Vault

I have minimized image use to hold page count below 300 (the limit for LinkedIn posts).

For each of the sub-sections, start the resource instantiation process by clicking the menu icon in the upper-left of the Azure Portal interface, and then click “**+ Create a resource**.”

Application Registration

(aka “Service Principal,” “Client”)

Create Application Registrations using **Azure Active Directory**. Complete the following steps:

- Navigate to “**Azure Active Directory**”
- Select “**App Registrations**” in the **Manage** group of the navigation pane
- Click “**+ New Registration**”
- Complete the “**Register an application**” form, enter a meaningful **Name** and then click **Register**
- Generate a Key Vault Secret for each Client Identifier and Client Secret

Data Lake

Create Resource

Complete the following steps:

- Complete the “**Create a Storage Account**” form, **Basics** tab.
- Complete the “**Create a Storage Account**” form, **Advanced** tab, check “**Enable Hierarchical Namespace**” in the “**Data Lake Storage Gen2**” grouping.
- Click the “**Review + create**” button, review configuration, and then click the **Create** button.

Container

Complete the following steps:

- Navigate to your Data Lake, and then **Containers** in the “**Data storage**” group
- Click “**+ Container**” and enter a meaningful **Name** value in the resulting pop-out
- Click the **Create** button

Storage Explorer

Complete the following steps:

- Download and install the Microsoft Azure Storage Explorer app from <https://azure.microsoft.com/en-us/features/storage-explorer/>
- Launch the app and provide Azure credentials.
- Click the “**Connect to Azure storage...**” link
- On the resulting pop-up, click the “**Add an Azure Account**” radio button and then click **Next**
- Authenticate with Azure credentials

Sample Data

Complete the following steps:

- Download sample files from a site like <https://filesamples.com/formats/csv>
- Return to the “Microsoft Azure **Storage Explorer**” app and navigate to your container
- Click the **Upload** button and select “Upload Files” in the resulting pop-up menu
- Click the ellipses button to the right of the “**Selected Files**” box
- Navigate to **Downloads**, select your sample data files, and then click the **Open** button
- Complete the “**Upload Files**” pop-up, click the **Upload** button

Data Migration Assistant

Complete the following steps:

- Browse to <https://www.microsoft.com/en-us/download/details.aspx?id=53595>
- Download, open, and install the “**User Installer**” for your platform

Data Share

Note: To demonstrate sharing data to a target, you must instantiate a second ADX instance.

Complete the following steps:

- Complete the “**Create Data Share**” form
- Click the “**Review + create**” button, validate settings, and then click the **Create** button

Sent Shares

Complete the following steps:

- Navigate to your Data Share, and then “**Sent Shares**” in the “**Data Share**” group
- Click “**+ Create**” and then complete the “**Sent Shares**” form
 - On the “**1. Details**” tab, including “**Share Type**” value “**In-place**” and then click the **Continue** button
 - On the “**2. Datasets**” tab, click the “**Add datasets**,” then select “**Azure Data Explorer**” on the resulting “**Select dataset type**” pop-out, and finally click **Next**
 - Complete the resulting “**Azure Data Explorer**” pop-out and then click the **Next** button
 - On the next pop-out, check the box next to your cluster and then click **Next**
 - On the final pop-out, click “**Add datasets**” and then **Continue**
 - On the “**3. Recipients**” tab, click “**Add recipient**”
 - In the resulting interface, enter an **Email** value and then click **Continue**
 - On the “**4. Review + Create**” tab, click **Create**

The designated email recipient can expect to receive an invitation email. Clicking the “**View invitation >**” link in the email will open the “**Data Share Invitations**” page where the invitee can complete the form and then click to “**Accept and Configure**.”

Received Shares

Complete the following steps:

- Navigate to your Data Share, and then “**Received Shares**” in the “**Data Share**” group
- Click the Datasets tab and then check the box next to your Data Explorer cluster
- Click “**+ Map to target**”, complete the resulting pop-out and then click “**Map to target**”

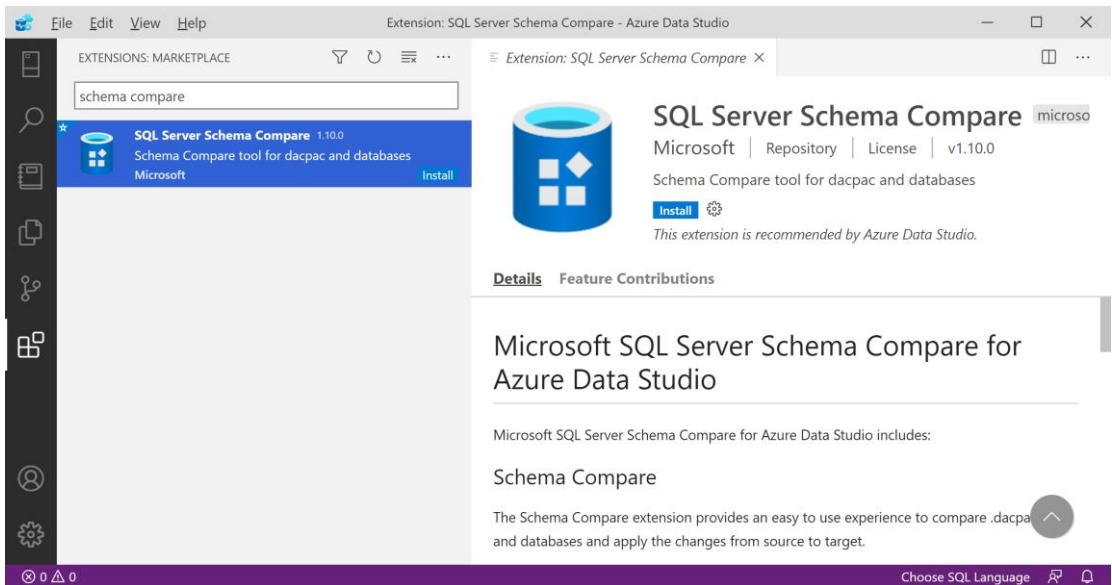
Data Studio

Complete the following steps:

- Browse to <https://docs.microsoft.com/en-us/sql/azure-data-studio/download-azure-data-studio?view=sql-server-ver15>
- Download, open, and install the “**User Installer**” for your platform

Add Extensions

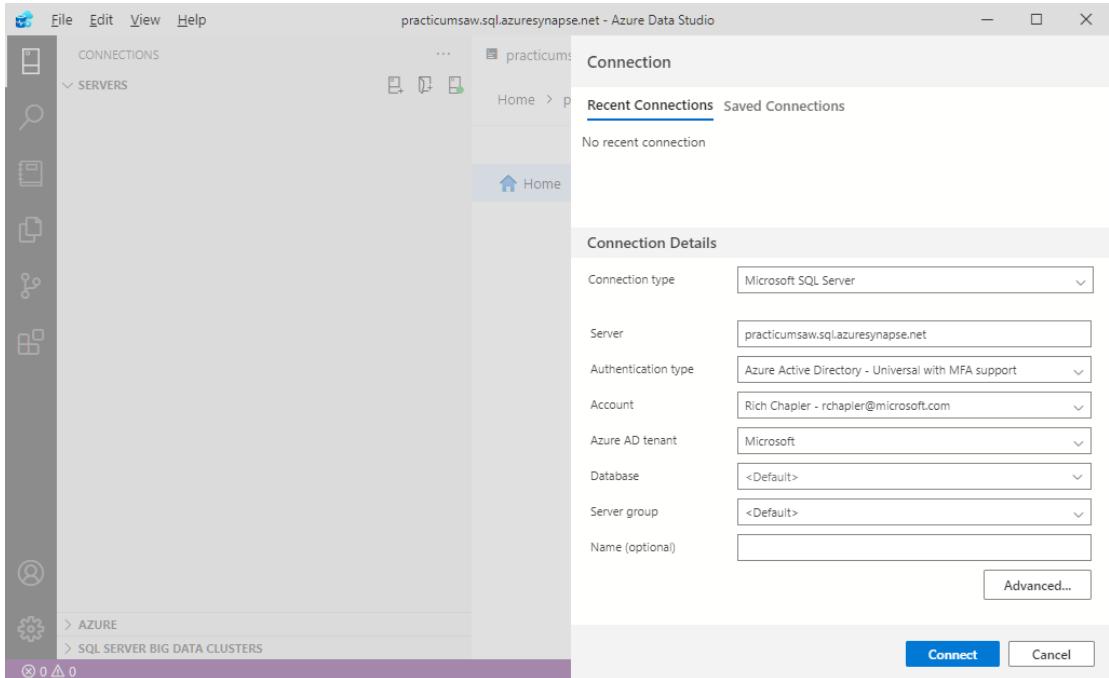
Launch Data Studio and then navigate to **Extensions** in the navigation pane.



Search for “schema compare” and install the “**SQL Server Schema Compare**” extension.

Create Connection

Navigate to **Connections** in the navigation pane.



Click the “Create Connection” icon (first of three) to the right of the **SERVERS** section header.

Complete the “**Connection Details**” pop-out, including:

Connection Type	Confirm default value, “ Microsoft SQL Server ”
Server	Example format: myserver.sql.azuresynapse.net
Authentication Type	Select “ Azure Active Directory – Universal with MFA support ”
Account	Confirm credentials
Azure AD Tenant	Confirm tenant

Click the **Connect** button.

Databricks

Complete the “**Create an Azure Databricks workspace**” form, including:

Pricing Tier	Select “ Premium... ” to enable features that we will need for this exercise
---------------------	-------------------------------------------------------------------------------------

Click the “**Review + create**” button, review configuration, and then click the **Create** button.

New Cluster

Navigate to the **Overview** page in Databricks, click the “**Launch Workspace**” button, and login if required.

The screenshot shows the Azure Databricks workspace. At the top, there's a navigation bar with tabs for 'practicumdb - Microsoft Azure' and 'Databricks'. Below the navigation bar, the main header reads 'Azure Databricks'. On the left, a sidebar menu includes icons for Home, Workspace, Recents, Data, Clusters, Jobs, Models, and Search. The 'Clusters' icon is highlighted. The main content area has three main sections: 'Explore the Quickstart Tutorial' (with a note to spin up a cluster in 5 minutes), 'Import & Explore Data' (with a 'Drop files or click to browse' area), and 'Create a Blank Notebook' (with a note to start querying and visualizing data). Below these sections, there are three tabs: 'Common Tasks', 'Recents', and 'Documentation'. The 'Common Tasks' tab is active, showing links for New Notebook, Create Table, New Cluster, New Job, New MLflow Experiment, Import Library, and Read Documentation. The 'Recents' tab shows a placeholder for recent files. The 'Documentation' tab lists links for Documentation, Release Notes, and Getting Started.

In the “**Common Tasks**” grouping, click “**New Cluster**.”

Create Cluster

New Cluster

Cluster Name: practicumdbc

Cluster Mode: Standard

Pool: None

Databricks Runtime Version: Runtime: 7.4 (Scala 2.12, Spark 3.0.1)

Autopilot Options:

- Enable autoscaling
- Terminate after 120 minutes of inactivity

Worker Type: Standard_DS3_v2

Driver Type: Same as worker

This Runtime version supports only Python 3.

Create Cluster

Complete the “Create Cluster” form and then click the “Create Cluster” button.

New Notebook

Return to the start page and then, in the “Common Tasks” grouping, click “New Notebook.”

The screenshot shows the Azure Databricks start page. On the left, there's a sidebar with icons for Home, Workspace, Recents, Data, Clusters, Jobs, Models, and Search. The main area has tabs for Common Tasks, Recents, and Documentation. In the Common Tasks section, there's a "Create Notebook" button. A modal dialog titled "Create Notebook" is open, showing fields for Name (set to "practicumdb"), Default Language (set to "Python"), and Cluster (set to "practicumdbc"). There are "Cancel" and "Create" buttons at the bottom of the dialog. To the right of the dialog, there's a "Create a Blank Notebook" section with a "Create" button.

Enter a meaningful name and click the **Create** button.

The screenshot shows the Azure Databricks workspace. The sidebar is identical to the start page. The main workspace shows a notebook titled "practicumdbn (Python)". The notebook interface includes a toolbar with various icons, a code editor window with a single line of code "1", and a status bar at the bottom with the text "Shift+Enter to run shortcuts".

Make note of the URL {i.e., <http://adb-21458...>} for use in the next section.

Secret Scope

Navigate to your Key Vault, and then click **Properties** in the **Settings** group of the navigation pane.

practicumkv | Properties

Name	practicumkv
Sku (Pricing tier)	Standard
Location	westus2
Vault URI	https://practicumkv.vault.azure.net/
Resource ID	/subscriptions/[REDACTED]
Subscription ID	[REDACTED]
Subscription Name	rchapler
Directory ID	[REDACTED]
Directory Name	Microsoft
Soft-delete	
Days to retain deleted vaults	90
Purge protection	<input checked="" type="radio"/> Disable purge protection (allow key vault and objects to be purged during retention period) <input type="radio"/> Enable purge protection (enforce a mandatory retention period for deleted vaults and vault objects)

Make note of the values in “Vault URI” and “Resource ID.”

You will replace **{databricksInstance}** with the start of the URL in your workspace, from my exercise, example: <https://adb-2154823451042175.15.azuredatabricks.net/#secrets/createScope>

The screenshot shows the Microsoft Azure Databricks portal with the URL <https://adb-2154823451042175.15.azuredatabricks.net/?o=2154823451042175#secrets/createScope>. The page title is "Create Secret Scope". On the left, there is a sidebar with icons for Home, Workspace, Recents, Data, Clusters, Jobs, and Models. The main content area has a heading "Create Secret Scope" with "Cancel" and "Create" buttons. Below this, a note says "A store for secrets that is identified by a name and backed by a specific store type. [Learn more](#)". There are two sections: "Scope Name" (containing "practicumdbss"), "Manage Principal" (containing "Creator"), "Azure Key Vault" (with "DNS Name" set to "https://practicumkv.vault.azure.net/" and "Resource ID" set to "/46ed/resourceGroups/practicumrg/providers/Microsoft.KeyVault/vaults/practicumkv").

Complete the “**Create Secret Scope**” form, including values for:

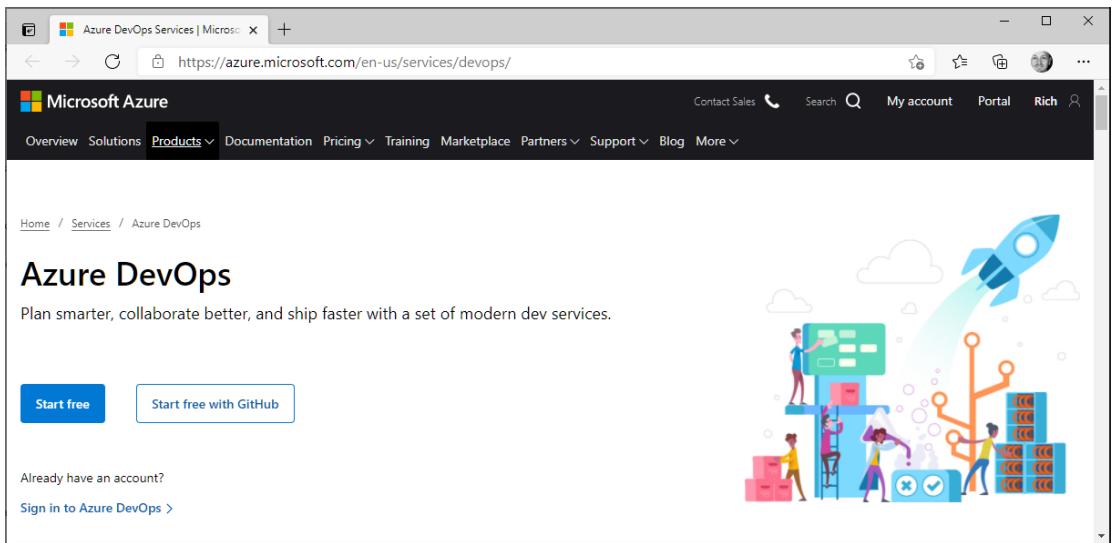
DNS Name Paste the copied “**Vault UI**” value

Resource ID Paste the copied “**Resource ID**” value

Click the **Create** button.

DevOps

Browse to <https://azure.microsoft.com/en-us/services/devops/>



Click the “**Start free with GitHub**” button.

On the resulting Azure DevOps form, click the “**New organization**” link on the navigation pane.

Click the **Continue** button.

Complete the “**Almost done...**” form and then click the **Continue** button.

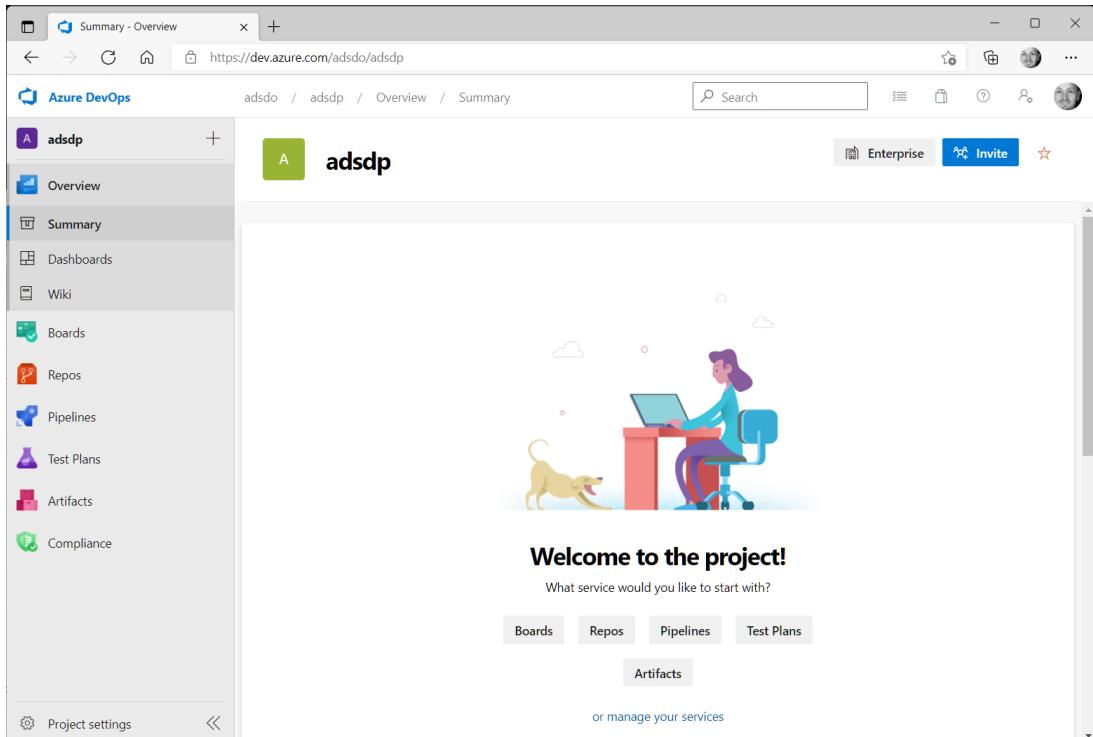
Create Project

Complete the “**Create a project to get started**” form, including:

Visibility

Select the value that best aligns with your requirements

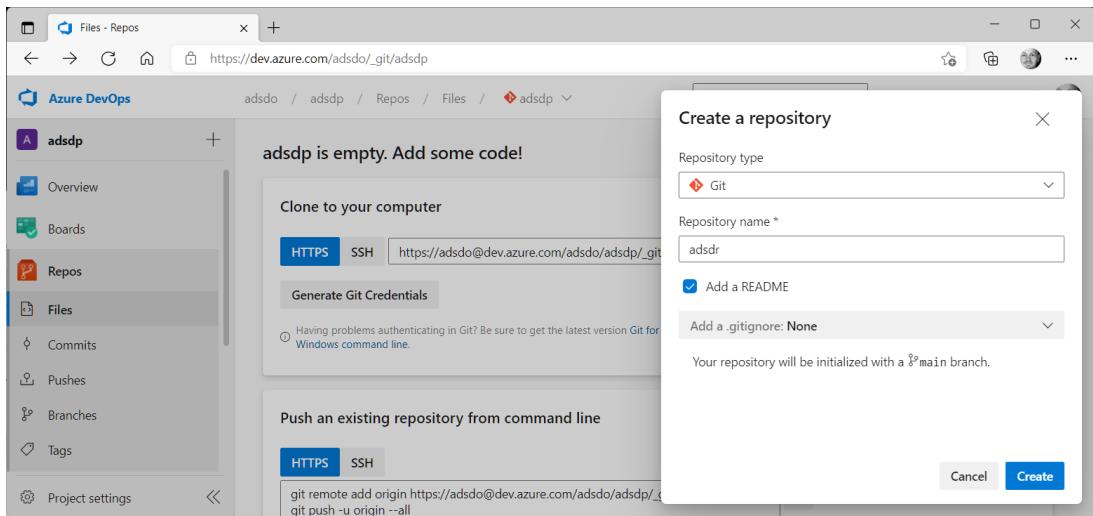
Click the “**+ Create project**” button.



When processing is complete, you will be directed to a “Welcome to the project!” screen.

Create Repository

Click **Repos** in the navigation pane, then click the “+” button at the top-right of the navigation pane and select “**New repository**” from the resulting drop-down menu.



Complete the “**Create a repository**” form, including:

Repository Type	Confirm default selection, Git
------------------------	---------------------------------------

Click the “**Create**” button.

Event Hub

(aka “Event Hub Namespace”)

Complete the “**Create Namespace**” form, including:

Pricing Tier

Select the pricing tier appropriate for your use case

Click the “**Review + create**” button, review configuration, and then click the **Create** button.

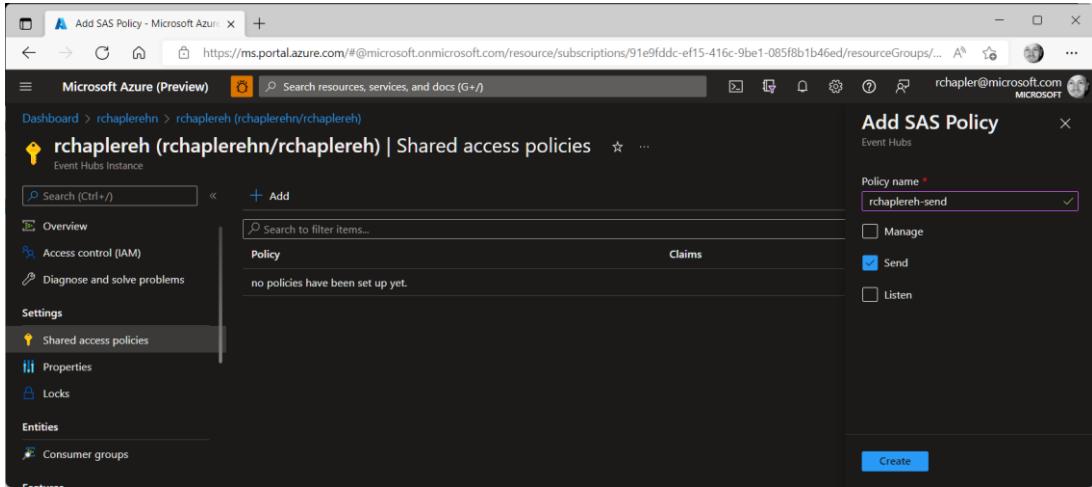
On the Overview page of the newly created Event Hub Namespace, click “+ Event Hub”

Complete the “**Create Event Hub**” form.

Click the “**Review + create**” button, review configuration, and then click the **Create** button.

Shared Access Policies

Navigate to the Event Hub Namespace (not a specific Event Hub) and click on “**Shared access policies**” in the **Settings** group of the navigation pane. Click “+ Add.”



On the resulting pop-out, enter a meaningful value for “Policy name” and click to select appropriate permissions {e.g., **Send**}.

Click the **Create** button.

The screenshot shows the Microsoft Azure portal interface for managing Shared access policies. On the left, the navigation bar includes 'Dashboard', 'rchaplerehn', 'Event Hubs Namespace', 'Overview', 'Activity log', 'Access control (IAM)', 'Tags', 'Diagnose and solve problems', 'Events', 'Settings' (selected), 'Scale', 'Geo-Recovery', and 'Networking'. The main content area is titled 'SAS Policy: RootManageSharedAccessKey' and shows the following details:

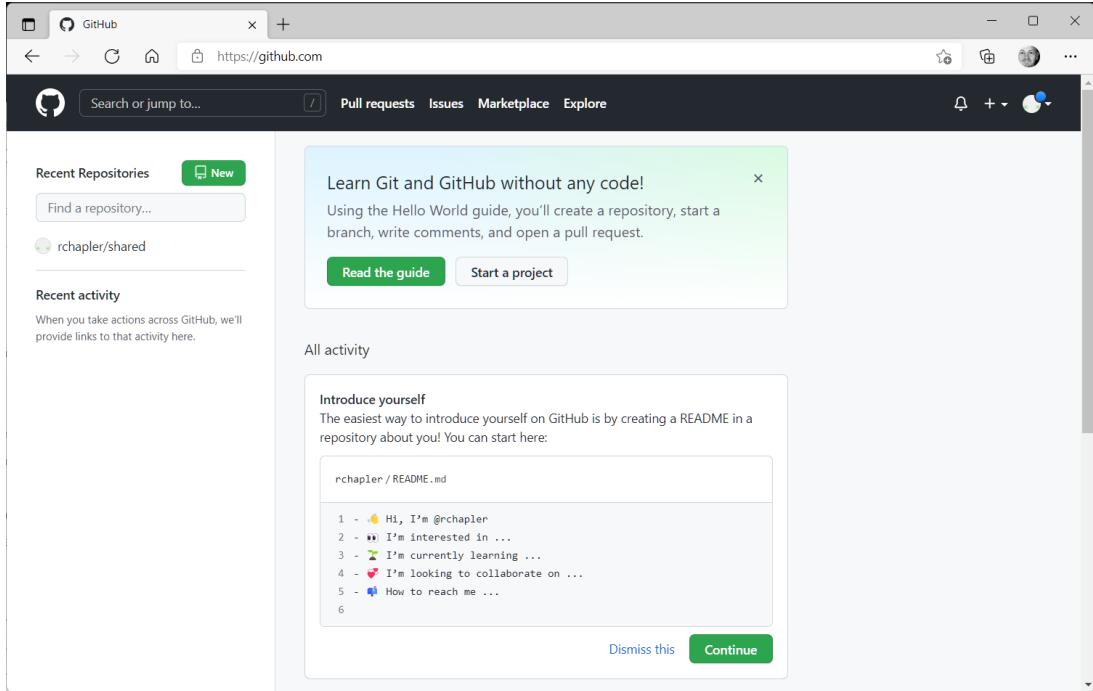
- Policy:** RootManageSharedAccessKey
- Claims:** Manage, Send, Listen
- Primary key:** a14VwmBc48dur5GxLczSckKQ7vtvNYHRIk4AUhrDeXk=
- Secondary key:** Z+BJLOFTH52Dja8ChdUd8DnCrj1eA7s5oWTw3yPQQ=
- Connection string-primary key:** Endpoint=sb://rchaplerehn.servicebus.windows.net/SharedA...
- Connection string-secondary key:** Endpoint=sb://rchaplerehn.servicebus.windows.net/SharedA...

Return to the added policy to copy keys and connection strings.

GitHub Enterprise

Browse to <https://github.com/login> and sign in (or sign up, as needed).

Create Repository



The screenshot shows the GitHub Enterprise homepage. On the left, there's a navigation pane with 'Recent Repositories' and a search bar. A green 'New' button is positioned next to the 'Recent Repositories' heading. Below it is a 'Find a repository...' search input field. To the right of the navigation is a main content area. A prominent green callout box titled 'Learn Git and GitHub without any code!' provides instructions on how to use the Hello World guide to create a repository, start a branch, write comments, and open a pull request. It includes two buttons: 'Read the guide' and 'Start a project'. Further down, there's a section titled 'Introduce yourself' with a sub-section 'README.md'. It shows a sample README file content with numbered items 1 through 6, each preceded by an emoji icon. At the bottom of this section are 'Dismiss this' and 'Continue' buttons.

Click the **New** button to the right of “**Recent Repositories**” in the navigation pane.

The screenshot shows the GitHub interface for creating a new repository. At the top, the title 'Create a New Repository' is visible, along with the URL 'https://github.com/new'. The main heading 'Create a new repository' is centered above a brief description: 'A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.' Below this, the 'Owner' field is set to 'rchapler' and the 'Repository name' field is 'adsgr'. A note suggests using short and memorable names like 'cuddly-octo-spork'. There is a 'Description (optional)' input field which is currently empty. Under 'Visibility', the 'Private' option is selected, indicated by a radio button and the text 'You choose who can see and commit to this repository.'. Below this, there's a section titled 'Initialize this repository with:' containing several optional checkboxes: 'Add a README file', 'Add .gitignore', and 'Choose a license'. Each checkbox has a descriptive subtitle underneath. At the bottom of the form is a prominent green 'Create repository' button.

© 2022 GitHub, Inc. Terms Privacy Security Status Docs Contact GitHub Pricing API Training Blog About

Complete the “Create a new repository” form, including:

Public or Private Choose Private (unless you have a specific reason for using Public)

Click the “Create repository” button.

Key Vault

Complete the “**Create Key Vault**” form, including:

Pricing Tier	Confirm default selection, Standard
Days to Retain...	Confirm default value, 90

Click the “**Review + create**” button, review configuration, and then click the **Create** button.

Key Vault Secret

Open a new tab and navigate to your Key Vault.

Click **Secrets** in the **Settings** group of the navigation pane and then click the “**+ Generate/Import**” button.

Complete the “**Create a secret**” form, including:

Value	Paste the previously copied secret value
-------	------------------------------------------

Click the **Create** button.

Log Analytics

Navigate to your Resource Group and click the “**+ Create**” button.

On the resulting “**Create a resource**” form, search for and select “**Log Analytics Workspace**” from the “**Search services and marketplace**” textbox.

On the resulting “**Log Analytics Workspace**” form, click the **Create** button.

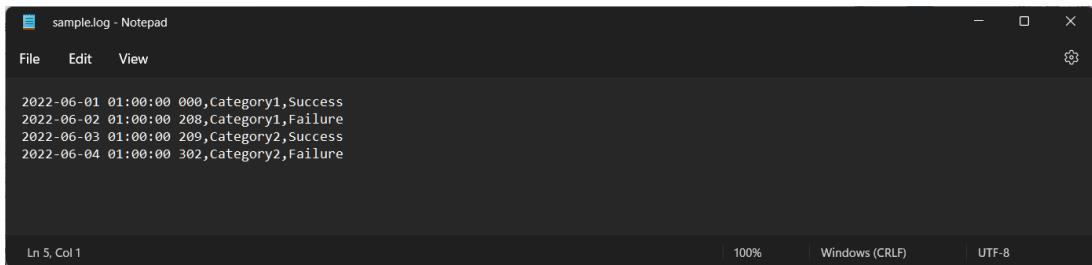
Complete the resulting “**Create Log Analytics workspace**” form on the **Basics** tab.

Click the “**Review + create**” button, review configuration, and then click the **Create** button.

Custom Log

Note: Collection of data from custom logs implies use of an agent on an on-prem device.

Use a text editor to create a sample log file.



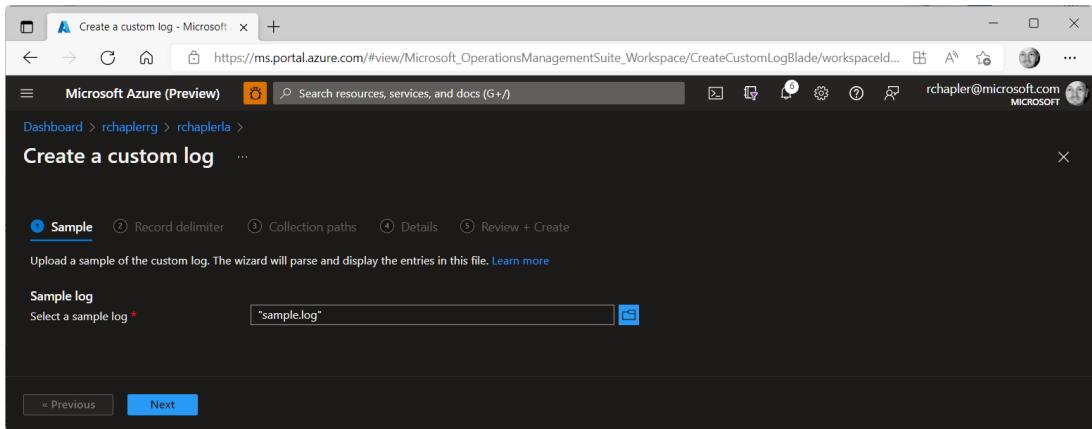
The screenshot shows a Windows Notepad window titled "sample.log - Notepad". The window contains the following log entries:

```
2022-06-01 01:00:00 000,Category1,Success
2022-06-02 01:00:00 208,Category1,Failure
2022-06-03 01:00:00 209,Category2,Success
2022-06-04 01:00:00 302,Category2,Failure
```

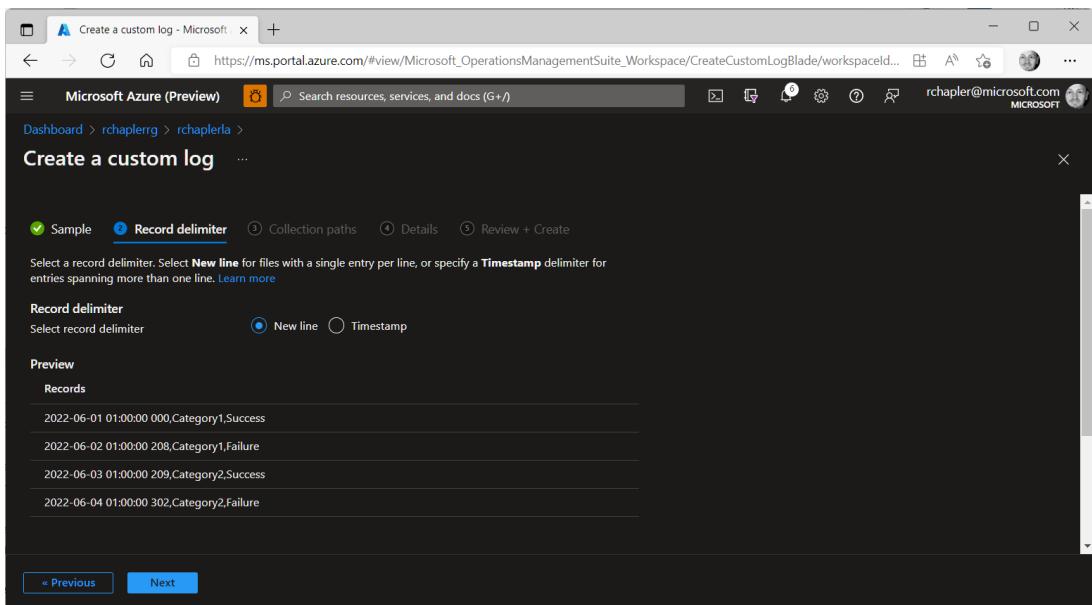
The status bar at the bottom of the Notepad window displays "Ln 5, Col 1", "100%", "Windows (CRLF)", and "UTF-8".

Navigate to Log Analytics and then click “**Custom logs**” in the **Settings** group of the navigation pane.

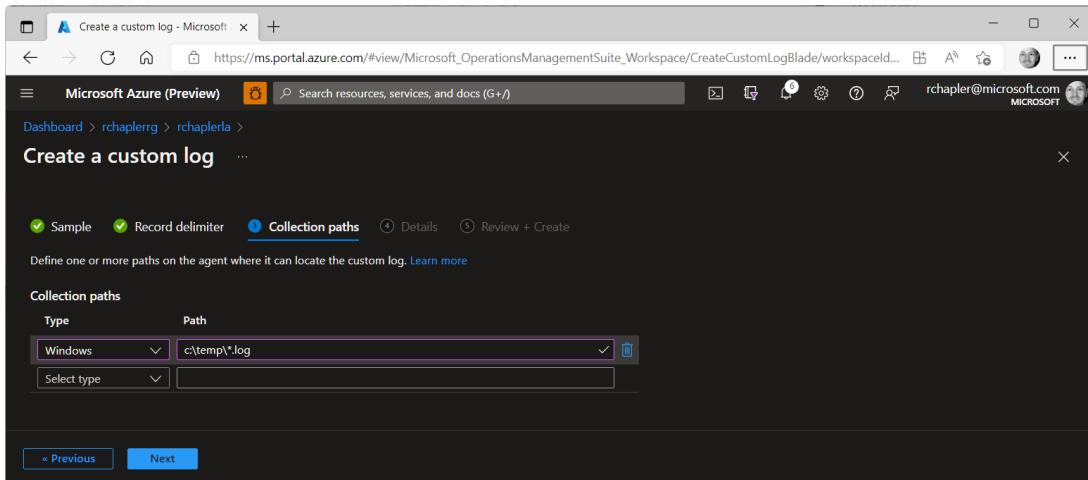
Click “**+ Add custom log**” on the resulting page.



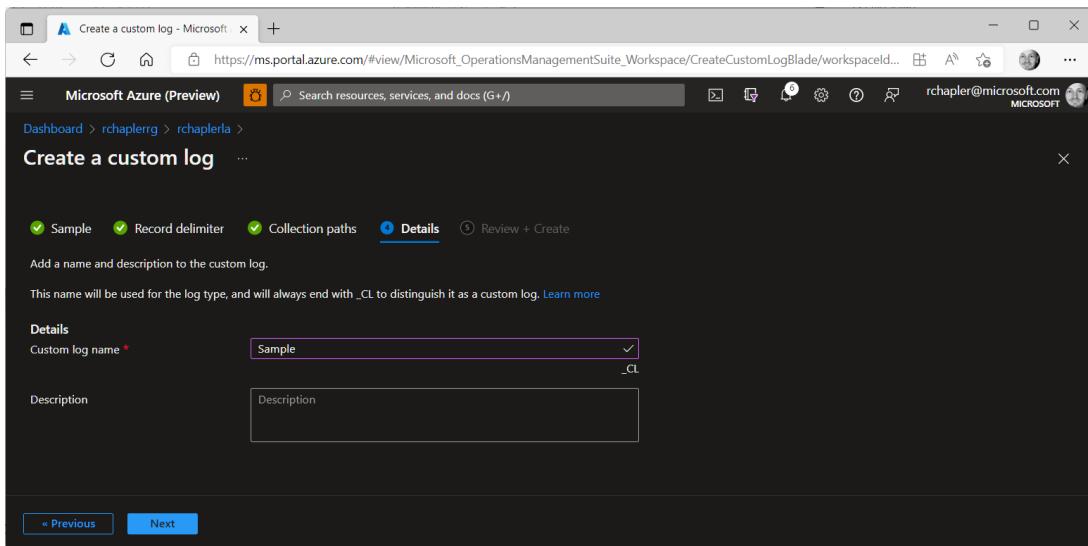
Complete the “Create a custom log” form, **Sample** tab, select your sample log file, confirm successful upload, and then click the **Next** button.



Complete the “Create a custom log” form, “Record delimiter” tab, confirm default setting “New line” and then click the **Next** button.



Complete the “Create a custom log” form, “Collection paths” tab, select values for **Type** and **Path** describing the location where log files will be found (over time). Click the **Next** button.



Complete the “Create a custom log” form, **Details** tab, enter a “Custom log name” value and then click the **Next** button.

Complete the “Create a custom log” form, “Review + create” tab, review configuration, and then click the **Create** button.

Logic App

Complete the “**Create Logic App**” form, including:

Plan	Select the plan type appropriate for your use case <i>Consumption will work in most cases</i>
Windows Plan...	Click “ Create new ” and provide a name for an App Service Provider
SKU and Size	Click “ Change size ” and pick values appropriate to your use case
Zone Redundancy	Select the value appropriate for your use case <i>Disabled will work in most cases</i>

Click the “**Review + create**” button, review configuration, and then click the **Create** button.

Metrics Advisor

Complete the “**Create Metrics Advisor**” form, including:

Pricing Tier	Select S0
Bring Your Own...	Confirm default selection, “No”
I confirm...	Check the box

Click the “**Review + create**” button, review configuration, and then click the **Create** button.

Add Permissions to Data Explorer

Navigate to Data Explorer, and then click **Permissions** in the navigation pane.

Click “**+ Add**” on the resulting page

Select **AllDatabasesViewer** from the resulting drop-down menu.

Search for and then click to **Select** Metrics Advisor.

Purview

(aka "Microsoft Purview", "Azure Purview")

Complete the “Create Purview account” form.

Managed Resource Group...	This additional Resource Group will hold ancillary resources created specifically for Synapse
----------------------------------	-----------------------------------------------------------------------------------------------

Note: Because you are likely to have more than one managed Resource Group, consider using a consistent naming pattern {e.g., "<UseCase>mrg-p" for Purview and "<UseCase>mrg-x" for X}

Click the “Review + Create” button, review configuration, and then click the **Create** button.

Add Access Policy to Key Vault

Navigate to the Key Vault.

The screenshot shows the Microsoft Azure portal interface. The user is viewing the 'practicumkv' key vault. The left navigation pane shows options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Events, Keys, Secrets, Certificates, Access policies, and Networking. The 'Access policies' option is under the 'Settings' group. The main content area displays the 'Essentials' section with details such as Resource group (practicumrg), Location (West US 2), Subscription (rchapler), and Vault URI (https://practicumkv.vault.azure.net/). The 'Access policies' section is expanded, showing a table with columns 'Name', 'Principal type', 'Start time', and 'End time'. There is a single row with 'Name' as 'rchapler' and 'Principal type' as 'User'. At the bottom of this section, there is a note: 'Click here to add tags'.

Click “Access Policies” in the **Settings** group of the navigation pane.

The screenshot shows the 'Access policies' section of the Azure portal. On the left, a sidebar lists 'Overview', 'Activity log', 'Access control (IAM)', 'Tags', 'Diagnose and solve problems', 'Events', 'Keys', 'Secrets', 'Certificates', and 'Access policies'. The 'Access policies' item is selected. The main area has a search bar and buttons for 'Save', 'Discard', and 'Refresh'. It includes sections for 'Enable Access to:' (checkboxes for Azure VM deployment, ARM template deployment, and Azure Disk Encryption), 'Permission model' (radio buttons for 'Vault access policy' and 'Azure role-based access control', with 'Vault access policy' selected), and a '+ Add Access Policy' button. A table titled 'Current Access Policies' is shown with columns: Name, Email, Key Permissions, Secret Permissions, Certificate Permissions, and Action. The table currently displays one row labeled 'APPLICATION'.

Click the “+ Add Access Policy” link.

The screenshot shows the 'Add access policy' dialog. On the left, there are dropdown menus for 'Configure from template (optional)', 'Key permissions' (set to '0 selected'), 'Secret permissions' (set to '2 selected' with 'Select all' checked), and 'Certificate permissions'. Below these are sections for 'Select principal *' (dropdown menu) and 'Authorized application' (radio button). A large blue 'Add' button is at the bottom left. On the right, a search bar finds 'practicump'. A list of results shows 'practicump' with ID 'd0206ad4-fa5d-4923-982a-4cf48bcf3559' and the status 'Selected'. A 'Selected items' list contains the same entry. A 'Select' button is at the bottom right.

Complete the “Add access policy” form, including:

Secret Permissions Select **Get** and **List** in the drop-down menu

Select Principal Search for, and then **Select** the managed identity for Purview

Click the **Add** button.

The screenshot shows the Microsoft Azure portal interface. The left sidebar has a 'Key vault' icon selected under 'Access policies'. The main content area is titled 'practicumkv | Access policies'. It displays a message: 'Please click the 'Save' button to commit your changes.' Below this, there's a section 'Enable Access to:' with three checkboxes: 'Azure Virtual Machines for deployment', 'Azure Resource Manager for template deployment', and 'Azure Disk Encryption for volume encryption'. The 'Vault access policy' radio button is selected under 'Permission model'. A table titled 'Current Access Policies' shows one entry: 'practicump' with '0 selected' under 'Key Permissions', '2 selected' under 'Secret Permissions', and '0 selected' under 'Certificate Permissions'. There is a 'Delete' button at the bottom right of the table row.

Click the **Save** button.

Connect Key Vault

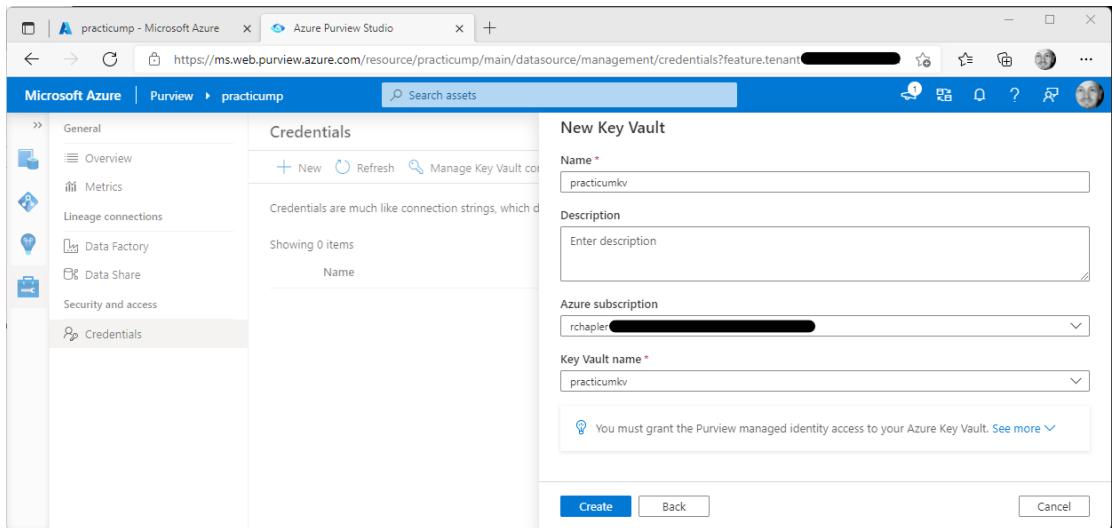
Navigate to Purview and click the “**Open Purview Studio**” button.

Select the **Management** icon on the navigation pane.

Click **Credentials** in the “**Security and access**” group of the resulting menu.

Click the “**Manage Key Vault connections**” button on the resulting page.

Click the “**+ New**” button the resulting pop-out.



Complete the “**New Key Vault**” pop-out, including:

Key Vault Name

Select your Key Vault

Click the **Create** button.

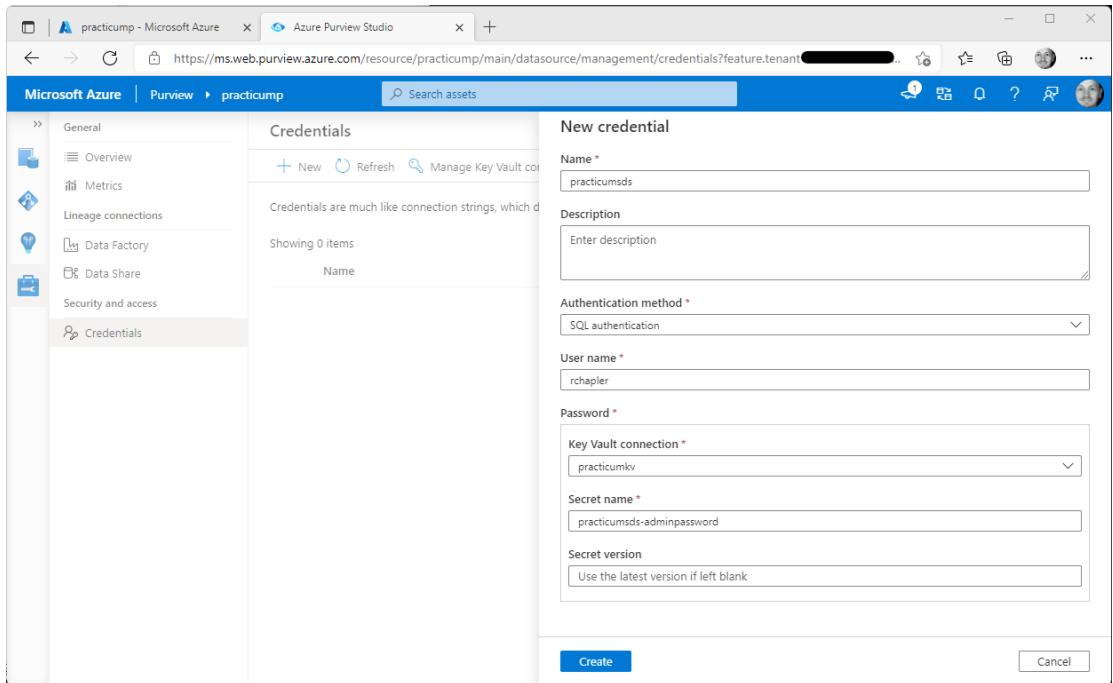
On the resulting “**Confirm granting access**” pop-up, review and then click the **Confirm** button.

Confirm the addition of Key Vault on the “**Manage Key Vault connections**” pop-out.

Click the **Close** button.

Add New Credential

On the **Credentials** form, click the “**+ New**” button.



The screenshot shows the Azure Purview Studio interface. In the top navigation bar, there are tabs for 'practicump - Microsoft Azure', 'Azure Purview Studio', and a search bar. Below the navigation is a left sidebar with icons for General, Overview, Metrics, Lineage connections, Data Factory, Data Share, Security and access, and Credentials (which is selected). The main content area has a title 'Credentials' with a sub-section 'New credential'. The 'New credential' form contains the following fields:

- Name ***: practicumsds
- Description**: Enter description
- Authentication method ***: SQL authentication
- User name ***: rchabler
- Password ***: (password field)
- Key Vault connection ***: practicumkv
- Secret name ***: practicumsds-adminpassword
- Secret version**: Use the latest version if left blank

At the bottom right of the form are 'Create' and 'Cancel' buttons.

On the resulting “**New Credential**” pop-out, including:

Authentication	Select “SQL authentication”
User Name	Enter the “server admin login” value used during creation of the SQL Database Server
Key Vault Connection	Select your key vault
Secret Name	Select your secret

Click the **Create** button.

*Note: Purview requires “**Storage Blob Data Reader**” permissions to scan a Data Lake for blobs.*

Resource Group

Complete the “**Create a Resource Group**” form, including a value for:

Region	Select a region appropriate for your situation; take into consideration that: <ul style="list-style-type: none">• Some regions {e.g., West US and East US} see higher demand than others• Creation of resources in the same region offers best performance and lowest cost
---------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Click the “**Review + create**” button, review configuration, and then click the **Create** button.

Role Assignment

Create using PowerShell

Complete the following steps:

- Navigate to Cloud Shell (mount to new / existing storage, if required)
- Select **PowerShell** from the “**Select Environment**” drop-down menu
- Update and execute the following command to create a Service Principal and give it Contributor access to the subscription:

```
az ad sp create-for-rbac --name "<UseCase>ar" --role contributor --scopes  
/subscriptions/{Subscription GUID}
```

```
az ad sp create-for-rbac --name "rchaplerar" --role contributor --scopes  
/subscriptions/91e9fddc-ef15-416c-9be1-085f8b1b46ed/resourceGroups/rchaplerrg
```

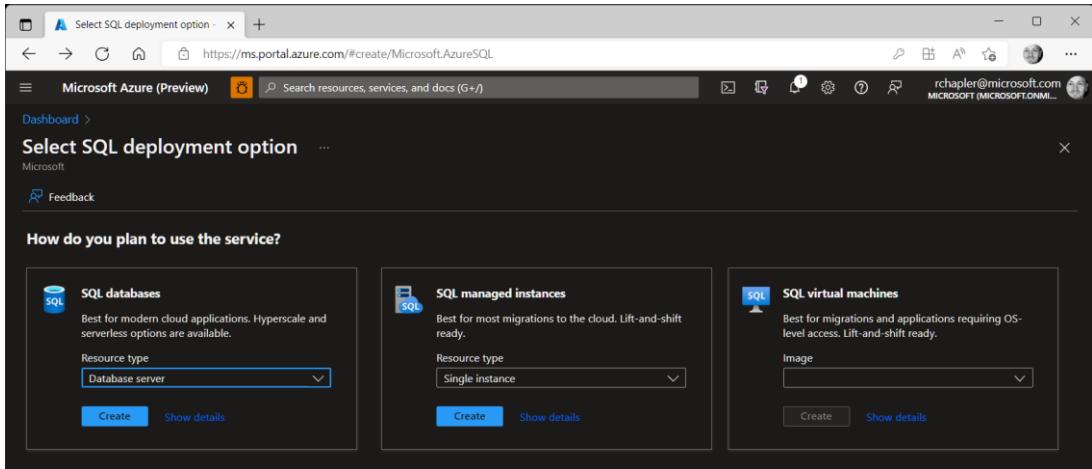
You can expect a response like the following redacted example:

```
{  
  "appId": "{ClientId}",  
  "displayName": "rchaplerar",  
  "password": "{Password}",  
  "tenant": "{TenantId}"  
}
```

SQL

(aka “Azure SQL”, “Azure SQL Server”, “Azure SQL Database”)

Create Database Server



Complete the “Select SQL deployment option” form, “SQL databases” grouping, select “**Database Server**” from the Resource Type drop-down menu and then click the **Create** button.

Complete the “Create SQL Database” form, **Basics** tab, including:

Authentication Method	Select “ Use only Azure Active Directory (Azure AD) authentication ” <i>Note: Though “SQL authentication” may be appropriate to some use cases, I do not typically recommend it</i>
Set Azure AD Admin	Click the “ Set admin ” link and on the resulting pop-out, search for and select the appropriate item

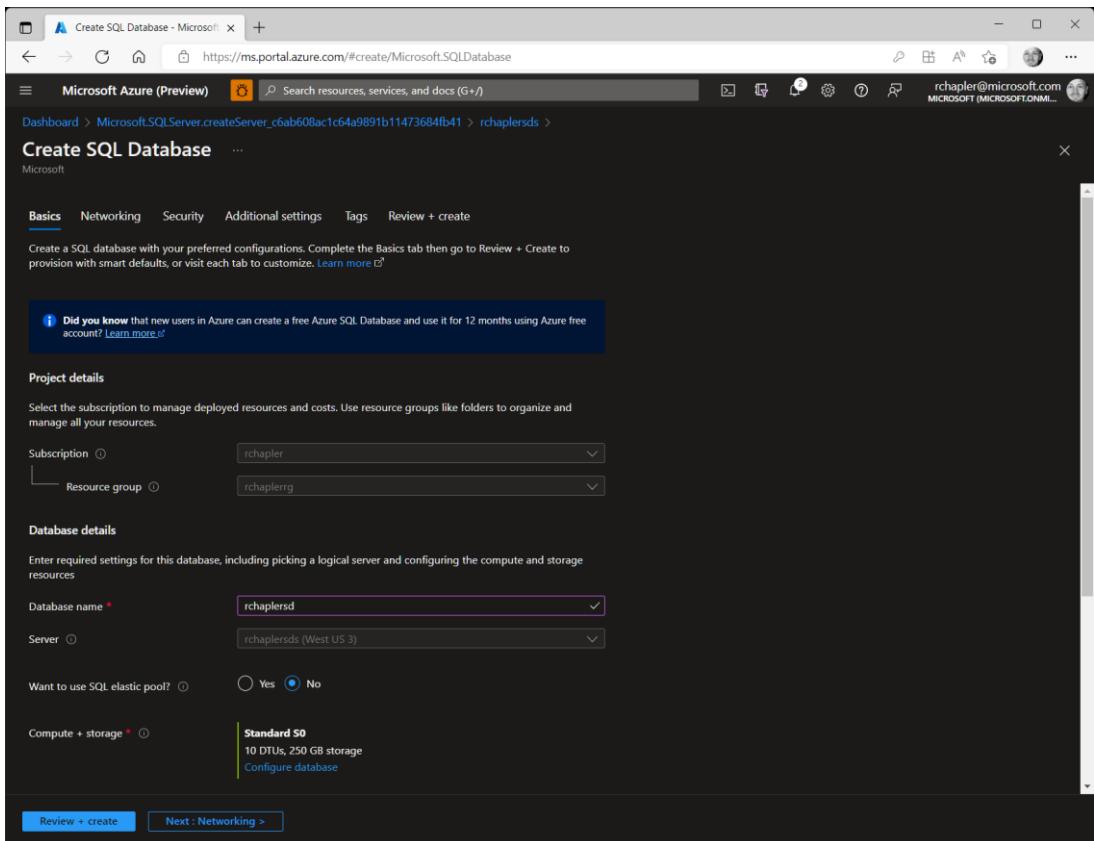
Navigate to the **Networking** tab, and including:

Allow Azure services...	Select Yes <i>Note: This setting may be inappropriate for some use cases</i>
--------------------------------	----------------------------------------------------------------------------------------

Click the “**Review + create**” button, review configuration, and then click the **Create** button.

Create Database

On the **Overview** page of the new Azure SQL Database Server, click the “+ Create database” button.



Complete the “Create SQL Database” form, **Basics** tab, including:

...SQL elastic pool?	Confirm default value, No
Compute + Storage	Click the “Configure database” link and review / update default values on the resulting Configure page <i>Note: Your use case should drive your choice, but barring good reason, I typically stay with default Service Tier and minimum values for DTUs and Data Max Size</i>
...Storage Redundancy	Select appropriate value

Add a sample database, if needed, on the “Additional settings” tab.

Click the “Review + create” button, review configuration, and then click the **Create** button.

Create User

Note: This can be used for both System-Assigned Managed Identities and User-Assigned Managed Identities

Navigate to your Azure SQL Database and select “**Query Editor...**” from the navigation pane.

Modify and then execute the following T-SQL:

```
CREATE USER [<UseCase><ResourceAbbreviation>] FROM EXTERNAL PROVIDER
```

```
ALTER ROLE db_datareader ADD MEMBER [<UseCase><ResourceAbbreviation>]
```

Storage Account

Complete the “**Create Storage Account**” form, including:

Performance	Confirm default radio button selection, “Standard”
Account Kind	Confirm default selection, “StorageV2 (general purpose v2)”
Replication	Confirm default selection, “Read-access geo-redundant storage (RA-GRS)”

Click the “**Review + create**” button, review configuration, and then click the **Create** button.

Synapse

(aka “Azure Synapse Analytics”)

Create Synapse Workspace

Complete the “Create Synapse workspace” form, including:

Managed Resource Group	This additional Resource Group will hold ancillary resources created specifically for Synapse
<i>Note: Because you are likely to have more than one managed Resource Group, consider using a consistent naming pattern {e.g., “<UseCase>mrg-s” for Synapse and “<UseCase>mrg-x” for X}</i>	
Select Data Lake Storage Gen2	Confirm default, “From subscription”
Account Name	Select an existing data lake or click the “Create new” link to create a new data lake
File System Name	Select an existing data lake or click the “Create new” link to create a new data lake file system
Assign myself...	Check to assign necessary permissions

Click the “Next: Security >” button and then select the “Use only Azure Active Directory (Azure AD) authentication” radio button.

Note: Though local authentication may be appropriate to some use cases, I do not typically recommend it

Click the “Review + create” button, review configuration, and then click the **Create** button.

Configure Repository

Navigate to Synapse Studio, click the **Manage** navigation icon, select “**Git configuration**” from the “**Source control**” grouping in the resulting navigation and then click the “**Configure**” button.

On resulting “**Configure a repository**” pop-out, including:

Repository Type	Select “ Azure DevOps Git ” (or “GitHub...”) from the drop-down menu
Azure Active Directory	Select the value appropriate for your organization

Click the **Continue** button.

On the second “**Configure a repository**” pop-out, including:

Azure DevOps Organization...	Select your DevOps account
Project Name	Select your DevOps project
Repository Name	Select the repo created with your DevOps project
Collaboration Branch	Drop-down menu, click “ + Create new ”, enter an appropriate name in the resulting pop-up, and click Create
Publish Branch	Confirm default, workspace_publish
Root Folder	Confirm default, /
Import Existing Resources...	Unchecked

Click the **Apply** button.

Create Credentials

Note: This exercise requires a User-Assigned Managed Identity

Navigate to Synapse Studio, click the **Manage** navigation icon, select **Credentials** from the **Security** grouping in the resulting navigation. Click the “**+ New**” button.

On resulting “**New credential**” pop-out, including:

Type	Select “ User Assigned Managed Identity ”
User Assigned Managed Identities	Select your User Assigned Managed Identity

Click the **Create** button. **Publish** or **Commit** change (as appropriate).

Create Integration Runtime

Navigate to Synapse Studio, click the **Manage** navigation icon, select “**Integration runtimes**” from the **Integration** grouping in the resulting navigation. Click the “**+ New**” button.

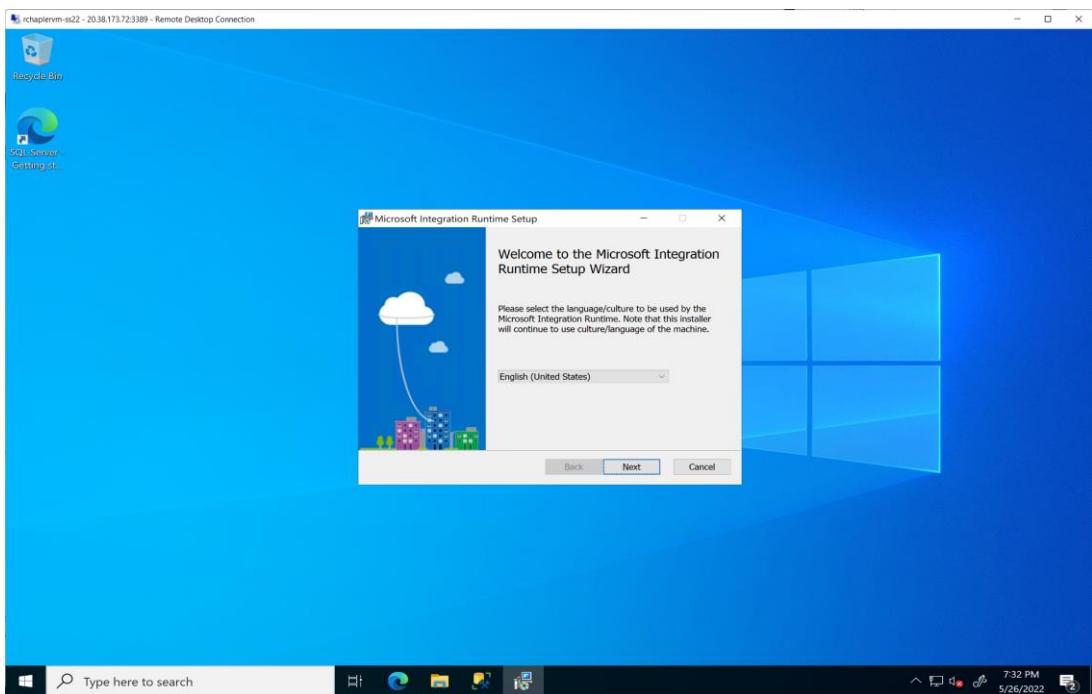
On the resulting pop-out, select “**Azure, Self-Hosted**” and then click the **Continue** button.

On the next pop-out, select “**Self-Hosted**” and then click the **Continue** button.

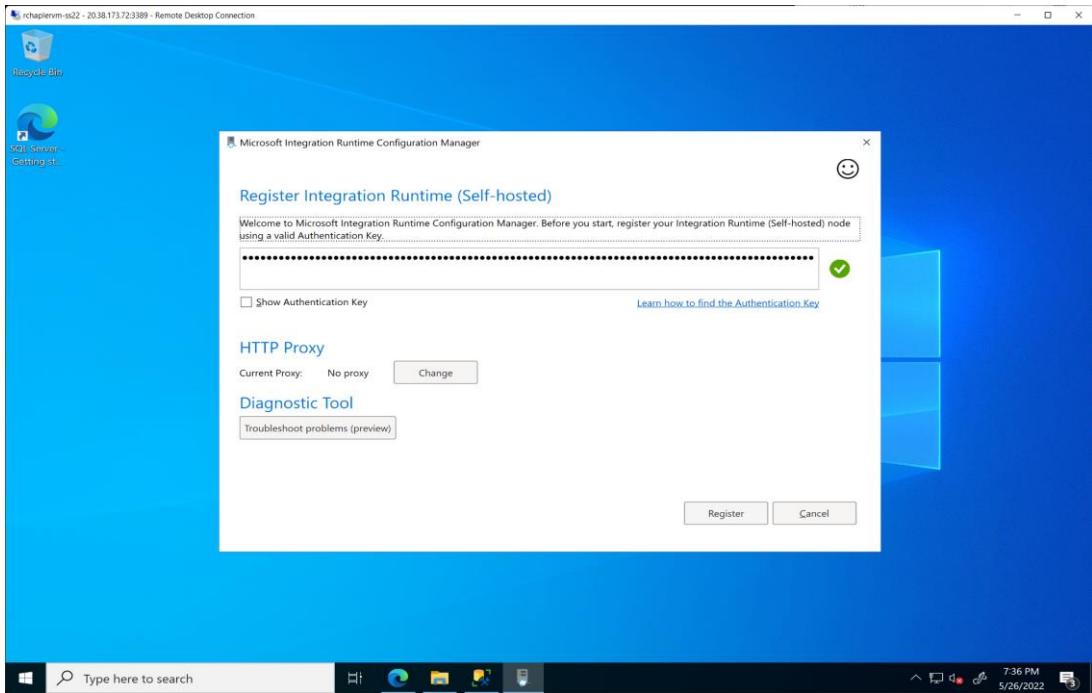
On the next pop-out, enter a value for **Name** and then click the **Create** button.

If you plan to install the Self-Hosted Integration Runtime on any machine other than the machine on which you’re working, you will select Option 2. For this exercise, I am installing it on the same virtual machine that I prepared for SQL Server.

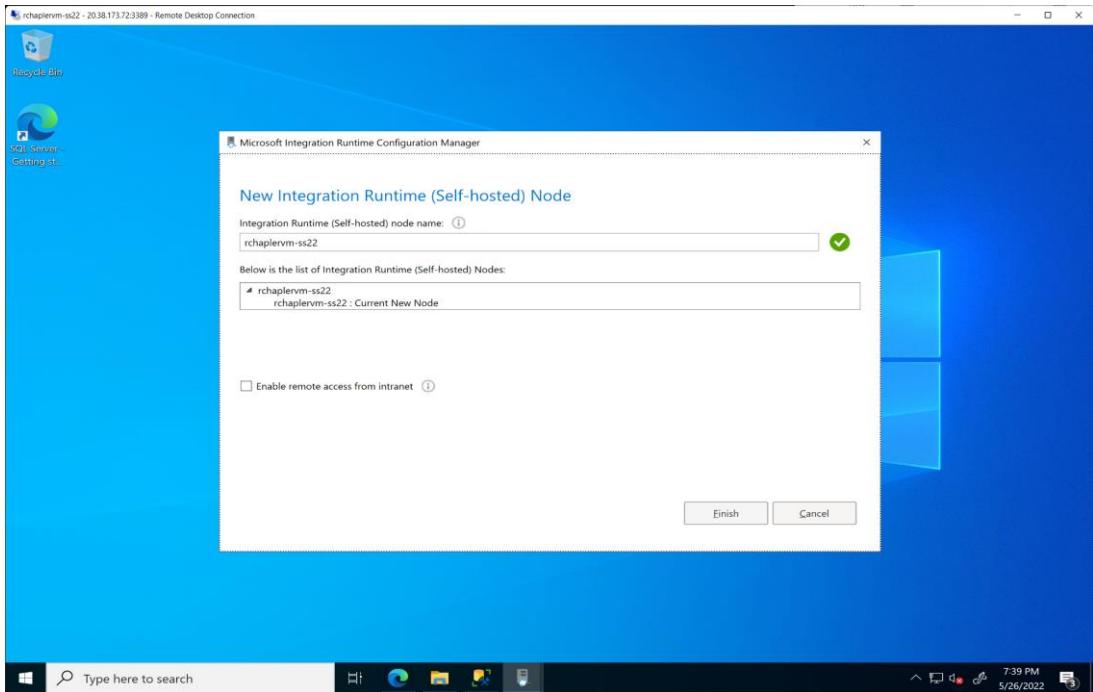
Click the “Download and install integration runtime” link and after download is complete, open the downloaded file on the Self-Hosted Integration Runtime machine.



Finish the generic installation process.



The “**Microsoft Integration Runtime Configuration Manager**” will open when installation is complete. Copy the **Key1** value from the Synapse pop-out {i.e., “**Step 2**”} and paste it into the “...valid Authentication Key” textbox. Click the **Register** button.



Review the final configuration values and then click the **Finish** button. After initialization is complete, you can **Close** the window.

Create Linked Services

Note: This exercise requires a User-Assigned Managed Identity, related Synapse Credentials, and a Self-Hosted Integration Runtime

Navigate to Synapse Studio, click the **Manage** navigation icon, select “**Linked services**” from the “**External connections**” grouping in the resulting navigation. Click the “**+ New**” button.

Search for and then select the resource for which you are creating a linked service; examples:

- Azure Data Explorer (Kusto)
- Azure Data Lake Storage Gen2
- Azure Key Vault
- Azure SQL Database
- SQL Server

Click the **Continue** button.

Complete the “**New linked service...**” pop-out, including:

Connect via...	Confirm default selection, “ AutoResolveIntegrationRuntime ”
-----------------------	---------------------------------------------------------------------

And including resource specific items...

Azure Data Explorer (Kusto)

Connect via...	Confirm default selection, “ AutoResolveIntegrationRuntime ”
-----------------------	---------------------------------------------------------------------

Authentication Method	Select “Managed Identity”
------------------------------	---------------------------

Selection Method	Select “Enter manually”
-------------------------	-------------------------

Endpoint	Enter the URL for your Data Explorer Pool {e.g., <a href="https://<UseCase>dep.<UseCase>s.kusto.azuresynapse.net">https://<UseCase>dep.<UseCase>s.kusto.azuresynapse.net }
-----------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Database	Select your Data Explorer Database
-----------------	------------------------------------

Azure Data Lake Storage Gen2

Connect via...	Confirm default selection, “ AutoResolveIntegrationRuntime ”
-----------------------	---------------------------------------------------------------------

Selection Method	Confirm default selection, “ From Azure subscription ”
-------------------------	---------------------------------------------------------------

Authentication Method	Confirm default selection, “ Account key ”
------------------------------	---------------------------------------------------

Storage Account Name	Select your Data Lake
-----------------------------	-----------------------

Azure Key Vault

Connect via...	Confirm default selection, “ AutoResolveIntegrationRuntime ”
Selection Method	Confirm default selection, “ From Azure subscription ”
Azure Key Vault Name	Select your Key Vault

Azure SQL Database

Connect via...	Confirm default selection, “ AutoResolveIntegrationRuntime ”
Selection Method	Confirm default selection, “ From Azure subscription ”
Server Name	Select your Azure SQL Server
Database Name	Select your Azure SQL Database
Authentication Type	Select “ System Assigned Managed Identity ” or “ User Assigned Managed Identity ” (depending on use case requirements)
Credentials	Select your Synapse Credentials

SQL Server

Connect via...	Select your Self-Hosted Integration Runtime
Server Name	LoREM
Database Name	Enter “AdventureWorks2019”
Authentication Type	Select “Windows authentication”
User Name	Enter your credentials
Password	

Click “**Test connection**” to confirm successful connection and then click the **Create and Publish (or Commit)** button.

Create Pools

Data Explorer Pool

Navigate to your Synapse Analytics workspace, click the **Open** link on the “**Open Synapse Studio**” rectangle and then click the **Manage** icon in the navigation pane.

Select “**Data Explorer pools...**” in the “**Analytics pools**” grouping of the resulting navigation and then click the “**+ New**” button.

On the resulting “**Create Data Explorer pool**” pop-out, **Basics** tab, including:

Workload	Select an appropriate option {e.g., “ Compute optimized ” as we will not have a large volume of data}
Size	Select an appropriate option {e.g., “ Extra Small ” as we will not demonstrate expensive computation}

Navigate to the “**Create Data Explorer pool**” pop-out, “**Additional settings**” tab, and then including:

Autoscale	Select an appropriate option {e.g., Disabled as we will not require Autoscale for this demonstration}
Number of Instances	Select an appropriate option (I usually choose the least expensive option for demonstrations)
Estimated Price	Confirm the estimated cost per hour that Azure generates based on your selected options

Click the “**Review + create**” button, review configuration, and then click the **Create** button.

Create Database

Navigate to your Synapse Analytics workspace, click the **Open** link on the “**Open Synapse Studio**” rectangle and then click the **Data** icon in the navigation, the + button in the header and then “**Data Explorer database...**” in the resulting drop-down menu.

On the resulting “**Create Data Explorer database...**” pop-out, including:

Pool Name	Select your Data Explorer Pool
Default Retention...	Confirm default, 365
Default Cache...	Confirm default, 31

Click the **Create** button.

Add Permissions

Navigate to your Data Explorer Pool and click **Permissions** in the “**Security + networking**” group in the navigation pane.

Click “+ Add” on the resulting page and select **AllDatabasesViewer** for read permissions or **AllDatabasesAdmin** for write, etc. permissions from the resulting drop-down menu.

Search for and select your Synapse managed instance.

Add Sample Data

Sample 1, Product Table

Navigate to your Synapse Analytics workspace, click the **Open** link on the “**Open Synapse Studio**” rectangle click the **Data** icon in the navigation, expand “**Data Explorer Databases...**”, and then select your Data Explorer Pool.

Right click on your Data Explorer Database {e.g., <UseCase>ded} and select “New KQL script” from the resulting drop-down menu.

Execute the following logic to create a new table named Product:

```
.create table Product ( ProductId:int, ProductNumber: string, Name:string, ListPrice: decimal,  
ModifiedDate:datetime )
```

Sample 2, StormEvents Data

Follow the instructions in the “Quickstart: Ingest sample data into Azure Data Explorer” article (<https://docs.microsoft.com/en-us/azure/data-explorer/ingest-sample-data>) to populate sample data that we can surface in Power Apps.

Familiarize yourself with the resulting data for use in later sections.

Dedicated SQL Pool

Navigate to Synapse.

Click the “+ New dedicated SQL pool” button.

Enter a meaningful name in “**Dedicated SQL pool name**” and choose a pricing tier.

Click the “**Review + create**” button, review configuration, and then click the **Create** button.

Serverless SQL Database

Navigate to your Synapse Analytics workspace and then click the **Open** link on the “**Open Synapse Studio**” rectangle.

Click the **Data** icon in the navigation pane.

Click the **Workspace** tab. Click the + button and “**SQL database**” in the resulting drop-down menu.

Complete the “**Create SQL database**” pop-out, including:

Select SQL Pool	Confirm default, Serverless
Type	

Click the **Create** button.

Apache Spark Pool

Navigate to your Synapse Analytics workspace and then click the **Open** link on the “**Open Synapse Studio**” rectangle.

Click the **Manage** icon and then “**Apache Spark pools**” in the navigation pane.

Click the “**New Apache Spark pool**” button and on the resulting pop-out, including:

Isolated Compute	Confirm default, Disabled
Node Size Family	Confirm default, “ Memory Optimized ”
Node Size	Select “Small (4 vCores / 32 GB)”
Autoscale	Select Disabled
Number of Nodes	Slide to lowest possible value (to minimize demonstration cost)
Estimated Price	Review final “Est. cost per hour” and view pricing details, as desired

Click the “**Review + create**” button, validate settings, and then click the **Create** button.

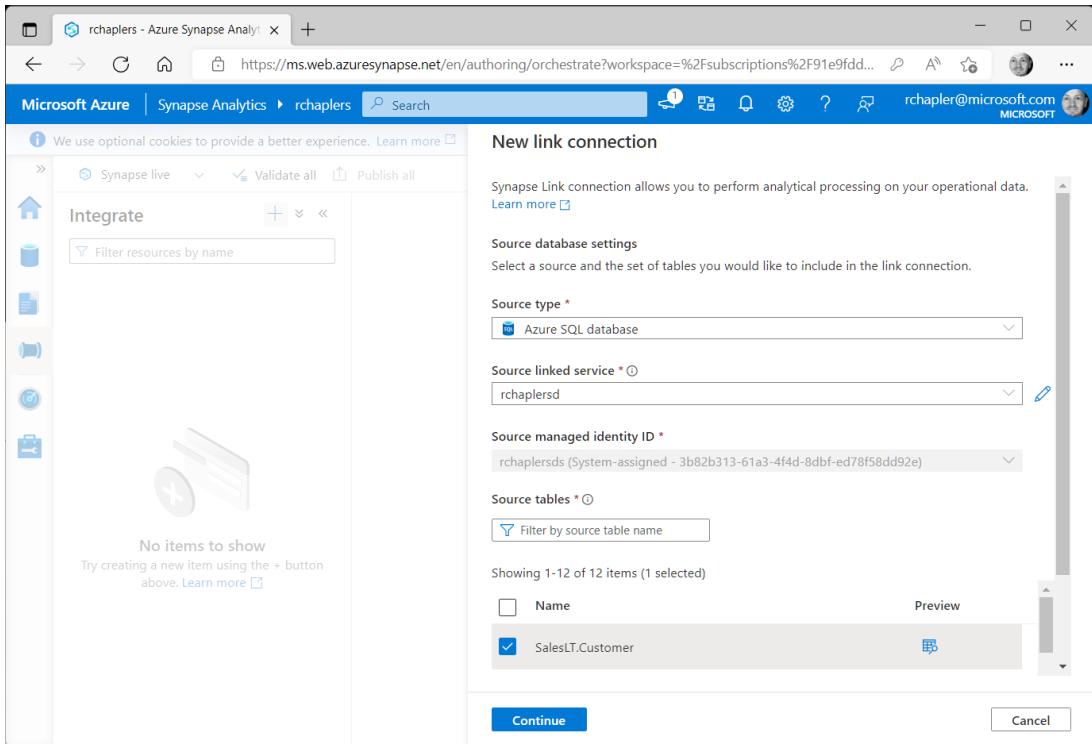
Create Link Connections

Note: These instructions require:

- an Azure SQL Server / Database with 1) “**Pricing Tier**” set to **Premium**, 2) **Identity** >> “**System-Assigned Managed Identity**” set to **ON**, and 3) access granted {i.e., USER created / **db_datareader** ROLE altered}
- an Azure Synapse Linked Service for the Azure SQL Database (using a system-assigned managed identity)
- an Azure Synapse, Dedicated SQL Pool

Source Database Settings

Navigate to Synapse Studio, click the **Integrate** navigation icon, click “+” and select “Link Connection...” from the resulting drop-down menu.



On the resulting “New link connection” > “Source database settings” pop-out, including:

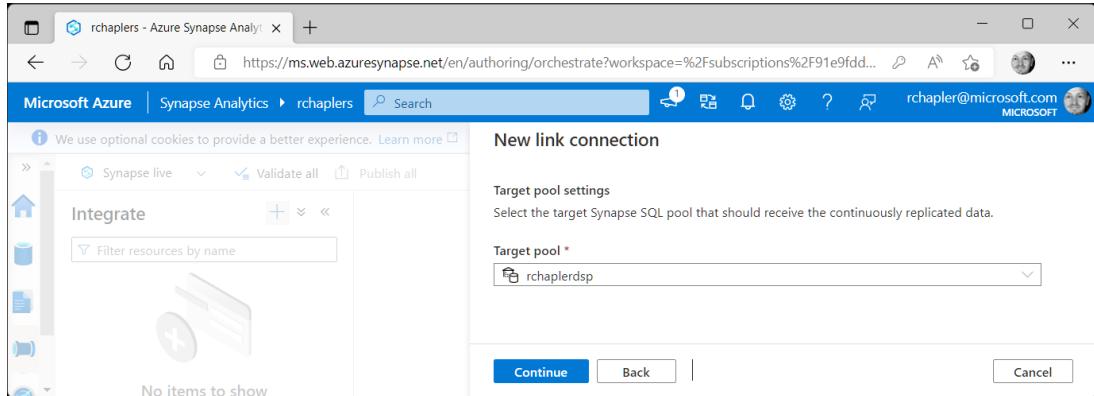
Source Type Select “**Azure SQL database**”

Source Linked Service Select the Linked Service for your Azure SQL Database

Source Tables Select tables for replication

Click the **Continue** button.

Target Pool Settings

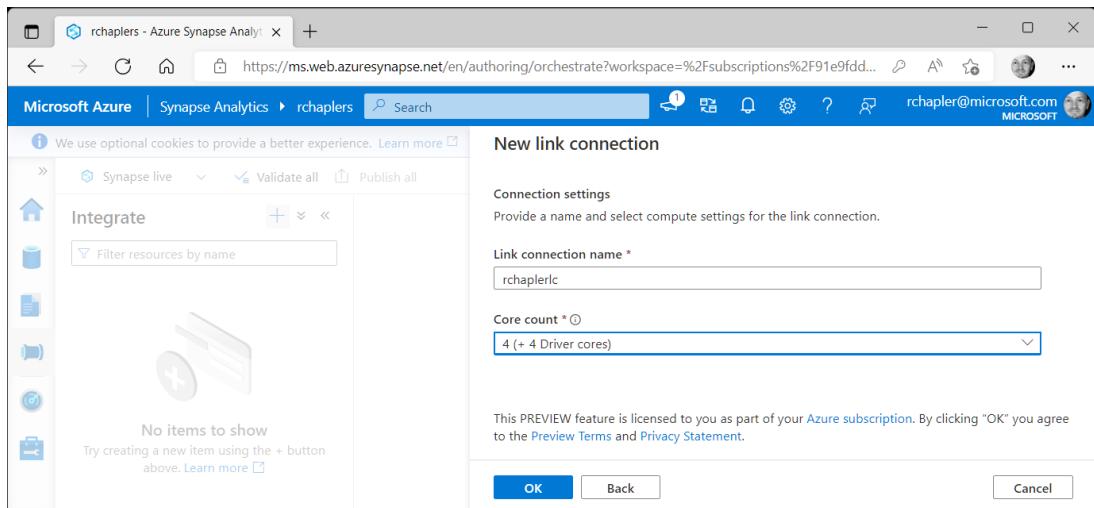


The screenshot shows the Microsoft Azure Synapse Analytics interface. On the left, there's a sidebar with icons for Home, Databases, Tables, Functions, and Pipelines. The main area is titled 'Integrate' with a sub-section 'Target pool settings'. A dropdown menu shows 'rchaplerdsp' as the selected target pool. At the bottom right of the main area are 'Continue', 'Back', and 'Cancel' buttons.

On the resulting “**New link connection**” > “**Target pool settings**” pop-out, including:

Target Pool Select your Dedicated SQL Pool

Click the **Continue** button.



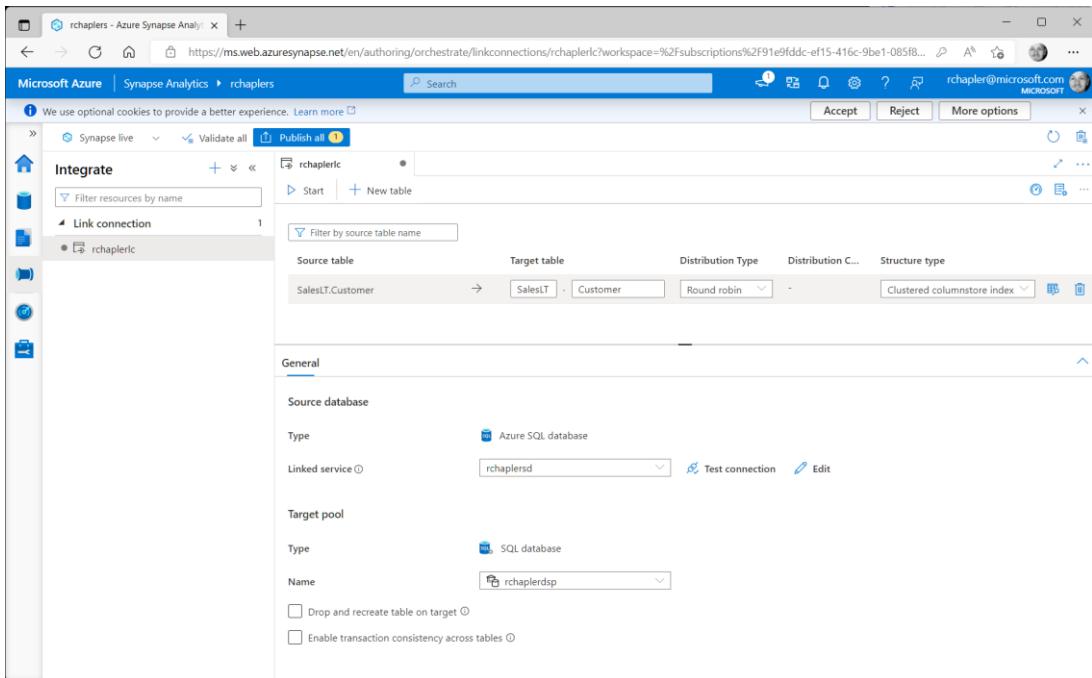
The screenshot shows the 'New link connection' dialog for 'Connection settings'. It includes fields for 'Link connection name' (set to 'rchaplerlc') and 'Core count' (set to '4 (+ 4 Driver cores)'). Below these fields, a note states: 'This PREVIEW feature is licensed to you as part of your Azure subscription. By clicking "OK" you agree to the [Preview Terms](#) and [Privacy Statement](#)'. At the bottom are 'OK', 'Back', and 'Cancel' buttons.

On the resulting “**New link connection**” > “**Connection settings**” pop-out, including:

Core Count Select an appropriate value (consider the minimum to reduce cost)

Click the **OK** button.

All settings are presented in what feels like a description for recurring data movement (albeit not a Pipeline or Data Flow).



Click “**Publish All**” and then click the **Start** button.

Add Access Policy to Key Vault

Navigate to Key Vault, click “**Access Configuration**” in the **Settings** group of the navigation pane and then click the “**Go to access policies**” button on the resulting page. Click “**+ Create**” link and on the resulting “**Create an access policy**” form, **Permissions** tab, including:

Secret Permissions Select **Get** and **List**

Click the **Next** button to navigate to the **Principal** tab, and including:

Select Principal Search for, and then **Select** the managed identity for your Synapse Workspace

Navigate to the “**Review + Create**” tab, review configuration, and then click the **Create** button.

User-Assigned Managed Identity

Complete the “**Create User Assigned Managed Identity**” form.

Click the “**Review + create**” button, validate settings, and then click the **Create** button.

Virtual Machines

Create using Azure Portal

Use Case: Customer XYZ shared the following requirements:

- Mimic hard-to-replicate environments {e.g., on-prem SQL Server 2008 R2...}
- Create and configure using Portal

Complete the “**Create virtual machine**” form, **Basics** tab, including:

Availability Options	Confirm default, “No infrastructure redundancy required”
Security Type	Confirm default, Standard
Image	Search for and select the “ SQL Server 2008 R2... ” image
Azure Spot Instance	Confirm default, unchecked
Size	Select the SKU appropriate to your use case requirement (I typically start with the cheapest option)

Further down on the “**Create virtual machine**” form, **Basics** tab, including:

Username	Self-explanatory
Password	
...Inbound Ports	Confirm defaults, “Allow Selected Ports” and “RDP (3389)”

Complete the “**Create virtual machine**” form, “**SQL Server settings**” tab, including:

SQL Connectivity	Select “Public (Internet)”
Port	Confirm default, 1433
SQL Authentication	Click to Enable and then confirm default (which will match your previously entered Administrator values)
Login Name	
Password	

Click the “**Review + create**” button, validate settings, and then click the **Create** button.

Configure SQL Server

Connect to the new virtual machine (using RDP) to complete image-specific configuration.

SQL Server Browser

Enable SQL Server Browser to allow connection to the SQL Server:

- Open **Services**, search for and double-click to open “**SQL Server Browser**”
- On the **General** tab, change “**Startup Type**” to **Automatic**, and then click the **Apply** button
- Click the **Start** button

Configure IE ESC

Disable Internet Explorer Enhanced Security Configuration to allow downloading to the demonstration VM:

- Open **Server Manager** and click the “**Configure IE ESC**” link in the “**Server Summary**” > “**Security Information**” interface grouping
 - Alternatively, navigate to “**Local Server**” and you will find “**IE Enhanced Security Configuration**” in the **Properties** grouping
- Complete the “**Internet Explorer Enhanced Security Configuration**” pop-up, click **Off** under Administrators and Users, and then click the **OK** button

Sample Database

Browse to <https://docs.microsoft.com/en-us/sql/samples/adventureworks-install-configure> and complete the following steps:

- Download the appropriate version of the AdventureWorks sample database {e.g., “**AdventureWorks2008R2-oltp.bak**”}
- Open **SQL Server Management Studio**, right-click on **Databases**, and click “**Restore Database**” in the resulting drop-down menu
- On the “**Restore Database**” pop-up, enter **AdventureWorks** in the “**To database**” text box, “**Destination for restore**” interface grouping
- On the “**Restore Database**” pop-up, click the “**From device**” radio button in the “**Source for restore**” interface grouping
- Click the **ellipses** button and in the resulting “**Specify Backup**” pop-up, click the **Add** button
- Browse to the downloaded BAK file; you may have to copy the BAK from the Downloads folder to another location (like c:\temp) to make it accessible
- Back on the “**Restore Database**” pop-up, check the box next to added item in “**Select the backup sets to restore**” and then click the **OK** button

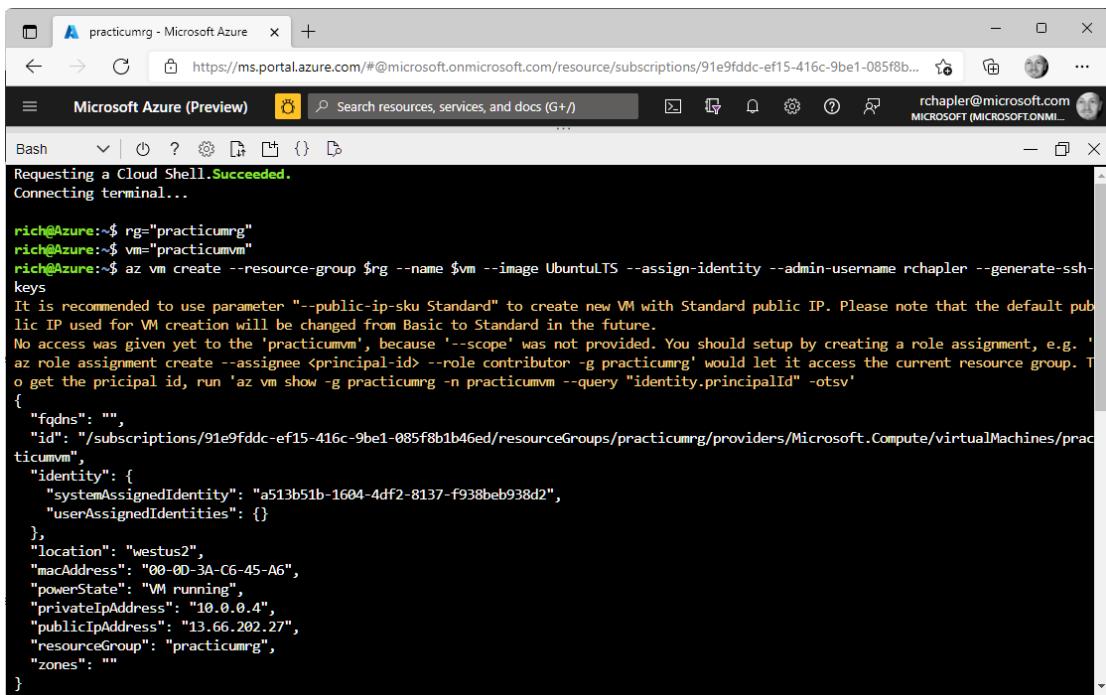
When you have successfully completed these steps, you will see a pop-up that says “**The restore of database ‘AdventureWorks’ completed successfully.**”

Create using Cloud Shell (Bash)

Use Case: Customer XYZ shared the following requirements:

- Mimic hard-to-replicate environments {e.g., on-prem Ubuntu (Linux)}
- Create and configure using Cloud Shell

Navigate to Cloud Shell and select **Bash** from the “Select Environment” drop-down menu.



The screenshot shows a Microsoft Azure Cloud Shell interface. The title bar says "practicumrg - Microsoft Azure". The main area is a terminal window titled "Bash". The terminal output shows the following sequence of commands and their results:

```
Requesting a Cloud Shell. Succeeded.
Connecting terminal...
rich@Azure:~$ rg="practicumrg"
rich@Azure:~$ vm="practicumvm"
rich@Azure:~$ az vm create --resource-group $rg --name $vm --image UbuntuLTS --assign-identity --admin-username rchapler --generate-ssh-keys
It is recommended to use parameter "--public-ip-sku Standard" to create new VM with Standard public IP. Please note that the default public IP used for VM creation will be changed from Basic to Standard in the future.
No access was given yet to the 'practicumvm', because '--scope' was not provided. You should setup by creating a role assignment, e.g. 'az role assignment create --assignee <principal-id> --role contributor -g practicumrg' would let it access the current resource group. To get the principal id, run 'az vm show -g practicumrg -n practicumvm --query "identity.principalId" -otsv'
{
  "fqdns": "",
  "id": "/subscriptions/91e9fddc-ef15-416c-9be1-085f8b1b46ed/resourceGroups/practicumrg/providers/Microsoft.Compute/virtualMachines/practicumvm",
  "identity": {
    "systemAssignedIdentity": "a513b51b-1604-4df2-8137-f938beb938d2",
    "userAssignedIdentities": {}
  },
  "location": "westus2",
  "macAddress": "00-0D-3A-C6-45-A6",
  "powerState": "VM running",
  "privateIpAddress": "10.0.0.4",
  "publicIpAddress": "13.66.202.27",
  "resourceGroup": "practicumrg",
  "zones": ""
}
```

Update and execute the following command to set variable **rg** {i.e., the name of your Resource Group}:

```
rg=<UseCase>rg"
```

Update and execute the following command to set variable **vm** {i.e., the desired name for the new virtual machine}:

```
vm=<UseCase>vm"
```

Execute the following command to create your virtual machine:

```
az vm create --resource-group $rg --name $vm --image UbuntuLTS --assign-identity --admin-username rchapler --generate-ssh-keys
```

Successful execution will create the following resources:

- Virtual Machine
- Disk

- Network Security Group
- Public IP Address
- Network Interface
- Virtual Network

Update and execute the following command to connect to the server using SSH:

```
ssh -i ~/.ssh/id_rsa rchapler@[public ip address]
```

```
rich@Azure:~$ ssh -i ~/.ssh/id_rsa rchapler@13.66.202.27
The authenticity of host '13.66.202.27 (13.66.202.27)' can't be established.
ECDSA key fingerprint is SHA256:2psYqDje2y74/P36tyrkIBHa+HgHSdWgb9+UuA2gHs.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '13.66.202.27' (EDSA) to the list of known hosts.
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-1063-azure x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Fri Nov 19 15:18:45 UTC 2021

System load: 0.15      Processes:          112
Usage of /:   4.6% of 28.90GB  Users logged in:  0
Memory usage: 5%           IP address for eth0: 10.0.0.4
Swap usage:  0%

0 updates can be applied immediately.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

rchapler@practicumvm:~$
```

You will know you are successful if your prompt has changed to your equivalent of `rchapler@<UseCase>vm.`

Acquisition

In this section, we will discuss various methods of acquiring source data {e.g., migration, direct connection, incremental load, etc.}

Migration

Follow these instructions to migrate data from an on-prem instance of SQL Server to Azure SQL.

Use Case

Example notes:

- “We want to migrate some of many tables (schema and data) from an on-prem, SQL Server 2008 R2 database to Azure”
- “Once migrated, data will live only in Azure... the on-prem source will be deprecated”

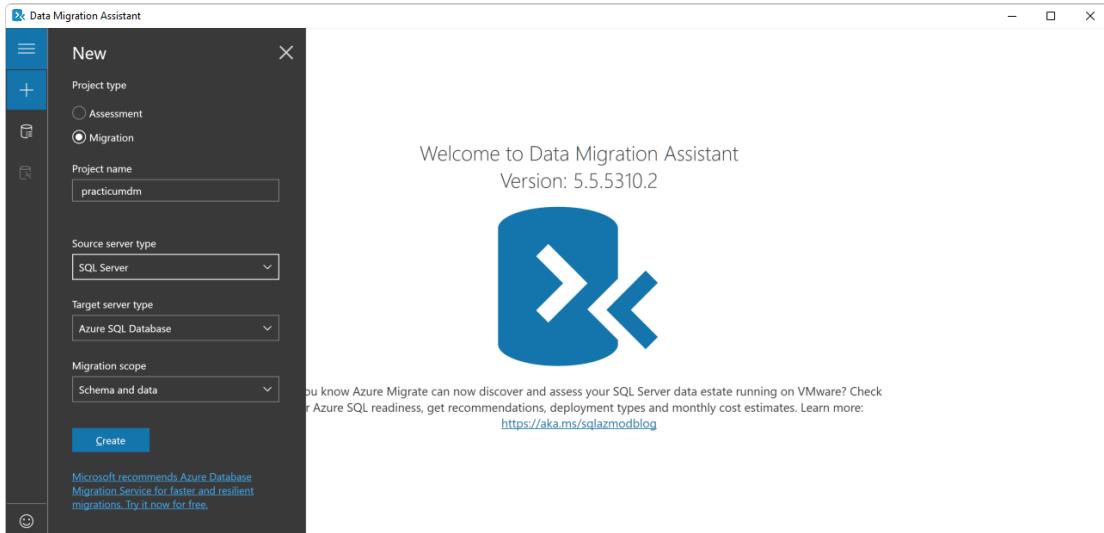
Prerequisites

This solution requires the following resources:

- Data Migration Assistant
- SQL
- Virtual Machine (with “SQL Server 2008 R2...” image and **AdventureWorks** sample database)

Create Migration Project

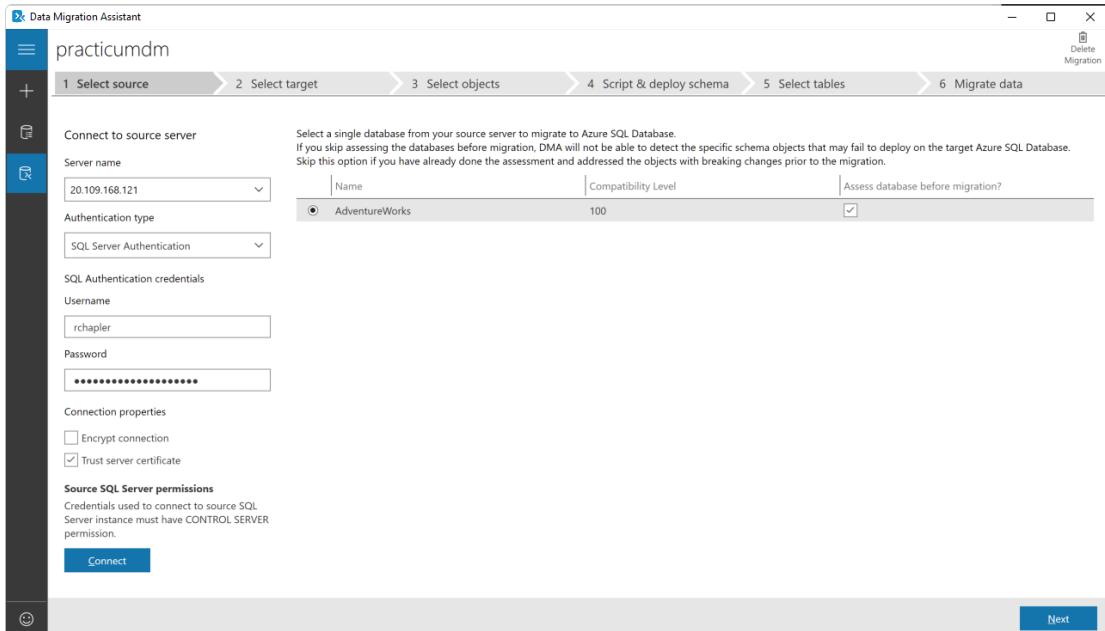
Open **Data Migration Assistant** and click + to create a new project



Complete the “**Create Data Share**” pop-out, including:

Project Type	Select the Migration radio button
Source Server Type	Select “SQL Server”
Target Server Type	Select “Azure SQL Database”
Migration Scope	Select “Schema and data”

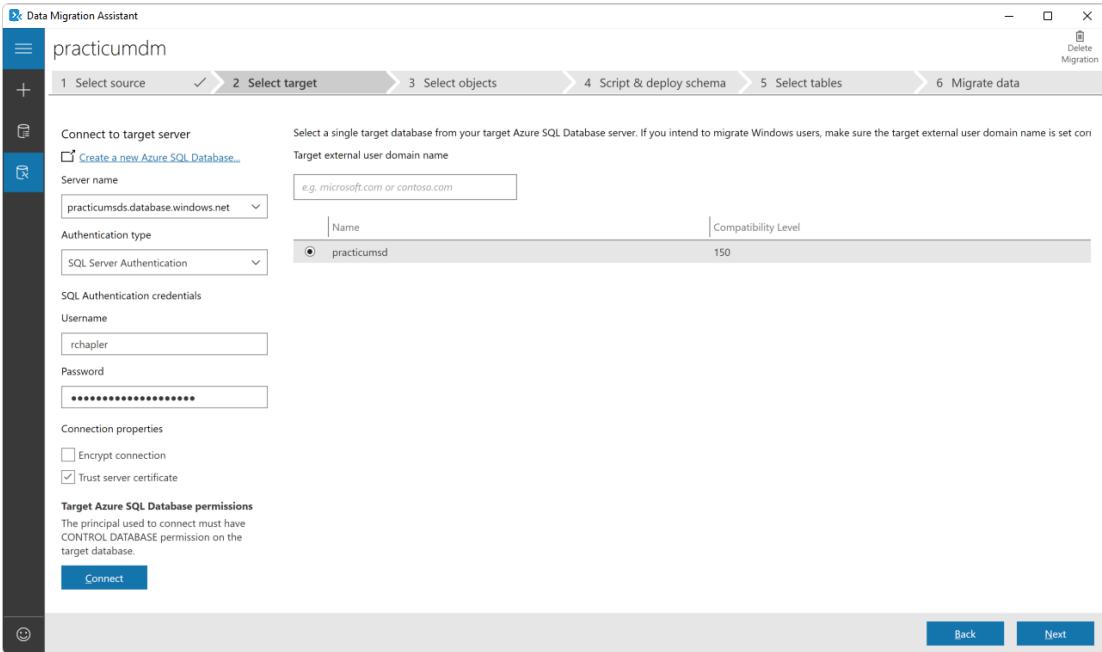
Click the **Create** button.



Complete the “**1. Select source**” form, including:

Server Name	Enter the “ Public IP address ” for your SQL Server 2008 virtual machine
Authentication Type	Select “ Windows Authentication ” and enter credentials
Username	
Password	
Encrypt Connection	Unchecked
Trust Server Certificate	Checked

Click the **Connect** button, confirm selection of **AdventureWorks** and then click the **Next** button.



Complete the “**2. Select target**” form, including:

Authentication Type Select “**SQL Server Authentication**” and enter credentials

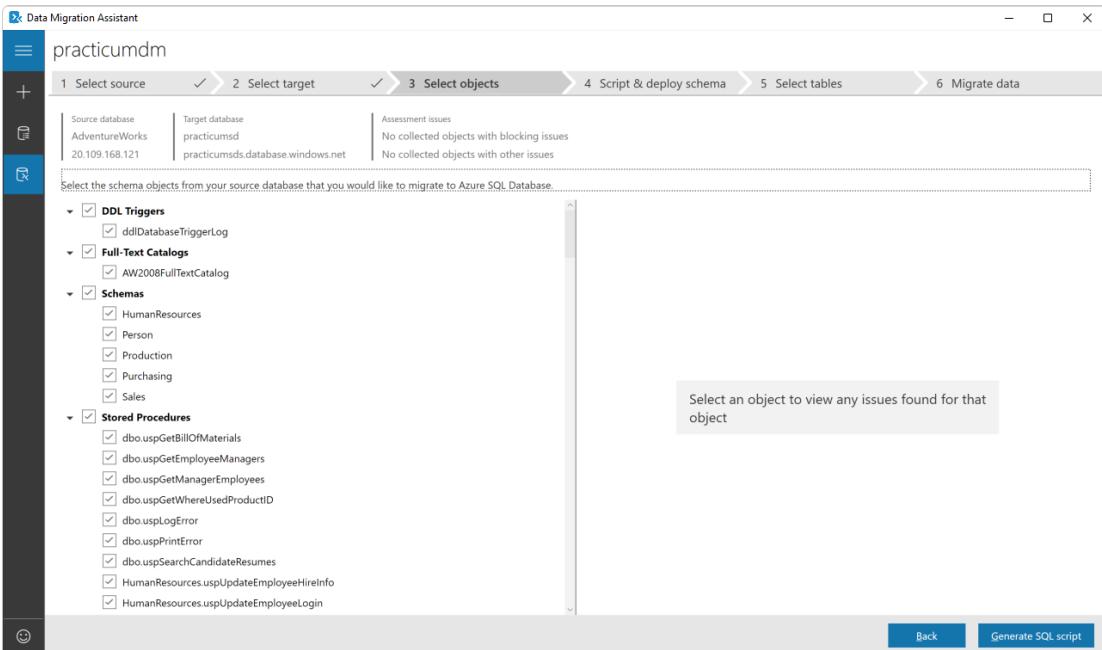
Username

Password

Encrypt Connection Unchecked

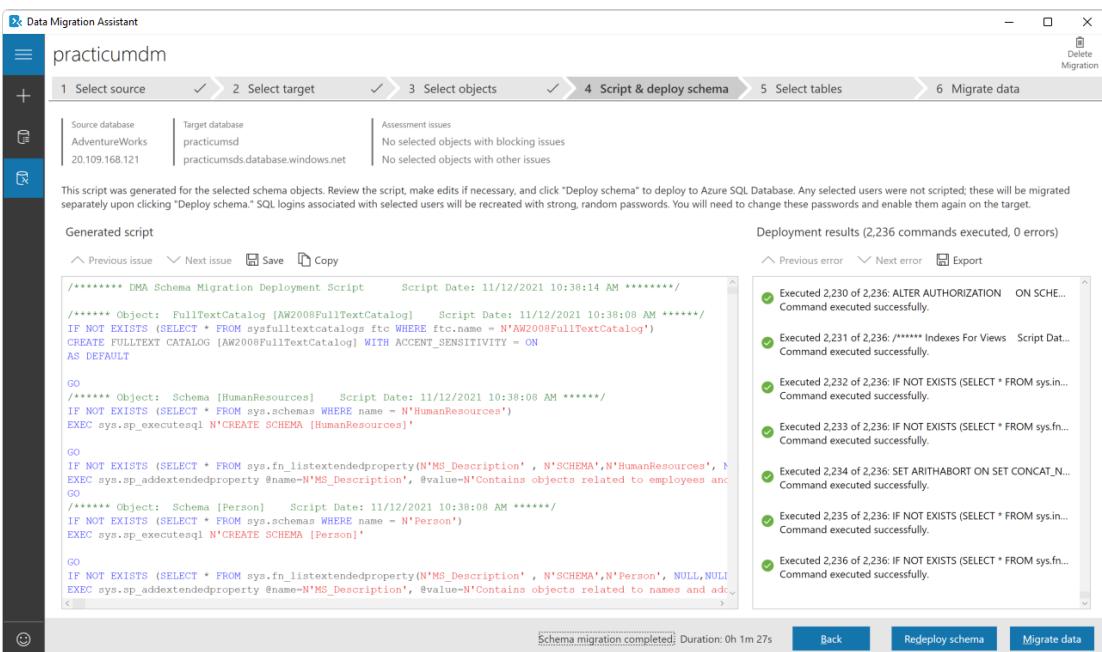
Trust Server Certificate Checked

Click the **Connect** button, confirm selection of your SQL database, and then click the **Next** button.



Complete the “**3. Select objects**” form, review schema objects, taking special note of any flagged items.

Click the “**Generate SQL script**” button in the bottom-right when you are ready to continue.



Complete the “**4. Script & deploy schema**” form, review the generated script.

Click the “Deploy schema” button when you are ready to execute the script and create schema on the target database.

Click the “Migrate data” button when you are ready to continue.

Select the tables containing data you would like to migrate to Azure SQL Database. Microsoft strongly recommends that you temporarily change your Azure SQL Database to performance level P15 during the migration process for the optimal migration experience.
[Learn more about performance tiers](#)

Table name	Row count	Ready to move
[HumanResources].[Department]	16	OK
[HumanResources].[Employee]	290	OK
[HumanResources].[EmployeeDepartmentHistory]	296	OK
[HumanResources].[EmployeePayHistory]	316	OK
[HumanResources].[JobCandidate]	13	OK
[HumanResources].[Shift]	3	OK
[Person].[Address]	19,614	OK
[Person].[AddressType]	6	OK
[Person].[BusinessEntity]	20,777	OK
[Person].[BusinessEntityAddress]	19,614	OK

Back Start data migration

Complete the “5. Select tables” form, review the list of tables and exclude items as appropriate.

Click the “Start data migration” button when you are ready to continue.

65 Server objects 0 In-progress 65 Successful 0 Warnings 0 Failed

Source database: AdventureWorks
Target database: practicumdm
20.109.168.121
practicumsds.database.windows.net

Status	Table name	Migration details
Success	[Person].[Address]	Migration successful. Duration: 0 hrs 0 mins 2 secs
Success	[Person].[AddressType]	Migration successful. Duration: 0 hrs 0 mins 1 secs
Success	[Person].[BusinessEntity]	Migration successful. Duration: 0 hrs 0 mins 1 secs
Success	[Person].[BusinessEntityAddress]	Migration successful. Duration: 0 hrs 0 mins 1 secs
Success	[Person].[BusinessEntityContact]	Migration successful. Duration: 0 hrs 0 mins 1 secs
Success	[Person].[ContactType]	Migration successful. Duration: 0 hrs 0 mins 0 secs

Allow time for the migration to successfully complete (as above).

No-Load

User connects to data in place, with no ETL process

External Data

Follow these instructions to explore methods of using data from data products external to Synapse.

Use Case

Example notes:

- “We want to use data via direct connection to the source”
- “We want to minimize investment in recurring data movement wherever possible”

Prerequisites

This solution requires the following resources:

- Data Lake (with sample data)
- Synapse (with Serverless SQL Database and Apache Spark Pool)

Alternate Methods

Navigate to your Synapse Analytics workspace, click the **Open** link on the “**Open Synapse Studio**” rectangle and then click the **Data** icon in the navigation pane.

The screenshot shows the Microsoft Azure Synapse Analytics workspace interface. The left sidebar has a 'Data' section with 'Workspace' and 'Linked' tabs. The 'Linked' tab is selected, showing a list of storage accounts under 'practicumsw'. One account, 'practicumdlc', is expanded, revealing its primary container 'practicumdlc (Primary)'. The main content area displays a table of files from this container:

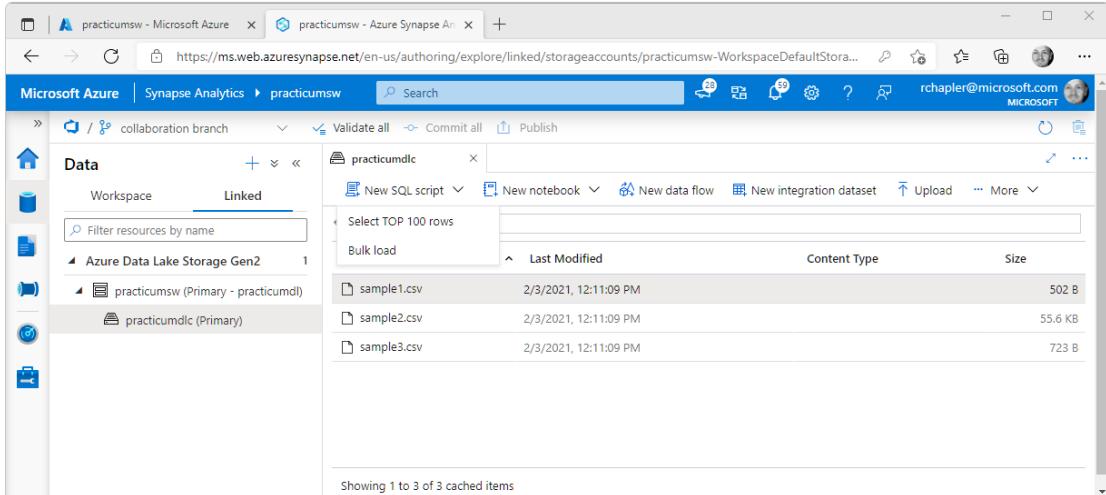
Name	Last Modified	Content Type	Size
sample1.csv	2/3/2021, 12:11:09 PM		502 B
sample2.csv	2/3/2021, 12:11:09 PM		55.6 KB
sample3.csv	2/3/2021, 12:11:09 PM		723 B

At the bottom of the table, it says 'Showing 1 to 3 of 3 cached items'.

Click the **Linked** tab and Expand navigation pane.

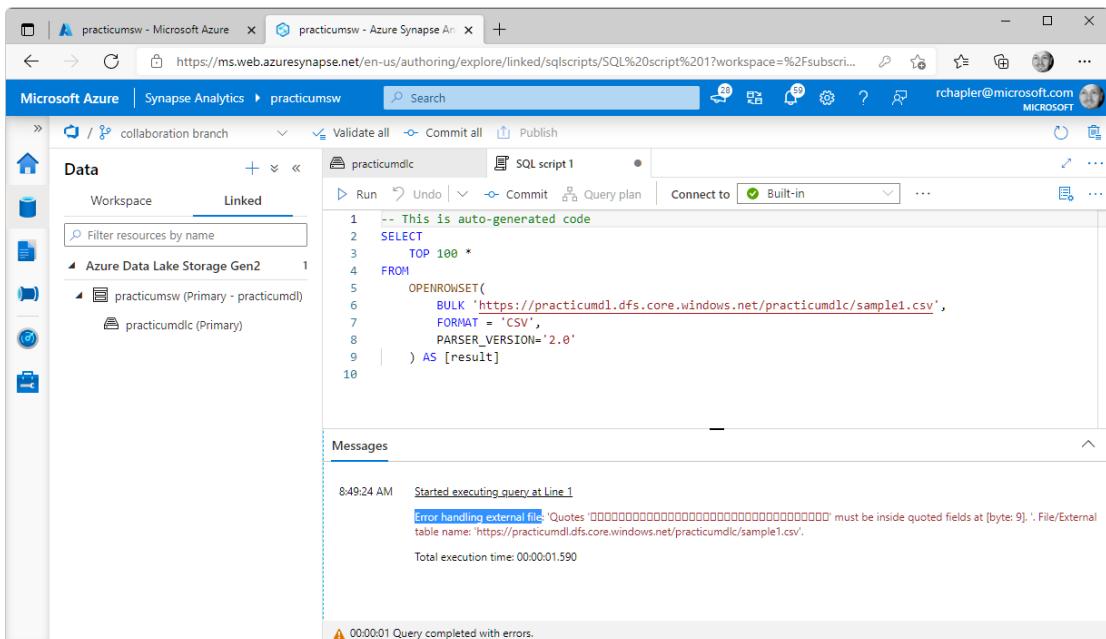
You will see that the Azure Data Lake Storage Gen2 specified during Synapse instantiation {i.e., <UseCase>dlc} surfaces with no additional configuration.

Method 1: SQL Script (OPENROWSET)



The screenshot shows the Microsoft Azure Synapse Analytics Data blade. On the left, there's a navigation pane with 'Data' selected, showing 'Workspace' and 'Linked'. Under 'Linked', it lists 'Azure Data Lake Storage Gen2' and 'practicumsw (Primary - practicumd1)'. Inside 'practicumsw', there's a 'practicumdlc (Primary)' folder. On the right, a table displays three CSV files: 'sample1.csv', 'sample2.csv', and 'sample3.csv', all modified on 2/3/2021 at 12:11:09 PM. The table has columns for 'Content Type' and 'Size'.

Click the “New SQL script” button and “SELECT TOP 100 rows” from the resulting drop-down menu.



The screenshot shows the Microsoft Azure Synapse Analytics SQL script editor. The top bar shows 'practicumsw - Microsoft Azure' and 'practicumsw - Azure Synapse Analytics'. The main area is titled 'practicumdlc' and contains a tab for 'SQL script 1'. The code pane contains the following SQL:

```
1 -- This is auto-generated code
2 SELECT
3     TOP 100 *
4 FROM
5     OPENROWSET(
6         BULK 'https://practicumd1.dfs.core.windows.net/practicumdlc/sample1.csv',
7         FORMAT = 'CSV',
8         PARSE_VERSION=2.0
9     ) AS [result]
10
```

The 'Messages' pane at the bottom shows the execution log:

- 8:49:24 AM Started executing query at Line 1
- Error handling external file: Quotes ' must be inside quoted fields at [byte: 9]'. File/External table name: 'https://practicumd1.dfs.core.windows.net/practicumdlc/sample1.csv'.
- Total execution time: 0:00:01.590

A warning message at the bottom says: 0:00:01 Query completed with errors.

The resulting auto-generated code uses OPENROWSET(...) and a built-in, serverless SQL Pool to pull data directly from the CSV file in the ADLS container.

When we run the code (assuming you are using the same sample files, of course), we get an error about quote-handling.

Given the formatting of the sample file, we cannot run this without changes to the code.

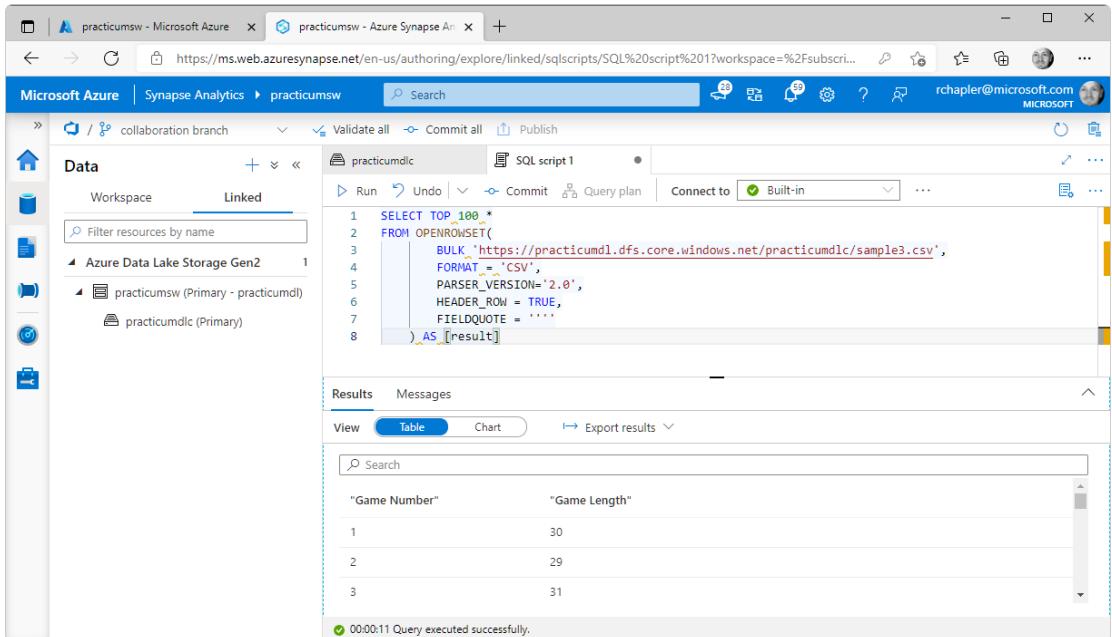
Update the code to:

```

SELECT TOP 100 *
FROM OPENROWSET(
    BULK 'https://<UseCase>d1.dfs.core.windows.net/<UseCase>dlc/sample3.csv',
    FORMAT = 'CSV',
    PARSE_VERSION='2.0',
    HEADER_ROW = TRUE,
    FIELDQUOTE = ''
) AS [result]

```

... and then, re-Run...



The screenshot shows the Microsoft Azure portal interface for a Synapse Analytics workspace named "practicumsw". The left sidebar displays "Data" resources, including "Workspace" and "Linked". A linked storage account "practicumsw (Primary - practicumdl)" is selected. In the center, a query editor window titled "practicumdlc" contains an SQL script:

```

1 SELECT TOP 100 *
2 FROM OPENROWSET(
3     BULK 'https://practicumdl.dfs.core.windows.net/practicumdlc/sample3.csv',
4     FORMAT = 'CSV',
5     PARSE_VERSION='2.0',
6     HEADER_ROW = TRUE,
7     FIELDQUOTE = ''
8 ) AS [result]

```

The "Results" tab is active, showing the output of the query:

"Game Number"	"Game Length"
1	30
2	29
3	31

A status message at the bottom indicates: "00:00:11 Query executed successfully."

Method 2: External Table (Serverless SQL Database)

Click the **Workspace** tab.

External Data Source

Expand navigation to and then right-click on “**External data sources**.”

The screenshot shows the Microsoft Azure Synapse Analytics Data workspace. In the left sidebar, under 'Data', the 'Workspace' tab is selected. Under 'External resources', 'External data sources' is expanded, showing 'New SQL script' and 'New external data source' as options in a dropdown menu. Below the menu, a message says 'Select an item' and 'Use the resource explorer to select or create a new item'.

Click “**New SQL script**”, and then “**New external data source**” in the drop-down menu.

The screenshot shows the Microsoft Azure Synapse Analytics Data workspace. In the left sidebar, under 'Data', the 'Workspace' tab is selected. Under 'External resources', 'External data sources' is selected. In the main area, a SQL script window is open with the following code:

```
CREATE EXTERNAL DATA SOURCE [ExternalDataSource] WITH
(
    LOCATION = 'https://practicumdl.blob.core.windows.net/practicumdlc'
)
```

The 'Results' tab shows the execution log:

```
9:29:07 AM Started executing query at Line 1
Total execution time: 00:00:03.449
00:00:03 Query executed successfully.
```

Replace <STORAGEACCOUNT> and <CONTAINER> in the auto-generated code and then **Run**.

External File Format

Expand navigation to and then right-click on “**External file formats**.”

The screenshot shows the Microsoft Azure portal interface for a Synapse Analytics workspace named 'practicumsw'. The left sidebar has 'Data' selected under 'Workspace'. In the main area, 'External file formats' is highlighted in the 'External resources' section. A tooltip 'Select an item' is visible over a large icon of two cylinders.

Click “New SQL script”, and then “New external file format” in the drop-down menu.

The screenshot shows the Microsoft Azure portal interface for a Synapse Analytics workspace named 'practicumsw'. The left sidebar has 'Data' selected under 'Workspace'. The main area displays a SQL script editor with the following code:

```
CREATE EXTERNAL FILE FORMAT [ExternalFileFormat] WITH
2 (
3     FORMAT_TYPE = DELIMITEDTEXT, FORMAT_OPTIONS ( FIELD_TERMINATOR = ',', FIRST_ROW = 2 )
4 )
```

The 'Results' tab shows the execution log:

```
10:24:53 AM Started executing query at Line 1
Total execution time: 00:00:02.705
```

At the bottom, a message indicates the query was executed successfully.

Append FORMAT_OPTIONS arguments FIELD_TERMINATOR and FIRST_ROW to the default FORMAT_TYPE argument and then Run.

External Table

Expand navigation to and then right-click on “External tables.”

The screenshot shows the Microsoft Azure Synapse Analytics Data Explorer interface. On the left, there's a navigation sidebar with icons for Home, Databases, External resources, Views, Schemas, and Security. The main area is titled 'Data' and has tabs for 'Workspace' and 'Linked'. Under 'External resources', there are sections for 'External data sources' and 'External file formats'. A context menu is open over the 'External tables' section, with 'New SQL script' and 'New external table' being the top items. Below the menu, there's a placeholder message 'Select an item' and a note 'Use the resource explorer to select or create a new item'.

Click “New SQL script”, and then “New external table” in the drop-down menu.

The screenshot shows the Microsoft Azure Synapse Analytics Data Explorer interface with a SQL script editor. The left sidebar shows the same navigation as before. The main area has a 'SQL script 1' tab. The code pane contains the following SQL script:

```
1 CREATE EXTERNAL TABLE [dbo].[ExternalTable]
2 (
3     [GameNumber] INT, [GameLength] INT
4 )
5 WITH
6 (
7     LOCATION = '/sample3.csv',
8     DATA_SOURCE = [ExternalDataSource],
9     FILE_FORMAT = [ExternalFileFormat]
10 )
11
```

The 'Results' tab at the bottom shows the execution log:

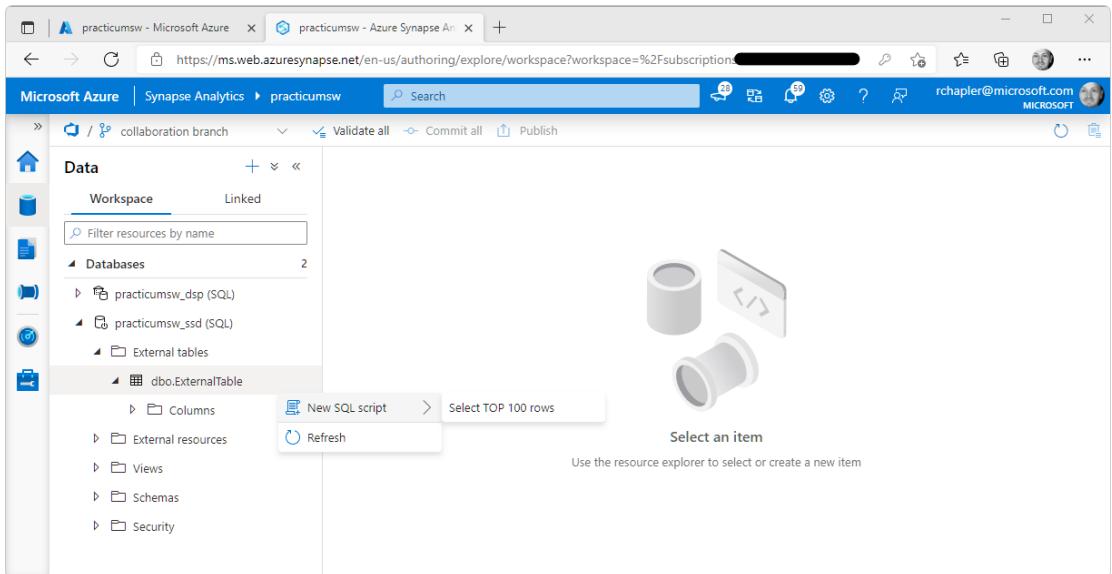
```
10:29:02 AM Started executing query at Line 1
Total execution time: 00:00:00.515
00:00:00 Query executed successfully.
```

Update the following items in the default code and then Run.

- **Schema Definition** ... replace `[id]` INT with columns matching the external data source
- **LOCATION** ... replace `'/folder/file'` with values matching those in your container
- **DATA_SOURCE** ... replace `[DataSource1]` with the name used in [Create External Data Source](#)
- **FILE_FORMAT** ... replace `[FileFormat1]` with the name used in [Create External File Format](#)

Confirm Success

Right-click on “External tables” and click Refresh.



Right-click on “**dbo.ExternalTable**”, click on “**New SQL script**”, and finally click on “**Select TOP 100 rows**” in the drop-down menu.

```
1 SELECT TOP (100) [GameNumber]
2 ,[GameLength]
3 | FROM [dbo].[ExternalTable]
```

GameNumber	GameLength
1	30
2	29
3	31

00:00:01 Query executed successfully.

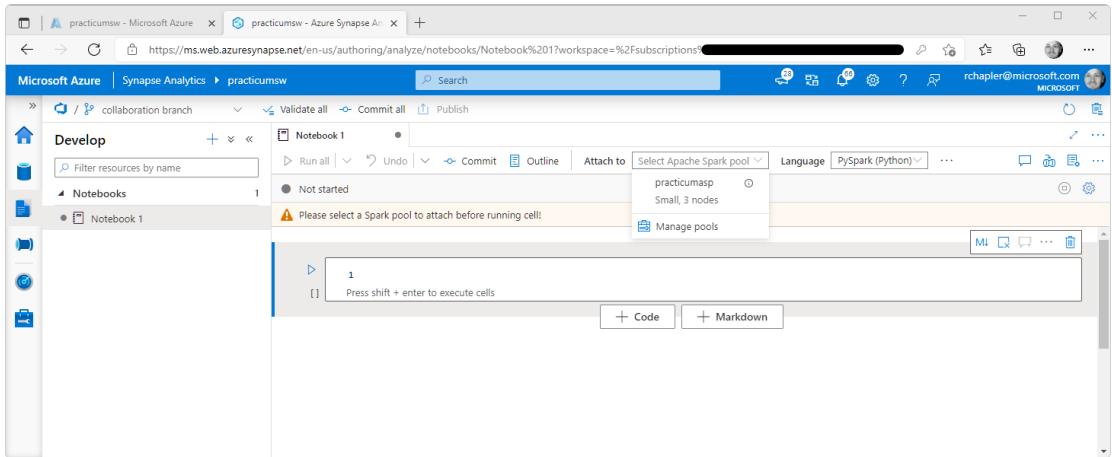
Click **Run**.

Method 3: Spark Notebook (Python)

Click the **Develop** tab.

Click the + icon just above and to the right of the “**Filter resources by name**” input.

Select **Notebook** from the resulting drop-down menu.



Select your Apache Spark Pool from the “**Attach to**” drop-down menu.

Paste the following code to Cell 1 and then click “**Run Cell**”:

```
%pyspark
theData = spark.read.load('abfss://<UseCase>d1c@<UseCase>d1.dfs.core.windows.net/sample1.csv',
format='csv', header=True)
theData.show(10)

print('Converted to Pandas...')
print(theData.toPandas())
```

practicumsw - Microsoft Azure practicumsw - Azure Synapse Analytics

https://ms.web.azuresynapse.net/en-us/authoring/analyze/notebooks/Notebook%201?workspace=%2Fsubscriptions%2F... rchaper@microsoft.com MICROSOFT

Microsoft Azure | Synapse Analytics | practicumsw | Search

collaboration branch | Validate all | Commit all | Publish

Develop | Notebook 1 | Ready

Notebooks | Notebook 1

```
1 %pyspark
2 theData = spark.read.load('abfss://practicumdlc@practicumdl.dfs.core.windows.net/sample1.csv', format='csv', header=True)
3 theData.show(10)
4
5 print('Converted to Pandas...')
6 print(theData.toPandas())
7
```

[1] ✓ 2 min 29 sec - Apache Spark session started in 2 min 6 sec 348 ms. Command executed in 23 sec 297 ms by rchaper on 1:21:45 PM, 9/23/21

> Job execution Succeeded | Spark 2 executors 8 cores | View in monitoring | Open Spark UI

...
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|Month| "Average"|"2005"|"2006"|"2007"|"2008"|"2009"|"2010"|"2011"|"2012"|"2013"|"2014"|"2015"|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
May	0.1	0	0	1	1	0	0	0	2	0	0	0
Jun	0.5	2	1	1	0	0	1	1	2	2	0	1
Jul	0.7	5	1	1	2	0	1	3	0	2	2	1
Aug	2.3	6	3	2	4	4	4	7	8	2	2	3
Sep	3.5	6	4	7	4	2	8	5	2	5	2	5
Oct	2.0	8	0	1	3	2	5	1	5	2	3	0
Nov	0.5	3	0	0	1	1	0	1	0	1	0	1
Dec	0.0	1	0	1	0	0	0	0	0	0	0	1
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+

Converted to Pandas...

	Month	"Average"	"2005"	"2006"	"2007"	"2008"	"2009"	"2010"	"2011"	"2012"	"2013"	"2014"	"2015"
0	May	0.1	0	0	...	2	0	0	0	0	0	0	0
1	Jun	0.5	2	1	...	2	2	0	1				
2	Jul	0.7	5	1	...	0	2	2	1				
3	Aug	2.3	6	3	...	8	2	2	3				
4	Sep	3.5	6	4	...	2	5	2	5				
5	Oct	2.0	8	0	...	5	2	3	0				
6	Nov	0.5	3	0	...	0	1	0	0				
7	Dec	0.0	1	0	...	0	0	0	0				

[8 rows x 13 columns]

+ Code | + Markdown

Mount Data Lake

Follow these instructions to establish connection to data in Azure Data Lake Storage.

Prerequisites

This solution requires the following resources:

- Data Lake (with “Storage Blob Data Reader” permissions set for your Application Registration)
- Databricks (with cluster, notebook, and secret scope)

Prepare Logic

Navigate to Databricks and the **<UseCase>dbn** notebook.

Add a new cell and paste the following code:

```
configs = {"fs.azure.account.auth.type": "OAuth",
           "fs.azure.account.oauth.provider.type":
           "org.apache.hadoop.fs.azurebfs.oauth2.ClientCredsTokenProvider",
           "fs.azure.account.oauth2.client.id": dbutils.secrets.get(scope="<UseCase>dbss", key="<UseCase>ar-clientid"),
           "fs.azure.account.oauth2.client.secret": dbutils.secrets.get(scope="<UseCase>dbss", key="<UseCase>ar-clientsecret"),
           "fs.azure.account.oauth2.client.endpoint": "https://login.microsoftonline.com/" +
           dbutils.secrets.get(scope="<UseCase>dbss", key="<UseCase>ar-tenantid") + "/oauth2/token"}

adlsAccount = "<UseCase>d1"
adlsContainer = "<UseCase>dlc"
adlsFolder = ""
mountPoint = "/mnt/<UseCase>

if not any(mount.mountPoint == mountPoint for mount in dbutils.fs.mounts()):
    dbutils.fs.mount(source = "abfss://" + adlsContainer + "@" + adlsAccount + ".dfs.core.windows.net/" +
                     adlsFolder, mount_point = mountPoint, extra_configs = configs )
```

(Partial) logic explanation:

- **<UseCase>dbss** ... refers to the Secret Scope
- **myClientId** ... refers to the Key Vault secret containing the “Application (client) ID”
- **myClientSecret** ... refers to the Key Vault secret containing the “Client Secret”
- **myTenantId** ... refers to the Key Vault secret containing the “Directory (tenant) ID”
- **<UseCase>dl** ... refers to your Data Lake
- **<UseCase>dlc** ... refers to the Data Lake Container
- **adlsFolder** ... placeholder / syntax for inclusion of a folder (null because it is not applicable in this instance)

Run “**Cmd 1.**”

The screenshot shows a Microsoft Azure Databricks notebook titled "practicumdbn". The notebook interface includes a sidebar with various icons for file operations, a toolbar at the top, and a main workspace where code is run. In the workspace, a cell titled "Cmd 1" contains the following Python code:

```
1 configs = {"fs.azure.account.auth.type": "OAuth",
2             "fs.azure.account.oauth.provider.type": "org.apache.hadoop.fs.azurebfs.oauth2.ClientCredsTokenProvider",
3             "fs.azure.account.oauth2.client.id": dbutils.secrets.get(scope="practicumdbss", key="practicumar-clientid"),
4             "fs.azure.account.oauth2.client.secret": dbutils.secrets.get(scope="practicumdbss", key="practicumar-clientsecret"),
5             "fs.azure.account.oauth2.client.endpoint": "https://login.microsoftonline.com/" +
6                 dbutils.secrets.get(scope="practicumdbss", key="practicumar-tenantid") + "/oauth2/token"}
7
8 adlsAccount = "practicumdl"
9 adlsContainer = "practicumdlc"
10 adlsFolder = ""
11 mountPoint = "/mnt/practicum"
12
13 if not any(mount.mountPoint == mountPoint for mount in dbutils.fs.mounts()):
14     dbutils.fs.mount( source = "abfss://" + adlsContainer + "@" + adlsAccount + ".dfs.core.windows.net/" + adlsFolder,
mount_point = mountPoint, extra_configs = configs )
```

Below the code cell, there is a message indicating the command took 26.98 seconds to run. A note at the bottom of the workspace says "Shift+Enter to run".

Quick Idea...

Consider organizing mount locations by associating them with a Databricks database entity; code example:

```
%sql
CREATE DATABASE myDatabase LOCATION "/mnt/<UseCase>/myDatabase"
```

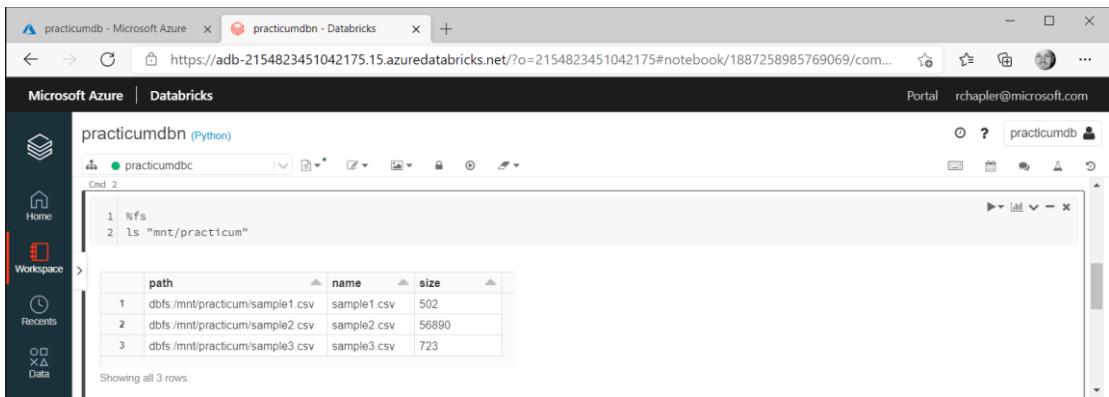
Confirm Success

Proof 1 ... Sample Files at Mount Point

Note: Running this logic requires "Storage Blob Data Reader" permissions on your data lake.

Add a cell below “**Cmd 1.**” In cell “**Cmd 2**”, paste and run the following code:

```
%fs
ls "mnt/<UseCase>"
```



```
%fs
ls "mnt/practicum"
```

	path	name	size
1	dbfs:/mnt/practicum/sample1.csv	sample1.csv	502
2	dbfs:/mnt/practicum/sample2.csv	sample2.csv	56890
3	dbfs:/mnt/practicum/sample3.csv	sample3.csv	723

You should see your sample files at the mount point.

Proof 2 ... Schema and Data

In your Databricks notebook, add a new cell, then paste and run the following code:

```
df = spark.read.csv("dbfs:/mnt/<UseCase>/sample1.csv")
df.printSchema()
df.show()
```

The screenshot shows the Microsoft Azure Databricks workspace interface. On the left is a sidebar with icons for Home, Workspace, Recents, Data, Clusters, Jobs, and Models. The main area is titled 'practicumdbn (Python)' and contains a code editor with the following content:

```

1 df = spark.read.csv("dbfs:/mnt/practicum/sample1.csv")
2 df.printSchema()
3 df.show()

root
|-- _c0: string (nullable = true)
|-- _c1: string (nullable = true)
|-- _c2: string (nullable = true)
|-- _c3: string (nullable = true)
|-- _c4: string (nullable = true)
|-- _c5: string (nullable = true)
|-- _c6: string (nullable = true)
|-- _c7: string (nullable = true)
|-- _c8: string (nullable = true)
|-- _c9: string (nullable = true)
|-- _c10: string (nullable = true)
|-- _c11: string (nullable = true)
|-- _c12: string (nullable = true)

+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| _c0 | _c1 | _c2 | _c3 | _c4 | _c5 | _c6 | _c7 | _c8 | _c9 | _c10 | _c11 | _c12 |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|Month| "Average"|"2005"|"2006"|"2007"|"2008"|"2009"|"2010"|"2011"|"2012"|"2013"|"2014"|"2015"
| May | 0.1| 0| 0| 1| 1| 0| 0| 0| 2| 0| 0| 0 |
| Jun | 0.5| 2| 1| 1| 0| 0| 1| 1| 2| 2| 0| 1 |
| Jul | 0.7| 5| 1| 1| 2| 0| 1| 3| 0| 2| 2| 1 |
| Aug | 2.3| 6| 3| 2| 4| 4| 4| 7| 8| 2| 2| 3 |
| Sep | 3.5| 6| 4| 7| 4| 2| 8| 5| 2| 5| 2| 5 |
| Oct | 2.0| 8| 0| 1| 3| 2| 5| 1| 5| 2| 3| 0 |
| Nov | 0.5| 3| 0| 0| 1| 1| 0| 1| 0| 1| 0| 1 |
| Dec | 0.0| 1| 0| 1| 0| 0| 0| 0| 0| 0| 0| 1 |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+

```

Below the code editor, a message says 'Command took 5.57 seconds -- by rchaphler@microsoft.com at 1/26/2021, 9:48:04 AM on practicumdb'. At the bottom, there is a note 'Shift+Enter to run shortcuts'.

Observations:

- Schema ... the schema is not defined, so the resulting interpretation calls out each field as data type string.
- Resultset ... data formatting {e.g., first row as headers, double quotes, etc.} has not been applied.

Localize Mounted Data

Follow these instructions to localize data from a Data Lake mount.

Prerequisites

This solution requires the following resources:

- Data Lake
- Databricks

Prepare Logic

Why Localize?

Databricks File System (DBFS) is a distributed file system mounted into a Databricks workspace and available on Databricks clusters. DBFS is an abstraction on top of scalable object storage and offers the following **benefits**:

- Allows you to mount storage objects so that you can **seamlessly access data without requiring credentials**.
- Allows you to interact with object storage using directory and file semantics instead of storage URLs.
- Persists files to object storage, so you will not lose data after you terminate a cluster.

[Databricks File System \(DBFS\) — Databricks Documentation](#) | December 16, 2020

Though it is not listed here, many customers refer to the following benefits when justifying the decision to localize data from Azure Data Lake Storage to Azure Databricks:

- Performance Gains
- Decreased Compute Cost
- Leverage Benefits of Partitioning

The Logic

Navigate to Databricks and the <UseCase>dbn notebook. Add a new cell and paste the following code:

```
import uuid
myUUID = uuid.uuid4().hex

spark.read.csv( "/mnt/<UseCase>/sample1.csv" ).write.format( "delta" ).save( "/mnt/" + myUUID )
spark.sql( "CREATE TABLE IF NOT EXISTS myDeltaTable_fromADLS USING DELTA LOCATION '/mnt/" + myUUID +
"/'" )
```

```

import uuid
myUUID = str(uuid.uuid4().hex)
spark.read.csv( "/mnt/practicum/sample1.csv" ).write.format( "delta" ).save( "/mnt/" + myUUID )
spark.sql( "CREATE TABLE IF NOT EXISTS mydeltatable_fromadls USING DELTA LOCATION '/mnt/" + myUUID + "/" )
```

(5) Spark Jobs
Out[3]: DataFrame[]
Command took 13.51 seconds -- by rchabler@microsoft.com at 1/26/2021, 10:26:28 AM on practicumbc

Shift+Enter to run shortcuts

Click on **Data** in the navigation pane. Click on the **default** database, then the **mydeltatable_fromadls** table.

Table: mydeltatable_fromadls

Details History Refresh

Description:
Created at: 2021-01-26 18:26:29
Last modified: 2021-01-26 18:26:32
Partition columns:
Number of files: 1
Size: 3.54 kB

Schema:

col_name	data_type	comment
_c0	string	
_c1	string	
_c2	string	
_c3	string	
_c4	string	
_c5	string	
_c6	string	
_c7	string	

Showing all 16 rows.

Sample Data:

_c0	_c1	_c2	_c3	_c4	_c5
1	Month	"Average"	"2005"	"2006"	"2007"
2	May	0.1	0	0	1
3	Jun	0.5	2	1	1
.

Observation: Previously noted schema and resultset challenges remain in the newly created table.

Simple Load

Synapse has many canned methods for loading and interacting with data.

Copy Data Tool

Follow these instructions to load data from Data Lake to Synapse.

Use Case

Example notes:

- “We want to load structured data from data lake to our Synapse SQL Pool”

Prerequisites

This solution requires the following resources:

- Data Lake (with container and sample data)
- Synapse (with Dedicated SQL Pool)

Note: These instructions also apply (with minor differences) to Azure Data Factory.

Use “Copy Data tool”

Navigate to your Synapse Analytics workspace, click the **Open** link on the “**Open Synapse Studio**” rectangle and then click the **Ingest** button.

Properties

The screenshot shows the Microsoft Azure Synapse Analytics Copy Data tool interface. The left sidebar lists steps: 1 Properties (selected), 2 Source, 3 Target, 4 Settings, and 5 Review and finish. The main area displays the 'Properties' step. It includes a brief description: 'Use Copy Data Tool to perform a one-time or scheduled data load from 90+ data sources. Follow the wizard experience to specify your data loading settings, and let the Copy Data Tool generate the artifacts for you, including pipelines, datasets, and linked services.' A 'Learn more' link is also present. Below this is a 'Properties' section with a sub-section 'Task type'. It shows two options: 'Built-in copy task' (selected) and 'Metadata-driven copy task (Preview)'. The 'Built-in copy task' section contains an icon of two blue cylinders and the text: 'You will get single pipeline to copy data from 90+ data source easily.' The 'Metadata-driven copy task' section contains an icon of a blue cube and a calendar, with the text: 'Metadata is required to be stored in external control tables to load data at large-scale.' Below the task type section, there is a note: 'You will get single pipeline to quickly copy objects from data source store to destination in a very intuitive manner.' Under 'Task cadence or task schedule *', there are three radio button options: 'Run once now' (selected), 'Schedule', and 'Tumbling window'. At the bottom are 'Previous' and 'Next >' buttons, and a 'Cancel' button.

Confirm default select, “Built-in copy task” and “Run once now”, then click the “**Next >**” button.

Source > Dataset

The screenshot shows the 'Copy Data tool' interface in the Microsoft Azure Synapse Analytics workspace 'practicum'. The left sidebar lists steps: Properties (selected), Source, Dataset, Configuration, Target, Settings, and Review and finish. The main panel is titled 'Source data store' and contains the following fields:

- Source type:** Azure Data Lake Storage Gen2
- Connection:** practicumdl (with 'Edit' and '+ New connection' buttons)
- Integration runtime:** AutoResolveIntegrationRuntime (with 'Edit' button)
- File or folder:** practicumdlc/sample1.csv (with 'Browse' button)
- Options:**
 - Binary copy
 - Recursively
 - Enable partition discovery
- Max concurrent connections:** (empty input field)
- Filter by last modified:** (empty input fields for Start time (UTC) and End time (UTC))

At the bottom are 'Previous' and 'Next >' buttons, and a 'Cancel' button.

Complete the “**Source data store**” form, including:

Source Type Select “Azure Data Lake Storage Gen2”

Connection Click the “+ New connection” button

The screenshot shows the Microsoft Azure Copy Data tool interface. On the left, a sidebar lists steps: Properties, Source, Dataset, Configuration, Target, Settings, and Review and finish. The main area is titled "New connection (Azure Data Lake Storage Gen2)". It includes fields for Name (practicumdl), Description, Connect via integration runtime (AutoResolveIntegrationRuntime), Authentication method (Managed Identity), Account selection method (From Azure subscription selected), Azure subscription (rchapler), Storage account name (practicumdl), and Test connection (To linked service selected). A note at the bottom states: "Managed identity name: practicumsrw Managed identity object ID: 53fcba03-a57d-451b-b453-8deb471a3c58 Grant workspace service managed identity access to your Azure Data Lake Storage Gen2. Learn more". At the bottom right, there are "Commit" and "Cancel" buttons, and a message "Connection successful" with a green checkmark.

Complete the “**New connection (Azure Data Lake Storage Gen2)**” form, including:

Connect via...	Confirm default selection, AutoResolveIntegrationRuntime
Account Selection Method	Confirm default selection, “ From Azure subscription ”
Storage Account Name	Select Data Lake

Click “**Test connection**” to confirm successful connection and then click the **Create** (or **Commit**) button.

Back on the “**Source data store**” form, click **Browse** to select container/file {e.g., “<UseCase>dlc/sample1.csv”}.

Click the “**Next >**” button.

Source > Configuration

The screenshot shows the "Copy Data tool" configuration interface in the Microsoft Azure Synapse Analytics portal. The left sidebar lists five steps: Properties (selected), Source, Dataset, Configuration, Target, Settings, and Review and finish. The main panel is titled "File format settings" and contains the following fields:

- File format**: Text format (dropdown)
- Column delimiter**: Comma (,) (dropdown)
- Row delimiter**: Line feed (\n) (dropdown)
- First row as header**: Checked checkbox
- Advanced**: Expander button
- Compression type**: None (dropdown)
- Additional columns**: New (button)

At the bottom of the panel are "Previous" and "Next >" buttons, and a "Cancel" button on the right.

Complete the “**File format settings**” form, confirm default configuration and the resulting data preview.

Make changes if required {e.g., in this example, I check “**First row as header**” to correct handling of incoming data}.

Click the “**Next >**” button.

Target > Dataset

The screenshot shows the 'Copy Data tool' interface in the Microsoft Azure portal. The left sidebar lists steps: Properties (checked), Source (checked), Target (selected), Dataset, Configuration, Settings, and Review and finish. The main panel is titled 'Destination data store' with the sub-instruction: 'Specify the destination data store for the copy task. You can use an existing data store connection or specify a new data store.' It includes dropdowns for 'Target type' (set to 'Azure Synapse Analytics') and 'Connection *' (set to 'Select...'), with a '+ New connection' button. Navigation buttons at the bottom include '< Previous', 'Next >', and 'Cancel'.

Complete the “**Destination data store**” form, including:

Source Type Select “Azure Synapse Analytics”

Connection Click the “+ New connection” button

The screenshot shows the Microsoft Azure Copy Data tool interface. On the left, a sidebar lists steps: Properties, Source, Target, Dataset, Configuration, Settings, and Review and finish. The main area is titled "New connection (Azure Synapse Analytics)". A note says: "Choose a name for your linked service. This name cannot be updated later." The "Name" field contains "practicumsa". The "Description" field is empty. Under "Connect via integration runtime", the dropdown shows "AutoResolveIntegrationRuntime". The "Connection string" tab is selected, showing "Account selection method: From Azure subscription". The "Azure subscription" dropdown shows "rchapler". The "Server name" dropdown shows "practicumsw (Synapse workspace)". The "Database name" dropdown shows "practicumsw_dsp". The "SQL pool" dropdown shows "Select SQL pool". The "Authentication type" dropdown shows "SQL authentication". The "User name" dropdown shows "rchapler". The "AKV linked service" dropdown shows "AzureKeyVault1". The "Secret name" dropdown shows "practicumsw-adminpassword". The "Secret version" dropdown shows "Use the latest version if left blank". Below these, there are sections for "Additional connection properties", "Annotations", and "Parameters". At the bottom are buttons for "Previous", "Next", "Commit", "Cancel", and "Test connection".

Complete the “**New connection (Azure Synapse Analytics)**” pop-out, including:

Connect via...

Confirm default selection, AutoResolveIntegrationRuntime

Account Selection Method	Confirm default selection, “From Azure subscription”
Server Name	Select Synapse Workspace
Database Name	Select your Dedicated SQL Pool
Authentication Type	Confirm default value, “SQL authentication”
User Name	Enter the “ SQL Server admin login ” value used during instantiation of Synapse
AKV Linked Service	Select the name of the Linked Service created for the Key Vault
Secret Name	Enter the Secret Name used to capture the Synapse Workspace administrator password

Click “**Test connection**” to confirm successful connection and then click the **Create** (or **Commit**) button.

The screenshot shows the Microsoft Azure Copy Data tool interface. The left sidebar lists steps: Properties (checked), Source (checked), Target (selected), Dataset, Configuration, Settings, and Review and finish. The main panel is titled 'Destination data store' and contains the following fields:

- Target type:** Azure Synapse Analytics
- Connection *:** practicumsw (dropdown with 'Edit' and 'New connection' buttons)
- Integration runtime *:** AutoResolveIntegrationRuntime (dropdown with 'Edit' button)
- Source:** Azure Data Lake Storage Gen2 file
- Target:** dbo (target schema) and Sample1 (target table, with '(auto-create)' note)
- Buttons:** < Previous, Next >, Cancel

Back on the “**Destination data source**” form, including target schema name {e.g., “dbo”} and target table name {e.g., “Sample1”}.

Click the “**Next >**” button.

Target > Configuration

The screenshot shows the 'Copy Data tool' configuration interface in the Microsoft Azure portal. The left sidebar lists steps: Properties, Source, Target (selected), Dataset, Configuration, Settings, Review and finish. The main area shows 'Column mapping' for 'Table mappings (1)'. It displays a list of columns from the source (Month, Average, 2005, 2006, 2007, 2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015) and their corresponding mappings to the destination (Month, Average, 2005, 2006, 2007, 2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015). Below this is the 'Azure Synapse Analytics sink properties' section, which includes a 'Pre-copy script' input field and an 'Advanced' link. At the bottom are 'Previous' and 'Next >' buttons, and a 'Cancel' button.

Review default column mappings; correct if required.

Click the “**Next >**” button.

Settings

The screenshot shows the 'Copy Data tool' settings page in the Microsoft Azure portal. The left sidebar lists steps: Properties (checked), Source (checked), Target (checked), Settings (checked), and Review and finish (unchecked). The main area is titled 'Settings' and contains the following fields:

- Task name *: CopyPipeline_8th
- Task description: (empty)
- Data consistency verification: (checkbox)
- Fault tolerance: (dropdown menu)
- Enable logging: (checkbox)
- Enable staging: (checkbox)
- Advanced** section:
 - Copy method: Bulk insert (selected)
 - Bulk insert table lock: No (selected)
- Data integration unit: Auto
- You will be charged # of used DIUs * copy duration * \$0.25/DIU-hour. Local currency and separate discounting may apply per subscription type. [Learn more](#)
- Degree of copy parallelism: (dropdown menu) with Edit checkbox checked

At the bottom are 'Previous' and 'Next >' buttons, and a 'Cancel' button.

On the **Settings** form, including:

Enable Staging	Unchecked
Copy Method	Bulk insert

Click the “**Next >**” button.

Summary

Screenshot of the Microsoft Azure Synapse Analytics Copy Data tool configuration summary page.

The left sidebar shows the pipeline steps: Properties, Source, Target, Settings, Review and finish, Review, and Deployment. The Source step is currently selected.

Summary
You are running pipeline to copy data from Azure Data Lake Storage Gen2 to Azure Synapse Analytics.

Properties

Task name	CopyPipeline_8th
Task description	

Source

Connection name	practicumdl
Dataset name	SourceDataset_8th
Column delimiter	,
Row delimiter	
Escape character	\
Quote char	"
First row as header	true
File name	sample1.csv

Target

Connection name	practicumsa
Dataset name	DestinationDataset_8th
Table name	dbo.Sample1

Copy settings

Timeout	7.00:00:00
Retry	0
Retry interval	30
Secure output	false
Secure input	false

Buttons at the bottom: < Previous, Next >, Cancel.

Review configuration and then click the “Next >” button.

Deployment

The screenshot shows the Microsoft Azure Synapse Analytics Copy Data tool interface. On the left, a vertical navigation bar lists steps: Properties, Source, Target, Settings, Review and finish, Review, and Deployment. The 'Source' step is currently selected, indicated by a checkmark. The main pane displays a flow from 'Azure Data Lake Storage Gen2' to 'Azure Synapse Analytics'. Below this, a message says 'Deployment complete'. A table titled 'Deployment step' shows two entries: 'Creating datasets' and 'Creating pipelines', both with a status of 'Succeeded' and green checkmarks. At the bottom are buttons for 'Finish', 'Edit pipeline', and 'Monitor'.

Monitor progress.

Click the **Finish** button on success.

Confirm Success

“**CopyPipeline...**” will be added to the list of **Pipelines** in the Synapse, **Integrate** section.

The screenshot shows the Microsoft Azure Synapse Analytics Integrate section. On the left, a sidebar shows a 'collaboration branch' and a list of activities including Synapse, Move & transform, Azure Data Explorer, Azure Function, Batch Service, Databricks, Data Lake Analytics, General, HDInsight, Iteration & conditionals, and Machine Learning. In the center, a pipeline named 'CopyPipeline_8th' is displayed. A modal window titled 'Copy data' shows a successful run of the 'Copy_8th' activity. Below the modal, the 'Output' tab of the pipeline details page is shown, displaying a table of pipeline runs. One run is listed: 'Copy_8th' (Type: Copy data), Run start: 2021-09-22T17:38:30.320, Duration: 00:00:10, Status: Succeeded, Integration runtime: DefaultIntegrationRuntime. A link 'View debug run consumption' is also present.

Click **Debug** and confirm successful execution.

Click the **Data** icon in the navigation pane.

The screenshot shows the Microsoft Azure Synapse Analytics workspace interface. The left sidebar has a 'Data' icon selected. The main area displays a SQL script editor with a dropdown menu open over a table named 'dbo.Sample1'. The dropdown menu includes options like 'New SQL script', 'New notebook', 'New data flow', 'New integration dataset', 'Refresh', and 'Aug'. The table 'dbo.Sample1' has columns for years from 2005 to 2011. The results pane shows the following data:

	"2005"	"2007"	"2008"	"2009"	"2010"	"2011"
0	1	1	0	0	0	0
1	1	0	0	1	1	1
2	1	1	2	0	1	3
3	3	2	4	4	4	7

At the bottom of the results pane, it says '00:00:00 Query executed successfully.'

Expand **Databases > {Dedicated SQL Pool} > Tables** and confirm that you can see your new table.

Right-click on your table, select “**New SQL script**”, and then “**Select TOP 100 rows**” in the drop-down menu.

Pipeline

Follow these instructions to load data from Data Explorer to Synapse.

Use Case

Example notes:

- “We want to load structured data from SQL to Data Explorer”

Prerequisites

This solution requires the following resources:

- Data Explorer (with cluster, database, and sample Product table)
- SQL (with sample data)
- Synapse (with linked services)

Note: These instructions also apply (with minor differences) to Azure Data Factory.

Confirm Linked Services

Navigate to your Synapse Analytics workspace and then click the **Open** link on the “**Open Synapse Studio**” rectangle. Select the **Manage** icon on the navigation and then “**Linked Services**” in the resulting navigation menu. Confirm linked service instantiation for: 1) Key Vault, 2) SQL and 3) Data Explorer.

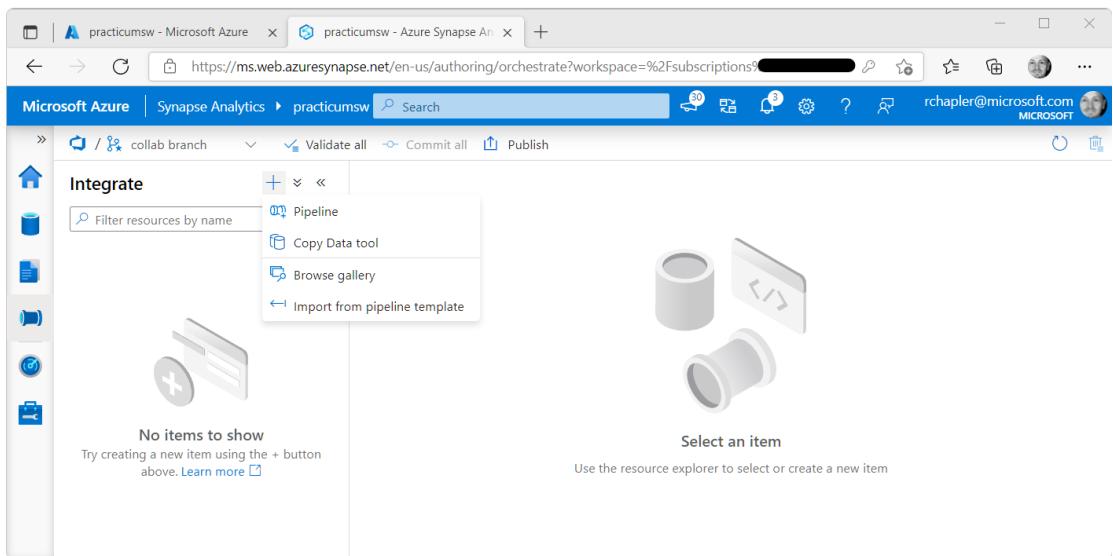
The screenshot shows the Microsoft Azure portal interface for a Synapse Analytics workspace named 'practicumsw'. The left sidebar navigation includes 'Analytics pools', 'External connections', 'Linked services' (which is currently selected), 'Integration', 'Access control', 'Code libraries', 'Source control', and 'Git configuration'. The main content area is titled 'Linked services' and contains a table listing five entries:

Name	Type	Related	Annotations
practicumde	Azure Data Explorer (Kusto)	0	
practicumkv	Azure Key Vault	1	
practicumsd	Azure SQL Database	0	
practicumsw-WorkspaceDefaultSqlServer	Azure Synapse Analytics	0	
practicumsw-WorkspaceDefaultStorage	Azure Data Lake Storage Gen2	0	

Note: you are likely to see system-generated linked services {e.g., "<UseCase>sw-WorkspaceDefault..."} in addition to instantiated items.

Create Pipeline

Click the **Integrate** icon in the navigation pane.



Click the **+** icon just above and to the right of the “**Filter resources by name**” input.

Select **Pipeline** from the resulting drop-down menu.

The screenshot shows the Microsoft Azure Synapse Analytics pipeline editor. On the left, the navigation pane is open with 'Integrate' selected. Under 'Pipelines', 'Pipeline 1' is selected. The main area shows a 'Copy data' activity named 'Copy data1'. Below it, there are tabs for General, Source (highlighted), Sink, Mapping, Settings, and User properties. The 'Source' tab has a 'Source dataset' dropdown set to 'Select...'. The top right of the interface has buttons for Commit, Save as template, Validate, Add trigger, and more.

Add a “**Copy data**” activity.

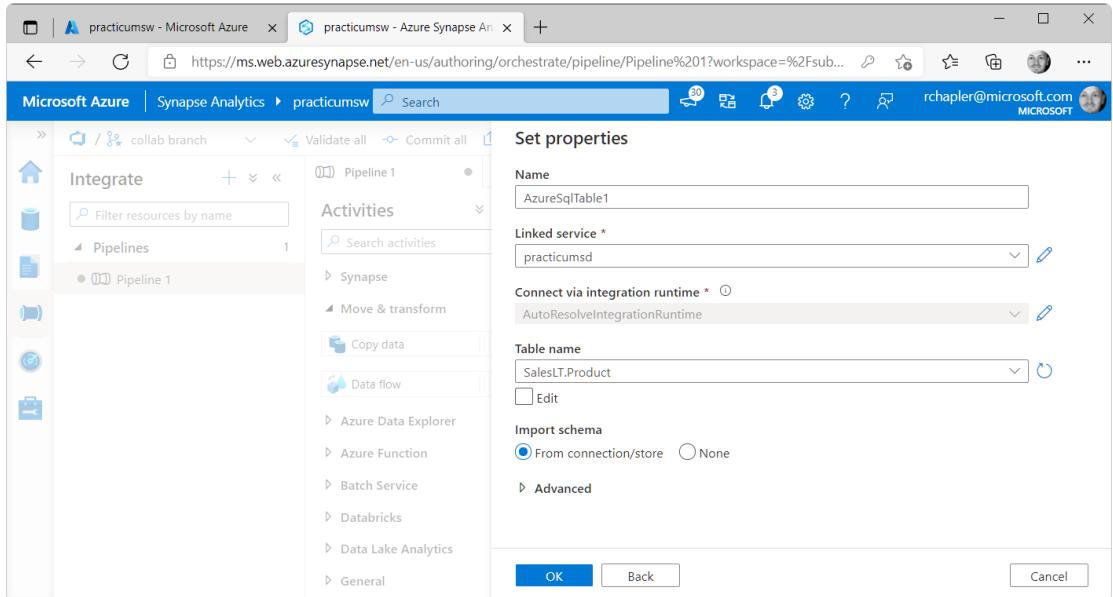
Source

On the **Source** tab, click the “**+ New**” button.

The screenshot shows the 'New integration dataset' dialog. At the top, it says 'New integration dataset'. Below that, a message says 'In pipeline activities and data flows, reference a dataset to specify the location and structure of your data within a data store. [Learn more](#)'. A search bar contains 'azure sql'. Below the search bar, tabs are shown: All (selected), Azure, Database, File, Generic protocol, NoSQL, Services and apps. Two options are listed: 'Azure SQL Database' and 'Azure SQL Database Managed Instance'. At the bottom are 'Continue' and 'Cancel' buttons.

Complete the “**New integration dataset**” pop-out, search for and then select “**Azure SQL Database**.”

Click the **Continue** button.



Complete the “**Set properties**” pop-out, including:

Linked Service Select your Azure SQL linked service

Table Name Select “SalesLT.Product”

Import Schema Confirm default selection, “From connection/store”

Leave all other settings with default values.

Click the **OK** button.

The screenshot shows the Microsoft Azure Synapse Analytics pipeline editor. On the left, the navigation pane is open with the 'Integrate' tab selected. Under 'Pipelines', 'Pipeline 1' is selected. The main workspace displays the 'Activities' section, which is currently expanded. A 'Copy data' activity is selected, shown in a preview window. The 'Source' tab is active, showing 'Source dataset' set to 'AzureSqlTable1'. Other settings include 'Use query' (Table selected), 'Query timeout (minutes)' set to 120, and 'Isolation level' set to 'None'. Below these, there are options for 'Partition option' (None selected) and 'Additional columns'. A note at the bottom says: 'Please preview data to validate the partition settings are correct before you trigger a run or publish the pipeline.'

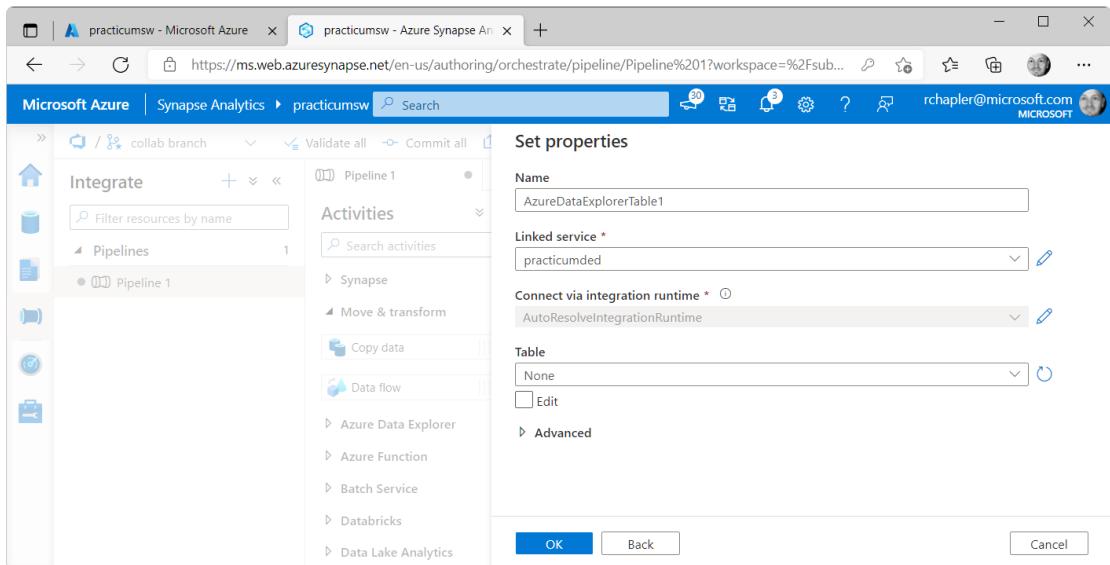
Sink

Click on the **Sink** tab and then click the “+ New” button.

The screenshot shows the Microsoft Azure Synapse Analytics pipeline editor with the 'Activities' section expanded. A 'New integration dataset' dialog is open over the workspace. The dialog title is 'New integration dataset' and it contains instructions: 'In pipeline activities and data flows, reference a dataset to specify the location and structure of your data within a data store.' Below this is a search bar labeled 'Select a data store' with 'data explorer' typed in. A list of data stores is shown, with 'Azure Data Explorer (Kusto)' highlighted. At the bottom of the dialog are 'Continue' and 'Cancel' buttons.

Complete the “New integration dataset” pop-out, search for and then select “Azure Data Explorer (Kusto).”

Click the **Continue** button.



Complete the “**Set properties**” pop-out, including:

Linked Service Select your Azure Data Explorer linked service

Table Name Select **Product**

Leave all other settings with default values.

Click the **OK** button.

The screenshot shows the Microsoft Azure Synapse Analytics Pipeline Editor. On the left, the 'Integrate' sidebar is open, showing a 'Pipelines' section with 'Pipeline 1' selected. The main area displays the 'Activities' pane, which is currently expanded to show the 'Move & transform' section. A 'Copy data' activity is selected and highlighted with a red circle. The pipeline canvas shows a single 'Copy data' component. Below the canvas, the 'Sink' tab is active in the configuration pane, showing the sink dataset as 'AzureDataExplorerTable1' and the database as 'practicumded'. Other tabs include General, Source, Mapping, Settings, and User properties.

Mapping

Click on the **Mapping** tab and then click the “Import schemas” button.

The screenshot shows the same Microsoft Azure Synapse Analytics Pipeline Editor interface, but now the 'Mapping' tab is selected in the configuration pane below the 'Sink' tab. The 'Import schemas' button is highlighted with a green checkmark. The mapping details show two source columns: 'ProductNumber' (Type: nvarchar) and 'ListPrice' (Type: money), both mapped to destination columns: 'ProductNumber' (Type: String) and 'ListPrice' (Type: Decimal). The 'Source' and 'Destination' dropdowns are also visible.

Confirm Success

Click **Validate** to confirm that there are no errors, and then click **Debug** to confirm that the pipeline runs successfully.

The screenshot shows the Microsoft Azure Synapse Analytics Pipeline Editor interface. On the left, the navigation pane is open with 'Integrate' selected, showing a list of activities including Synapse, Move & transform, Azure Data Explorer, Azure Function, Batch Service, Databricks, Data Lake Analytics, General, HDInsight, Iteration & conditionals, and Machine Learning. A pipeline named 'Pipeline 1' is selected. The main workspace displays a single activity named 'Copy data1'. Below the activity, there are tabs for Parameters, Variables, Settings, and Output. The Output tab is active, showing a table of the pipeline run. The table has columns: Name, Type, Run start, Duration, Status, and Integration runtime. One row is visible: 'Copy data1' (Type: Copy data), Run start: 2021-10-04T20:59:21.848, Duration: 00:00:15, Status: Succeeded (indicated by a green checkmark), and Integration runtime: DefaultIntegrationRuntime. At the top of the workspace, there are buttons for Commit, Save as template, Validate, Debug, and Add trigger. The status bar at the bottom indicates the pipeline run ID: 77b27627-6862-4dea-87ba-da3a3c9586a5.

Name	Type	Run start	Duration	Status	Integration runtime
Copy data1	Copy data	2021-10-04T20:59:21.848	00:00:15	Succeeded	DefaultIntegrationRuntime

Conditional Activity

Follow these instructions to demonstrate conditional logic in a pipeline.

Use Case

Example notes:

- “We want to include conditional handling in our pipeline”

Prerequisites

This solution requires the following resources:

- Synapse

Note: These instructions also apply (with minor differences) to Azure Data Factory.

Create Pipeline

Navigate to your Synapse Analytics workspace and then click the **Open** link on the “**Open Synapse Studio**” rectangle.

Click the **Integrate** icon in the navigation pane.

Click the **+** icon just above and to the right of the “**Filter resources by name**” input.

Select **Pipeline** from the resulting drop-down menu.

The screenshot shows the Microsoft Azure Synapse Analytics studio interface. The left sidebar has an 'Integrate' icon selected. In the main area, a pipeline named 'Pipeline 1' is being edited. The 'Activities' tab is active, showing a list of available activities: Synapse, Move & transform, Azure Data Explorer, Azure Function, Batch Service, Databricks, Data Lake Analytics, General, HDInsight, Iteration & conditionals, and Machine Learning. Below the activities list, there are tabs for 'Parameters', 'Variables', 'Settings', and 'Output'. Under 'Variables', there is a table with two rows. The first row has columns: 'Name' (Flag), 'Type' (Boolean), and 'Default value' (Value). The second row has columns: 'Name' (Result), 'Type' (String), and 'Default value' (Value). At the bottom of the 'Variables' section, there are buttons for '+ New' and 'Delete'.

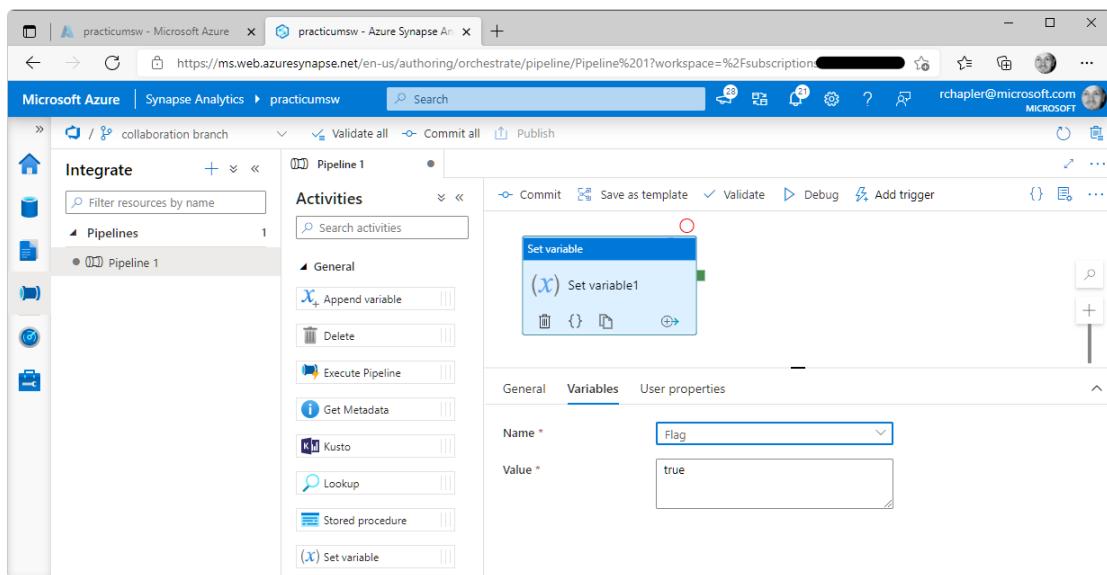
Click on the **Variables** tab.

Click “**+ New**” to add each of the following variables:

Flag (Boolean)	Will serve as a TRUE / FALSE trigger for the conditional result
Result (String)	For capturing an anecdotal result {e.g., "Success!"}

Activity 1, Set Variable

Expand **General** in the **Activities** bar, then drag-and-drop a “**Set variable**” component into the activity window.



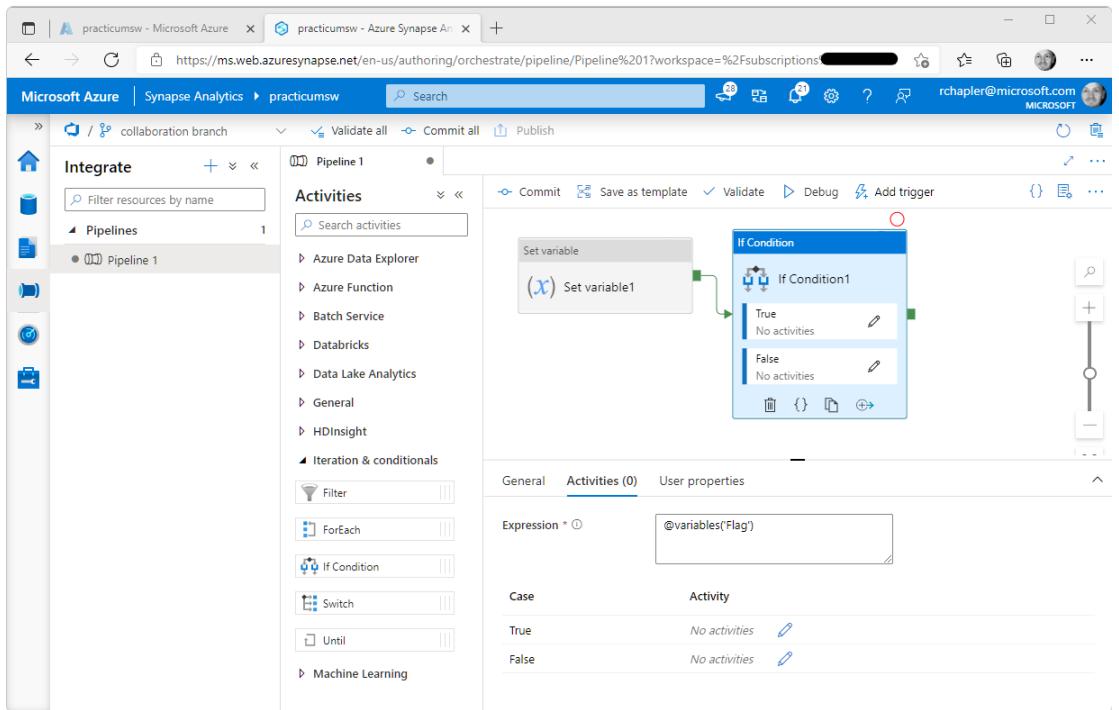
On the **Variables** tab of the “**Set Variable...**” component, including:

Name	Select Flag from the drop-down menu
Value	Enter true

Activity 2, If Condition

Expand “Iteration & conditionals” in the Activities bar.

Drag-and-drop an “**If Condition**” component into the pipeline.

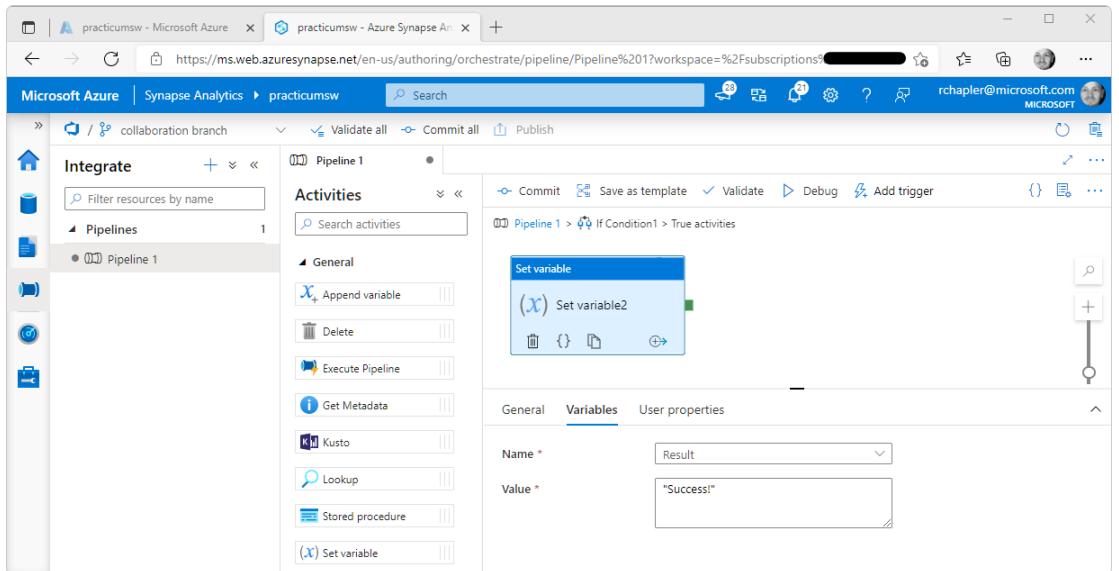


Create a dependency between the “**Set variable**” and “**If Condition**” components.

Click on the “**Activities (0)**” tab.

Click into the **Expression** textbox, and then enter the following: @variables('Flag')

Click on the pencil icon in the “**If Condition**” component, **True** selection.



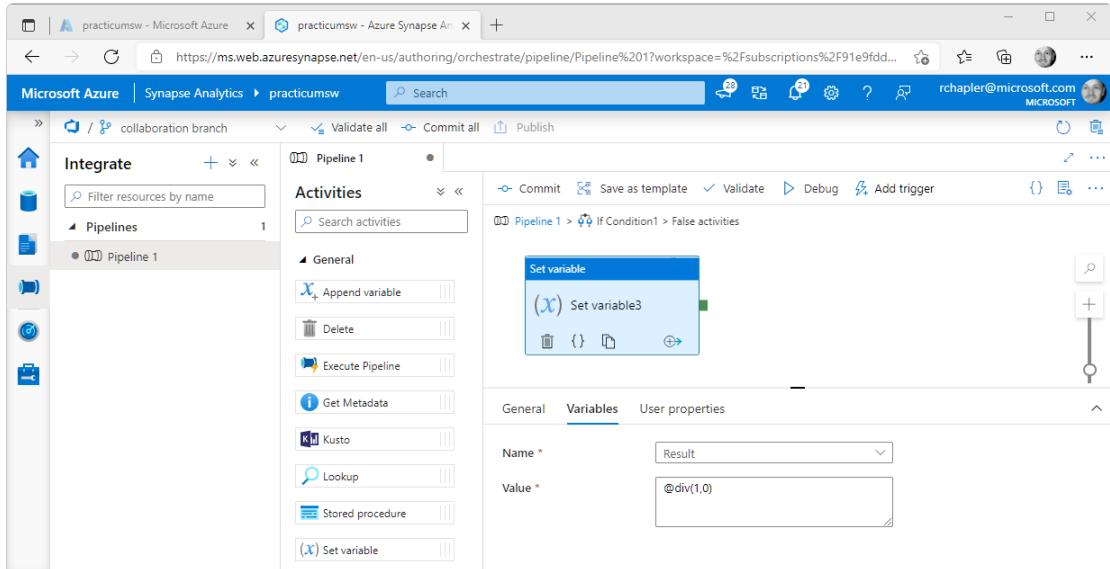
Expand **General** in the **Activities** bar, then drag-and-drop a “**Set variable**” component into the activity window.

On the **Variables** tab of the “**Set Variable...**” component, including:

Name	Select Result from the drop-down menu
Value	Enter “Success!”

Use cookie crumbs to navigate back to the main window.

Click on the pencil icon in the “**If Condition**” component, **False** selection.



Expand **General** in the **Activities** bar, then drag-and-drop a “**Set variable**” component into the activity window.

On the **Variables** tab of the “**Set Variable...**” component, including:

Name	Select Result from the drop-down menu
Value	Enter @div(1,0) ... this will force an error result

Confirm Success

Click **Debug**.

The screenshot shows the Microsoft Azure Synapse Analytics pipeline editor. On the left, the navigation pane is open with 'Integrate' selected, showing 'Pipelines' and 'Pipeline 1'. The main workspace displays a pipeline diagram with a 'Set variable' activity followed by an 'If Condition' activity. The 'Output' tab is selected, showing a table of pipeline runs. The first run, 'Pipeline run ID: 08bb1dff-2b6d-4023-8ab7-750cbcff2423', has all activities marked as 'Succeeded'. The status bar at the bottom right shows 'rchapler@microsoft.com MICROSOFT'.

You should see **Succeeded** messages given current variable settings.

Next, we will make a pipeline change that will force pipeline failure.

The screenshot shows the Microsoft Azure Synapse Analytics pipeline editor. The pipeline diagram is identical to the previous one, but the 'Output' tab is not selected. Instead, the 'Variables' tab is selected for the 'Set variable1' activity. The 'Value' field is set to 'false', which is highlighted with a red circle. The status bar at the bottom right shows 'rchapler@microsoft.com MICROSOFT'.

Navigate to the “Set variable1” component, **Variables** tab.

Replace the **true** value with **false** and then click **Debug**.

You will see **Failed** messages given current variable settings.

Incremental Load

There is more than one way to incrementally load of data. Your chosen method will hinge on questions like...

- Does the source offer “delta comparison”-type functionality?
- Does the source include both identifier and waterline column? If not, could you use some form of a CHECKSUM functionality?
- Does the source include a “soft delete” column (and corresponding timestamp)?
- Does the Azure resource you plan to use {e.g., pipeline, data flow, etc.} provide for connection to the source? Is data staging required?
- Is it cheaper to separately load an initial batch of data {e.g., historical}?

The following sections explore use cases and corresponding methods (based on these criteria).

Pipeline

Follow these instructions to establish incremental load using a Synapse Pipeline to move data from a SQL database to a Data Explorer database.

Use Case

Example notes:

- “We want to efficiently load data from an **Oracle** operational data source”
- “We want to capture a time series of delete, insert, and update events and provide for “time travel” through data”

Through additional research we learn that the Oracle currently lacks a corresponding Data Flow connector, so we must use a Pipeline.

Prerequisites

This solution requires the following resources:

- SQL (with sample database) ... using this **in lieu of Oracle** given complexity of instantiation
- Synapse with Data Explorer Pool, Data Explorer Database, and Linked Services for Data Explorer and SQL

Note: These instructions also apply (with minor differences) to Azure Data Factory and Azure Data Explorer.

Prepare Source

Navigate to SQL, click the “**Query editor...**” navigation icon, and authenticate. Add IP to firewall settings, if necessary. Execute the following logic:

```
CREATE TABLE [dbo].[Product_Pending] ( [_id] varchar(16), [_op] varchar(8) )
```

Prepare Destination

Navigate to Synapse Studio, click the **Develop** navigation icon, click “+” and select “**KQL script**” from the resulting drop-down menu.

Create Destination Table

Execute the following logic:

```
.create table _product (
    ProductId:int
    , ProductNumber:string
    , Name:string
    , ListPrice:decimal
    , ModifiedDate:datetime
    , _id:string
    , _op:string
    , _dt:datetime
) with ( folder = 'bronze' )
```

(Partial) logic explanation:

- Included columns should match their source counterpart (name and equivalent data type)
- The “**_id**” column provides a string-based, generic capture of various identifier data types {e.g., int, guid, etc.} and situations where unique identification may require concatenation of multiple fields {e.g., “123-456”}
- The “**_op**” column provides for flagging of **delete**, **insert**, or **update** operations
- The “**_dt**” column provides standard capture of the waterline date value for a given row
- “**with (folder = ‘bronze’)**” groups destination tables {i.e., the raw data} into a “**bronze**” folder

Create Materialized View (“..._Latest”)

Execute the following logic:

```
.create materialized-view with ( folder = 'silver', backfill = true ) _product_latest on table _product {  
    _product  
    | extend _opdt = ingestion_time()  
    | summarize arg_max(_opdt, *) by _id  
}
```

(Partial) logic explanation:

- “**backfill = true**” will include all existing records (as opposed to only including records ingested after view creation)
- “**extend _opdt...**” surfaces ingestion_time()
- “**...arg_max(_opdt, *) by _id**” finds and returns the row where _opdt is maximized for a given _id

Note: Materialized Views are not instantaneously populated... if you backfill a large amount of data and then run a query against the materialized view quickly after, you are likely to see incomplete results. If it is important to access the data quickly {i.e., soon after addition}, consider using a Function rather than a Materialized View.

Create User-Facing Function

Simplify user experience {i.e., “I want to see the latest data and I do not care about deleted rows!”} by preparing a Function in Data Explorer.

Execute the following logic:

```
create-or-alter function product( excludeDeletes: bool = true )  
{ _product_latest | where _op != iif( excludeDeletes, 'delete', '' ) }
```

Create _Waterline Table

Execute the following logic:

```
.create table _product_waterline ( _id : string, _dt : datetime ) with ( folder = 'bronze' )
```

The second column might also be named and data-typed for values like “_cs” {i.e., checksum}, “_hash”, “_hex” depending on what you need to use to waterline the source.

Create _Pending Function

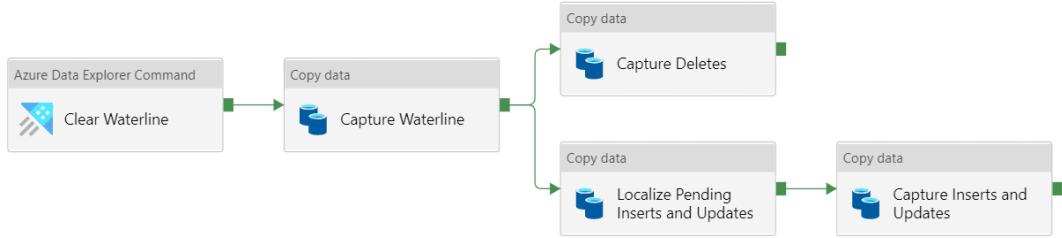
Execute the following logic:

```
.create function _product_pending with (folder='bronze')
{
    _product_latest
    | project latest_dt = _dt, latest_id = _id, latest_op = _op
    | join kind = fullouter ( _product_waterline | project waterline_id = _id, waterline_dt = _dt )
        on $left.latest_id == $right.waterline_id
    | project _id = case(
        isnull(waterline_dt) and latest_op != 'delete', latest_id // delete
        , isnull(latest_dt), waterline_id // insert
        , latest_dt != waterline_dt, waterline_id // update
        , '' )
    , _op = case(
        isnull(waterline_dt) and latest_op != 'delete', 'delete'
        , isnull(latest_dt), 'insert'
        , latest_dt != waterline_dt, 'update'
        , '' )
    | where not(isempty(_id)) and _op in ('delete', 'insert', 'update')
}
```

Note: If you are using an alternative to waterline “_dt” {e.g., checksum, hex, hash, etc.}, you may have to replace the use of ISNULL with ISEMPTY

Create Pipeline

When we are finished, our pipeline will look and function as snipped below.



Navigate to Synapse Studio, click the **Integrate** navigation icon, click “+” and select **Pipeline** from the resulting drop-down menu.

Activity 1: Clear Waterline

This activity will clear any previously created waterline data.

Expand “**Azure Data Explorer**” in the **Activities** bar, then drag-and-drop an “**Azure Data Explorer Command**” component into the activity window.

On the **Connection** tab, select the Data Explorer Linked Service from the drop-down menu.

On the **Command** tab, paste the following KQL in the Command textbox.

```
.clear table _product_waterline data
```

Click **Debug** and monitor result to confirm successful progress.

Activity 2: Capture Waterline

This activity will capture a dataset that will be used to identify deleted, inserted, or updated records. Capture of _waterline data to database is necessary in Pipelines because unlike Data Flow, there is no in-flow use of data.

Expand “**Move & Transform**” in the **Activities** bar, drag-and-drop a “**Copy data**” component into the activity window and then create a dependency from the “**Clear Waterline**” component.

On the **Source** tab, including:

Source Dataset	Create a new “ Azure SQL Database ” dataset using the SQL linked service and the “ None ” table {e.g., “<UseCase>sd_None”}
Use Query	Select the Query radio button
Query	Paste the following T-SQL: <pre>SELECT ProductId _id,MAX(ModifiedDate) _dt FROM SalesLT.Product GROUP BY ProductId</pre>

No additional configuration is required on this tab.

On the **Sink** tab, including:

Sink Dataset	Create a new “ Azure Data Explorer (Kusto) ” dataset using the Data Explorer linked service and the “_product_waterline” table {e.g., “<UseCase>ded_product_waterline”}
Database and Table	Confirm values

No additional configuration is required on this tab.

On the **Mapping** tab, click the “**Import schemas**” button.

Publish changes and then click **Debug** and monitor result to confirm successful progress.

Activity 3: Capture Deletes

This activity will **capture data associated with pending delete operations** (using already-captured data in Data Explorer since the data is no longer available at the source).

Expand “**Move & Transform**” in the **Activities** bar, drag-and-drop a “**Copy data**” component into the activity window and then create a dependency from the “**Capture Pending**” component.

On the **Source** tab, including:

Source Dataset	Select the <UseCase>ded_product_pending dataset
Use Query	Select the Query radio button
Query	Paste the following parameterized KQL: <pre>_product_pending where _op == 'delete' join _product on _id project ProductId, ProductNumber, Name, ListPrice, ModifiedDate, _id, _dt, _op = _op1</pre>

No additional configuration is required on this tab.

On the **Sink** tab, including:

Sink Dataset	Create an “ Azure Data Explorer (Kusto) ” dataset using the Data Explorer linked service and the “ _product ” table {e.g., “<UseCase>ded_product”}
Database and Table	Confirm values

No additional configuration is required on this tab.

On the **Mapping** tab, click the “**Import schemas**” button.

Click **Debug** and monitor result to confirm successful progress.

Activity 4: Localize Pending Inserts and Updates

This activity will localize “_pending” data for delete operations to the source for a SQL JOIN.

Expand “**Move & Transform**” in the **Activities** bar, drag-and-drop a “**Copy data**” component into the activity window and then create a dependency from the “**Capture Pending**” component.

On the **Source** tab, including:

Source Dataset	Select the <UseCase>ded_product_pending dataset
Use Query	Select the Query radio button
Query	Paste the following KQL: <code>_product_pending where _op in ('insert', 'update')</code>

No additional configuration is required on this tab.

On the **Sink** tab, including:

Sink Dataset	Create a new “ Azure SQL Database ” dataset using the SQL linked service and the “ dbo.Product_Pending ” table {e.g., “<UseCase>sd_Product_Pending”}
Pre-Copy Script	Paste the following T-SQL: <code>TRUNCATE TABLE [dbo].[Product_Pending]</code>

No additional configuration is required on this tab.

On the **Mapping** tab, click the “**Import schemas**” button.

Click **Debug** and monitor result to confirm successful progress.

Activity 5: Capture Inserts and Updates

This activity will capture data associated with pending insert and update operations (using data from the source system).

Expand “**Move & Transform**” in the **Activities** bar, drag-and-drop a “**Copy data**” component into the activity window and then create a dependency from the “**Capture Pending**” component.

On the **Source** tab, including:

Source Dataset	Select <UseCase>sd_None
Use Query	Select the Query radio button
Query	Paste the following T-SQL: <pre>SELECT x.[ProductId], x.[ProductNumber], x.[Name], x.[ListPrice], x.[ModifiedDate], x.[ProductId] [_id], x.[ModifiedDate] [_dt], p.[_op] FROM [SalesLT].[Product] x INNER JOIN [dbo].[Product_Pending] p ON x.[ProductId] = p._id</pre>

No additional configuration is required on this tab.

On the **Sink** tab, including:

Sink Dataset	Select <UseCase>ded_Product
Database and Table	Confirm values

No additional configuration is required on this tab.

On the **Mapping** tab, click the “**Import schemas**” button.

Click **Debug** and monitor result to confirm successful progress.

Confirm Success

The first and most obvious confirmation is that your completed pipeline runs successfully. Then, of course, confirm that data is correctly captured.

Confirmation 1, Record Count

Navigate to Synapse Studio, click the **Data** navigation icon, expand “**Data Explorer Databases...**” and your Data Explorer Pool. Right click on your Data Explorer Database {e.g., <UseCase>ded} and select “**New KQL script**” from the resulting drop-down menu.

Paste and Run the following logic to validate debug runs {i.e., with a non-zero record count}:

```
product | count
```

Confirmation 2, Delete Operation

Navigate to SQL, click the “**Query editor...**” navigation icon, and authenticate (adding IP to firewall if necessary). Execute the following logic:

```
SELECT MIN( [ProductID] ) FROM [SalesLT].[Product]
```

Use the returned value to execute the following logic:

```
DELETE FROM [SalesLT].[Product] WHERE [ProductID] = {MIN ProductId value}
```

Navigate to the Product pipeline in Synapse, click **Debug** and monitor result to confirm successful progress. Execute the following logic to confirm that the “delete” record is included in the Product table:

```
_product | where _op == 'delete'
```

Execute the following logic to confirm that you can see the full history for the deleted item:

```
_product | where _id == {deleted ProductId value}
```

sharedsw - Azure Synapse Analytics

https://ms.web.azuresynthesize.net/en-us/authoring/explore/workspace/kqlscripts/KQL%20script%201?workspace=%2Fsubscriptions%2F...

Microsoft Azure | Synapse Analytics > sharedsw

Synapse live | Validate all | Publish all

KQL script 1

Run Undo Publish Connect to shareddep Use database shareded

1 _product | where _id == 706

Results Messages

View Table Chart Export results

Search

ProductId	ProductNumber	Name	ListPrice	ModifiedDate	_id	_op	_dt
706	FR-R92R-58	HIL Road Frame...	1431.5	2008-03-11T10...	706	insert	201
706	FR-R92R-58	HIL Road Frame...	1431.5	2008-03-11T10...	706	delete	201

00:00:00 Query executed successfully.

The screenshot shows the Microsoft Azure Synapse Analytics Data Explorer interface. On the left, there's a navigation pane with icons for Home, Databases, Linked Services, and Data. Under 'Data', it shows 'Workspace' and 'Linked'. Below that is a tree view of 'Data Explorer Databases' under 'shareddep': 'shareded' (selected), 'External tables', 'Functions' (with 'product' listed), 'Materialized Views', and 'Tables'. In the main area, a KQL script is run against the 'shareded' database. The query is: '1 _product | where _id == 706'. The results show two rows from the 'product' table. The first row is an 'insert' operation with ProductId 706, ProductNumber 'FR-R92R-58', Name 'HIL Road Frame...', ListPrice 1431.5, ModifiedDate '2008-03-11T10...', and _id 706. The second row is a 'delete' operation for the same row. A message at the bottom indicates the query was executed successfully in 00:00:00.

Confirmation 3, Insert Operation

Navigate to SQL, click the “**Query editor...**” navigation icon, and authenticate (adding IP to firewall if necessary). Execute the following logic:

```
INSERT INTO [SalesLT].[Product] ([Name], [ProductNumber], [Color], [StandardCost], [ListPrice],  
[SellStartDate], [rowguid], [ModifiedDate])  
VALUES ('New Product ABC','ABC-123','bronze',1000,1500,GETDATE(),NEWID(),GETDATE())
```

Navigate to the Product pipeline in Synapse, click **Debug** and monitor result to confirm successful progress. Execute the following logic to confirm that the “insert” record is included in the Product table:

```
_product | where Name contains 'New Product'
```

Confirmation 4, Update Operation

Navigate to SQL, click “**Query Editor**” in the navigation, and then login (whitelist your IP address, if required). Execute the following logic to update a record:

```
UPDATE SalesLT.Product SET ModifiedDate=GETDATE() WHERE ProductID=(SELECT MAX([ProductID]) FROM  
[SalesLT].[Product])
```

Navigate to the Product pipeline in Synapse, click **Debug** and monitor result to confirm successful progress. Execute the following logic to confirm that the updated record is included in the Product table:

```
Product | where Name contains 'New Product'
```

Create Trigger

Establish a recurring schedule for your completed pipeline.

Navigate to the new pipeline in Synapse, and then click the “**Add trigger**” button and click “**New/Edit**” in the resulting drop-down menu.

On the resulting pop-out, click the “**Choose trigger...**” drop-down menu and select “**+ New**.”

Complete the “**New trigger**” pop-out, including:

Type	Select “ Schedule ”
Start Date	Confirm default value (consider using round hour / minute to avoid odd timing)
Time Zone	Confirm default value
Recurrence	Choose appropriate recurrence

No additional configuration is required . Review remaining tabs and then click the **Commit** button.

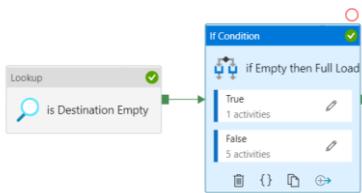
Special Circumstances

Not every source will align to the pattern described above; some of these unusual circumstances are described below.

Significant Initial Volume

Scenario: While implementing the incremental load pipeline, you recognize that processing the initial volume of source data through waterline, etc. processes is wasteful.

Proposed Solution: Consider amending your pipeline with the following precursor actions: 1) “sniff” of the destination {i.e., has any data previously been loaded}, and 2) conditional that with either initiate a simple load or an incremental load.



Lookup: Is Destination Empty?

Expand **General** in the **Activities** bar, then drag-and-drop a **Lookup** component into the activity window. On the **Settings** tab, including:

Source Dataset	Select the Data Explorer Dataset from the drop-down menu
First Row Only	Check
Query	Personalize and paste the following KQL: <code>_product_raw count</code>

If Condition: If Empty, then Full Load

Expand “**Iteration & Conditionals**” in the **Activities** bar, then drag-and-drop an “**If Condition**” component into the activity window. Create a dependency from the “**Sniff Destination**” component. On the **Activities** tab, enter the following **Expression** value:

```
@equals( activity('Sniff Destination').output.firstRow.Count, 0 )
```

Click on the pencil icon in the “**If Condition**” component, **True** selection and paste the logic for Simple Load {i.e., copy initial batch of data}.

Click on the pencil icon in the “**If Condition**” component, **False** selection and paste the logic for Incremental Load.

Slow Source

Scenario: While implementing the incremental load pipeline, you observe that the source server is severely under-powered ... running a simple waterline takes longer than a full trunc-and-load.

Proposed Solution #1: If you must evaluate delete, insert, and update events for all identifiers, consider starting the pipeline with trunc-and-load {i.e., dump and re-load of data in a destination table}. With data localized to Azure Data Explorer, all ensuing steps {e.g., waterline, pending changes, etc.} can be evaluated with functions.

Proposed Solution #2: If you only care about insert events (as in the case of a time series), consider starting the pipeline with an initial sniff of the destination table to find out the overall maximum waterline value, and then only pull those source records with a waterline value greater than that max.

Max Waterline

Expand **General** in the **Activities** bar, then drag-and-drop a **Lookup** component into the activity window. On the **Settings** tab, including:

Source Dataset	Select the Data Explorer Dataset from the drop-down menu
First Row Only	<input checked="" type="checkbox"/>
Query	Personalize and paste the following KQL: <code>_product_raw summarize theMax = max(modifieddate)</code>

Incremental Load

Expand “**Move & Transform**” in the **Activities** bar, drag-and-drop a “**Copy data**” component into the activity window and then create a dependency from the “**Sniff Destination**” component.

On the **Source** tab, including:

Source Dataset	Create a new “ Azure SQL Database ” dataset using the SQL linked service and the “ None ” table {e.g., “<UseCase>sd_None”}
Use Query	Select the Query radio button
Query	Paste the following dynamic expression: <code>@concat('SELECT * FROM [Product] WITH (NOLOCK)' ' , ' WHERE [ModifiedDate] > CAST(''' ' , string(activity('Max Waterline').output.firstRow.theMax) ' , '''' AS DATETIME)')</code>

No additional configuration is required on this tab.

CHECKSUM” vs. Waterline

If your source data lacks a waterline column, you will need to use “checksum”-type logic that can evaluate the data in one or more columns, return a coded value representing that data and provide for comparison against data in future evaluations of the one or more columns.

For example, your Source Query in “Capture Waterline” might look something like:

```
SELECT CONCAT(x.COL1, '') AS ID, STANDARD_HASH( x.COL2 || x. COL3 || x. COL4, 'SHA256' ) AS CS
FROM SCHEMA.TABLE x
```

This change will need to be considered for all related functions, etc.

Data Flow

Follow these instructions to demonstrate incremental load using Synapse Data Flows and Delta Lake.

Prerequisites

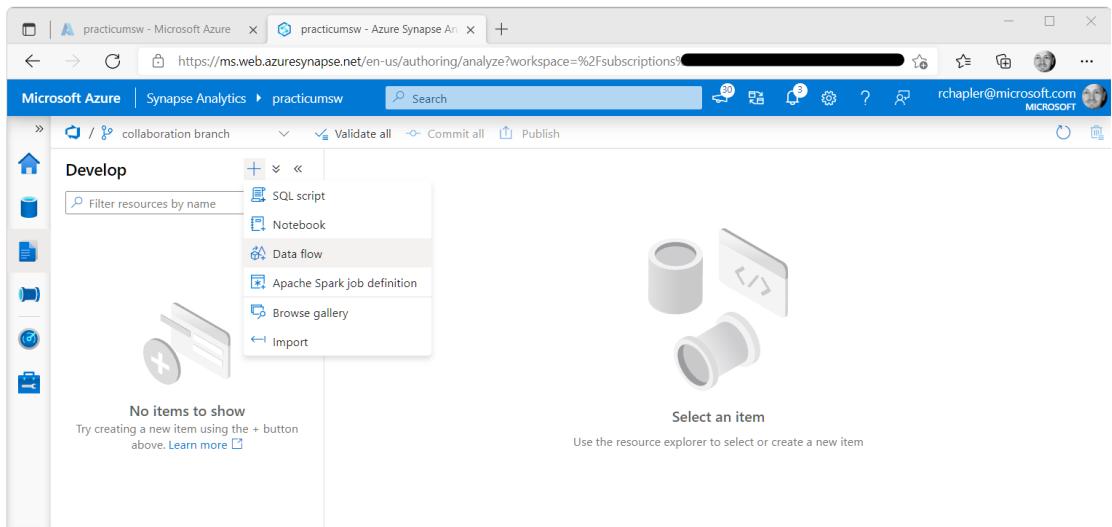
This solution requires the following resources:

- Data Lake
- SQL (with sample database)
- Synapse (with linked service for SQL and Apache Spark Pool)

Note: These instructions also apply (with minor differences) to Azure Data Factory.

Initial Load

This section describes how to load existing source data into Delta Lake (as a primer for Incremental Load).



Navigate to your Synapse Analytics workspace and then click the **Open** link on the “**Open Synapse Studio**” rectangle.

Create Data Flow

Click the **Develop** icon in the navigation pane.

Click the **+** icon just above and to the right of the “**Filter resources by name**” input.

Select “**Data flow**” from the resulting drop-down menu.

The screenshot shows the Microsoft Azure Synapse Analytics Data Flow blade. On the left, the navigation pane is open with 'Develop' selected, showing a list of 'Data flows'. One item, 'Initial Load', is currently selected and highlighted in grey. The main workspace displays a single step named 'Initial Load' with a status of '●'. Below this step, there is a dashed rectangle labeled 'Add Source'. At the top of the workspace, there are buttons for 'Commit', 'Validate', and 'Data flow debug' (which is turned on). To the right of the workspace, there is a 'Properties' panel. The 'General' tab is selected, showing the name 'Initial Load' and a description field. There are also tabs for 'Related', 'Parameters', and 'Settings'.

Complete the **Properties** pop-out form.

Activate “**Data flow debug**” to avoid wait time later in the process.

Source (sample database)

Click on “**Add Source**” in the dashed rectangle.

The screenshot shows the Microsoft Azure Synapse Analytics Data Flow blade with the 'practicumsd' source configuration open. The source is defined as an 'Integration dataset' linked to the 'practicumsd' linked service. The 'Source settings' tab is active, showing the output stream name 'practicumsd', source type 'Integration dataset', inline dataset type 'Azure SQL Database', and a successful test connection. Other tabs available include 'Source options', 'Projection', 'Optimize', 'Inspect', 'Data preview', and 'Description'.

Complete the “**Source settings**” tab, including:

Source Type	Select Inline
Inline Dataset Type	Select “ Azure SQL Database ”
Linked Service	Select <UseCase>sd
Sampling	Confirm default setting, Disable

Click “**Test connection**” to confirm successful configuration.

The screenshot shows the Microsoft Azure Synapse Analytics Data Flow blade. On the left, there's a navigation pane with 'Develop' selected. In the center, a pipeline named 'Initial Load' is shown, containing a single data flow step with 'practicumsd' as the source. The 'Source settings' tab is active at the bottom. Under 'Input', the 'Table' radio button is selected, and the 'Schema name' dropdown is set to 'SalesLT'. The 'Table name' dropdown is set to 'Product'. The 'Isolation level' dropdown is set to 'Read uncommitted'. There are tabs for 'Source options', 'Projection', 'Optimize', 'Inspect', 'Data preview', and 'Description'.

Complete the “**Source options**” tab, including:

Input	Confirm default setting, Table
Schema Name	Enter SalesLT
Table Name	Enter Product

The screenshot shows the Microsoft Azure Synapse Analytics Dataflow interface. On the left, the 'Develop' sidebar is open, showing a 'Data flows' section with one item named 'Initial Load'. The main area displays the 'Initial Load' data flow. The 'Projection' tab is selected, showing a schema with three columns: ProductID (integer), Name (string), and ProductNumber (string). The schema is defined as follows:

Column name	Type
ProductID	integer
Name	string
ProductNumber	string

On the **Projection** tab, click “**Import schema**” and then **Import** on the resulting pop-out.

The screenshot shows the Microsoft Azure Synapse Analytics Dataflow interface. The 'Data preview' tab is selected. It displays a table with 295 rows of data from the 'practicumsd' table. The columns are ProductID, Name, ProductNumber, and Color. The data includes rows such as ProductID 123 (Name: HL Road Frame - Black, 58, ProductNumber: FR-R92B-58, Color: Black), ProductID 680 (Name: Sport-100 Helmet, Red, ProductNumber: HL-U509-R, Color: Red), and ProductID 707 (Name: Sport-100 Helmet, Black, ProductNumber: HL-U509, Color: Black).

ProductID	Name	ProductNumber	Color
123	HL Road Frame - Black, 58	FR-R92B-58	Black
680	Sport-100 Helmet, Red	HL-U509-R	Red
707	Sport-100 Helmet, Black	HL-U509	Black
708	Mountain Bike Frame, M	CO-8000-M	Black

Navigate to the “**Data preview**” tab and then click **Refresh**.

Click the **Commit** button.

Sink (delta lake)

The screenshot shows the Microsoft Azure Synapse Analytics Dataflow interface. A 'Data flows' item is selected in the left sidebar. In the main area, there is a 'practicumsd' dataset (Data Lake source) connected to a 'practicumndl' dataset (Sink). The 'Sink' tab is active. The 'Destination' section shows the sink type as 'Sink'. Below it, a table displays data from the source dataset, with columns: ProductID, Name, ProductNumber, and Color. The table shows four rows of data: ProductID 680 (Name: HL Road Frame - Black, 58, ProductNumber: FR-R92B-58, Color: Black); ProductID 707 (Name: Sport-100 Helmet, Red, ProductNumber: HL-U509-R, Color: Red); ProductID 708 (Name: Sport-100 Helmet, Black, ProductNumber: HL-U509, Color: Black); and ProductID 709 (Name: Mountain Bike, Color: N/A, ProductNumber: N/A, Color: N/A).

Click the + in the bottom right of the Data Lake source, and then search for and select **Sink** from the resulting pop-up list.

The screenshot shows the Microsoft Azure Synapse Analytics Dataflow interface with the 'Sink' tab selected. The configuration includes:

- Output stream name:** practicumndl
- Incoming stream:** practicumsd
- Sink type:** Inline (selected)
- Inline dataset type:** Delta
- Linked service:** practicumsw-WorkspaceDefaultStorage
- Options:** Allow schema drift (checked), Validate schema (unchecked)

On the **Sink** tab, including:

Incoming Stream	Confirm default, <UseCase>sd
Sink Type	Select Inline
Linked Service	Select <UseCase>sw-WorkspaceDefaultStorage
Options	Confirm that “ Allow schema drift ” is checked

The screenshot shows the Microsoft Azure Synapse Analytics Data Flow blade. A pipeline named 'Initial Load' is selected. The 'Settings' tab is active, showing the following configuration:

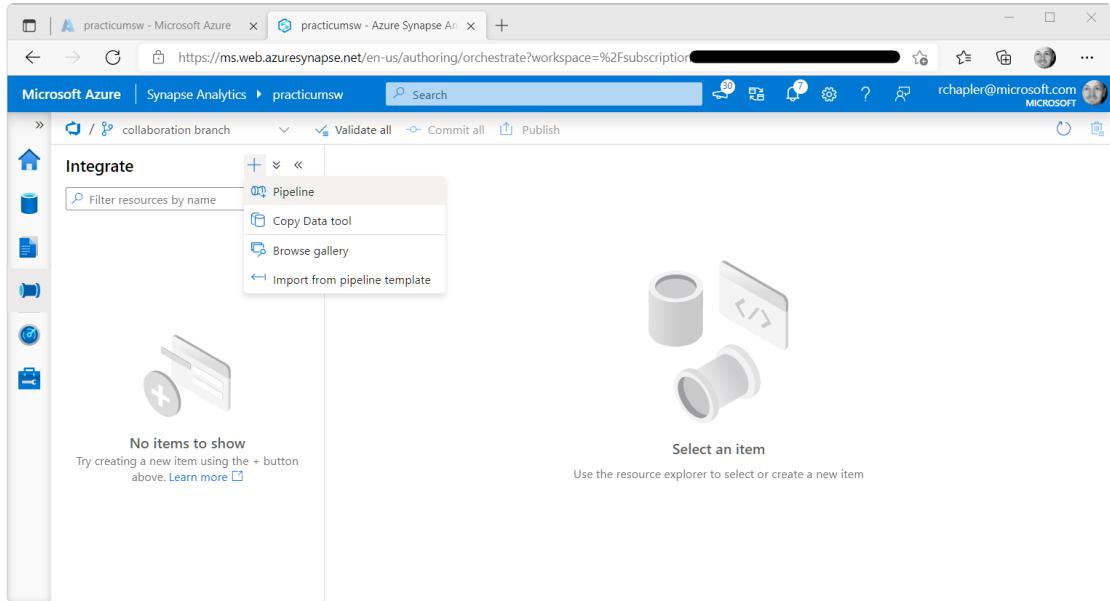
- Folder path:** practicumdlc / bronze/northwind/prc
- Compression type:** snappy
- Compression level:** Fastest
- Vacuum:** 0
- Table action:** None (radio button selected)
- Update method:**
 - Allow insert
 - Allow delete
 - Allow upsert
 - Allow update
- Delta options:**
 - Merge schema
 - Auto compact
 - Optimize write

On the **Settings** tab, including:

File System	Enter <UseCase>dlc
Folder Path	Enter “bronze/northwind/product”
Compression Type	Select snappy
Compression Level	Select Fastest
Auto Compact	Checked
Optimize Write	Checked

Click the **Commit** button.

Create Pipeline

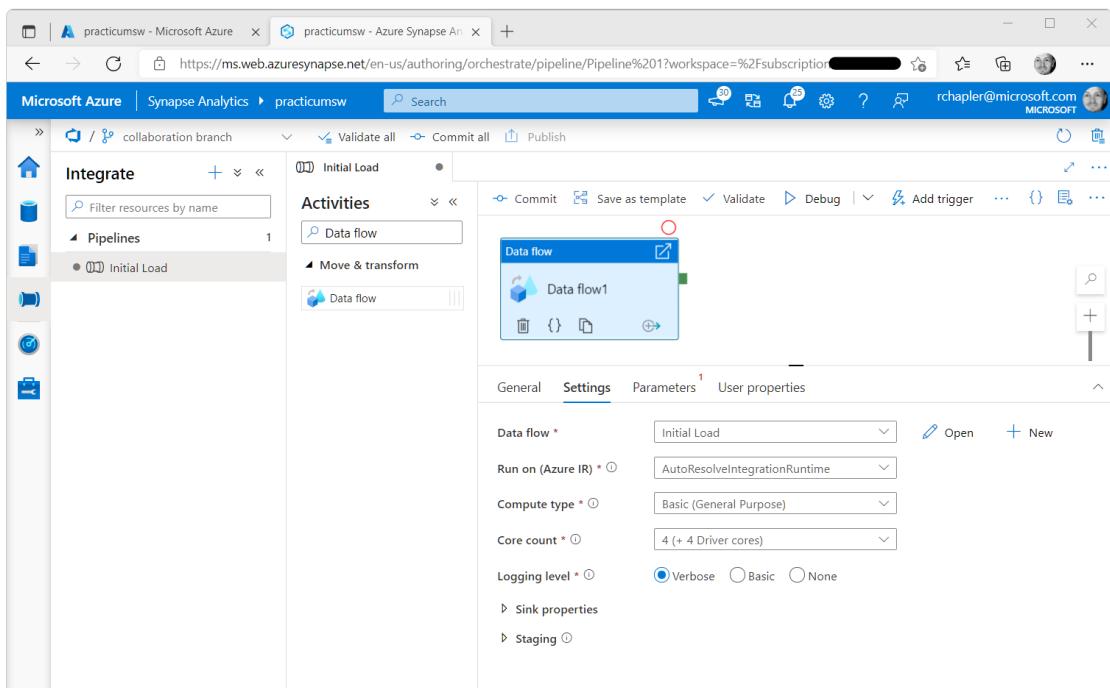


Click the **Integrate** icon in the navigation pane.

Click the + icon just above and to the right of the “**Filter resources by name**” input.

Select “**Pipeline**” from the resulting drop-down menu.

On the **Properties** pop-out, enter Name, “**Initial Load**.”



Search for, and then drag-and-drop a “**Data flow**” component into the activity window.

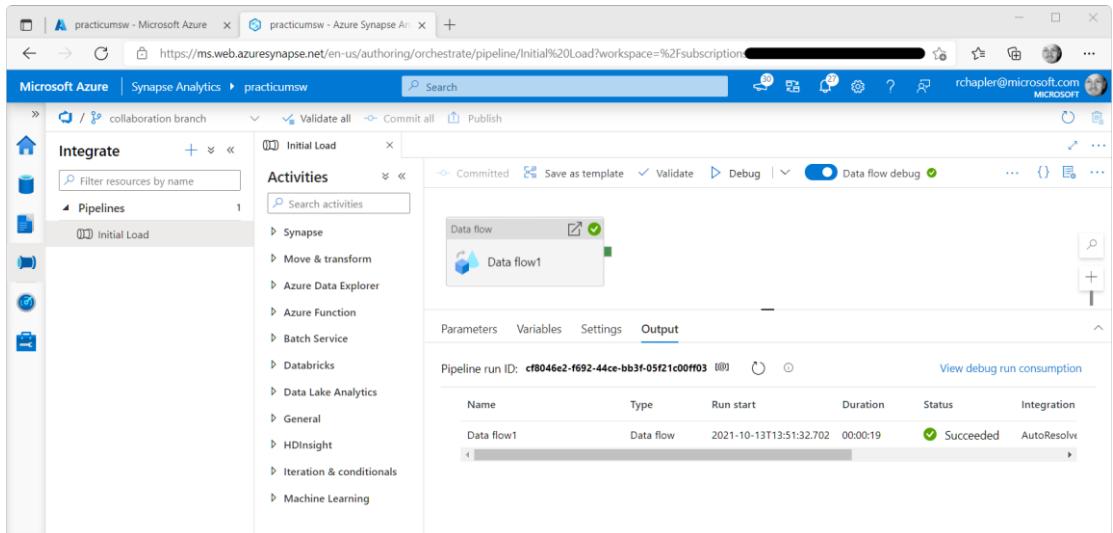
On the **Settings** tab of the “**Data flow...**” component, select your data flow.

Click the **Commit** button.

Confirm Success

Debug Pipeline / Data Flow

Click **Debug** and confirm successful execution.



Query Delta Lake

Navigate to Synapse, and then click the **Develop** icon in the navigation pane.

Click the + icon just above and to the right of the “**Filter resources by name**” input.

Select “**Notebook**” from the resulting drop-down menu.

In your new notebook, select your Apache Spark pool (if required) from the “**Attach to**” drop-down menu.

The screenshot shows the Microsoft Azure Synapse Analytics interface. In the left sidebar, under 'Develop', there are sections for 'Notebooks' (Notebook 1) and 'Data flows' (Initial Load). The main area displays a notebook titled 'Notebook 1'. The code cell contains the following PySpark code:

```
1 %%pyspark
2 df = spark.read.format('delta').load('abfss://practicumdl@practicumdl.dfs.core.windows.net/bronze/northwind/product')
3 print(df.count())
4
```

The output window shows the command was executed successfully in 3 minutes and 7 seconds, involving 87 ms, and was run by rchapler at 6:57:33 AM, 10/13/21. It also indicates 'Job execution Succeeded' with Spark 1 executors and 4 cores. There are 'View in monitoring' and 'Open Spark UI' links.

Run the following code to produce a count of records in the new Delta Lake:

```
%%pyspark
df =
spark.read.format('delta').load('abfss://<UseCase>d1c@<UseCase>d1.dfs.core.windows.net/bronze/northwind/
product')
print(df.count())
```

The screenshot shows the Microsoft Azure Synapse Analytics interface. The notebook 'Notebook 1' is displayed. The code cell contains the same PySpark code as the previous screenshot:

```
1 %%pyspark
2 df = spark.read.format('delta').load('abfss://practicumdl@practicumdl.dfs.core.windows.net/bronze/northwind/product')
3 display(df.limit(3))
4
```

The output window shows the command was executed successfully in 6 seconds, involving 906 ms, and was run by rchapler at 7:24:44 AM, 10/13/21. It indicates 'Job execution Succeeded' with Spark 1 executors and 4 cores. Below the code cell, there are tabs for 'View', 'Table', 'Chart', and 'Export results'. The 'Table' tab is selected, showing a table with the following data:

ProductID	Name	ProductNumber	Color	Sta
680	HL Road Frame - Black, 58	FR-R92B-58	Black	10!
707	Sport-100 Helmet, Red	HL-U509-R	Red	13.
708	Sport-100 Helmet, Black	HL-U509	Black	13.

Run the following code to display a few records in the new Delta Lake:

```
%%pyspark
df =
spark.read.format('delta').load('abfss://<UseCase>d1c@<UseCase>d1.dfs.core.windows.net/bronze/northwind/
product')
display(df.limit(3))
```

Incremental Load

This section describes how to provide for recurring capture of changes to source data.

Create Data Flow

Click the **Develop** icon in the navigation pane.

Click the + icon just above and to the right of the “**Filter resources by name**” input.

Select “**Data flow**” from the resulting drop-down menu.

Name your data flow “**Incremental Load**”

Source (delta lake)

In this step, we will add the Delta Lake (created in the previous section) as a source.

Click on “**Add Source**” in the dashed rectangle.

The screenshot shows the Microsoft Azure portal interface for Synapse Analytics. The left sidebar is titled 'Develop' and lists 'Notebooks' (1), 'Data flows' (2), and 'Initial Load'. The 'Data flows' item is expanded, showing 'Incremental Load' which is selected. The main workspace shows a data flow named 'Incremental Load'. The 'Source settings' tab is active, displaying the following configuration:

- Output stream name:** practicumdl
- Source type:** Integration dataset (selected)
- Inline dataset type:** Delta
- Linked service:** practicumsw-WorkspaceDefaultStorage (selected)
- Sampling:** Disable (selected)

Complete the “**Source settings**” tab, including:

Source Type	Select Inline
Inline Dataset Type	Select Delta
Linked Service	Select <UseCase>sw-WorkspaceDefaultStorage
Sampling	Confirm default setting, Disable

The screenshot shows the Microsoft Azure Synapse Analytics Dataflow blade. In the left sidebar, under 'Develop', there are sections for 'Notebooks' (1 item), 'Data flows' (2 items), and 'Data flows' sub-sections: 'Incremental Load' (selected) and 'Initial Load'. The main area displays an 'Incremental Load' configuration. At the top, there are buttons for 'Validate all', 'Commit all', 'Publish', 'Commit' (highlighted in blue), 'Validate' (highlighted in green), 'Data flow debug' (switched on), and 'Debug Settings'. Below this is a preview pane showing a single item named 'practicumdlc' with '0 total' columns. The 'Source options' tab is selected, showing the following configuration:

- Folder path**: practicumdlc / bronze/northwind/prc (with a 'Browse' button)
- Allow no files found**: Checked
- Compression type**: snappy
- Compression level**: Fastest
- Time travel**: Disable (radio button selected)

Complete the “**Source options**” tab, including:

File System	Enter <UseCase>dlc
Folder Path	Enter “bronze/northwind/product”
Allow No Files Found	Checked
Compression Type	Select snappy
Compression Level	Select Fastest
Time Travel	Confirm default, Disable

The screenshot shows the 'Import schema' dialog box from the Microsoft Azure Synapse Analytics interface. On the left, there's a sidebar with 'Develop' selected, showing 'Notebooks' (1), 'Data flows' (2), and 'Incremental Load'. The main area displays a dataset named 'practicumdl' with '0 total' columns. Below the dataset, there are sections for 'Source settings' and 'Source options'. To the right, the 'Import schema' configuration is shown, with fields for Date, Time, Numerical whole number, Numerical fraction, Boolean true, and Boolean false. At the bottom are 'Import' and 'Cancel' buttons.

On the **Projection** tab, click “**Import schema**” and then **Import** on the resulting pop-out.

Navigate to the “**Data preview**” tab and then click **Refresh**.

The screenshot shows the 'Data preview' tab of the Microsoft Azure Synapse Analytics interface. The top navigation bar includes 'Validate all', 'Commit all', 'Publish', 'Commit', 'Validate', 'Data flow debug' (which is turned on), and 'Debug Settings'. The main area shows the 'practicumdl' dataset with 17 total columns. Below the dataset, there are tabs for 'Source settings', 'Source options', 'Projection', 'Optimize', 'Inspect', and 'Data preview'. The 'Data preview' tab is active, showing a table with the following data:

ProductID	Name	ProductNumber	Color
680	HL Road Frame - Black, 58	FR-R92B-58	Black
707	Sport-100 Helmet, Red	HL-U509-R	Red
708	Sport-100 Helmet, Black	HL-U509	Black

Click the **Commit** button.

Source (sample database for this exercise)

In this step, we will add the SQL Northwind sample database (our primary source for this exercise).

Click on “**Add Source**” in the dashed rectangle.

The screenshot shows the Microsoft Azure Synapse Analytics Data Flow blade. A pipeline named "Incremental Load" is displayed. The first stage is "practicumdl" (Add source dataset), and the second stage is "practicumsd" (Integration dataset with 0 total columns). To the right of the stages, there is a dashed rectangle containing a plus sign, which is the "Add Source" button. The "Source settings" tab is selected, showing the following configuration:

- Output stream name: practicumsd
- Source type: Integration dataset (selected)
- Inline dataset type: Azure SQL Database
- Linked service: practicumsd (selected, with a green checkmark indicating a successful connection)
- Sampling: Disable (selected)

Complete the “**Source settings**” tab, including:

Source Type	Select Inline
Inline Dataset Type	Select “ Azure SQL Database ”
Linked Service	Select <UseCase>sd
Sampling	Confirm default setting, Disable

Click “**Test connection**” to confirm successful configuration.

The screenshot shows the Microsoft Azure Synapse Analytics Dataflow blade. On the left, there's a navigation pane with 'Develop' selected, showing 'Incremental Load' and 'Initial Load' under 'Data flows'. The main area displays a data flow diagram titled 'Incremental Load'. It consists of two main components: 'practicumdl' (Add source dataset) and 'practicumsd' (Add sink dataset). Below the diagram, the 'Source options' tab is selected, showing the following configuration:

Input	Confirm default setting, Table
Schema Name *	SalesLT
Table name *	Product
Isolation level	Read uncommitted

Complete the “**Source options**” tab, including:

Input	Confirm default setting, Table
Schema Name	Enter SalesLT
Table Name	Enter Product

The screenshot shows the Microsoft Azure Synapse Analytics Data Flow blade. On the left, there's a navigation pane with 'Develop' selected. In the center, an 'Incremental Load' pipeline is displayed. The pipeline consists of two main stages: 'practicumdl' (source dataset) and 'practicumsd' (target dataset). The 'practicumsd' stage is shown with 17 columns. Below the pipeline, the 'Projection' tab is active, showing a schema definition:

Column name	Type
ProductID	integer
Name	string
ProductNumber	string
Color	string
...	...

On the **Projection** tab, click “**Import schema**” and then **Import** on the resulting pop-out.

The screenshot shows the Microsoft Azure Synapse Analytics Data Flow blade with the 'Data preview' tab active. At the top, it displays summary statistics: Number of rows (100), * UPDATE (0), × DELETE (0), * UPSERT (0), & LOOKUP (0), and TOTAL (295). Below this, there's a toolbar with Refresh, Typecast, Modify, Map drifted, Statistics, and Remove buttons. The main area shows a preview of the data in the 'practicumsd' table:

ProductID	Name	ProductNumber	Color
680	HL Road Frame - Black, 58	FR-R92B-58	Black
707	Sport-100 Helmet, Red	HL-U509-R	Red

Navigate to the “Data preview” tab and then click Refresh.

Click the Commit button.

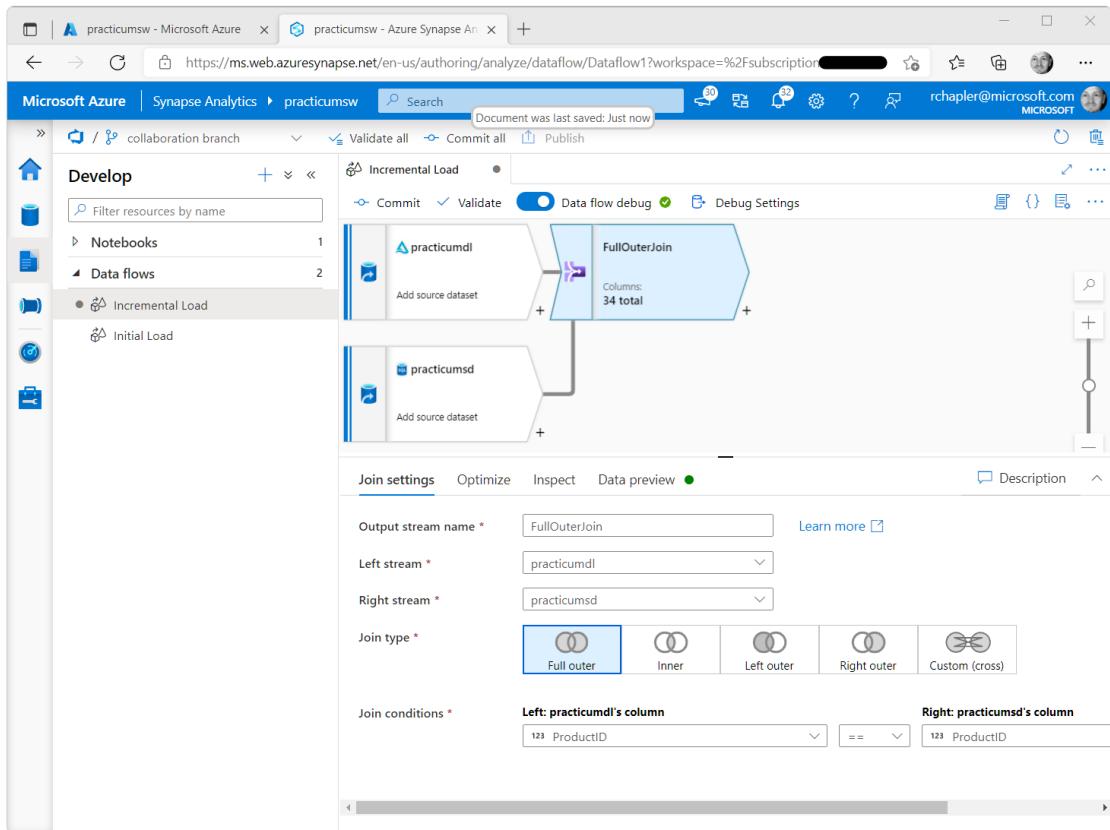
Join Sources

In this step, we will perform a full outer join between source datasets.

The screenshot shows the Microsoft Azure Synapse Analytics Data Flow blade. On the left, there's a navigation pane with 'Develop' selected, showing 'Notebooks' (1), 'Data flows' (2), 'Incremental Load', and 'Initial Load'. The main area displays a data flow titled 'Incremental Load'. It consists of two inputs: 'practicumdl' (Columns: 17 total) and 'practicumsd' (Add source dataset). A 'join' operation is being performed between them. Below the inputs, the 'Data preview' tab is selected, showing a table with the following data:

ProductID	Name	ProductNumber	Color
680	HL Road Frame - Black, 58	FR-R92B-58	Black
707	Sport-100 Helmet, Red	HL-U509-R	Red

Click the + in the bottom right of the SQL source, and then select **Join** from the resulting pop-up list.



Complete the “Join settings” tab, including:

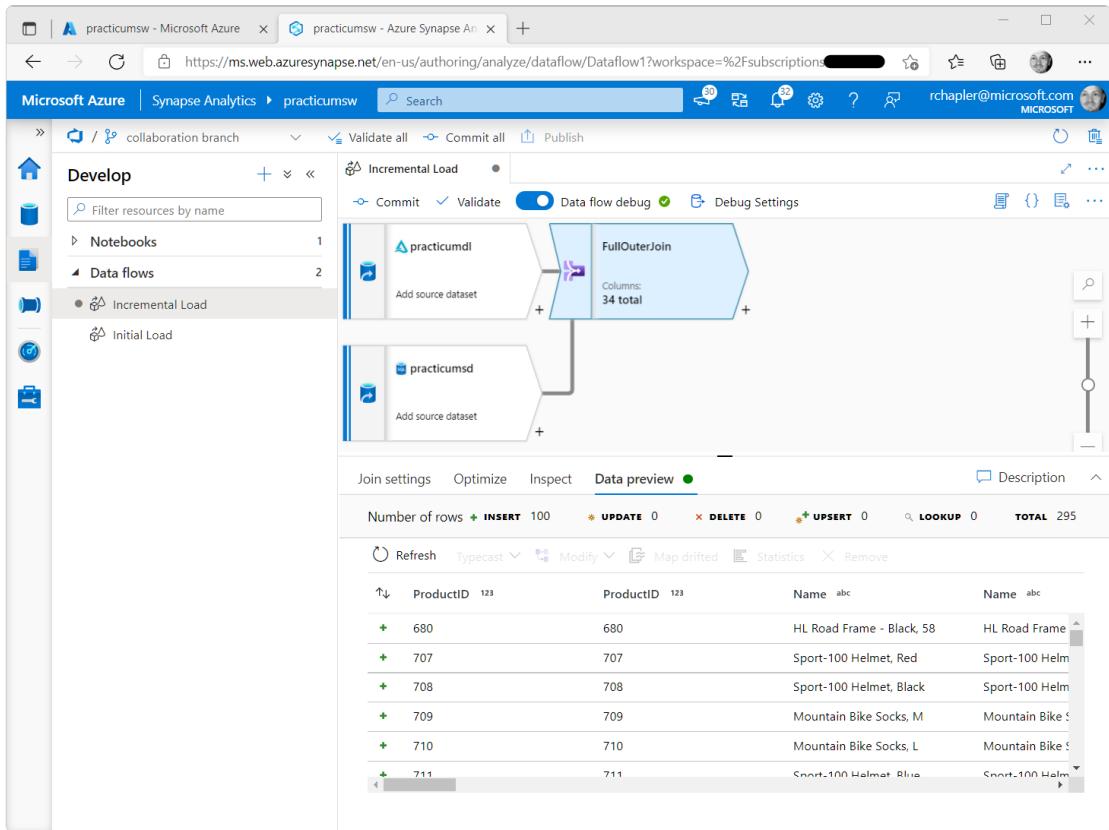
Left Stream Select <UseCase>sd

Right Stream Select AggregateWaterline

Join Type Select “Full outer”

Join Conditions Select ProductID for both Left... and Right... column

Navigate to the “Data preview” tab and then click Refresh.



Click the **Commit** button.

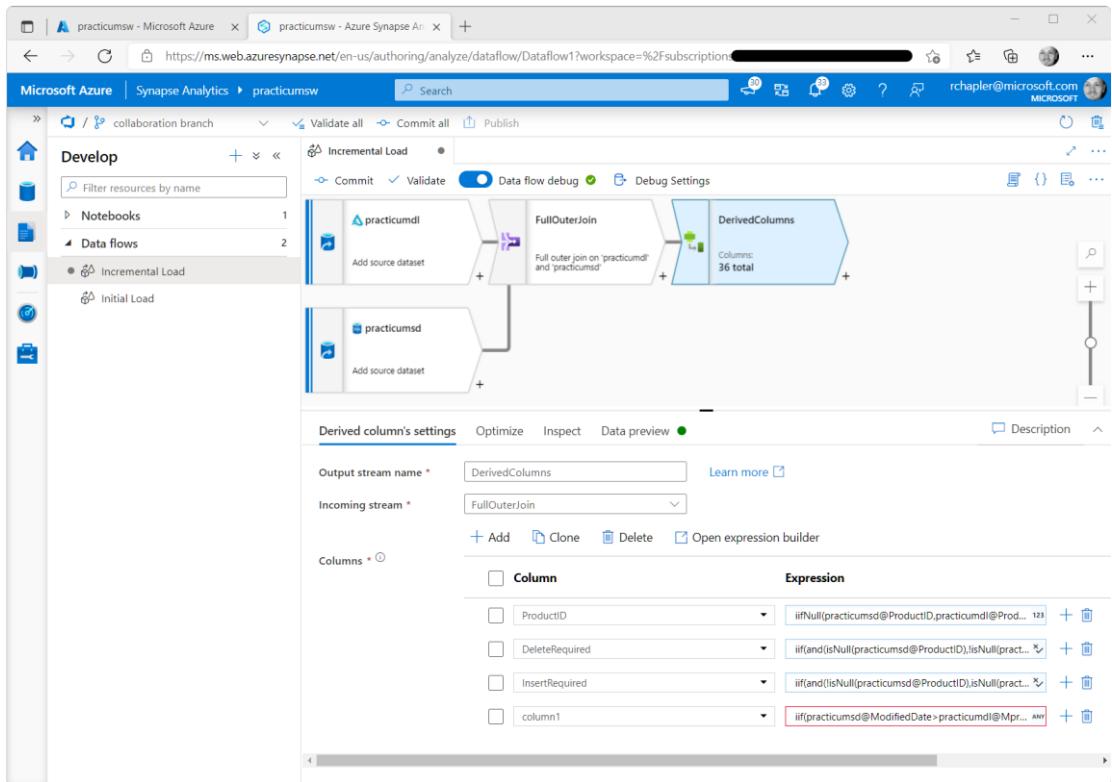
Derive Columns

In this step, we will add columns designed to characterize required Delta Lake changes.

The screenshot shows the Microsoft Azure Synapse Analytics Data Flow blade. On the left, the navigation pane shows 'Develop' selected, with 'Incremental Load' highlighted under 'Data flows'. The main area displays a data flow diagram titled 'Incremental Load'. It consists of two 'Add source dataset' components (labeled 'practicumdl' and 'practicumsd') connected to a 'FullOuterJoin' component. The 'FullOuterJoin' component has 34 total columns. A 'Schema modifier' panel is open, showing a 'Derived Column' option. Below the diagram, the 'Data preview' tab is selected, showing a table with the following data:

	ProductID	ProductID	Name	Name	ProductN
+	680	680	HL Road Frame - Black, S8	HL Road Frame - Black, S8	FR-R9,
+	707	707	Sport-100 Helmet, Red	Sport-100 Helmet, Red	HL-U5
+	708	708	Sport-100 Helmet, Black	Sport-100 Helmet, Black	HL-U5
+	709	709	Mountain Bike Socks, M	Mountain Bike Socks, M	SO-B9
+	710	710	Mountain Bike Socks, L	Mountain Bike Socks, L	SO-B9
+	711	711	Sport-100 Helmet, Blue	Sport-100 Helmet, Blue	HL-U5
+	712	712	AWC Logo Cap	AWC Logo Cap	CA-10

Click the + in the bottom right of the SQL source, and then select **Derived Column** from the resulting pop-up list.



Complete the “**Derived column’s settings**” tab, including:

Incoming Stream Confirm default selection

Columns ProductID

`iifNull(<UseCase>sd@ProductID,<UseCase>dl@ProductID)`

DeleteRequired

`iif(and(isNull(<UseCase>sd@ProductID),!isNull(<UseCase>dl@ProductID)),true(),false())`

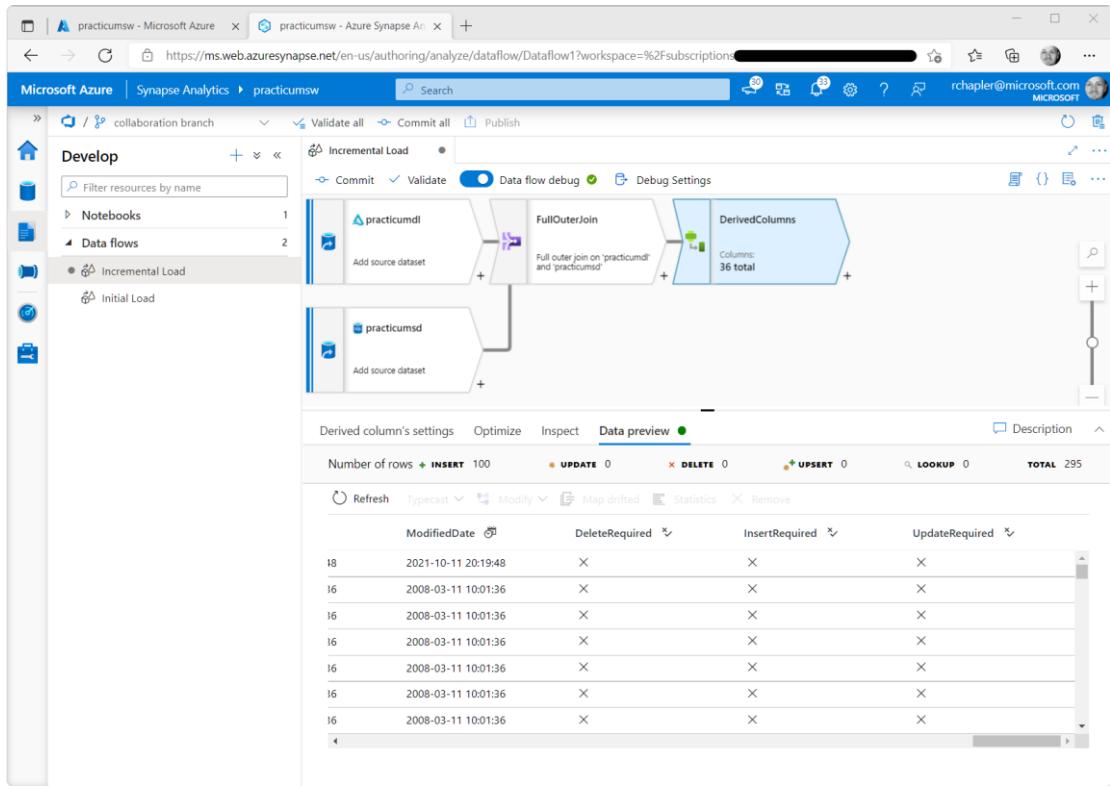
InsertRequired

`iif(and(!isNull(<UseCase>sd@ProductID),isNull(<UseCase>dl@ProductID)),true(),false())`

UpdateRequired

`iif(<UseCase>sd@ModifiedDate><UseCase>dl@ModifiedDate,true(),false())`

Navigate to the “**Data preview**” tab and then click **Refresh**.



Click the **Commit** button.

Note: I included this step to provide a straightforward way to preview and provide for the delete, insert and update of data... arguably, this step could be consolidated into Filter or AlterRows.

Filter Non-Actionable

In this step, we will remove rows that have no required action.

The screenshot shows the Microsoft Azure Synapse Analytics Data Flow interface. The pipeline is named 'Incremental Load'. It consists of two source datasets: 'practicumdl' and 'practicumsd', which are joined via a 'FullOuterJoin' operation. The output of this join is then processed by a 'DerivedColumns' operation, which creates 36 new columns. Finally, a 'Row modifier' operation is applied to the data. The 'Data preview' tab is selected, showing a sample of 100 rows from the final output. The columns include ProductID, Name, and various identifiers like FR-R92B-58, HL-U509-R, etc.

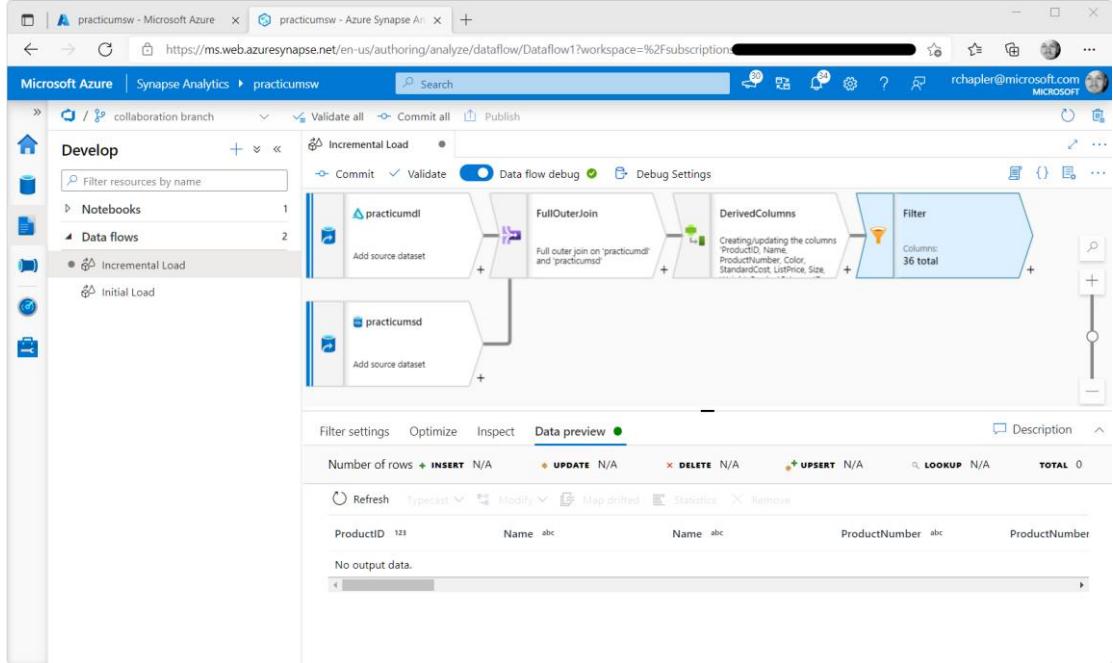
Click the **+** in the bottom right of the SQL source, and then select **Filter** from the resulting pop-up list.

The screenshot shows the same 'Incremental Load' pipeline as before, but with a 'Filter' component added after the 'DerivedColumns' operation. The 'Filter' component has a tooltip indicating it is 'Creating/Updating the columns ProductID, Name, ProductNumber, Color, StandardCost, ListPrice, Size'. The 'Filter settings' tab is selected, showing the configuration for the filter. The 'Output stream name' is set to 'Filter', the 'Incoming stream' is 'DerivedColumns', and the 'Filter on' condition is 'DeleteRequired||InsertRequired||UpdateRequired'.

Complete the “**Filter settings**” tab, including:

Incoming Stream	Confirm default selection
Filter On	DeleteRequired InsertRequired UpdateRequired

Navigate to the “**Data preview**” tab and then click **Refresh**.



Click the **Commit** button.

Note: Because we are doing an incremental load immediately after an initial load, on a sample database which doesn't have unexpected change, we see no current results in the Data Preview.

Change Examples

To make the remaining steps a bit more interesting, we will force changes to the sample database.

Navigate to your Azure SQL Database, click the “**Query editor (preview)**” link in the navigation, and login.

Delete Example

The screenshot shows the Microsoft Azure portal interface. The left sidebar is open, showing the 'Query editor (preview)' section under 'practicumsd (practicumsds/practicumsd)'. The main area is titled 'Query 1' and contains the following T-SQL code:

```
1 DELETE FROM [SalesLT].[Product] WHERE [ProductID]=730
```

Below the code, the 'Messages' tab is selected, displaying the message: 'Query succeeded: Affected rows: 1'. At the bottom of the messages area, there is a green status bar that says 'Query succeeded | 0s'.

Execute the following T-SQL:

```
DELETE FROM [SalesLT].[Product] WHERE [ProductID]=730
```

Note: I chose 730 because it did not trigger constraints; any row deletion will work.

Insert Example

The screenshot shows the Microsoft Azure portal interface. The left sidebar includes links for Overview, Activity log, Tags, Diagnose and solve problems, Quick start, and Query editor (preview). The main area displays a database named 'practicumsd' under the 'practicumsds/practicumsd' resource group. A message box indicates 'Showing limited object explorer here. For full capability please open SSDT.' The central pane contains a 'Query 1' editor with the following T-SQL code:

```
1 INSERT INTO [SalesLT].[Product]
2 ([Name],[ProductNumber],[Color],[StandardCost],[ListPrice],[SellStartDate]
3 VALUES
4 ('New Product ABC','ABC-123','Bronze',1000,1500,getdate(),newid(),getdate())
```

The 'Messages' tab shows the result: 'Query succeeded: Affected rows: 1'. A status bar at the bottom indicates 'Query succeeded | 0s'.

Execute the following T-SQL:

```
INSERT INTO [SalesLT].[Product]
([Name],[ProductNumber],[Color],[StandardCost],[ListPrice],[SellStartDate],[rowguid],[ModifiedDate])
VALUES
('New Product ABC','ABC-123','bronze',1000,1500,getdate(),newid(),getdate())
```

Update Example

The screenshot shows the Microsoft Azure portal interface. The left sidebar includes links for Overview, Activity log, Tags, Diagnose and solve problems, Quick start, and Query editor (preview). The main area displays the practicumsd database with a message about limited object explorer. The Query editor (preview) window is open, showing a query titled 'Query 1' with the following T-SQL code:

```
1 UPDATE [SalesLT].[Product]
2 SET [ListPrice] = 999.99, [ModifiedDate] = getdate()
3 WHERE ProductID=680
4
```

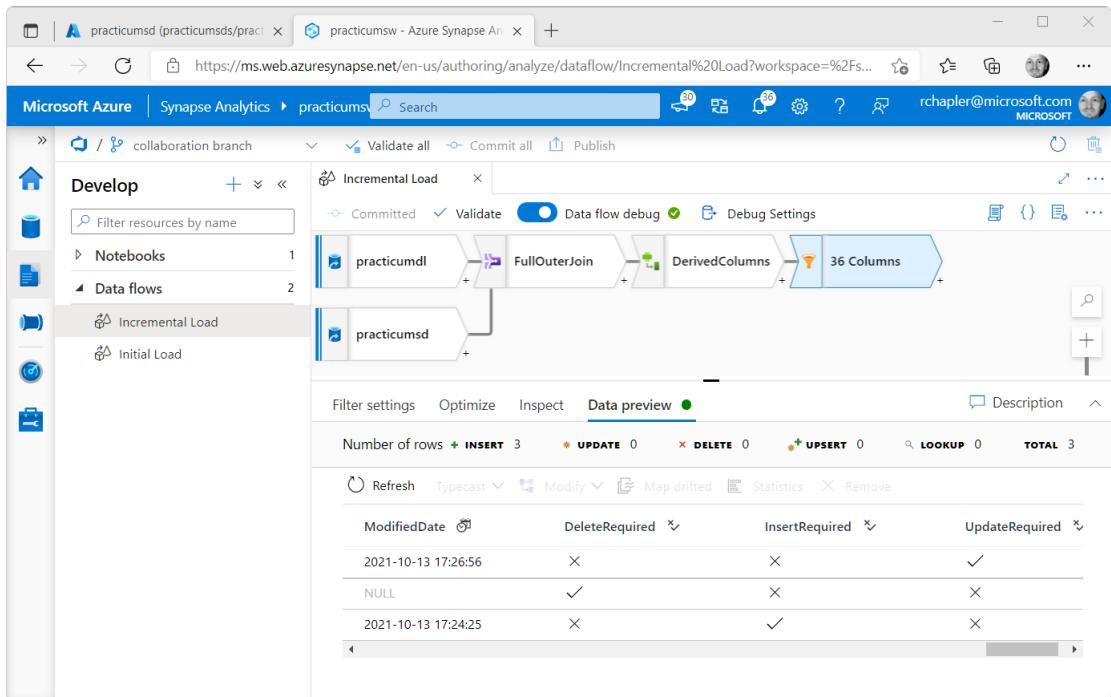
The 'Messages' tab shows the result: "Query succeeded: Affected rows: 1".

Execute the following T-SQL:

```
UPDATE [SalesLT].[Product]
SET [ListPrice] = 999.99, [ModifiedDate] = getdate()
WHERE ProductID=680
```

Confirm Changes

Return to the “**Incremental Load**” data flow and sequentially update with “**Data preview**” > **Refresh**.

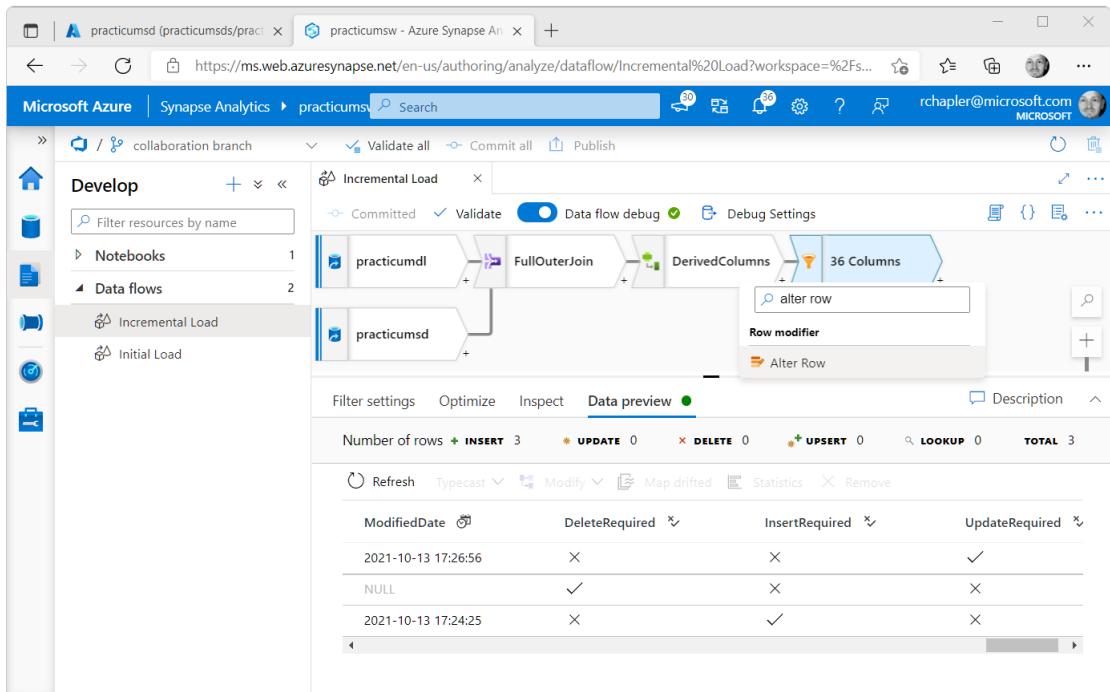


After update, you should see the three actions listed in **Filter > “Data preview.”**

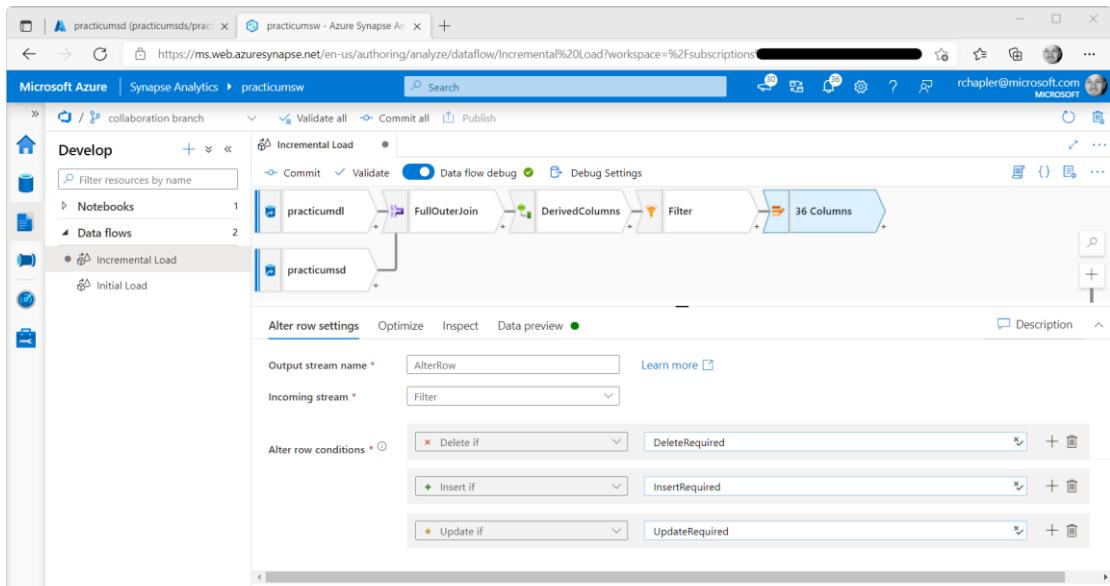
Finalize Data Flow

Alter Row

In this step, we will apply delete, insert, and update policies to the rows.



Click the + in the bottom right of the SQL source, and then select “**Alter Row**” from the resulting pop-up list.



Complete the “Alter row settings” tab, including:

Incoming Stream Confirm default selection

Alter Row Conditions “Delete if”
DeleteRequired

“Insert if”
InsertRequired

“Update if”
UpdateRequired

Navigate to the “Data preview” tab and then click Refresh.

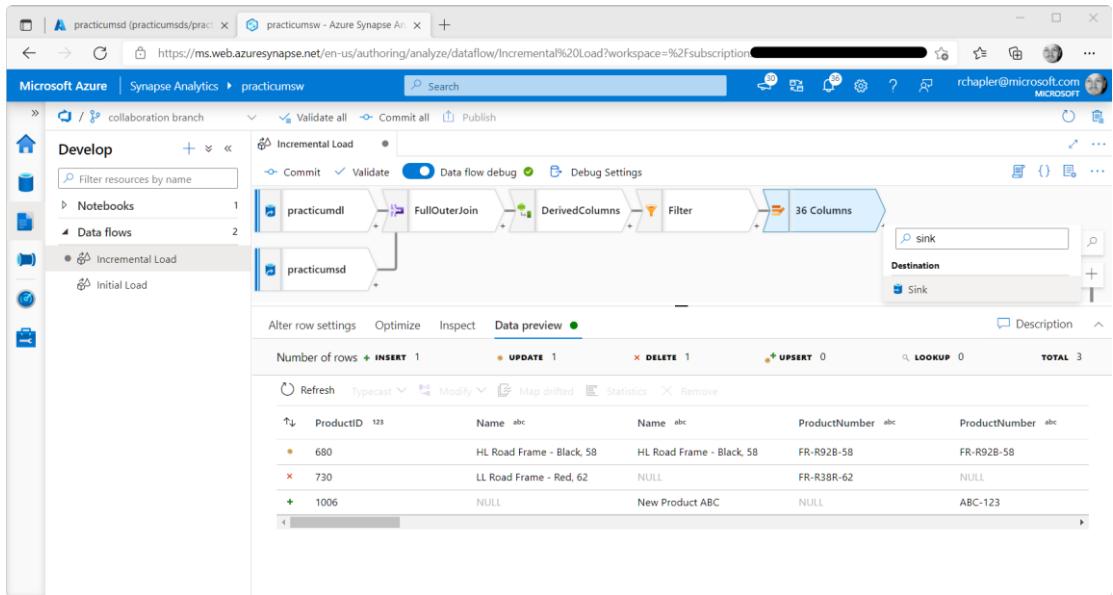
The screenshot shows the Azure Synapse Analytics Data Flow blade. On the left, there's a navigation pane with 'Develop' selected, showing 'Incremental Load' as the active data flow. The main area displays a data flow diagram with various components like 'FullOuterJoin', 'DerivedColumns', and 'Filter'. Below the diagram, the 'Data preview' tab is selected. It shows a table with the following data:

	ProductID	Name	ProductNumber
*	680	HL Road Frame - Black, 58	FR-R92B-58
*	730	LL Road Frame - Red, 62	FR-R38R-62
+	1006	NULL	ABC-123

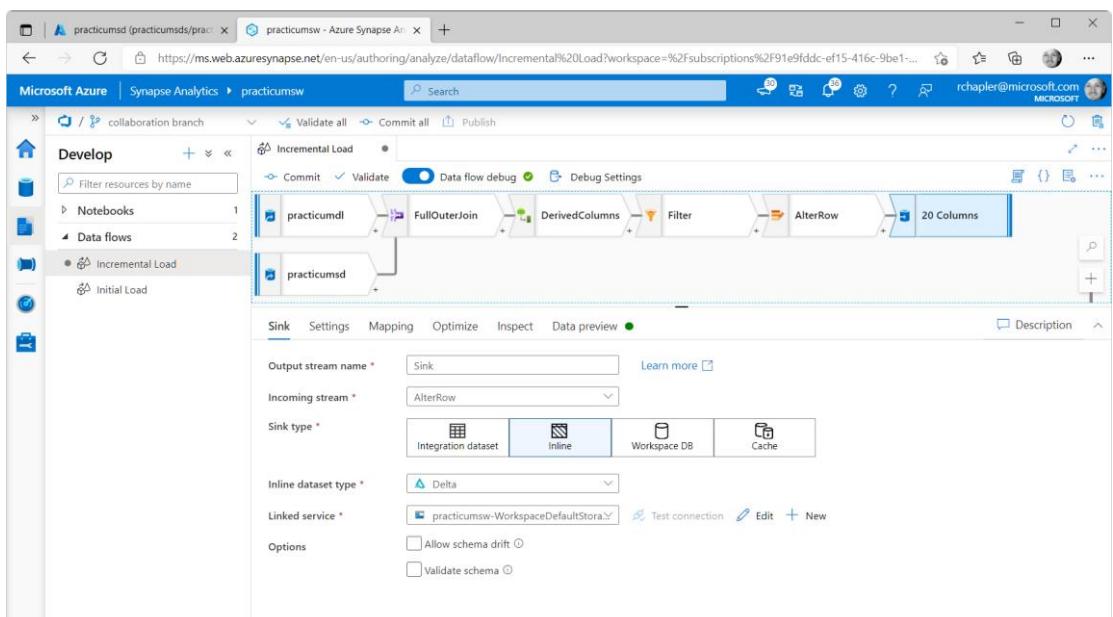
Click the **Commit** button.

Sink (delta lake)

In this step, we finalize the data flow.



Click the + in the bottom right of the SQL source, and then select **Sink** from the resulting pop-up list.

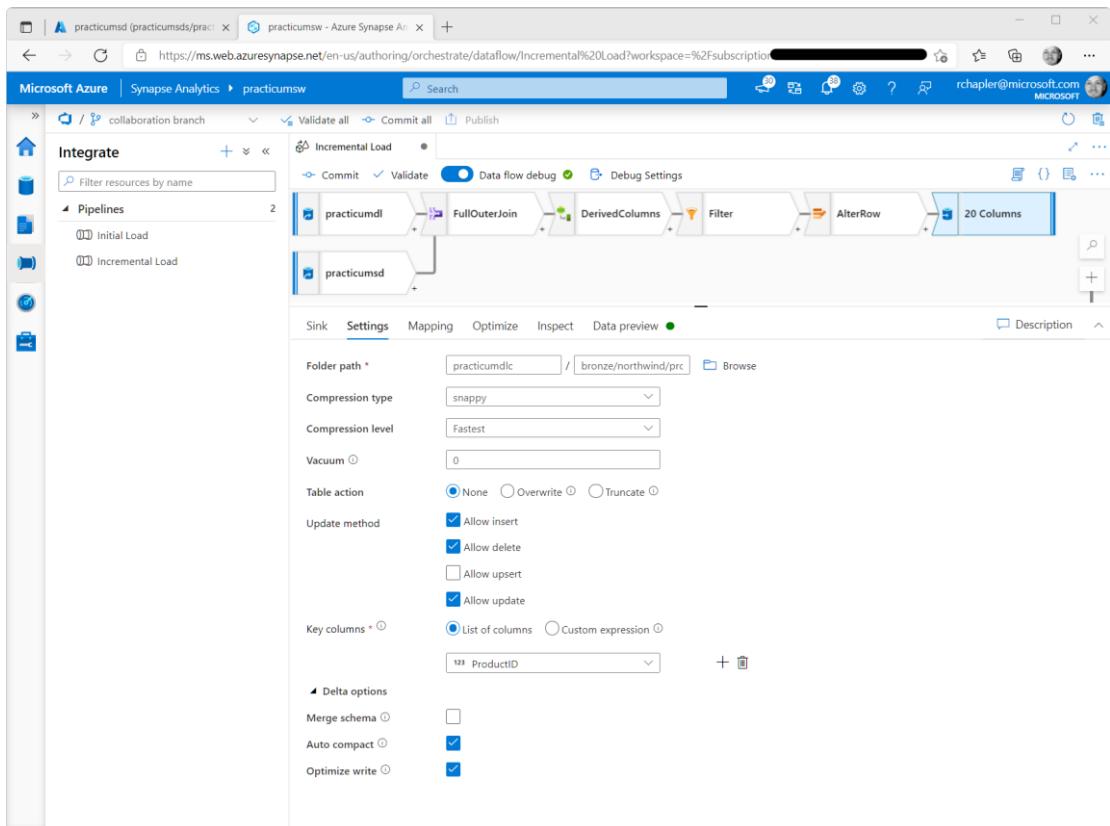


On the **Sink** tab, including:

Sink Type Select **Inline**

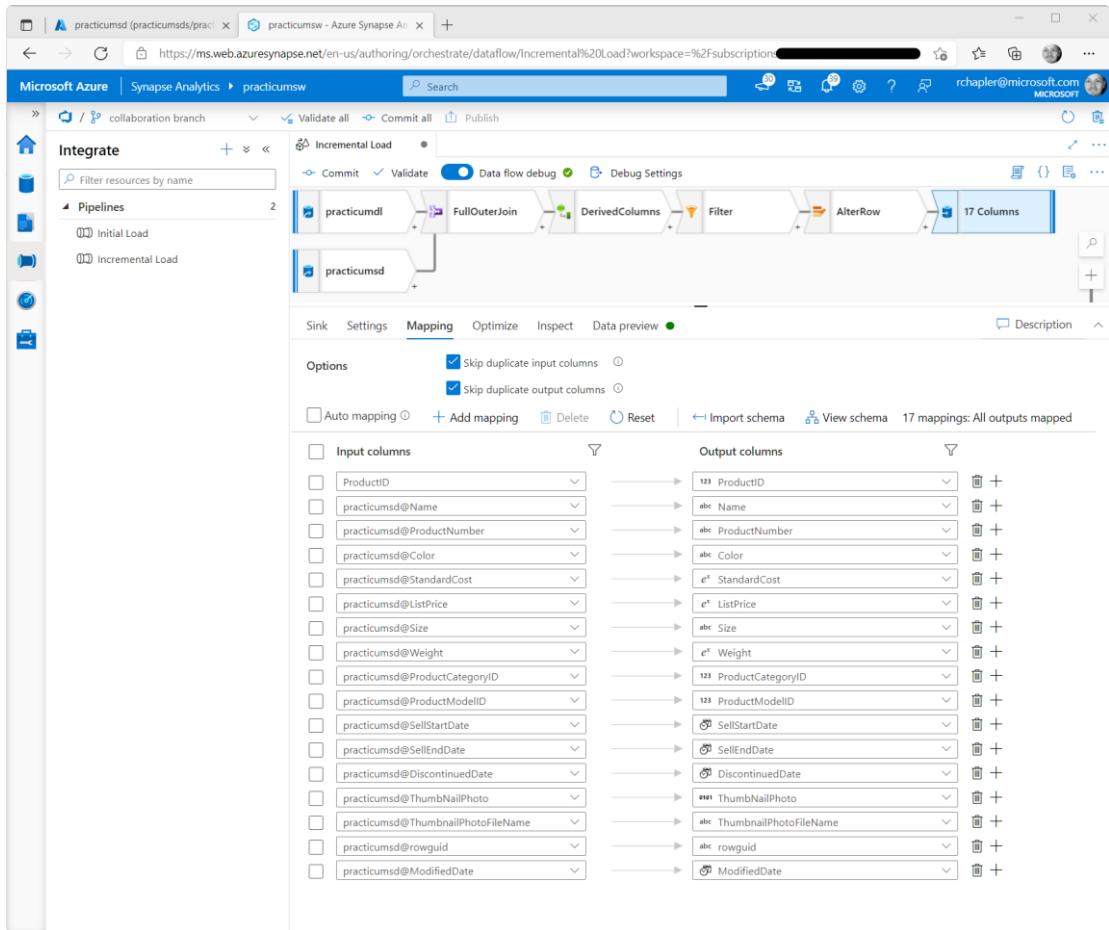
Inline Dataset Type Select **Delta**

Linked Service Select <UseCase>sw-WorkspaceDefaultStorage



On the **Settings** tab, including:

File System	Enter <UseCase>dlc
Folder Path	Enter “bronze/northwind/product”
Compression Type	Select snappy
Compression Level	Select Fastest
Update Method	“Allow insert”, “Allow delete”, and “Allow update” checked
Key Columns	Confirm default selection, “ List of columns ” and select ProductID
Auto Compact	Checked
Optimize Write	Checked

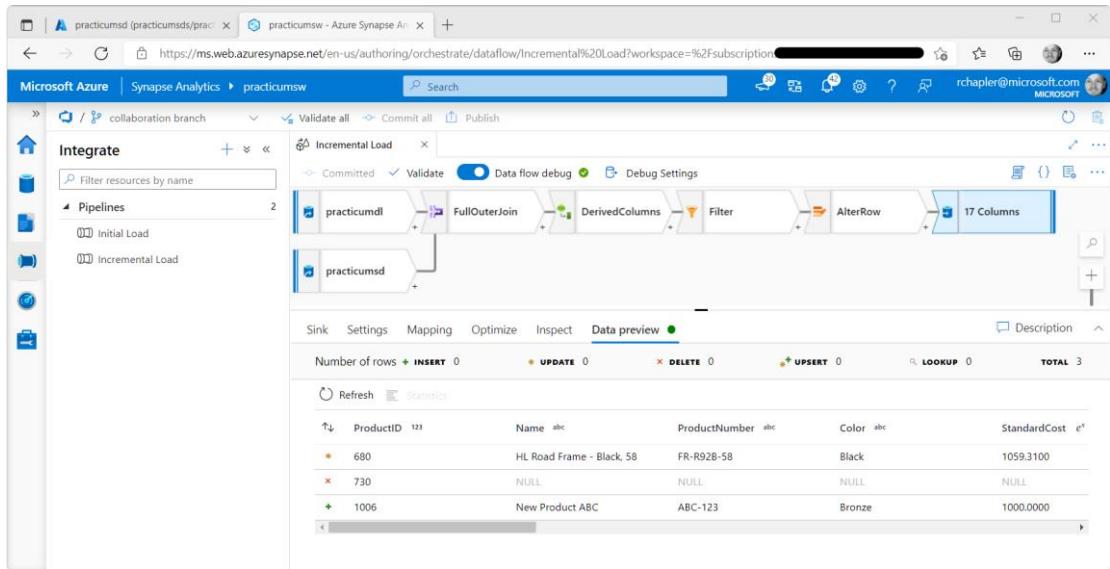


On the **Mapping** tab, un-check “Auto mapping”, click “**Import schema**” and then **Import** on the resulting pop-out.

Review the auto-generated list of columns and remove:

- Rows where “**Input columns**” begin with “<UseCase>dl...” (need data changes from the source, not the waterline dataset)
- Derived Columns DeleteRequired, InsertRequired, and UpdateRequired

Navigate to the “**Data preview**” tab and then click **Refresh**.



Click the **Commit** button.

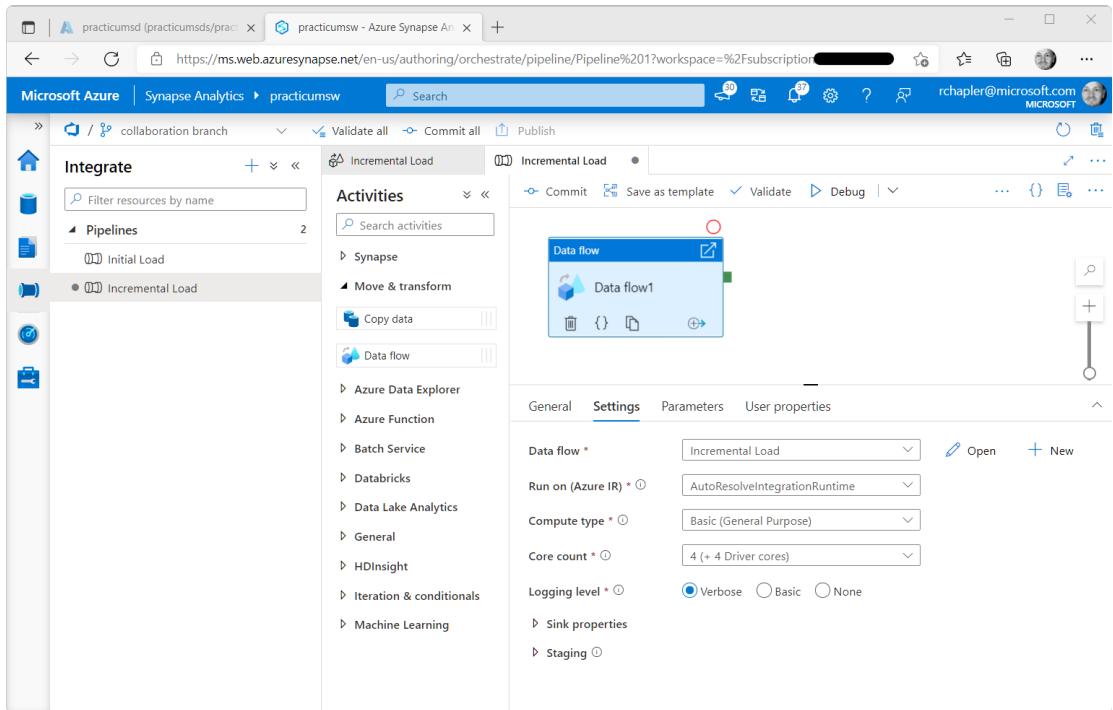
Create Pipeline

Click the **Integrate** icon in the navigation pane.

Click the **+** icon just above and to the right of the “**Filter resources by name**” input.

Select “**Pipeline**” from the resulting drop-down menu.

On the **Properties** pop-out, enter Name, “**Incremental Load**.”



Search for, and then drag-and-drop a “**Data flow**” component into the activity window.

On the **Settings** tab of the “**Data flow...**” component, select your data flow.

Click the **Commit** button.

Confirm Success

Debug Pipeline / Data Flow

Click **Debug** and confirm successful execution.

The screenshot shows the Microsoft Azure Synapse Analytics workspace for the 'practicumsw' workspace. In the left navigation pane, under the 'Integrate' section, 'Pipelines' is selected, showing 'Initial Load' and 'Incremental Load'. The main area displays an 'Incremental Load' pipeline named 'Incremental Load'. The pipeline details page shows a single Data flow named 'Data flow1'. Below the pipeline details, a table lists the run history for this pipeline.

Name	Type	Run start	Duration	Status	Integ
Data flow1	Data flow	2021-10-13T20:30:16.012	00:00:34	Succeeded	Autof

Review Produced Files

Click the **Data** icon in the navigation, and then the **Linked** tab.

Expand “Azure Data Lake Storage Gen2”, then <UseCase>sw to <UseCase>dlc.

Click to navigate to “<UseCase>dlc > bronze > northwind > product”

The screenshot shows the Microsoft Azure Synapse Analytics workspace for the 'practicumsw' workspace. In the left navigation pane, under the 'Data' section, 'Linked' is selected. The main area shows the file structure of the Azure Data Lake Storage Gen2 account 'practicumdlc'. The path 'practicumdlc > bronze > northwind > product' is selected. A table below lists the files in the 'product' folder.

Name	Last Modified	Content Type	Size
_delta_log	10/13/2021, 6:51:37 AM	Folder	
part-00000-6d4ac397-beee-4b62-9dfc-33458fb11fd6-c000.snappy.parquet	10/13/2021, 6:51:38 AM		147.3 KB
part-00000-82bf94ed-432a-4b12-958e-92309788c2b7-c000.snappy.parquet	10/13/2021, 1:30:25 PM		14.2 KB
part-00001-28ac0035-b7d0-48d7-b5ae-2cfb0e133fb-c000.snappy.parquet	10/13/2021, 1:30:25 PM		13.9 KB
part-00003-55c643f4-1b5b-4b1d-a69b-f7f57e6f843-c000.snappy.parquet	10/13/2021, 1:30:25 PM		7.5 KB
part-00004-3817f4e7-f32f-45b9-a797-5f94050729e5-c000.snappy.parquet	10/13/2021, 1:30:25 PM		13.8 KB
part-00005-c61bb8c7-177b-4a4d-8b3e-0f5a9f019fc4-c000.snappy.parquet	10/13/2021, 1:30:25 PM		14.9 KB

Review the files produced by delta lake.

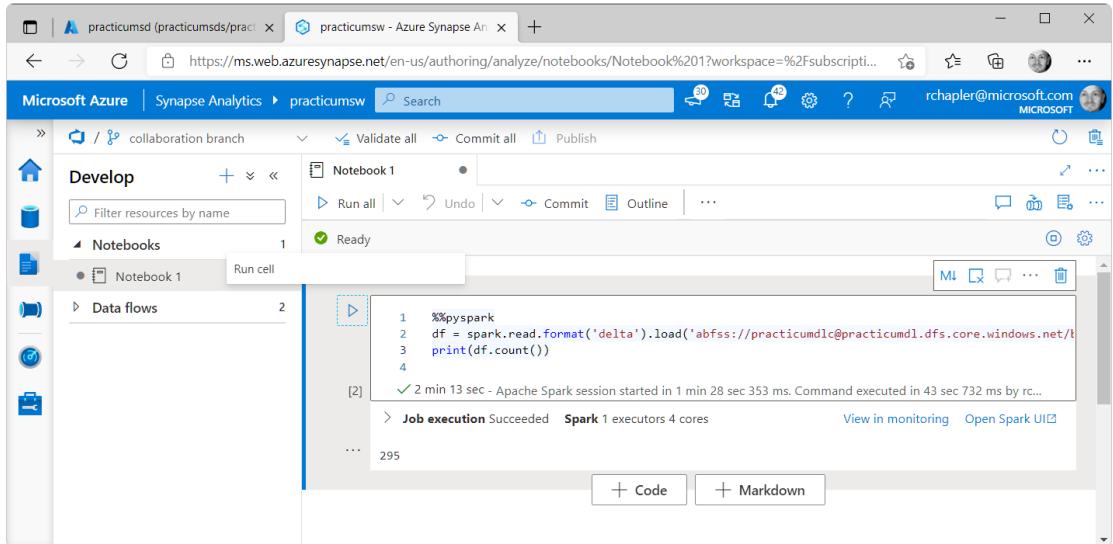
Query Record Count

Navigate to Synapse, and then click the **Develop** icon in the navigation pane.

Click the + icon just above and to the right of the “**Filter resources by name**” input.

Select “**Notebook**” from the resulting drop-down menu.

In your new notebook, select your Apache Spark pool (if required) from the “**Attach to**” drop-down menu.

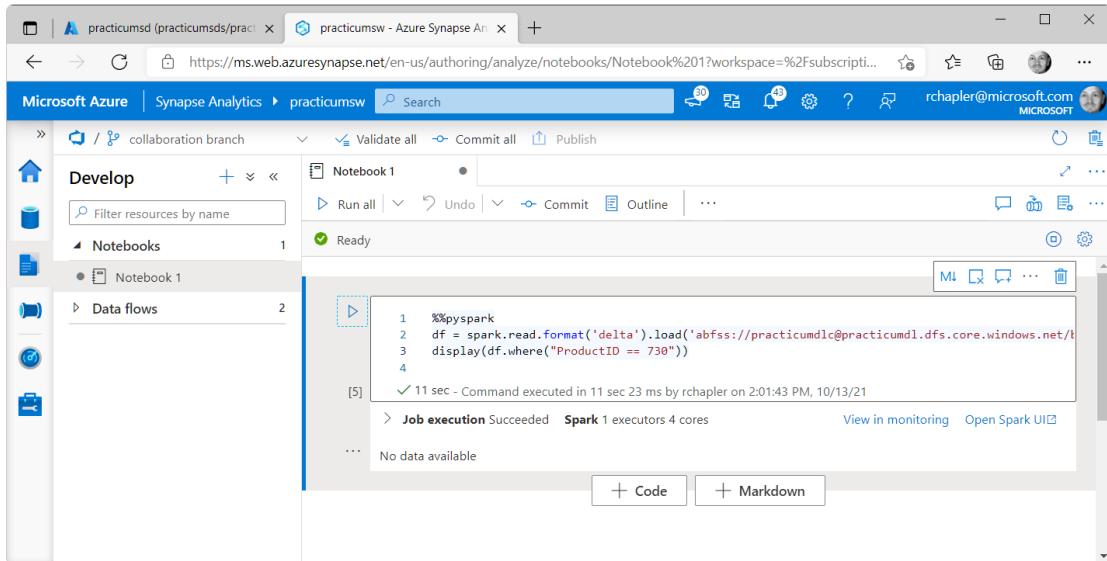


Run the following code to produce a count of records in the new Delta Lake:

```
%pyspark
df =
spark.read.format('delta').load('abfss://<UseCase>dlc@<UseCase>d1.dfs.core.windows.net/bronze/northwind/
product')
print(df.count())
```

Note: Expected value is 295... we started with 295, deleted one and inserted one.

Confirm Delete Example



The screenshot shows the Microsoft Azure Synapse Analytics interface. On the left, there's a navigation sidebar with 'Develop' selected, showing 'Notebooks' (1) and 'Data flows' (2). The main area is titled 'Notebook 1'. At the top of the notebook, there's a toolbar with 'Run all', 'Undo', 'Commit', 'Outline', and other options. Below the toolbar, a status bar says 'Ready'. A code cell contains the following Python code:

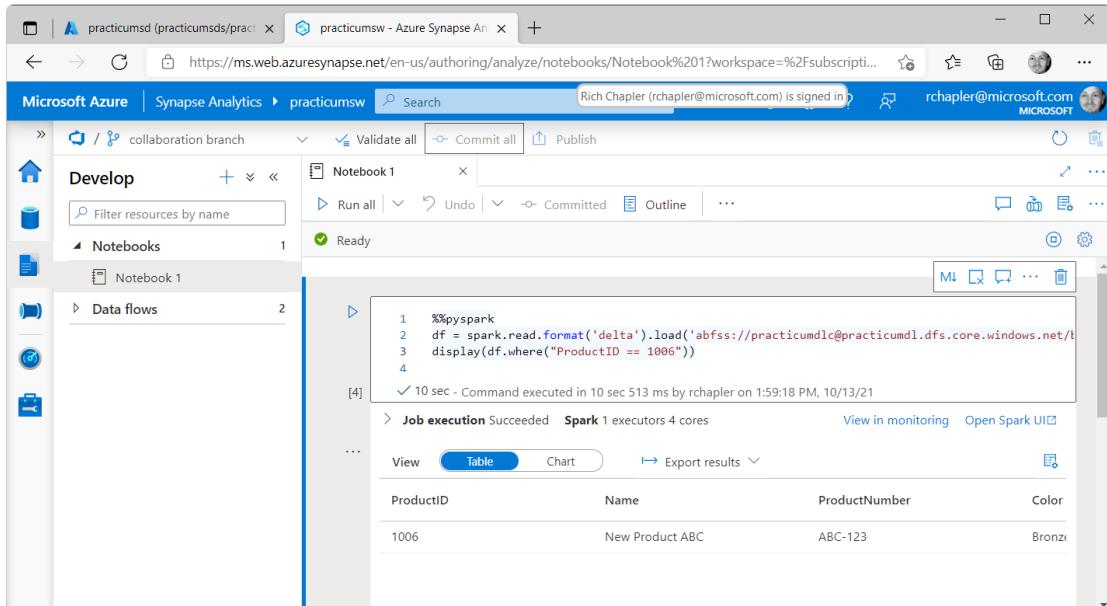
```
1 %%pyspark
2 df = spark.read.format('delta').load('abfss://practicumdlc@practicumdl.dfs.core.windows.net/northwind/bronze')
3 display(df.where("ProductID == 730"))
4
```

A message box indicates a successful execution: '✓ 11 sec - Command executed in 11 sec 23 ms by rchaper on 2:01:43 PM, 10/13/21'. Below the message, it says 'Job execution Succeeded' with 'Spark 1 executors 4 cores'. There are buttons for 'View in monitoring' and 'Open Spark UI'. At the bottom of the code cell, there are '+ Code' and '+ Markdown' buttons.

Run the following code to confirm our update example:

```
%%pyspark
df =
spark.read.format('delta').load('abfss://<UseCase>dlc@<UseCase>dl.dfs.core.windows.net/bronze/northwind/
product')
display(df.where("ProductID == 730"))
```

Confirm Insert Example



The screenshot shows the Microsoft Azure Synapse Analytics interface. The left sidebar shows 'Develop' selected with 'Notebooks' (1) and 'Data flows' (2). The main area is titled 'Notebook 1'. The toolbar includes 'Run all', 'Undo', 'Committed', 'Outline', and other options. The status bar says 'Ready'. A code cell contains the same Python code as the previous example:

```
1 %%pyspark
2 df = spark.read.format('delta').load('abfss://practicumdlc@practicumdl.dfs.core.windows.net/northwind/bronze')
3 display(df.where("ProductID == 1006"))
4
```

A message box indicates a successful execution: '✓ 10 sec - Command executed in 10 sec 513 ms by rchaper on 1:59:18 PM, 10/13/21'. Below the message, it says 'Job execution Succeeded' with 'Spark 1 executors 4 cores'. There are buttons for 'View in monitoring' and 'Open Spark UI'. The result is displayed as a table:

ProductID	Name	ProductNumber	Color
1006	New Product ABC	ABC-123	Bronze

Run the following code to confirm our update example:

```
%%pyspark
df =
spark.read.format('delta').load('abfss://<UseCase>dlc@<UseCase>d1.dfs.core.windows.net/bronze/northwind/
product')
display(df.where("ProductID == 1006"))
```

Confirm Update Example

The screenshot shows the Microsoft Azure Synapse Analytics notebook interface. On the left, there's a sidebar with 'Develop' selected, showing 'Notebooks' (1) and 'Data flows' (2). The main area is titled 'Notebook 1' and contains the following code:

```
1 %%pyspark
2 df = spark.read.format('delta').load('abfss://practicumdlc@practicumdl.dfs.core.windows.net/bronze/northwind/
product')
3 display(df.where("ProductID == 680"))
4
```

Below the code, a message indicates a successful execution: **[3]** ✓ 11 sec - Command executed in 10 sec 642 ms by rchaper on 1:54:40 PM, 10/13/21. It also shows job details: **Job execution Succeeded**, **Spark 1 executors 4 cores**. There are buttons for **View in monitoring** and **Open Spark UI**.

The results are displayed in a table:

ProductID	Name	ProductNumber	Color	StandardCost	ListPrice
680	HL Road Frame - Black, 58	FR-R92B-58	Black	1059.3100	999.9900

Run the following code to confirm our update example:

```
%%pyspark
df =
spark.read.format('delta').load('abfss://<UseCase>dlc@<UseCase>d1.dfs.core.windows.net/bronze/northwind/
product')
display(df.where("ProductID == 680"))
```

API Load

Logic Apps > Event Hub > Data Explorer

Use Case

Example notes:

- “We want to stream weather data from a publicly available API”

Prerequisites

This solution requires the following resources:

- Event Hub (with Primary Key associated with Shared Access Policy for “send”)
- Logic App (using Consumption plan)
- Data Explorer Cluster (or Synapse Data Explorer Pool) and Data Explorer Database

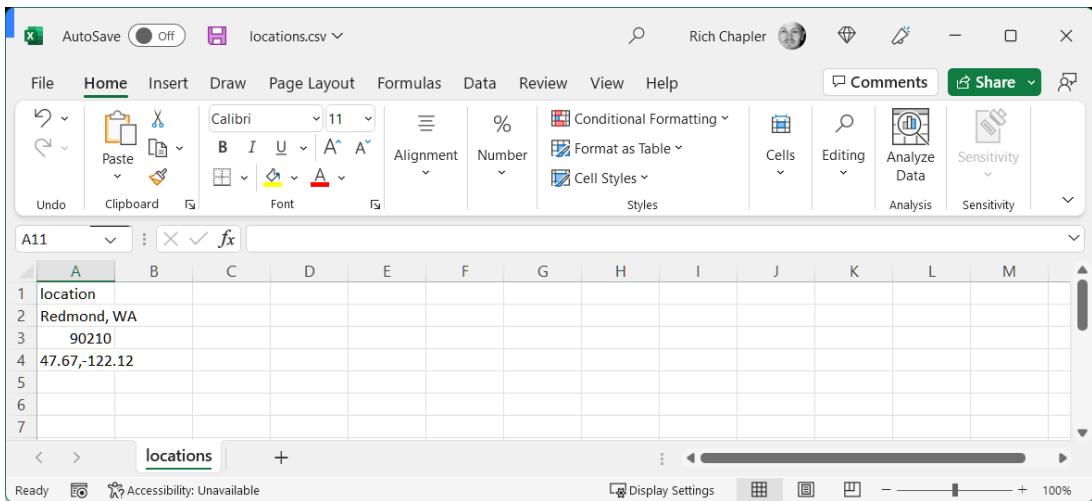
Prepare Destination

We will need two tables for this exercise:

- **Locations** ... a proxy for a table that might exist in your environment; for example: you have a “Service Calls” table that includes location information regarding service calls (for which you want to correlate weather data)
- **Weather** ... a table that will be home to regularly captured data from MSN Weather

In this section, we will prepare Locations.

Open Excel, create a blank workbook and add a few rows with location information... valid entries include City, Region, State, Country, Landmark, PostalCode and Lat.Long.



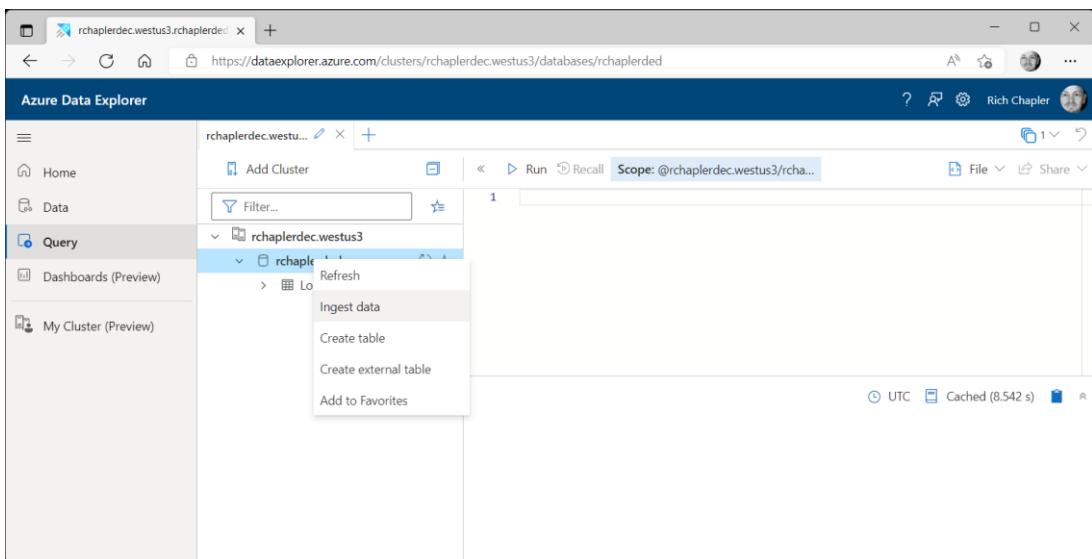
The screenshot shows a Microsoft Excel window with the following details:

- Title Bar:** Shows "AutoSave Off" and the file name "locations.csv".
- Menu Bar:** Includes File, Home, Insert, Draw, Page Layout, Formulas, Data, Review, View, and Help.
- Toolbar:** Standard Excel toolbar with icons for Undo, Redo, Paste, Clipboard, Font, Alignment, Number, Conditional Formatting, Format as Table, Cell Styles, Cells, Editing, Analyze Data, and Sensitivity.
- Worksheet Area:** A11 is selected. The data in column A is:
 - 1 location
 - 2 Redmond, WA
 - 3 90210
 - 4 47.67,-122.12
 - 5
 - 6
 - 7
- Bottom Navigation:** Shows tabs for "locations" and "+".
- Status Bar:** Displays "Ready", "Accessibility: Unavailable", "Display Settings", and a zoom level of 100%.

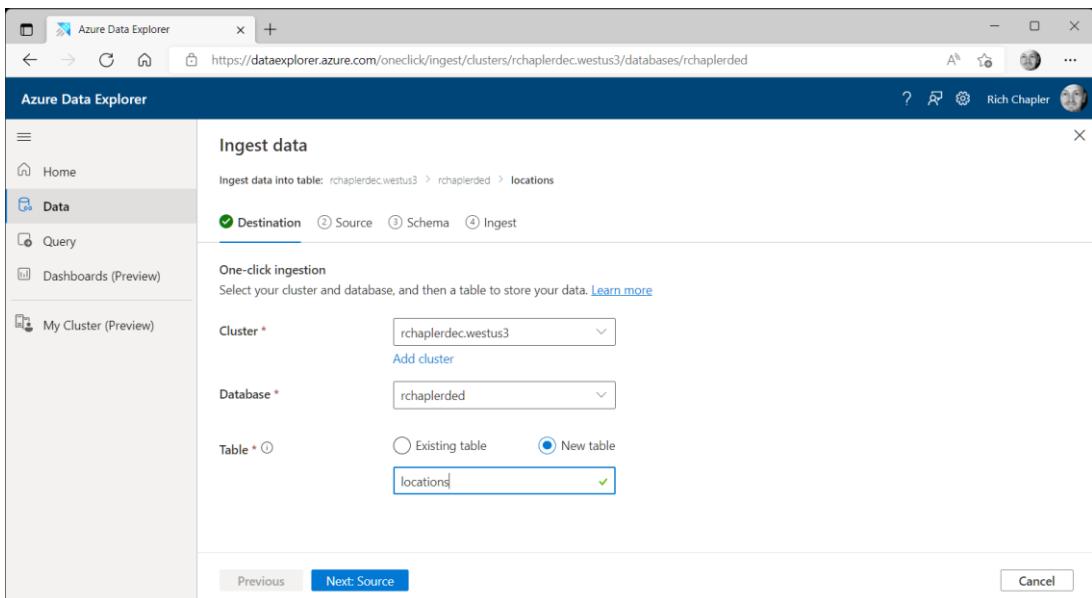
Note that I have included a header value in cell A1 as column name.

Save as “locations.csv” on your Desktop.

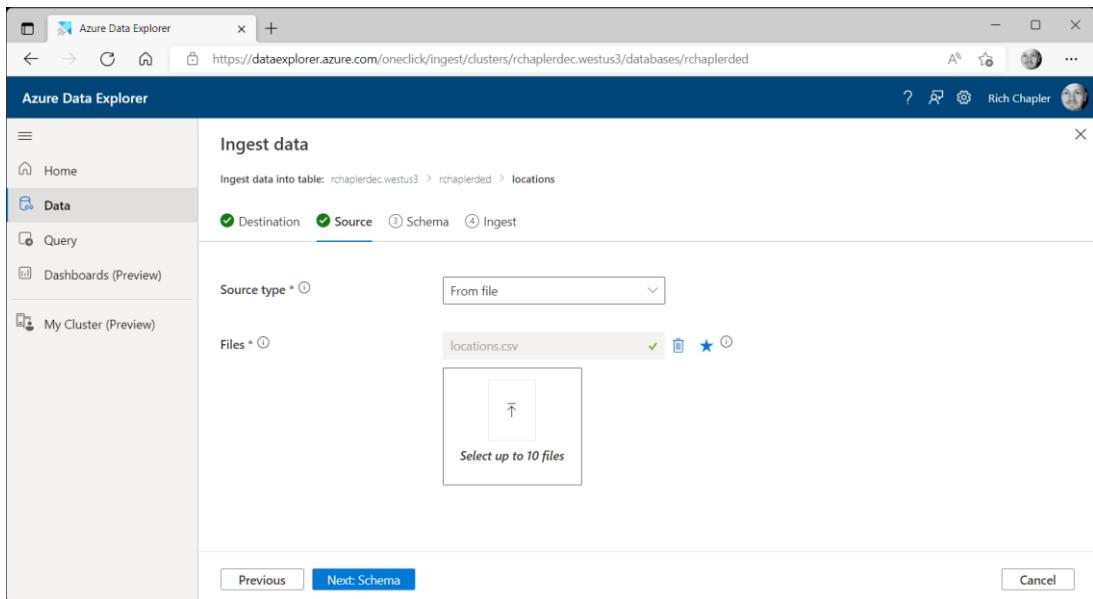
Navigate to <https://dataexplorer.azure.com/>



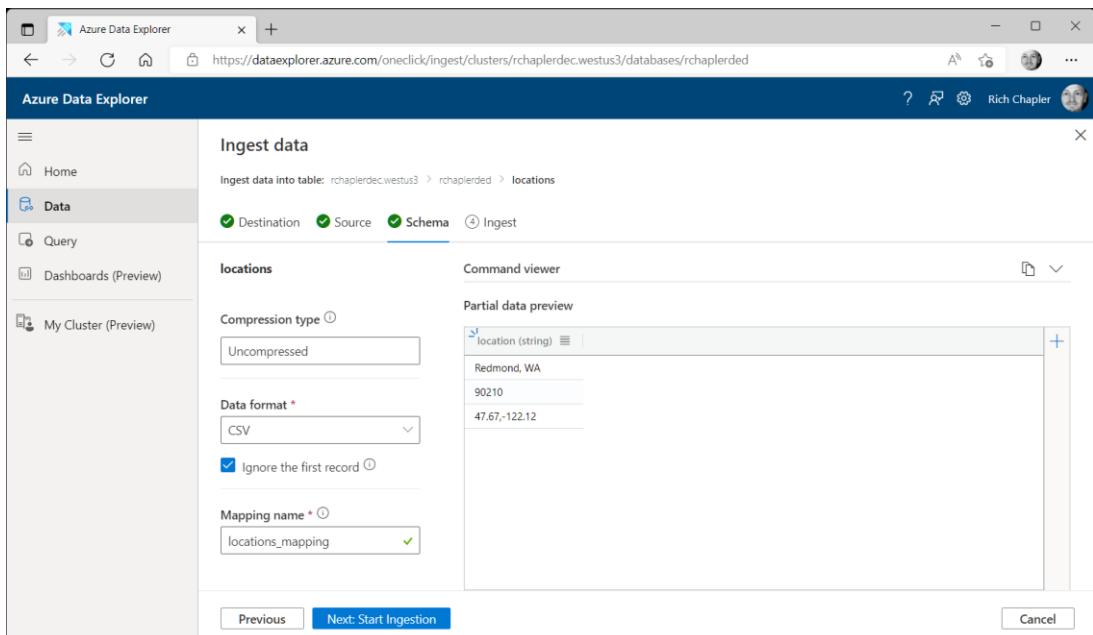
Right-click on your database and select “**Ingest data**” from the resulting drop-down menu.



Complete the “**Ingest data**” form, **Destination** tab, confirm Cluster and Database values, select the “**New table**” radio button, enter “**locations**” in the **Table** textbox, and then click the “**Next: Source**” button.



Complete the “**Ingest data**” form, **Source** tab, confirm Source type “**From file**”, browse to and select the previously created “locations.csv”, and then click the “**Next: Schema**” button.



Complete the “**Ingest data**” form, **Schema** tab, review / update the generated schema definition (as needed) and then click the “**Next: Start Ingestion**” button.

Azure Data Explorer interface showing the results of a data ingestion task. The main message is "Data ingestion completed". Below it, under "Ingestion Preparation", there is a green checkmark and the text "Ingestion Preparation". Under "Ingestion", it says "1 total blobs, 1 succeeded • 0 failed". Under "Data preview", there is a table with one row: "Redmond, WA", "90210", and "47.67,-122.12".

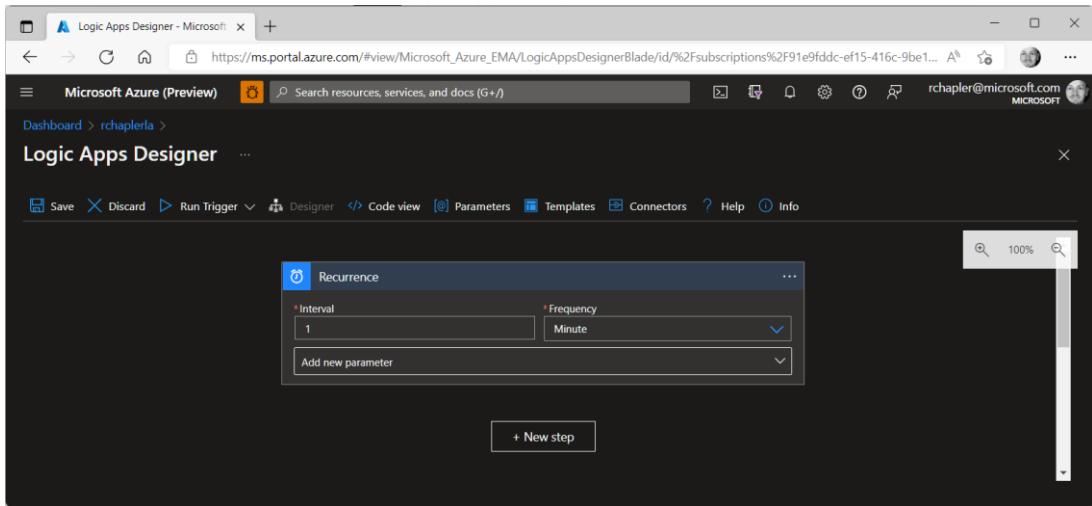
Confirm successful data ingestion.

Request Data

Logic Apps Designer interface showing the "Introducing Azure Logic Apps" page. It features a video player, icons for databases, files, charts, and clouds, and a "Watch on YouTube" button. To the right, there is a summary of Logic Apps benefits: "Building integration solutions is easier than ever. Logic Apps brings speed and scalability into the enterprise integration space. The ease of use of the designer, variety of available triggers and actions, and powerful management tools make centralizing your APIs simpler than ever. As businesses move towards digitalization, Logic Apps allows you to connect legacy and cutting-edge systems together." Below this, a section titled "Start with a common trigger" lists eight common triggers:

- When a message is received in a Service Bus queue
- When an HTTP request is received
- When a new tweet is posted
- When an Event Grid resource event occurs
- Recurrence
- When a new email is received in Outlook.com
- When a new file is created on OneDrive
- When a file is added to FTP server

Navigate to your Logic App and select **Recurrence** from the “Start with a common trigger” page.

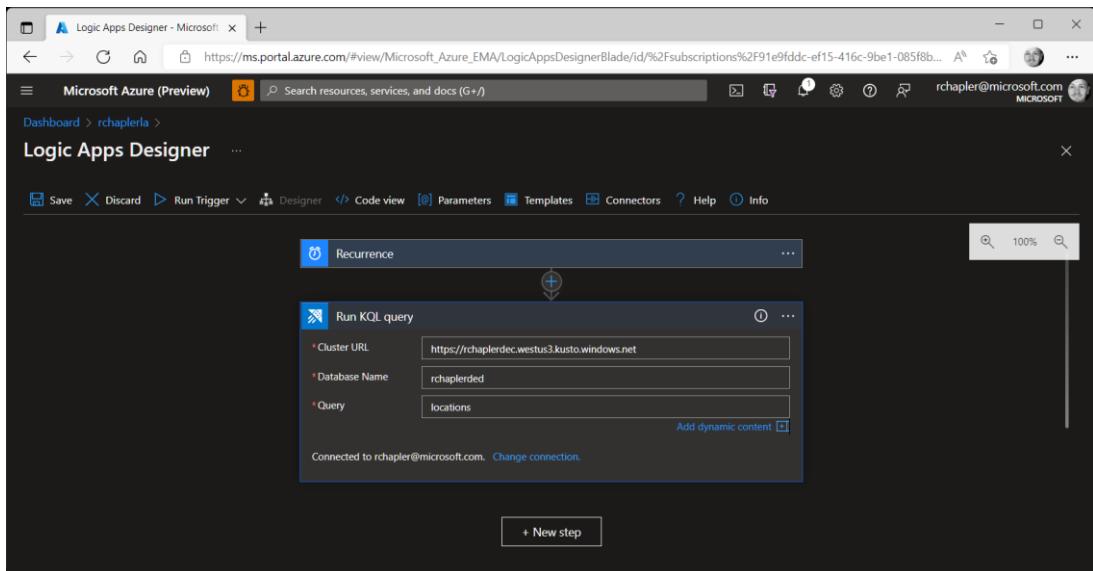


Complete the “**Logic Apps Designer**” form, **Recurrence** trigger, including:

Interval Enter 1

Frequency Select **Minute**

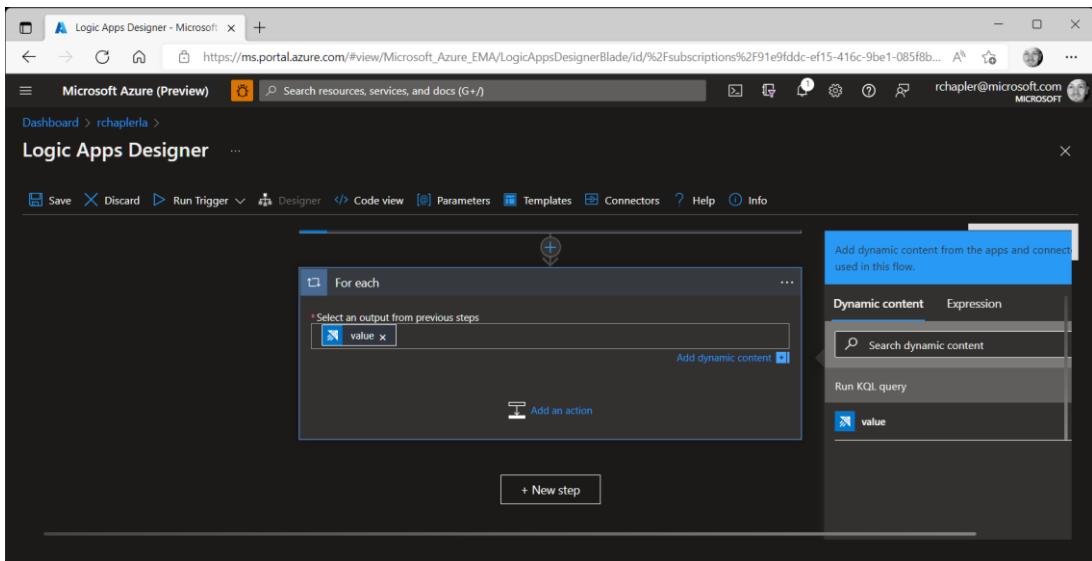
Click the “**+ New step**” button. On the resulting “**Choose an operation**” component, search for and select “**Azure Data Explorer**.” On the **Actions** tab, select “**Run KQL query**.” Select your **Tenant** and sign in, as needed.



Complete the “**Logic Apps Designer**” form, “**Run KQL query**” action, including:

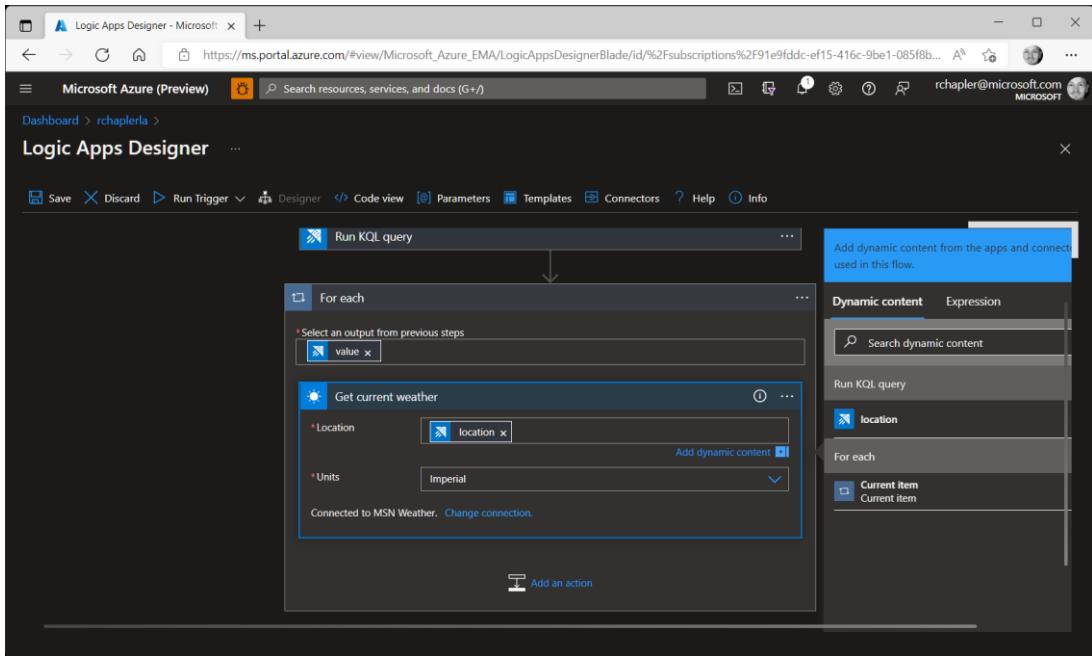
Cluster Name	Enter the URI of your Data Explorer cluster (or pool)
Database Name	Enter the name of your Data Explorer database
Query	Enter locations

Click the “**+ New step**” button.



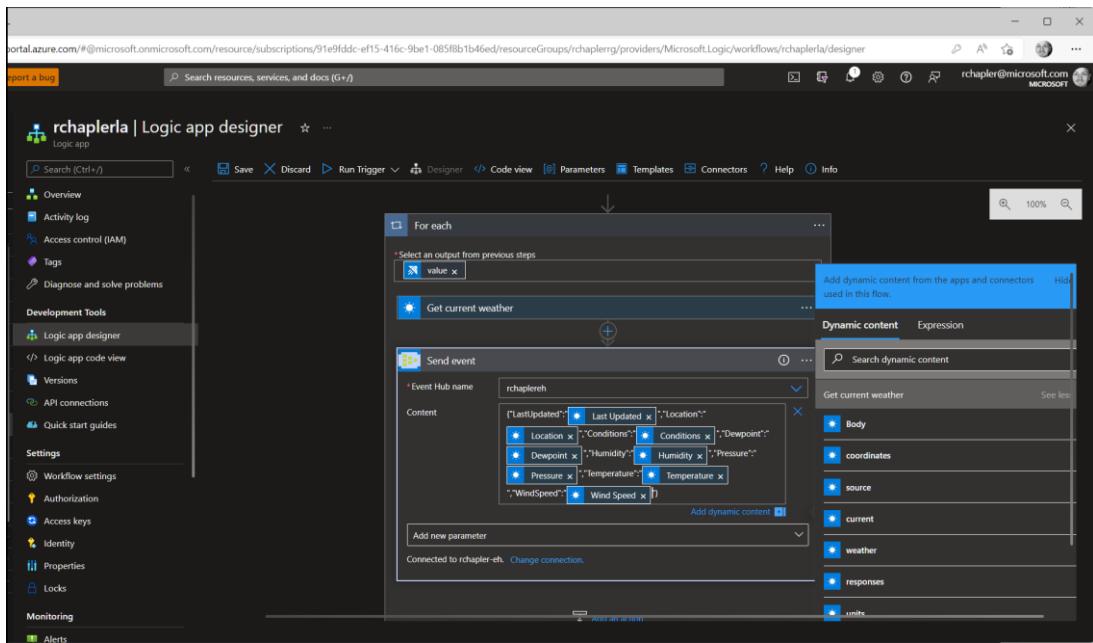
On the resulting “Choose an operation” component, search for and select “Control.” On the Actions tab, select “For each.” Click on the “Select an output from previous steps” textbox and then select dynamic content selection “value” from the “Run KQL query” group of the resulting “Dynamic content” pop-out.

Click the “Add an action” link.



On the resulting “**Choose an operation**” component, search for and select “**MSN Weather**.” On the **Actions** tab, select “**Get current weather**.” In the Location textbox, select dynamic content selection “**location**” from the “**Run KQL query**” group of the resulting “**Dynamic content**” pop-out.

Click the “**Add an action**” link.



On the resulting “Choose an operation” component, search for and select “Event Hubs.” On the Actions tab, select “Send event.” In the Location textbox, select dynamic content selection “location” from the “Run KQL query” group of the resulting “Dynamic content” pop-out.

Complete the “Logic Apps Designer” form, “Event Hubs” action, including:

Authentication Type Select “Access Key”

Connection String Enter the connection string for your Event Hub

Note: browse to the Shared Access Key on the Event Hub Namespace to easily copy the Event Hub Connection String

Click the **Create** button. When processing is complete, select your Event Hub from the “Event Hub Name” drop-down menu. Next, click the “Add new parameter” drop-down menu, and select **Content**.

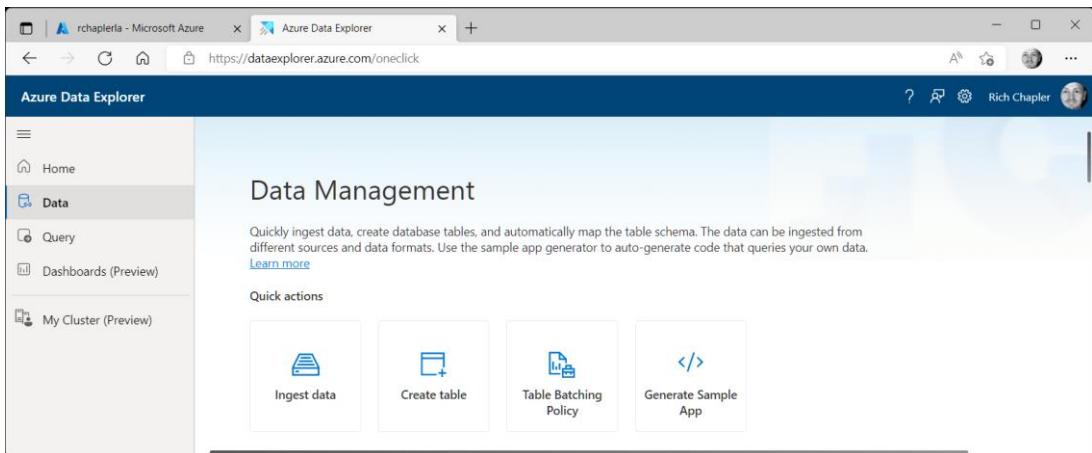
In the resulting **Content** textbox, construct a string using the following base string with bracketed values replaced with dynamic content.

```
{"LastUpdated": "{LastUpdated}", "Location": "{Location}", "Conditions": "{Conditions}", "Dewpoint": "{Dewpoint}", "Humidity": "{Humidity}", "Pressure": "{Pressure}", "Temperature": "{Temperature}", "WindSpeed": "{WindSpeed}"}
```

Click **Save** and then “Run Trigger” > **Run** to test. If everything is correct, the run should finish quickly, and you will see green checkmark circles next to all boxes.

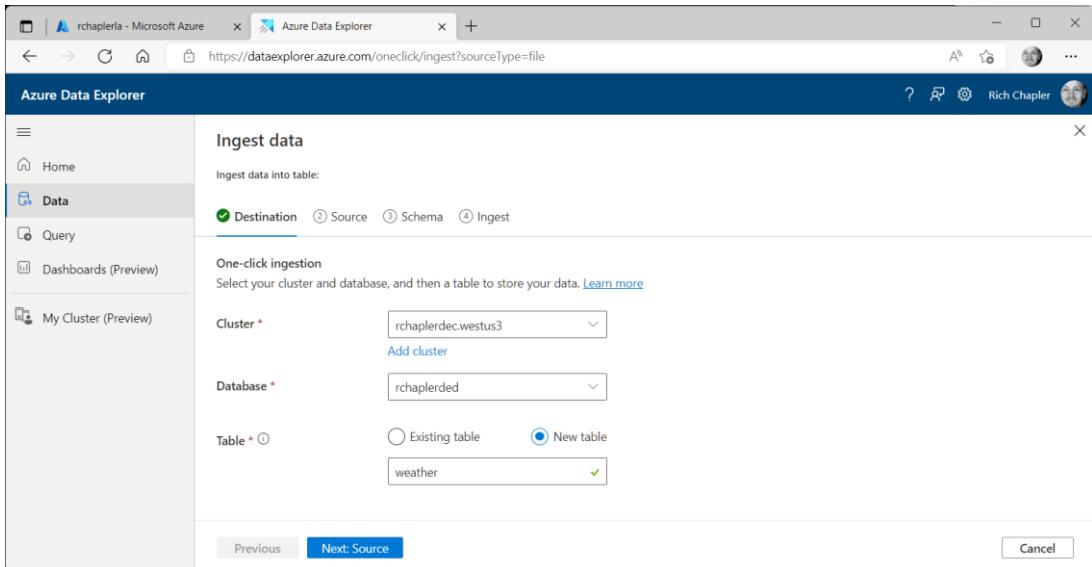
Ingest Data

Navigate to <https://dataexplorer.azure.com/>



The screenshot shows the Azure Data Explorer interface. The left sidebar has a 'Data' menu item selected. The main area is titled 'Data Management' with a sub-instruction: 'Quickly ingest data, create database tables, and automatically map the table schema. The data can be ingested from different sources and data formats. Use the sample app generator to auto-generate code that queries your own data.' Below this are four 'Quick actions' buttons: 'Ingest data', 'Create table', 'Table Batching Policy', and 'Generate Sample App'.

Click **Data** in the navigation pane and then the “**Ingest Data**” button on the resulting “**Data Management**” page.



The screenshot shows the 'Ingest data' configuration page. The 'Destination' tab is active. It displays fields for 'Cluster' (set to 'rchaplerdec.westus3'), 'Database' (set to 'rchaplerdec'), and 'Table' (set to 'weather'). The 'New table' radio button is selected. At the bottom, there are 'Previous', 'Next: Source' (highlighted in blue), and 'Cancel' buttons.

Complete the “**Ingest data**” form, **Destination** tab, confirm Cluster and Database values, select the “**New table**” radio button, enter “**weather**” in the **Table** textbox, and then click the “**Next: Source**” button.

The screenshot shows the 'Ingest data' form in the Azure Data Explorer interface. The left sidebar has 'Data' selected. The main area is titled 'Ingest data' with a sub-section 'Ingest data into table:'. The 'Source' tab is active, indicated by a blue underline. Below it, there are tabs for 'Destination', 'Schema', and 'Ingest'. The 'Source type' dropdown is set to 'Event Hub'. Under 'Data Connection', fields include 'Subscription' (rcchapler), 'Event Hub namespace' (rchaplerehn), 'Event Hub' (rchaplereh), 'Data connection name' (rcchaplerded-rchaplereh), 'Consumer group' (\$Default), 'Compression' (None), and 'Event system properties'. At the bottom are buttons for 'Previous', 'Next: Schema' (which is blue and underlined), and 'Cancel'.

Complete the “**Ingest data**” form, **Source** tab, including:

Source Type	Select “Event Hub”
Event Hub Namespace	Select your Event Hub Namespace
Event Hub	Select your Event Hub
Data Connection Name	Confirm system-selected value
Consumer Group	Confirm default selection, “\$Default”
Compression	Confirm default selection, None

Click the “**Next: Schema**” button.

The screenshot shows the 'Ingest data' form in the Azure Data Explorer. The left sidebar has 'Data' selected. The main area is titled 'Ingest data' with a sub-section 'Ingest data into table: weather (rchaplerded-rchaplerdeh)'. The 'Schema' tab is active, indicated by a blue underline. Below it are 'Destination', 'Source', and 'Ingest' tabs, all with green checkmarks. Under 'Destination', 'Source', and 'Schema', there are several configuration options like 'Enable streaming ingestion', 'Data batching latency' (set to 40), and 'Data format' (set to 'TXT'). There are also two checked checkboxes: 'Include events with the selected format' and 'Include events without formats'. On the right, a 'Command viewer' pane shows a partial data preview with a JSON array of weather data. Buttons at the bottom include 'Previous', 'Next: Start Ingestion' (highlighted in blue), and 'Cancel'.

Complete the “**Ingest data**” form, **Schema** tab, review / update the generated schema definition (as needed) and then click the “**Next: Start Ingestion**” button.

The screenshot shows the Azure Data Explorer Query blade. The left sidebar has 'Query' selected. The main area shows a query editor with the following code:

```
1 weather
2 | take 3
```

Below the query, the results are displayed in a table named 'Table 1'. The table has three columns: 'Stats', 'Search', and 'UTC'. The 'data' column shows the following JSON array:[{"LastUpdated": "2022-06-15T16:24:34+00:00", "Location": "Redmond, WA", "Conditions": "Cloudy", "Dewpoint": 47, "Humidity": 61}, {"LastUpdated": "2022-06-15T16:21:23+00:00", "Location": "90210, CA", "Conditions": "Cloudy", "Dewpoint": 61, "Humidity": 61}, {"LastUpdated": "2022-06-15T16:24:34+00:00", "Location": "16537 NE 74th St, Redmond, WA 98052, United States", "Conditions": "Cloudy", "Dewpoint": 47, "Humidity": 61}, {"LastUpdated": "2022-06-15T16:27:28+00:00", "Location": "90210, CA", "Conditions": "Cloudy", "Dewpoint": 61, "Humidity": 61}, {"LastUpdated": "2022-06-15T16:24:34+00:00", "Location": "Redmond, WA", "Conditions": "Cloudy", "Dewpoint": 47, "Humidity": 61}, {"LastUpdated": "2022-06-15T16:24:34+00:00", "Location": "16537 NE 74th St, Redmond, WA 98052, United States", "Conditions": "Cloudy", "Dewpoint": 47, "Humidity": 61}, {"LastUpdated": "2022-06-15T16:27:28+00:00", "Location": "90210, CA", "Conditions": "Mostly cloudy", "Dewpoint": 61, "Humidity": 61}, {"LastUpdated": "2022-06-15T16:24:34+00:00", "Location": "Redmond, WA", "Conditions": "Cloudy", "Dewpoint": 47, "Humidity": 61}, {"LastUpdated": "2022-06-15T16:24:34+00:00", "Location": "16537 NE 74th St, Redmond, WA 98052, United States", "Conditions": "Cloudy", "Dewpoint": 47, "Humidity": 61}

Confirm successful data ingestion.

...using Databricks

Follow these instructions to **source data from a REST API**.

Prerequisites

This solution requires the following resources:

- Databricks
- Key Vault

Stage Resources

For this exercise, we will use “**Current Weather**” data from <https://openweathermap.org/api>.

The screenshot shows the OpenWeatherMap API landing page. At the top, there's a navigation bar with links for Guide, API, Pricing, Maps, FAQ, Partners, Blog, Marketplace, Sign in, and Support. Below the navigation bar, the main content area has a heading "Weather API". Underneath, a message encourages users to sign up for monthly subscriptions or use the free account. Three service options are listed: "Current Weather Data", "Hourly Forecast 4 days", and "One Call API". Each service has a brief description and two buttons: "API doc" and "Subscribe".

Service	Description	Buttons
Current Weather Data	Access current weather data for any location including over 200,000 cities. We collect and process weather data from different sources such as global and local weather models, satellites, radars and vast network of weather stations. JSON, XML, and HTML formats. Available for both Free and paid subscriptions.	API doc, Subscribe
Hourly Forecast 4 days	Hourly forecast is available for 4 days. Forecast weather data for 96 timestamps. Higher geographic accuracy. JSON and XML formats. Available for Developer, Professional and Enterprise accounts.	API doc, Subscribe
One Call API	Make one API call and get current, forecast and historical weather data. Minute forecast for 1 hour. Hourly forecast for 48 hours. Daily forecast for 7 days. Historical data for 5 previous days. National weather alerts. JSON format. Available for both Free and paid subscriptions.	API doc, Subscribe

I chose this as my sample source because they have a Free subscription, their data is relatable, and it is easy to use.

Navigate to <https://openweathermap.org/pric>.

The screenshot shows the OpenWeatherMap Pricing page. At the top, there's a navigation bar with links for Guide, API, Pricing, Maps, FAQ, Partners, Blog, Marketplace, Sign in, and Support. Below the navigation is a search bar with the placeholder "Weather in your city". The main content area has a heading "Pricing" and a sub-heading "Current weather and forecasts collection". A table displays five subscription plans:

Free	Startup 40 USD / month	Developer 180 USD / month	Professional 470 USD / month	Enterprise 2,000 USD / month
Get API key	Subscribe	Subscribe	Subscribe	Subscribe
60 calls/minute 1,000,000 calls/month	600 calls/minute 10,000,000 calls/month	3,000 calls/minute 100,000,000 calls/month	30,000 calls/minute 1,000,000,000 calls/month	200,000 calls/minute 5,000,000,000 calls/month
Current Weather	Current Weather	Current Weather	Current Weather	Current Weather

Click the “**Get API key**” button in the **Free** section and complete the “**Create a New Account**” process.

When you’re successfully created an account, making an API call for “**Current Weather**” data is as simple as:

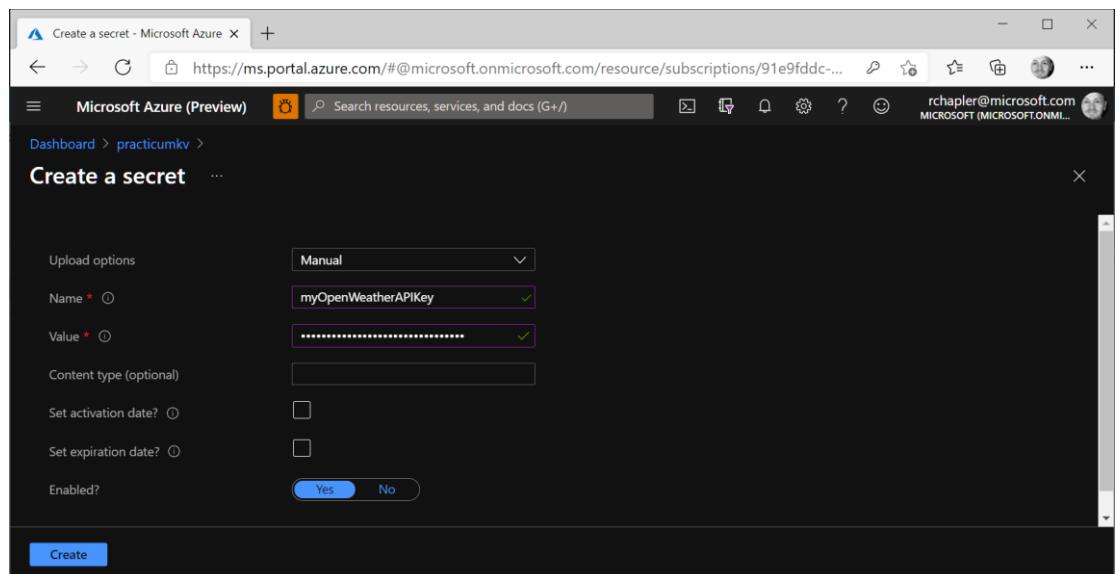
```
api.openweathermap.org/data/2.5/weather?q={city name}&appid={API key}
```

The screenshot shows a browser window with the URL `api.openweathermap.org/data/2.5/weather?q=redmond&appid=...`. The page title is "Not secure". The page content displays a JSON response for the weather in Redmond:

```
{"coord":{"lon":-122.1215,"lat":47.674}, "weather":[{"id":804,"main":"Clouds","description":"overcast clouds","icon":"04d"}], "base":"stations", "main": {"temp":279.53,"feels_like":277.4,"temp_min":278.15,"temp_max":280.93,"pressure":1007,"humidity":65,"sea_level":1007,"grnd_level":1006}, "visibility":10000,"wind":{"speed":0.27,"deg":241}, "clouds":{"all":100}, "dt":1611870000, "sys": {"type":3,"id":2010401,"country":"US","sunrise":1611848385,"sunset":1611882185}, "timezone":-28800, "id":5808079, "name":"Redmond", "cod":200}
```

Add Secret to Key Vault

Add your API Key to Key Vault.



The screenshot shows the 'Create a secret' page in the Microsoft Azure portal. The URL in the address bar is <https://ms.portal.azure.com/#@microsoft.onmicrosoft.com/resource/subscriptions/91e9fddc-...>. The page title is 'Create a secret - Microsoft Azure'. The main form fields are:

- Upload options:** Manual (dropdown menu)
- Name ***: myOpenWeatherAPIKey (text input field with a green checkmark)
- Value ***: (text input field with a green checkmark)
- Content type (optional):** (empty text input field)
- Set activation date?**: (checkbox)
- Set expiration date?**: (checkbox)
- Enabled?**: Yes (radio button selected)

At the bottom left is a blue 'Create' button.

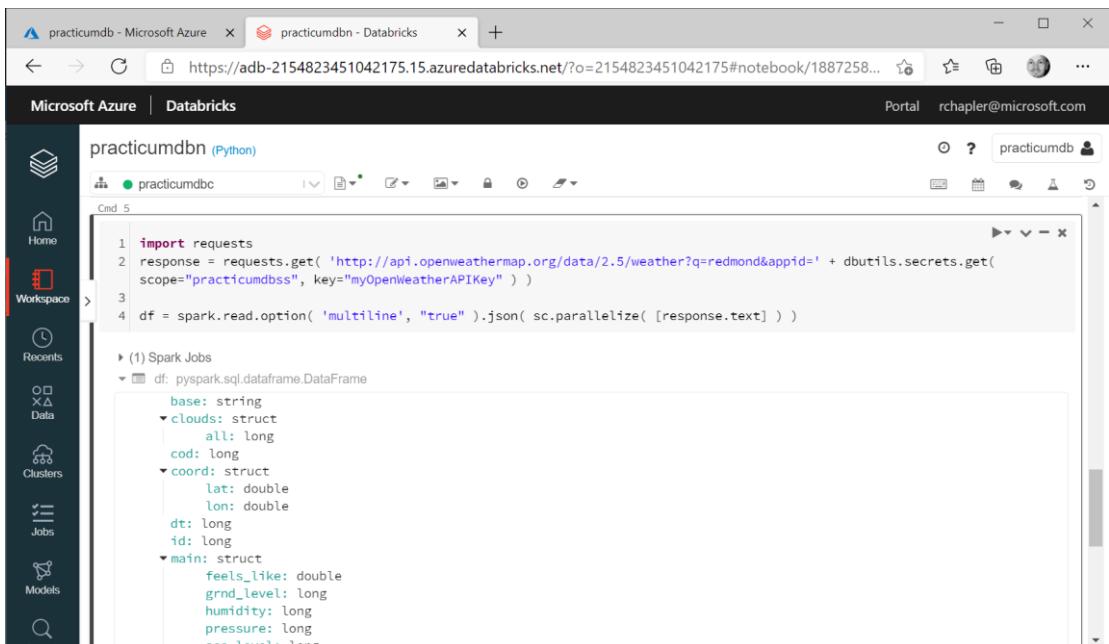
Prepare Logic

Navigate to Databricks and the <UseCase>dbn notebook. Add a new cell and paste the following code:

```
import requests
response = requests.get( 'http://api.openweathermap.org/data/2.5/weather?q=redmond&appid=' + 
dbutils.secrets.get( scope="<UseCase>dbss", key="myOpenWeatherAPIKey" ) )

df = spark.read.option( 'multiline', "true" ).json( sc.parallelize( [response.text] ) )
```

Run the cell.



The screenshot shows the Microsoft Azure Databricks interface. On the left is a sidebar with icons for Home, Workspace (selected), Recent, Clusters, Jobs, Models, and Search. The main area has a title bar 'practicumdb - Microsoft Azure' and 'practicumdb - Databricks'. Below the title bar is a URL 'https://adb-2154823451042175.15.azuredatabricks.net/?o=2154823451042175#notebook/1887258...'. The workspace contains a notebook titled 'practicumdbn (Python)'. A code cell contains the provided Python code. Below the code, a 'Spark Jobs' section is expanded, showing a single job with a DataFrame named 'df'. The schema for 'df' is displayed as follows:

```
base: string
  clouds: struct
    all: long
  cod: long
  coord: struct
    lat: double
    lon: double
  dt: long
  id: long
  main: struct
    feels_like: double
    grnd_level: long
    humidity: long
    pressure: long
    sea_level: long
```

Another Python example (including POST, headers, and body):

```
import json
stringJSON = '{"data": [{"1212":0,"1227":0,..."ZZ9":0}]}'
theJSON = json.loads(stringJSON)
response = requests.post(
'http://eedfb2cf-deb1-4260-971d-d7a6c308d9b2.eastus.azurecontainer.io/score'
, headers={'CONTENT-TYPE': 'application/json'}
, json = theJSON
)
response.json()
```

...from Purview

Use Case

Example notes:

- “Our current governance solution lacks connectors for some Azure resource types”
- “We want to use Purview to collect source metadata from Azure resources”
- “Can we port Purview-collected metadata into our corporate-approved solution?”

Prerequisites

This solution requires the following resources:

- Application Registration (with a client secret)
- Data Lake (with Container)
- Purview
- Synapse (with linked service and dataset for your Data Lake)

Prepare Purview

Navigate to **Purview Governance Portal**, then **Data Map**, and then **Collections**.

Select the collection that is home to the metadata you plan to query.

Select the **Role Assignments** tab and add your Application Registration to “**Collection admins**”, “**Data source admins**”, and “**Data curators**.”

Create Linked Service

Navigate to Synapse Studio, click the **Manage** navigation icon, select “**Linked services**” from the “**External connections**” grouping in the resulting navigation. Click the “**+ New**” button.

Complete the “**New linked service...**” pop-out, including:

Connect via...	Confirm default selection, “ AutoResolveIntegrationRuntime ”
Base URL	Modify and enter: <code>https://{{Purview Instance Name}}.purview.azure.com/scan/datasources/{{Purview Data Source Name}}?api-version=2022-02-01-preview</code>
Authentication Type	Select Anonymous

Click “**Test connection**” to confirm successful connection and then click the **Save** button.

Create Dataset

Navigate to Synapse Studio and click the **Data** navigation icon.

Select “**Integration datasets**” from the **Linked** tab in the resulting navigation.

Click “+” and select “**Integration dataset**” from the resulting drop-down menu.

Complete the “**New integration dataset**” pop-out, search for and select **REST**.

Click the **Continue** button.

Complete the “**Set properties**” pop-out, including:

Linked Service	Select the REST Linked Service (created in the prior step)
-----------------------	------------------------------------------------------------

Click the **OK** button.

On the resulting screen, click “**Test connection**” to confirm successful connection.

Create Pipeline

When we are finished, our Synapse Pipeline will look and function as snipped below.



Navigate to Synapse Studio, click the **Integrate** navigation icon, click “+” and select **Pipeline** from the resulting drop-down menu.

Activity 1: Get Token

This activity will make a REST API call to <http://login.microsoftonline.com> and get a bearer token.

Expand **General** in the **Activities** bar, then drag-and-drop a **Web** component into the activity window. On the **Settings** tab, including:

URL	Modify and enter: https://login.microsoftonline.com/{TenantId}/oauth2/token
Method	Select POST
Headers	Click “ + Add ” and enter key-value pair: content-type, application/x-www-form-urlencoded
Body	Modify and enter: <code>grant_type=client_credentials&client_id={Client Identifier}&client_secret={Client Secret}&resource=https://purview.azure.net</code>

Click **Debug** and monitor result to confirm successful progress.

Activity 2: Get Data

For this exercise, focused simply on demonstrating end-to-end solution functionality, we are going to pull information about data sources and capture the result in a file in the Data Lake.

Many other API options are discussed in the Purview API documentation:

<https://docs.microsoft.com/en-us/rest/api/purview/>

Expand “**Move & Transform**” in the **Activities** bar, drag-and-drop a “**Copy data**” component into the activity window and then create a dependency from the “**Get Bearer Token**” component.

On the **Source** tab, including:

Source Dataset	Select your REST dataset
Request Method	Select GET
Additional Headers	<p>Click “+ Add” and enter key-value pair: content-type, application/json; charset=utf-8</p> <p>Click “+ Add” and enter key-value pair (with dynamic content): Authorization, @concat('Bearer ',activity('Get Token').output.access_token)</p>

No additional configuration is required on this tab.

On the **Sink** tab, including:

Sink Dataset	Select your Data Lake dataset
---------------------	-------------------------------

No additional configuration is required on this tab.

Confirm Success

Click **Debug** and monitor result to confirm successful progress.

Special Load

Batch Upsert

Follow these instructions to provide for recurring upsert of data.

Prerequisites

This solution requires the following resources:

- Data Lake
- Databricks

CREATE TABLE

Navigate to Databricks and the <UseCase>dbn notebook.

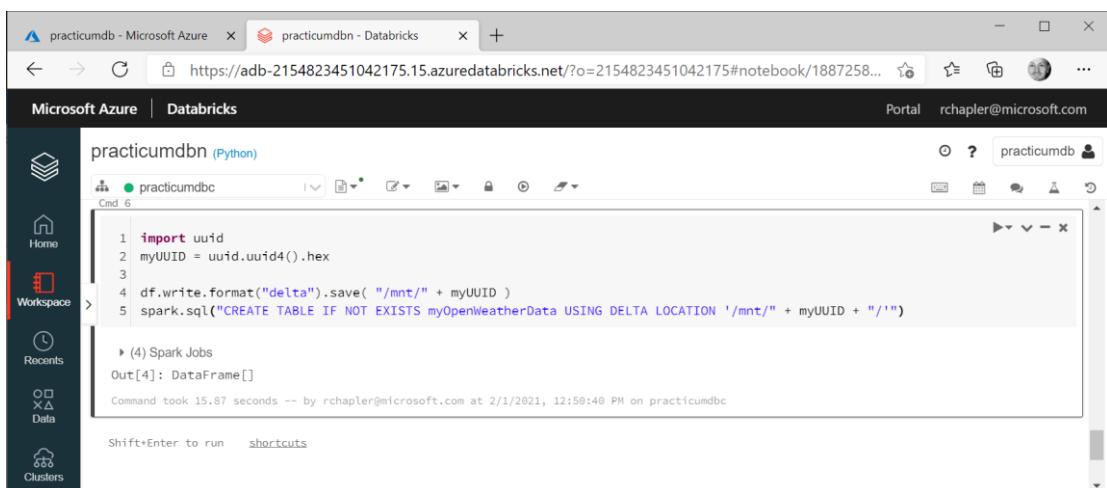
Add a new cell and paste the following code:

```
import uuid
myUUID = uuid.uuid4().hex

df.write.format("delta").save( "/mnt/" + myUUID )
spark.sql("CREATE TABLE IF NOT EXISTS myOpenWeatherData USING DELTA LOCATION '/mnt/" + myUUID + "'")
```

Notes:

- We are using dataframe **df** created in [Objective 3 ... The Logic](#)
- We are using format **delta** to provide for future UPSERT operations.



The screenshot shows the Microsoft Azure Databricks workspace. The left sidebar has icons for Home, Workspace (selected), Recents, Data, and Clusters. The main area shows a notebook titled "practicumdbn (Python)". A code cell in the notebook contains the following Python code:

```
import uuid
myUUID = uuid.uuid4().hex
df.write.format("delta").save( "/mnt/" + myUUID )
spark.sql("CREATE TABLE IF NOT EXISTS myOpenWeatherData USING DELTA LOCATION '/mnt/" + myUUID + "'")
```

The output of the cell shows:

- ▶ (4) Spark Jobs
- Out[4]: DataFrame[]

Below the cell, it says "Command took 15.87 seconds -- by rchabler@microsoft.com at 2/1/2021, 12:50:40 PM on practicumdb". At the bottom of the notebook interface, there is a note: "Shift+Enter to run shortcuts".

MERGE INTO

Navigate to Databricks and the <UseCase>dbn notebook. Add a new cell and paste the following code:

```
import requests
import uuid

response = requests.get( 'http://api.openweathermap.org/data/2.5/weather?q=redmond&appid=' + dbutils.secrets.get( scope="<UseCase>dbss", key="myOpenWeatherAPIKey" ) )
df = spark.read.option( 'multiline', "true" ).json( sc.parallelize( [response.text] ) )

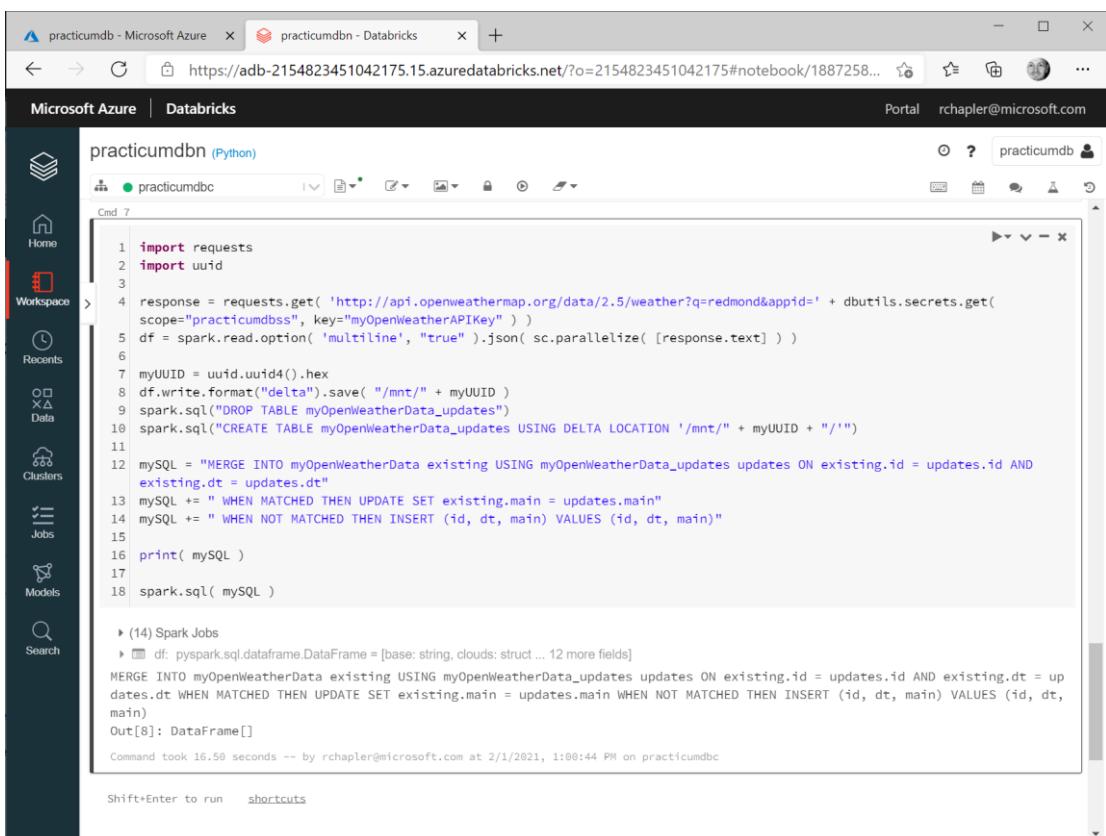
myUUID = uuid.uuid4().hex
df.write.format("delta").save( "/mnt/" + myUUID )
spark.sql("DROP TABLE myOpenWeatherData_updates")
spark.sql("CREATE TABLE myOpenWeatherData_updates USING DELTA LOCATION '/mnt/" + myUUID + "'")

mySQL = "MERGE INTO myOpenWeatherData existing USING myOpenWeatherData_updates updates ON existing.id = updates.id AND existing.dt = updates.dt"
mySQL += " WHEN MATCHED THEN UPDATE SET existing.main = updates.main"
mySQL += " WHEN NOT MATCHED THEN INSERT (id, dt, main) VALUES (id, dt, main)"

print( mySQL )

spark.sql( mySQL )
```

Run the cell.



The screenshot shows the Azure Databricks interface. On the left is the sidebar with icons for Home, Workspace, Recents, Data, Clusters, Jobs, Models, and Search. The main area has two tabs: 'practicumdb - Microsoft Azure' and 'practicumdb - Databricks'. The 'practicumdb - Databricks' tab is active, showing a notebook titled 'practicumdb (Python)'. The notebook contains the provided Python code. The code uses the requests library to get weather data from OpenWeatherMap, saves it as a delta table, and then performs a MERGE INTO operation to update or insert data into the 'myOpenWeatherData' table. The output of the cell shows the generated SQL query and its execution results, including a DataFrame object named 'df' and a command execution summary.

```
practicumdb - Microsoft Azure x prakticumdb - Databricks x +  
https://adb-2154823451042175.azuredatabricks.net/?o=2154823451042175#notebook/1887258...  
Microsoft Azure | Databricks  
practicumdb (Python)  
practicumdbc  
Cnd 7  
1 import requests  
2 import uuid  
3  
4 response = requests.get( 'http://api.openweathermap.org/data/2.5/weather?q=redmond&appid=' + dbutils.secrets.get( scope="practicumdbss", key="myOpenWeatherAPIKey" ) )  
5 df = spark.read.option( 'multiline', "true" ).json( sc.parallelize( [response.text] ) )  
6  
7 myUUID = uuid.uuid4().hex  
8 df.write.format("delta").save( "/mnt/" + myUUID )  
9 spark.sql("DROP TABLE myOpenWeatherData_updates")  
10 spark.sql("CREATE TABLE myOpenWeatherData_updates USING DELTA LOCATION '/mnt/" + myUUID + "'")  
11  
12 mySQL = "MERGE INTO myOpenWeatherData existing USING myOpenWeatherData_updates updates ON existing.id = updates.id AND existing.dt = updates.dt"  
13 mySQL += " WHEN MATCHED THEN UPDATE SET existing.main = updates.main"  
14 mySQL += " WHEN NOT MATCHED THEN INSERT (id, dt, main) VALUES (id, dt, main)"  
15  
16 print( mySQL )  
17  
18 spark.sql( mySQL )  
  
▶ (14) Spark Jobs  
▶ df: pyspark.sql.dataframe.DataFrame = [base: string, clouds: struct ... 12 more fields]  
MERGE INTO myOpenWeatherData existing USING myOpenWeatherData_updates updates ON existing.id = updates.id AND existing.dt = updates.dt WHEN MATCHED THEN UPDATE SET existing.main = updates.main WHEN NOT MATCHED THEN INSERT (id, dt, main) VALUES (id, dt, main)  
Out[8]: DataFrame[]  
Command took 16.50 seconds -- by rchapler@microsoft.com at 2/1/2021, 1:00:44 PM on practicumdbc  
Shift+Enter to run    shortcuts
```

Confirm Success

Navigate to Databricks and the <UseCase>dbn notebook. Add a new cell and paste the following code:

```
%sql  
SELECT * FROM myOpenWeatherData
```

Run the cell.

The screenshot shows the Microsoft Azure Databricks workspace interface. On the left, there is a sidebar with icons for Home, Workspace (which is selected), Recent, Data, Clusters, Jobs, Models, and Search. The main area has two tabs: 'practicumdbn (Python)' and 'practicumdbn (SQL)'. The Python tab shows a command history and a table view of the data. The SQL tab shows the executed SQL code and its output. The output of the SQL code is a DataFrame containing the following data:

	base	clouds	cod	coord	dt	id	main
1	stations	["all": 100]	200	{"lat": 47.674, "lon": -122.1215}	1612211690	5808079	{"feels_like": 280.49, "grnd_level": 1004, "h": 282.59, "temp_min": 281.48}
2	null	null	null	null	1612212650	5808079	{"feels_like": 279.62, "grnd_level": 1005, "h": 282.59, "temp_min": 281.48}

Below the table, it says 'Showing all 2 rows.'

At the bottom of the SQL tab, it says 'Command took 1.18 seconds -- by rchapler@microsoft.com at 2/1/2021, 1:22:13 PM on practicumdbc'.

Note that the row created using MERGE INTO has a limited number of populated cells since only some of the possible cells were included in WHEN NOT MATCHED THEN INSERT (id, dt, main)... we can, of course, expand this to include all fields about which we care.

Synchronize Unstructured Data

Server-to-Cloud

(featuring Linux and AzCopy)

Follow these instructions to automate recurring synchronization of unstructured data {e.g., image files} from an on-prem Linux server to Azure Storage.

Use Case: Customer XYZ shared the following requirements:

- **Unstructured data (images) are regularly loaded to an **on-prem Linux server****
 - This configuration is owned by a separate group and cannot be changed to load directly to Azure
- Synchronization of new data must occur **as soon as it becomes available**

Prerequisites

This solution requires the following resources:

- Storage Account with container named **images** and Shared Access Token (with **Read, Write, Delete, List** permissions)
- Virtual Machine (Linux/Ubuntu) ... intended to mimic the customers' on-prem configuration

Stage Sample Data

In this section, we will load three sample images from an online traffic cam source ([1139--9 \(320x240\) \(fl511.com\)](https://fl511.com/map/Cctv/1139--9)) onto our Linux virtual machine.



Navigate to Azure Cloud Shell.

Connect using SSH

Update and execute the following command to connect to the server using SSH:

```
ssh -i ~/.ssh/id_rsa rchapler@{public ip address}
```

Create Images Directory

Execute the following command to create a directory for our sample images:

```
mkdir images
```

Get Sample Images

Execute the following command to get a sample image from the traffic cam source:

```
wget -O images/01.jpg https://f1511.com/map/cctv/1139--9
```

Update and execute the previous command to get additional, updated sample images (every ~minute); example:

```
wget -O images/02.jpg https://f1511.com/map/cctv/1139--9
```

```
wget -O images/03.jpg https://f1511.com/map/cctv/1139--9
```

Execute the following commands to list sample files in the **images** directory:

```
cd images  
ls
```

Prepare AzCopy

In this section, we will download, install, and test AzCopy on our Linux virtual machine.

Download AzCopy

Execute the following command to download the AzCopy installer:

```
wget https://aka.ms/downloadazcopy-v10-linux
```

You can expect a response like:

```
--2021-11-23 16:21:50-- https://aka.ms/downloadazcopy-v10-linux  
Resolving aka.ms (aka.ms)... 23.63.47.52  
Connecting to aka.ms (aka.ms)|23.63.47.52|:443... connected.  
HTTP request sent, awaiting response... 301 Moved Permanently  
Location: https://azcopyvnext.azureedge.net/release20211027/azcopy_linux_amd64_10.13.0.tar.gz  
[following]  
--2021-11-23 16:21:50--  
https://azcopyvnext.azureedge.net/release20211027/azcopy\_linux\_amd64\_10.13.0.tar.gz  
Resolving azcopyvnext.azureedge.net (azcopyvnext.azureedge.net)... 104.86.182.19, 104.86.182.43,  
2600:1409:9800:21::17d8:93b8, ...  
Connecting to azcopyvnext.azureedge.net (azcopyvnext.azureedge.net)|104.86.182.19|:443... connected.  
HTTP request sent, awaiting response... 200 OK  
Length: 11876012 (11M) [application/gzip]
```

```
Saving to: 'downloadazcopy-v10-linux'

downloadazcopy-v10-linux
100%[=====] =====
==>] 11.33M ---KB/s  in 0.08s

2021-11-23 16:21:50 (141 MB/s) - 'downloadazcopy-v10-linux' saved [11876012/11876012]
```

Extract Download

Execute the following command to extract the downloaded GZIP file:

```
tar -xvf downloadazcopy-v10-linux
```

You can expect a response like:

```
azcopy_linux_amd64_10.13.0/
azcopy_linux_amd64_10.13.0/NOTICE.txt
azcopy_linux_amd64_10.13.0/azcopy
```

AzCopy Sync

Update and execute the following command to test file synchronization using AzCopy:

```
./azcopy_linux_amd64_10.13.0/azcopy sync "/home/rchapler/images"
"https://<UseCase>sa.blob.core.windows.net/images?sp=rwdl&st=2021-11-23T20:23:01Z&se=2022-10-
01T03:23:01Z&spr=https&sv=2020-08-04&sr=c&sig=U3o98QJ1NJ8wAldkjBBJv8Eken4N3R1rr50gLxsq6PE%3D" --
recursive
```

You can expect a response like:

```
INFO: Any empty folders will not be processed, because source and/or destination doesn't have full
folder support
```

```
Job ffac3e11-2fc7-ee4f-5216-95fd028b1a51 has started
Log file is located at: /home/rchapler/.azcopy/ffac3e11-2fc7-ee4f-5216-95fd028b1a51.log
```

```
3 Files Scanned at Source, 0 Files Scanned at Destination
```

```
Job ffac3e11-2fc7-ee4f-5216-95fd028b1a51 Summary
Files Scanned at Source: 3
Files Scanned at Destination: 0
Elapsed Time (Minutes): 0.0334
Number of Copy Transfers for Files: 3
Number of Copy Transfers for Folder Properties: 0
Total Number Of Copy Transfers: 3
Number of Copy Transfers Completed: 3
Number of Copy Transfers Failed: 0
Number of Deletions at Destination: 0
Total Number of Bytes Transferred: 58702
Total Number of Bytes Enumerated: 58702
Final Job Status: Completed
```

Navigate to the **images** container in your Storage Account and confirm file synchronization.

The screenshot shows the Microsoft Azure Storage Container blade for the 'images' container. The left sidebar has a 'Settings' section with options like Shared access tokens, Access policy, Properties, Metadata, and Editor (preview). The main area displays a table of blobs:

Name	Modified	Access tier	Archive status	Blob type	Size	Lease state
01.jpg	11/23/2021, 9:46:03 ...	Hot (Inferred)		Block blob	19.69 KiB	Available
02.jpg	11/23/2021, 9:46:03 ...	Hot (Inferred)		Block blob	18.56 KiB	Available
03.jpg	11/23/2021, 9:46:03 ...	Hot (Inferred)		Block blob	19.07 KiB	Available

Prepare Script

In this section, we will prepare a script and confirm success.

Create Script with Vim

Execute the following command to open a text editor:

```
vim script.sh
```

In the editor, type **i** to change to **INSERT** mode.

Update and paste the following command lines into the text editor:

```
#!/bin/sh
./azcopy_linux_amd64_10.13.0/azcopy sync "/home/rchapler/images"
"https://<UseCase>sa.blob.core.windows.net/images?sp=rwdl&st=2021-11-23T20:23:01Z&se=2022-10-01T03:23:01Z&spr=https&sv=2020-08-04&sr=c&sig=U3o98QJ1NJ8wAldkjBBJv8Eken4N3R1rr50gLxsq6PE%3D" --recursive
```

```
#!/bin/sh
./azcopy_linux_amd64_10.13.0/azcopy sync "/home/rchapler/images" "https://practicumsa.blob.core.windows.net/images?sp=rwdl&st=2021-11-23T20:23:01Z&se=2022-10-01T03:23:01Z&spr=https&sv=2020-08-04&sr=c&sig=U3o98QJ1NJ8wAldkjBBJv8Eken4N3R1rr50gLxsq6PE%3D" --recursive
```

Switch to command mode by pressing the **ESC** key.

Press the : (colon) key to open the prompt bar in the bottom left corner of the window.

Type **x** after the colon and press the **Enter** key to save changes and exit.

Back in the Azure Cloud Shell, update and execute the following command to grant permissions to the new script:

```
chmod +x /home/rchapler/script.sh
```

Confirm Success

In the Cloud Shell, update and paste the following command line.

```
/home/rchapler/script.sh
```

Successful processing of the new script will produce a response like:

```
INFO: Any empty folders will not be processed, because source and/or destination doesn't have full  
folder support
```

```
Job cb947969-d113-624c-6ac6-56274d66baac has started  
Log file is located at: /home/rchapler/.azcopy/cb947969-d113-624c-6ac6-56274d66baac.log
```

```
4 Files Scanned at Source, 0 Files Scanned at Destination
```

```
Job cb947969-d113-624c-6ac6-56274d66baac Summary  
Files Scanned at Source: 4  
Files Scanned at Destination: 4  
Elapsed Time (Minutes): 0.0334  
Number of Copy Transfers for Files: 0  
Number of Copy Transfers for Folder Properties: 0  
Total Number Of Copy Transfers: 0  
Number of Copy Transfers Completed: 0  
Number of Copy Transfers Failed: 0  
Number of Deletions at Destination: 0  
Total Number of Bytes Transferred: 0  
Total Number of Bytes Enumerated: 0  
Final Job Status: Completed
```

Schedule Cron Job

In this section, we will schedule a cron job and confirm success.

Schedule Cron Job

In the Cloud Shell, execute the following command to **open the cron table** {i.e., job scheduler}:

```
crontab -e
```

In Crontab, update and paste the following command line.

```
* * * * * /home/rchapler/script.sh
```

(Partial) logic explanation:

- **Schedule** ... “* * * * *” (in this case, “every minute”)
- **Command** ... “/home/rchapler/script.sh”

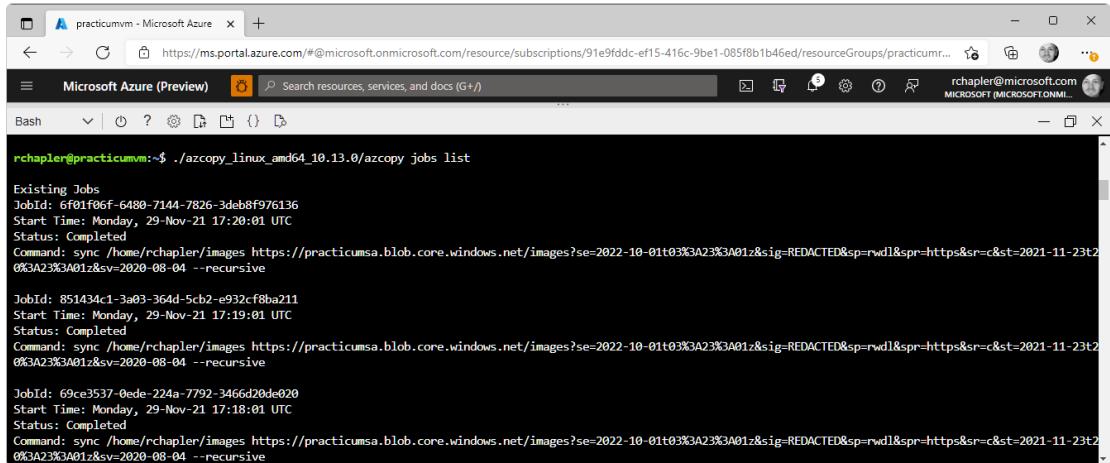
Press CTRL-X on the keyboard to exit crontab; save when prompted.

Back in the Cloud Shell, you should see the message “crontab: installing new crontab”

Confirm Success

In the Cloud Shell, update and execute the following command line to review recent runs of AzCopy.

```
./azcopy_linux_amd64_10.13.0/azcopy jobs list
```



The screenshot shows a Microsoft Azure Cloud Shell interface. The terminal window displays the command `./azcopy_linux_amd64_10.13.0/azcopy jobs list` and its output. The output lists three existing jobs, each with a unique JobId, start time, status (Completed), and the command used to sync files from a local directory to a blob storage endpoint. The most recent execution is at the top of the list.

```
rchapler@practicumvm:~$ ./azcopy_linux_amd64_10.13.0/azcopy jobs list

Existing Jobs
JobId: 6f01f06f-6480-7144-7826-3deb8f976136
Start Time: Monday, 29-Nov-21 17:20:01 UTC
Status: Completed
Command: sync /home/rchapler/images https://practicumsa.blob.core.windows.net/images?se=2022-10-01t03%3A23%3A01z&sig=REDACTED&sp=rwdl&spr=https&sr=c&st=2021-11-23t0%3A23%3A01z&sv=2020-08-04 --recursive

JobId: 851434c1-3a03-364d-5cb2-e932cf8ba211
Start Time: Monday, 29-Nov-21 17:19:01 UTC
Status: Completed
Command: sync /home/rchapler/images https://practicumsa.blob.core.windows.net/images?se=2022-10-01t03%3A23%3A01z&sig=REDACTED&sp=rwdl&spr=https&sr=c&st=2021-11-23t0%3A23%3A01z&sv=2020-08-04 --recursive

JobId: 69ce3537-0ede-224a-7792-346620de020
Start Time: Monday, 29-Nov-21 17:18:01 UTC
Status: Completed
Command: sync /home/rchapler/images https://practicumsa.blob.core.windows.net/images?se=2022-10-01t03%3A23%3A01z&sig=REDACTED&sp=rwdl&spr=https&sr=c&st=2021-11-23t0%3A23%3A01z&sv=2020-08-04 --recursive
```

More recent executions are shown first {i.e., you will want to scroll the top of the results to see the most recent execution}.

Confirm Synchronization

Execute the following command to get a new sample image from the traffic cam source:

```
wget -O images/10.jpg https://f1511.com/map/cctv/1139--9
```

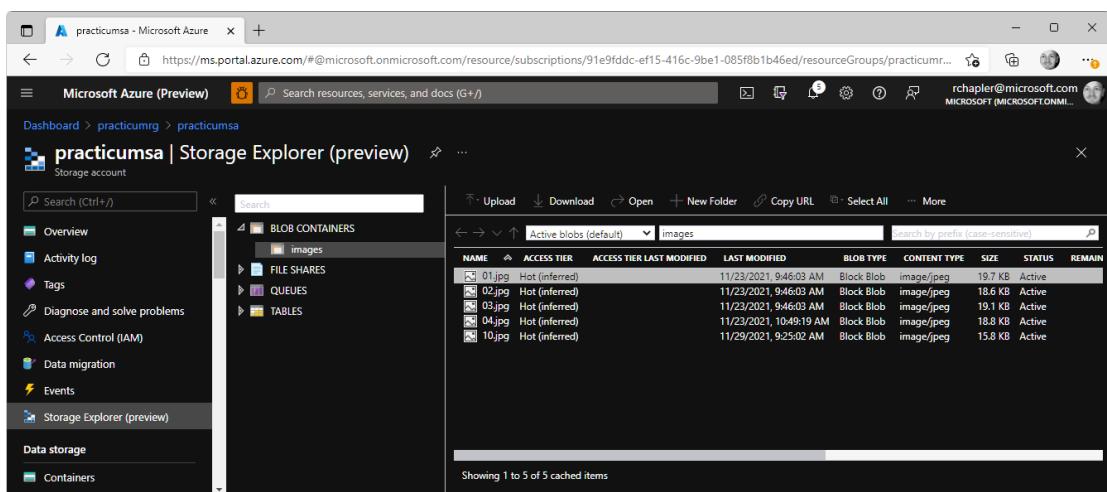
You can expect a response like:

```
--2021-11-29 17:24:45-- https://f1511.com/map/cctv/1139--9
Resolving f1511.com (f1511.com)... 172.67.4.44, 104.22.25.76, 104.22.24.76, ...
Connecting to f1511.com (f1511.com)|172.67.4.44|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 16149 (16K) [image/jpeg]
Saving to: 'images/10.jpg'

images/10.jpg
100%[=====] 15.77K --.-.
KB/s in 0.06s

2021-11-29 17:24:45 (251 KB/s) - 'images/10.jpg' saved [16149/16149]
```

After waiting a minute for the scheduled synchronization, navigate to your storage account, and then **Storage Explorer**.



The screenshot shows the Microsoft Azure Storage Explorer interface. On the left, there's a sidebar with various navigation options like Overview, Activity log, Tags, Diagnose and solve problems, Access Control (IAM), Data migration, Events, and Storage Explorer (preview). Under Storage storage, it lists Containers, Queues, and Tables. The main area shows a tree view of blob containers: BLOB CONTAINERS > images. Below this, a table lists Active blobs (default) with the following data:

NAME	ACCESS TIER	ACCESS TIER LAST MODIFIED	LAST MODIFIED	BLOB TYPE	CONTENT TYPE	SIZE	STATUS	REMAIN
01.jpg	Hot (inferred)	11/23/2021, 9:46:03 AM	11/23/2021, 9:46:03 AM	Block Blob	image/jpeg	19.7 KB	Active	
02.jpg	Hot (inferred)	11/23/2021, 9:46:03 AM	11/23/2021, 9:46:03 AM	Block Blob	image/jpeg	18.6 KB	Active	
03.jpg	Hot (inferred)	11/23/2021, 9:46:03 AM	11/23/2021, 9:46:03 AM	Block Blob	image/jpeg	19.1 KB	Active	
04.jpg	Hot (inferred)	11/23/2021, 10:49:19 AM	11/23/2021, 10:49:19 AM	Block Blob	image/jpeg	18.8 KB	Active	
10.jpg	Hot (inferred)	11/29/2021, 9:25:02 AM	11/29/2021, 9:25:02 AM	Block Blob	image/jpeg	15.8 KB	Active	

At the bottom, it says "Showing 1 to 5 of 5 cached items".

You should see the new file {i.e., “10.jpg”} included in the **images** container.

On-Prem to On-Prem

Follow these instructions to migrate files using Azure File Sync and a storage account.

Use Case

Work with your customer to understand and document requirements; example notes:

- Implementing a third-party Content Management System (CMS)
- Must migrate **petabytes of unstructured data**, including media, image, and Portable Document Format (PDF) files
- **Source** server is an **on-prem** server, and **destination** server is a **virtual machine**
- It is critical that **file metadata remains unchanged** during migration {e.g., Created On datetime}

Prerequisites

This solution requires the following resources:

- File Sync (aka Storage Sync Services)
- Storage Account
- Virtual Machine 1... to mimic the customers' source server (on-prem)
- Virtual Machine 2... to mimic the customers' destination server (virtual machine)

Follow the instructions in the “Infrastructure-as-Code” section to create templates, create a pipeline using the following YAML, and then deploy necessary pre-requisites:

```
trigger:
  branches:
    include:
      - refs/heads/main
  paths:
    include:
      - ARMTemplates
  jobs:
    - job: sa
      displayName: 'Instantiate Storage Account'
      pool:
        vmImage: windows-latest
      steps:
        - task: AzureResourceManagerTemplateDeployment@3
          inputs:
            ConnectedServiceName: a775d3ec-e4f1-46ed-905b-ddd26a59356c
            subscriptionName: 91e9fddc-ef15-416c-9be1-085f8b1b46ed
            resourceGroupName: <UseCase>
            location: West US
            csmFile: ARM Templates/StorageAccount.json
    - job: sss
      displayName: 'Instantiate File Sync (aka Storage Sync Service)'
      pool:
        vmImage: windows-latest
      steps:
        - task: AzureResourceManagerTemplateDeployment@3
          inputs:
            ConnectedServiceName: a775d3ec-e4f1-46ed-905b-ddd26a59356c
            subscriptionName: 91e9fddc-ef15-416c-9be1-085f8b1b46ed
            resourceGroupName: <UseCase>
```

```

location: West US
csmFile: ARM Templates/FileSync.json
- job: vm1
displayName: 'Instantiate Virtual Machine #1'
pool:
vmImage: windows-latest
steps:
- task: AzureResourceManagerTemplateDeployment@3
displayName: 'ARM Template deployment: <UseCase>vm01'
inputs:
ConnectedServiceName: a775d3ec-e4f1-46ed-905b-ddd26a59356c
subscriptionName: 91e9fddc-ef15-416c-9be1-085f8b1b46ed
resourceGroupName: <UseCase>vm01
location: West US
csmFile: ARM Templates/VirtualMachine.json
- job: vm2
displayName: 'Instantiate Virtual Machine #2'
pool:
vmImage: windows-latest
steps:
- task: AzureResourceManagerTemplateDeployment@3
displayName: 'ARM Template deployment: <UseCase>vm02'
inputs:
ConnectedServiceName: a775d3ec-e4f1-46ed-905b-ddd26a59356c
subscriptionName: 91e9fddc-ef15-416c-9be1-085f8b1b46ed
resourceGroupName: <UseCase>vm02
location: West US
csmFile: ARM Templates/VirtualMachine.json

```

Pipelines - Run 20211222.3

#20211222.3 Update Objective - Synchronize Unstructured Data, On-Prem to On-Prem

Manually run by Rich Chapter

Repository and version

- infrastructure
- main → 067bd41e

Time started and elapsed

- Today at 5:32 AM
- 6m 53s

Related

- 0 work items
- 0 artifacts

Tests and coverage

- Get started

Name	Status	Duration
Instantiate Storage Account	Queued	
Instantiate File Sync (aka Storage Sync Service)	Queued	
Instantiate Virtual Machine #1	Success	3m 29s
Instantiate Virtual Machine #2	Running	2m 39s

Install Agent

Navigate to the first virtual machine, click the **Connect** button, and select **RDP** from the resulting drop-down menu.

On the resulting “**Connect with RDP**” form, click the “**Download RDP File**” button.

Click the “**Open file**” link under your RDP file in the **Downloads** drop-down menu.

Click the **Connect** button on the resulting “**Remote Desktop Connection**” pop-up.

In the “**Windows Security**” pop-up, click the “**More choices**” link, then “**Use a different account**” in the resulting choices, then click **OK**.

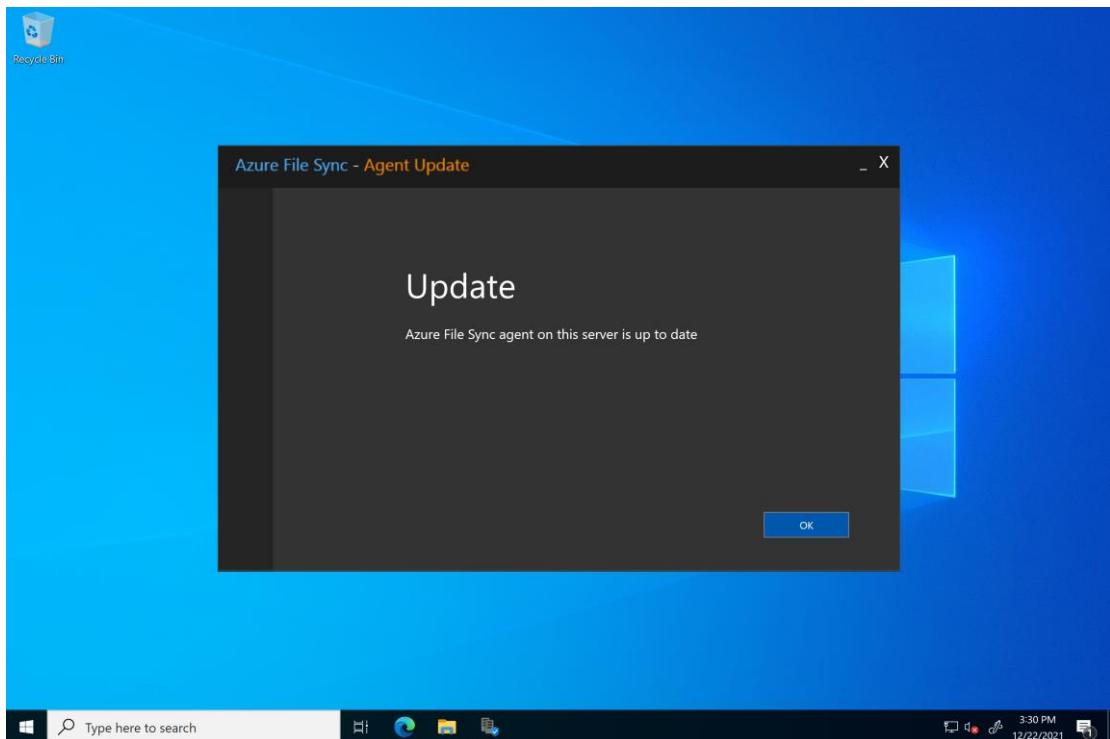
Back at the top of the “**Windows Security**” pop-up, enter credentials included in the YAML template.

Note: “User name” should include the name of the server {e.g., “<UseCase>vm01\Admin123”}.

Once connected to the virtual machine, browse to [Download Azure File Sync Agent from Official Microsoft Download Center](#).

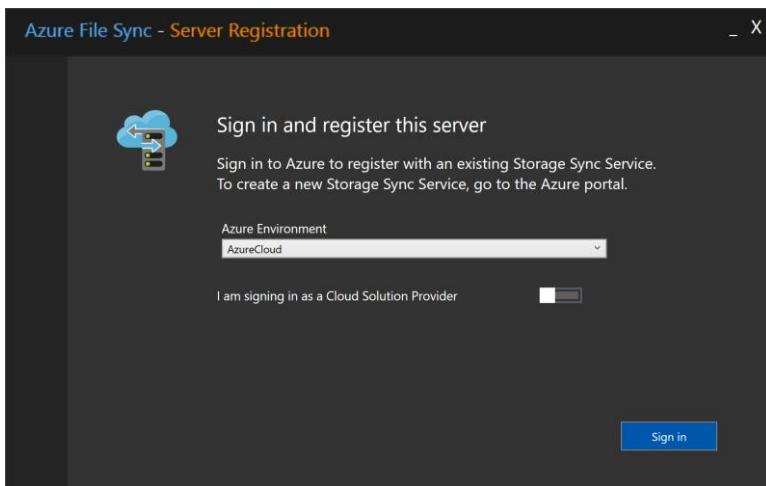
Click the **Download** button, and select the appropriate download {e.g., **StorageSyncAgent_WS2022.msi**} and click the **Next** button.

Open the downloaded file and complete the installation.



The Azure File Sync, server registration utility will open.

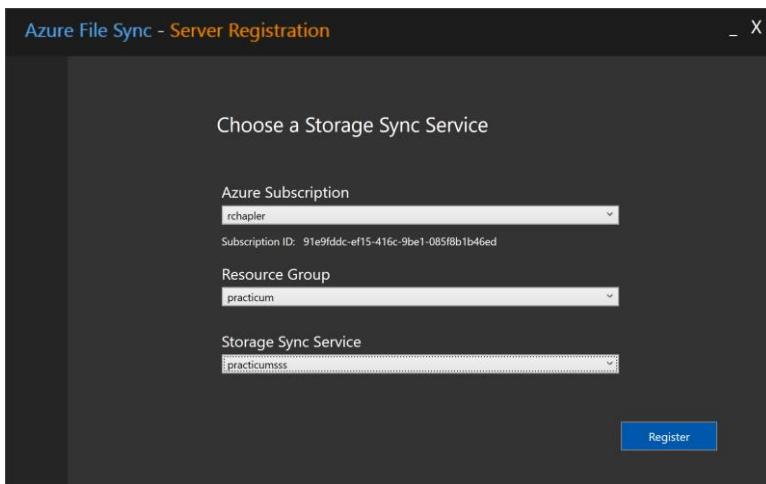
Click the OK button on the Update page.



Confirm “**Azure Environment**” selection and click the “**Sign in**” button on the “**Sign in and register this server**” page.

Complete the sign in process.

Note: You will likely have to add trusted sites and repeat the sign in process since this is a new, unconfigured server.



Complete the “**Choose a Storage Sync Service**” form, including:

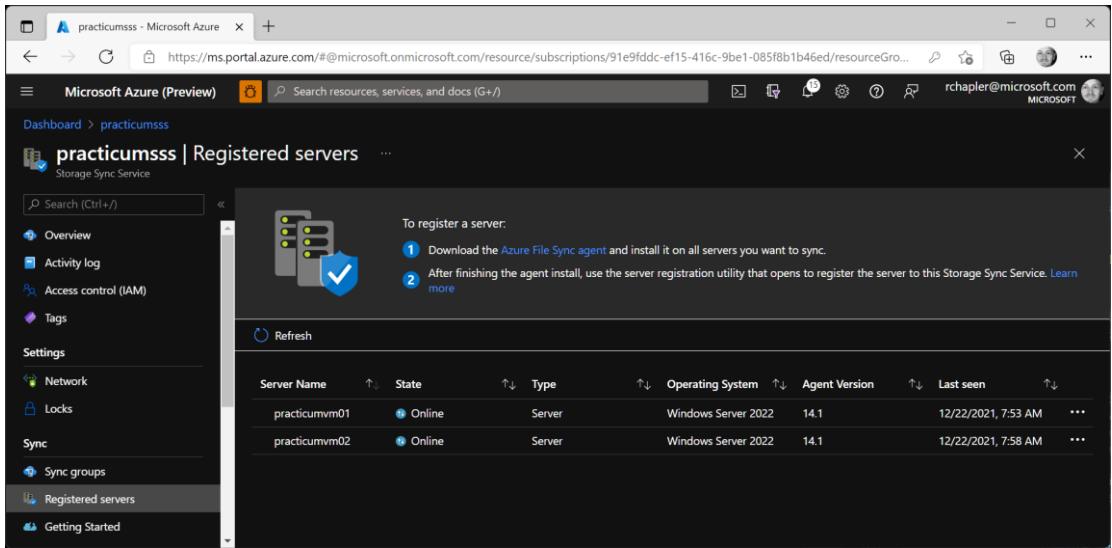
Storage Sync Service Select your storage sync service

Click the Register button and allow time for the “Network connectivity test” to reach “Passed. You are all set.”

Close (or minimize) the “**Remote Desktop Connection**” window.

Repeat this process for the second virtual machine.

Confirm Success



The screenshot shows the Microsoft Azure Storage Sync Service Registered servers page. The left sidebar includes options like Overview, Activity log, Access control (IAM), Tags, Settings (Network, Locks), Sync (Sync groups, Registered servers, Getting Started), and a Getting Started link. The main area displays a table of registered servers:

Server Name	State	Type	Operating System	Agent Version	Last seen	...
practicumvm01	Online	Server	Windows Server 2022	14.1	12/22/2021, 7:53 AM	...
practicumvm02	Online	Server	Windows Server 2022	14.1	12/22/2021, 7:58 AM	...

A sidebar on the right provides instructions for registering a server:

- Download the Azure File Sync agent and install it on all servers you want to sync.
- After finishing the agent install, use the server registration utility that opens to register the server to this Storage Sync Service. [Learn more](#)

Navigate to Storage Sync Service, then “**Registered Servers**” in the **Sync** group of the navigation pane.

You should see your two registered servers.

Create File Share

Open your Storage Account, then “**File Shares**” in the “**Data storage**” group of the navigation pane.

Click the “**+ File share**” button and enter a meaningful **Name** value in the resulting pop-out.

Click the **Create** button.

Create Sync Group

Navigate to Storage Sync Service, then “**Registered Servers**” in the **Sync** group of the navigation pane.

Click the “**+ Sync group**” button.

Complete the “**Choose a Storage Sync Service**” form, including:

Storage Account Click the “**Select storage account**” button and select your storage account on the resulting screen

Azure File Share Select your storage account, file share

Click the **Create** button.

Add Server Endpoint

Click into the new Sync Group and then click the “**Add server endpoint**” button at the top of the screen.

On the resulting “**Add server endpoint**” pop-out, including:

Registered Server Select your first virtual machine

Path Enter “**C:\Temp**”

Click the **Create** button and then repeat the process for the second virtual machine.

Confirm Success

Connect to your first virtual machine and then open File Explorer.

Navigate to C:\Temp, right-click and select New > Bitmap Image in the resulting drop-down menus.

Take note of the metadata {e.g., **Created** datetime}.

Connect to your second virtual machine and then open File Explorer.

You should see a synchronized copy of the “New Bitmap Image.bmp” file and metadata will match.

Repeat this exercise with a file update {e.g., rename} and a file delete.

Deployment

The following sections describe instantiation and deployment of the Azure resources necessary for code deployment.

Some Basics

Before we dive into full deployment of a data solution, some basics must be covered:

- Branches
- Pull Requests
- ARM Templates
- Schema Comparison

This solution requires the following resources:

- Service Principal
- A code repository... DevOps (with project and repository) or GitHub (with repository)
- Synapse or Data Factory (with configured repository)

Branches

Every developer should begin work on an assigned Task with the creation of a branch {e.g., “rchapler_12345” where “rchapler” refers to the developer alias and “12345” refers to the Task Identifier} based on the latest version of the code in the collaboration branch {e.g., “development”}.

Branches can be created using DevOps, GitHub, Synapse, or Data Factory. For this exercise, we will create a branch using Synapse.

Navigate to Synapse Studio, click the **Integrate** navigation icon, click the branch drop-down menu and select “**+ New branch**.”

On the resulting “Create a new branch” pop-up, including:

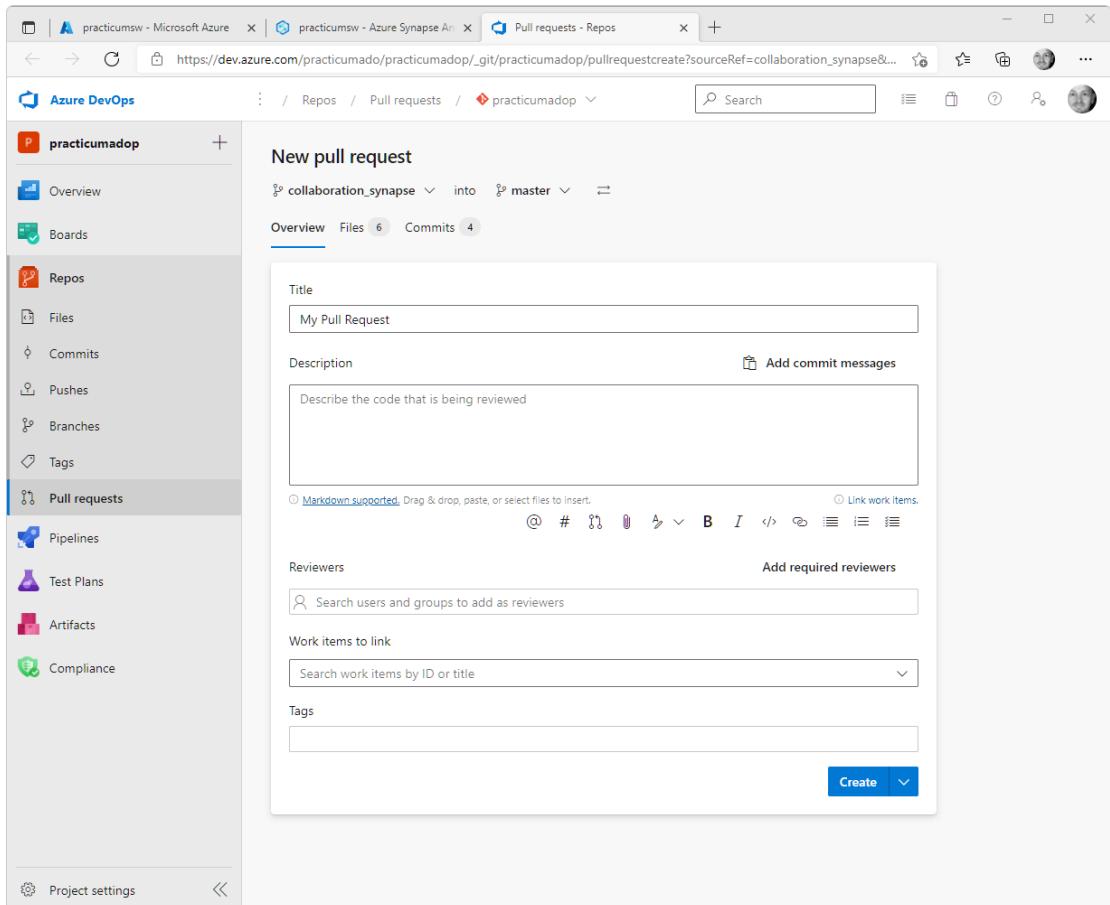
Branch Name	Consider a branch name that will inform the Development and Deployment Managers {e.g., begin with an email alias like “rchapler”, include a timestamp like “20220601”, and maybe include a Task Identifier like “12345”}
Base On	Select the appropriate starting point for work on the assigned task; in most cases Developers will select the collaboration branch

Click the **Create** button.

To follow the subsequent instructions for creating a Pull Request, consider making a minor change to the code in your branch.

Pull Requests

Navigate to Synapse Studio, click the **Integrate** navigation icon, click the branch drop-down menu and select “**Create pull request.**” A third browser tab (pointing to <https://dev.azure.com...>) will open.

A screenshot of the Azure DevOps interface showing the 'New pull request' form. The left sidebar shows the project structure with 'practicumadop' selected. The main area displays the 'New pull request' form with the following fields filled:

- Title:** My Pull Request
- Description:** Describe the code that is being reviewed
- Branches:** collaboration_synapse into master
- Reviewers:** Search users and groups to add as reviewers
- Work items to link:** Search work items by ID or title
- Tags:** (empty)

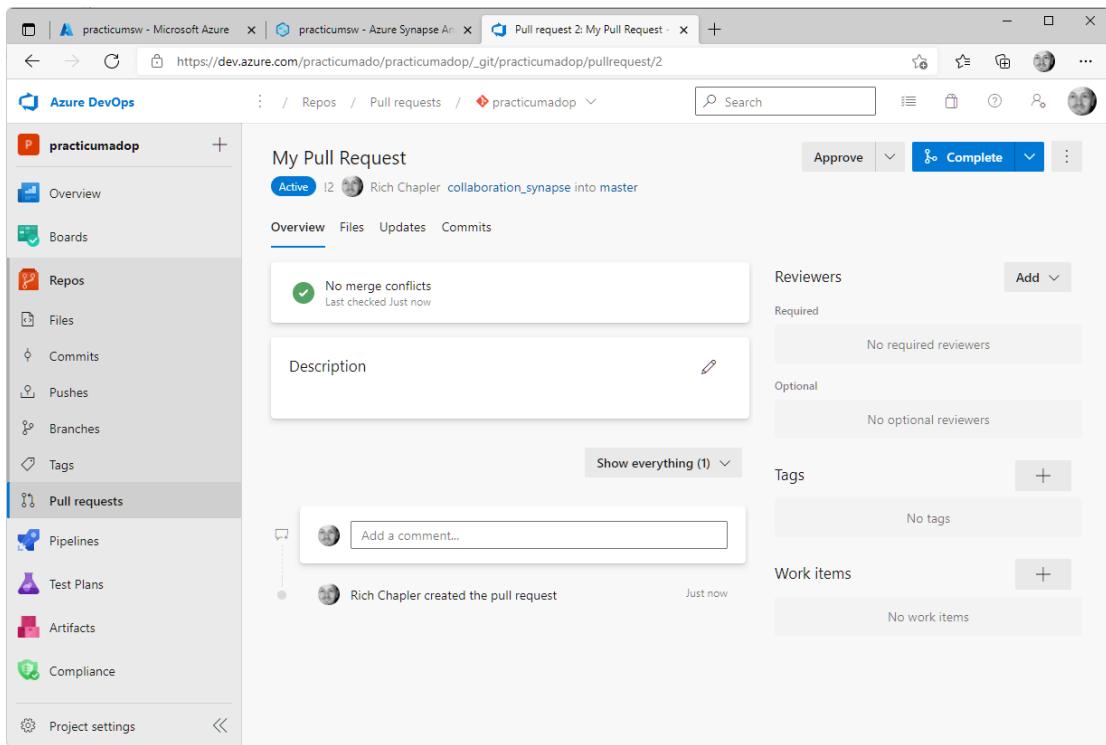
A 'Create' button is at the bottom right of the form.

On the second “**New pull request**” form, including:

Branches	{collaboration branch} into {master branch} ... you may have to create a master branch if one does not exist
Title	Enter a title that is meaningful for you (and aligned with your naming standards)

Including items like **Reviewers** and “**Work items...**” as appropriate.

Click the **Create** button.



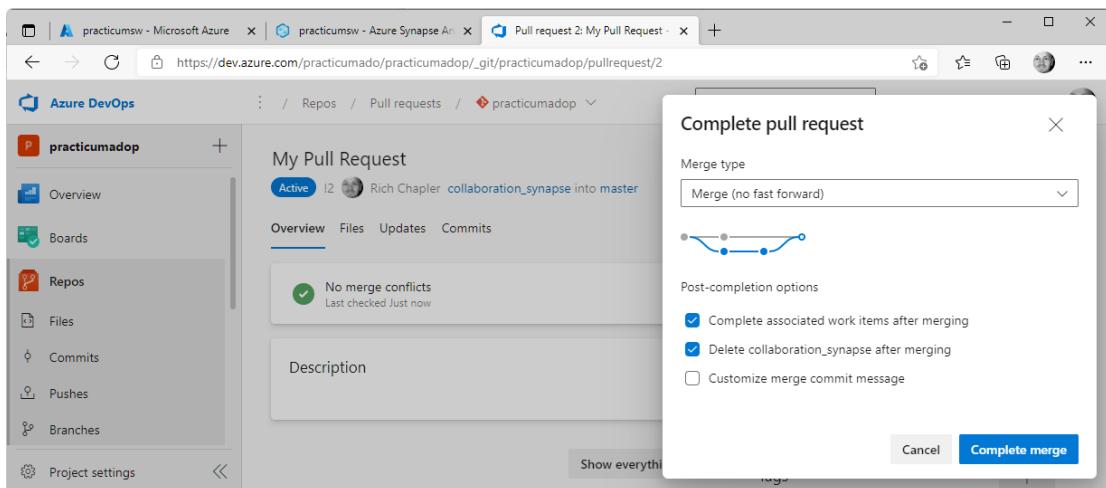
The screenshot shows the 'My Pull Request' page in Azure DevOps. The left sidebar is titled 'practicumadop' and includes 'Overview', 'Boards', 'Repos' (selected), 'Files', 'Commits', 'Pushes', 'Branches', 'Tags', 'Pull requests' (selected), 'Pipelines', 'Test Plans', 'Artifacts', 'Compliance', and 'Project settings'. The main content area is titled 'My Pull Request' for 'practicumadop' with the branch 'collaboration_synapse' into 'master'. It shows an 'Active' status with 12 commits by Rich Chapler. Below this are tabs for 'Overview' (selected), 'Files', 'Updates', and 'Commits'. A summary box says 'No merge conflicts Last checked Just now'. To the right are sections for 'Reviewers' (Required: No required reviewers, Optional: No optional reviewers), 'Tags' (No tags), and 'Work items' (No work items). A comment input field says 'Add a comment...'. A message at the bottom says 'Rich Chapler created the pull request Just now'.

At this point (and if specified), designated reviewers would be notified by email, and they will need to review and approve for the Pull Request to move forward.

Additional configuration can provide for validation of other gating criteria {e.g., inclusion of text patterns that might be secret like an account key}.

We have not included gating factors in this example.

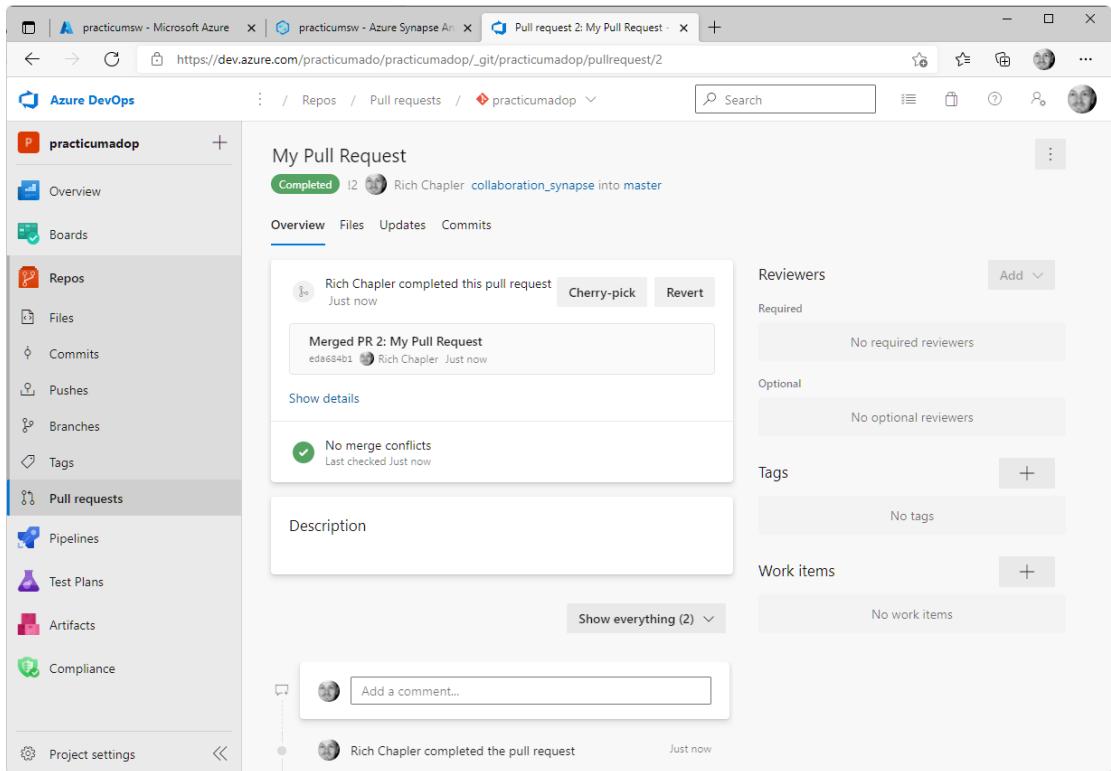
Click the **Complete** button.



The screenshot shows the 'My Pull Request' page with the 'Complete' button highlighted. A modal dialog titled 'Complete pull request' is open. It has a 'Merge type' dropdown set to 'Merge (no fast forward)'. Below it is a diagram showing a merge commit between two branches. Under 'Post-completion options', there are three checkboxes: 'Complete associated work items after merging' (checked), 'Delete collaboration_synapse after merging' (checked), and 'Customize merge commit message' (unchecked). At the bottom are 'Cancel' and 'Complete merge' buttons.

Confirm selections in the “**Complete pull request**” pop-out.

Click the “**Complete merge**” button.



The screenshot shows the Azure DevOps interface for a pull request. The left sidebar is titled "practicumadop" and includes options like Overview, Boards, Repos, Pull requests (which is selected), Pipelines, Test Plans, Artifacts, and Compliance. The main content area is titled "My Pull Request" and shows a completed merge from "Rich Chapler" to "collaboration_synapse" into "master". Below this, there are sections for "Reviewers" (Required: No required reviewers; Optional: No optional reviewers), "Tags" (No tags), and "Work items" (No work items). A "Description" field is present with a "Show everything (2) ▾" button. At the bottom, a comment input field says "Add a comment..." and a log entry states "Rich Chapler completed the pull request Just now".

Note: Post-completion options such as “Cherry-pick” and Revert provide for operational control even after a code merge.

ARM Templates

In this section, we will prepare a bare-minimum template, and then manually deploy that template.

Create Template

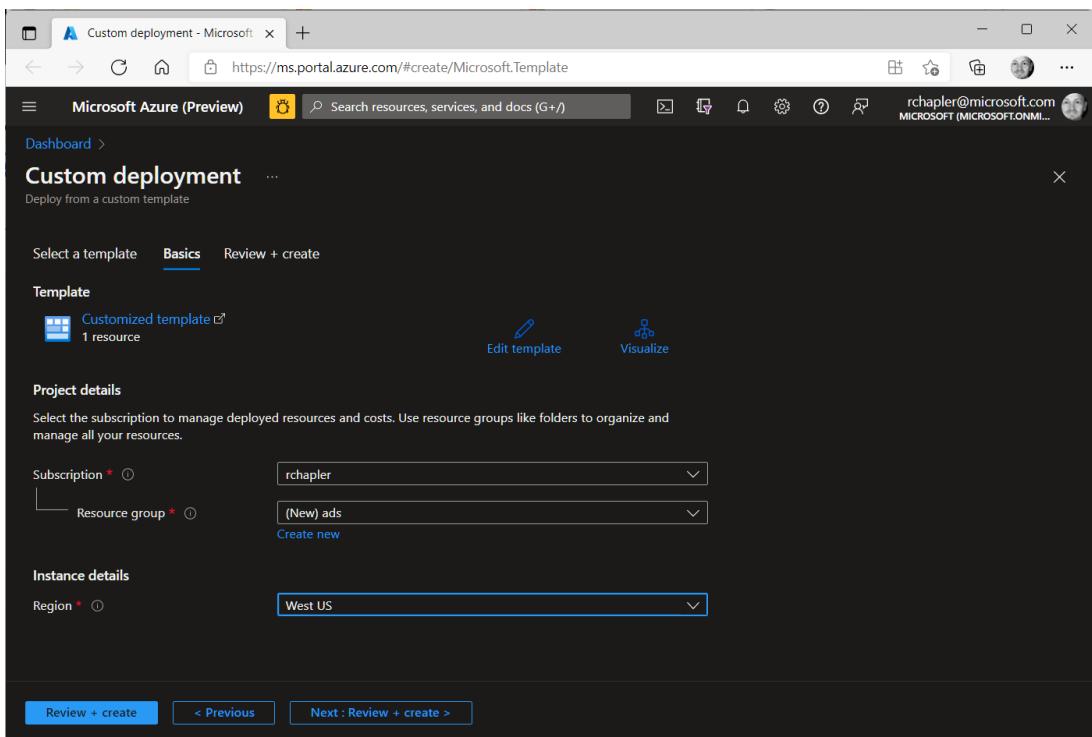
Navigate to <https://ms.portal.azure.com/#create/Microsoft.Template> and then click on the “**Build your own template in the editor**” link. On the resulting “**Edit template**” screen, replace the default JSON with the following prepared, “bare minimum” JSON:

```
{  
  "$schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",  
  "contentVersion": "1.0.0.0",  
  "resources": [  
    {  
      "type": "Microsoft.Storage/storageAccounts",  
      "apiVersion": "[providers('Microsoft.Storage','storageAccounts').apiVersions[0]]",  
      "name": "[concat(resourceGroup().name,'sa')]",  
      "location": "[resourceGroup().location]",  
      "sku": { "name": "Standard_LRS" },  
      "kind": "StorageV2"  
    }  
  ]  
}
```

(Partial) logic explanation:

[concat...]	Concatenation of the Resource Group name and acronym for the resource {i.e., “sa” for storage account}
Location	Use of “[resourceGroup().location]” ensures common location specification
AccountType	“Standard_LRS” is the cheapest option (and fine for demonstration) Every resource will have specific properties and codes best derived from online documentation

Review the pasted JSON and then click the **Save** button.



The screenshot shows the 'Custom deployment' page in the Microsoft Azure portal. The 'Basics' tab is selected. In the 'Template' section, there is a 'Customized template' card with one resource. Below it, under 'Project details', the subscription is set to 'rchapler' and the resource group is '(New) ads'. Under 'Instance details', the region is set to 'West US'. At the bottom, there are navigation buttons: 'Review + create' (highlighted in blue), '< Previous', and 'Next : Review + create >'.

Complete the “Custom deployment” form, including:

Resource Group Select existing or “**Create new**”

Region Select a region appropriate for your situation

Click the “**Review + create**” button, validate settings, and then click the **Create** button. When the deployment is complete, click the “**Go to resource**” button to review the created Storage Account.

Export Template

In this section, we will export the deployed template to see what Azure adds.

Navigate to the newly created Storage Account, and then click “**Export template**” in the **Automation** group of the navigation pane.

```
$schema: "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",
contentVersion: "1.0.0.0",
parameters: {
  "storageAccounts_adssa_name": {
    "defaultValue": "adssa",
    "type": "String"
  }
},
variables: {},
resources: [
  {
    "type": "Microsoft.Storage/storageAccounts",
    "apiVersion": "2021-08-01",
    "name": "[parameters('storageAccounts_adssa_name')]",
    "location": "westus",
    "sku": {
      "name": "Standard_LRS",
      "tier": "Standard"
    },
    "properties": {
      "blobService": {
        "deleteRetentionPolicy": {
          "enabled": true,
          "days": 30
        }
      },
      "fileService": {
        "deleteRetentionPolicy": {
          "enabled": true,
          "days": 30
        }
      },
      "queueService": {
        "deleteRetentionPolicy": {
          "enabled": true,
          "days": 30
        }
      },
      "tableService": {
        "deleteRetentionPolicy": {
          "enabled": true,
          "days": 30
        }
      }
    }
  }
]
```

There is far more JSON than we included in the bare bones JSON!

Complete the following steps:

- Click the **Download** button and a file named “ExportedTemplate-ads.zip” will be written to the **Downloads** folder on your device
- Open this ZIP file and extract the “**template.json**” and “**parameters.json**” files
- Navigate to <https://ms.portal.azure.com/#create/Microsoft.Template>
- Click on the “**Build your own template in the editor**” link
- Click “**Load file**”
- Navigate to the “**template.json**” file and click the **Open** button

When you have successfully completed these steps, you will be able to review and re-use the exported template content.

Deploy Template

...via DevOps Pipeline

In this section, we will create, save, and run a DevOps Pipeline that deploys an ARM template.

Upload Template

Upload the ARM Template to the DevOps repo by completing the following steps:

- Navigate to DevOps, select your project, and then click on **Repos** in the navigation pane
- Click the + icon just to the right of your project name and select “**New repository**” from the resulting drop-down menu
- In the resulting “**Create a repository**” pop-out, enter a “**Repository name**” {e.g., “infrastructure”} and then click the **Create** button
- Initialize the new repository with a ReadMe file; you can delete this file once additional files have been uploaded
- Click the vertical ellipses button in the upper-right of the screen, and then “**Upload file(s)**” in the resulting drop-down menu
- Drag-and-drop the **StorageAccount.json** template and click the **Commit** button

On successful completion, you can expect to receive an email reporting “**Build succeeded.**”

Navigate to your Resource Group and confirm Storage Account creation.

Create Pipeline

Navigate to DevOps, click on **Pipelines** in the navigation pane, and then click the “**New Pipeline**” button.

The screenshot shows the 'New pipeline' creation interface in Azure DevOps. On the left, the navigation pane is visible with 'Pipelines' selected. The main area has a heading 'Where is your code?' and a list of source options: 'Azure Repos Git (YAML)', 'Bitbucket Cloud (YAML) Hosted by Atlassian', 'GitHub (YAML) Home to the world's largest community of developers', 'GitHub Enterprise Server (YAML) The self-hosted version of GitHub Enterprise', 'Other Git Any generic Git repository', and 'Subversion Centralized version control by Apache'. A note at the bottom says 'Use the classic editor to create a pipeline without YAML.'

On the Connect tab {i.e., “New Pipeline ... Where is your code?”} form, click the “**Use classic editor**” link.

Complete the “**Select a source**” pop-out, select “**Azure Repos Git**” and then including:

Team Project	Select your DevOps project
Repository	Select your DevOps repository {e.g., infrastructure }
Default Branch...	Confirm default selection

Click the **Continue** button.

The screenshot shows the 'Select a build pipeline template' page. The left sidebar shows 'practicumadop' and other project options. The main area has a large right-pointing arrow icon and the text 'Choose a template'. To the right, there's a 'Select a template' section with a search bar and an 'Empty job' option, and a 'Configuration as code' section with a note about YAML files.

Click “Empty job” on the “Select a template” pop-out.

The screenshot shows the Azure DevOps Pipelines interface. On the left, the 'Pipelines' menu is selected. In the center, a pipeline named 'practicumadop-Cl' is displayed. The pipeline consists of two tasks: 'Get sources' and 'Agent job 1'. The 'Agent job 1' task is currently selected. On the right, there are configuration fields: 'Name' set to 'practicumadop-Cl', 'Agent pool' set to 'Azure Pipelines', and 'Agent Specification' set to 'windows-2019'. A section for 'Parameters' is present with a note that the pipeline doesn't have any parameters yet.

Confirm default values for **Name**, **Agent Pool**, and **Agent Specification**.

The screenshot shows the same Azure DevOps Pipelines interface as before, but with a different view. The 'Add tasks' dialog is open on the right, showing a search bar with 'arm' typed in. Below it, the 'Marketplace' tab is selected, displaying a list of extensions. The first extension listed is 'ARM template deployment' by Microsoft Corporation, which is described as deploying an ARM template to all deployment scopes. An 'Add' button is visible next to its description.

Click the “+” button to the right of “Agent Job 1.”

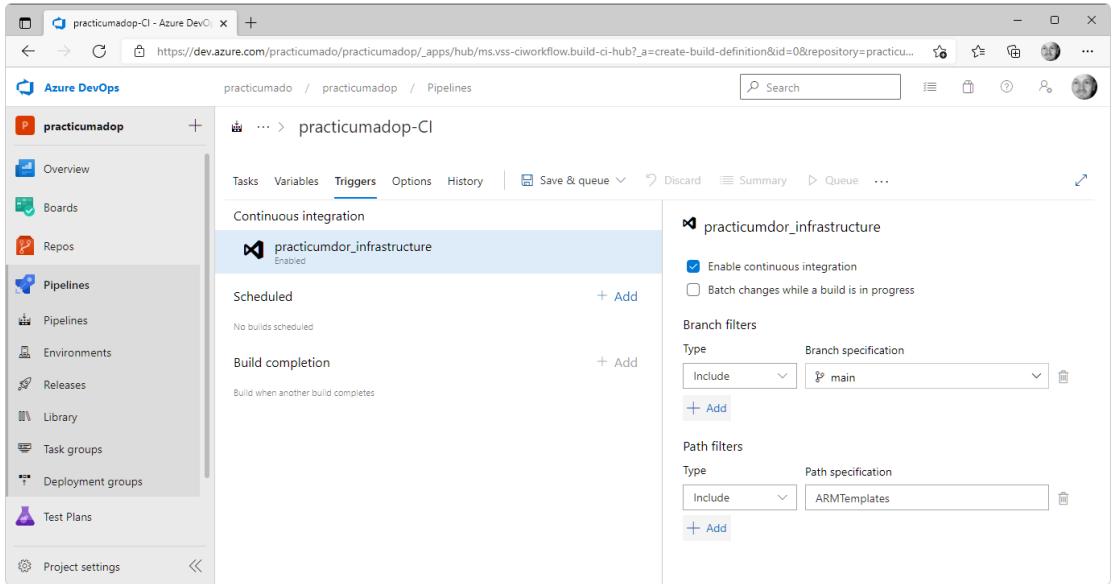
Search for and then select “**ARM template deployment**.”

Click the **Add** button.

Complete the “**ARM template deployment**” pop-out, including:

Deployment Scope	Confirm default, “ Resource Group ”
Action	Select “Create or update Resource Group”
Template	Browse to and select the “ StorageAccount.json ” template created in the prior section
Deployment Mode	Confirm selection, Incremental

Click on the **Triggers** tab.



Check the “**Enable continuous integration**” checkbox; new input controls will surface.

Branch Filter	Confirm defaults, Type: Include and Branch Specification: Main
Path Filter	Add Type: Include and Path Specification: “ ARMTemplates ”

Click on the “**Save & queue**” button, and then click “**Save & queue**” in the resulting drop-down menu.

Confirm values on the resulting “**Run pipeline**” pop-out, and then click the “**Save and run**” button.

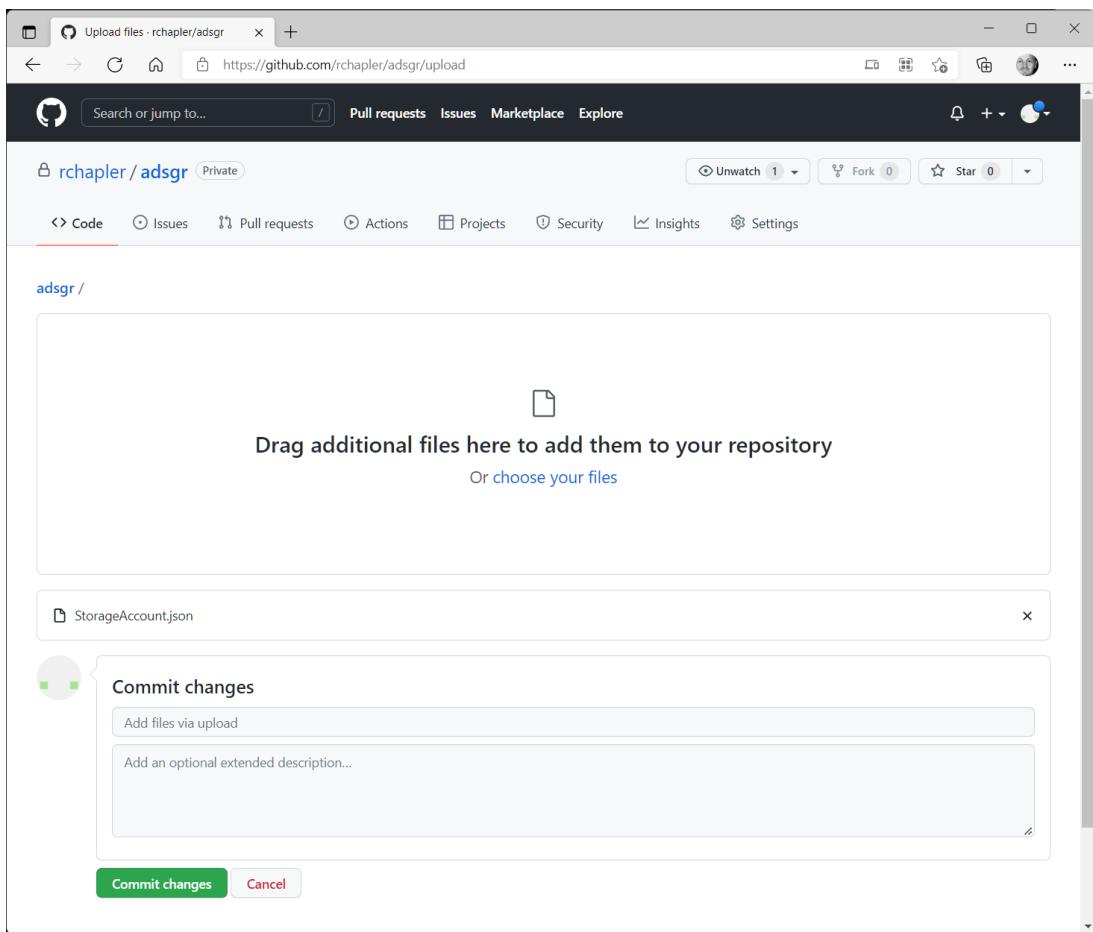
...via GitHub Action

In this section, we will create, save, and run a GitHub Action that deploys an ARM template.

Upload Template

Modify (with your values) and navigate to the following URL:

<https://github.com/rchapler/<UseCase>gr/upload>



Click the “choose your files” link, select the “StorageAccount.json” file created in [Create Template](#), and finally, click the “Commit changes” button at the bottom of the page.

Prepare Credentials

Use a text editor to modify the following string:

```
{  
  "clientId": "{App GUID, aka Client Id}",  
  "clientSecret": "{Password String, aka Client Secret "rbac"}",  
  "subscriptionId": "{SubscriptionId GUID}",  
  "tenantId": "{TenantId GUID}"  
}
```

You should use the values produced when you created the Service Principal.

Create Secret

Modify and navigate to the following URL:

[https://github.com/rchapler/shared/settings/secrets/actions.](https://github.com/rchapler/shared/settings/secrets/actions)

Click the “**New repository secret**” button.

Complete the “**Actions secrets / new secret**” form, including:

Value	Paste the credential string prepared in the prior step
	AZURELOGINCRED\$

The screenshot shows the GitHub Actions secrets creation interface. On the left, there's a sidebar with sections like General, Access, Collaborators, Code and automation (with Actions, Webhooks, Pages), Security, Code security and analysis, Deploy keys, and Secrets (which is currently selected). The main area is titled "Actions secrets / New secret". It has fields for "Name" (containing "AZURELOGINCRED\$") and "Value" (containing a JSON object). At the bottom right, there's a green "Add secret" button.

```
{  
  "clientId": "[App GUID, aka Client Id]",  
  "clientSecret": "[Password String, aka Client Secret \"dbac\"]",  
  "subscriptionId": "[Subscription GUID]",  
  "tenantId": "[Tenant GUID]"  
}
```

Click the “**Add secret**” button.

Repeat for “SUBSCRIPTIONID” secret.

Create Workflow

The screenshot shows the GitHub Actions interface for creating a new workflow. At the top, the URL is https://github.com/rchapler/adsgsr/actions/new?category=none&query=simple+workflow. The menu bar includes Pull requests, Issues, Marketplace, and Explore. Below the menu, the repository rchapler / adsgsr (Private) is selected. The Actions tab is highlighted. The main content area is titled "Get started with GitHub Actions" and contains a search bar with the query "simple workflow". It shows two workflow options: "Simple workflow" (By GitHub) and "Manual workflow" (By GitHub Actions). Both options have a "Configure" button. The footer includes links for Terms, Privacy, Security, Status, Docs, Contact GitHub, Pricing, API, Training, Blog, and About.

Click **Actions** in the menubar, search for “**Simple workflow**”, and then click the **Configure** button.

The screenshot shows a GitHub repository page for 'rchabler / adsgr'. The user is editing a file named 'rchabler20220217.yml' under the '.github/workflows' directory. The file contains a YAML-based GitHub Action workflow. On the right side of the editor, there is a sidebar titled 'Marketplace' which lists several actions available for use in the workflow.

```
name: rchabler20220217
on: [push, pull_request]
jobs:
  build:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v2
      - name: Azure Login
        uses: Azure/login@v1.1
        with:
          creds: ${{ secrets.AZURELOGINCREDS }}
      - name: Run ARM deploy
        uses: azure/arm-deploy@v1
        with:
          subscriptionId: "{Subscription GUID}"
          resourceGroupName: "ads"
          template: ./StorageAccount.json
```

Enter a meaningful name, then modify and paste the following YAML.

```
name: rchabler20220216
on: workflow_dispatch
jobs:
  build:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v2

      - name: AzureLogin
        uses: Azure/login@v1.1
        with:
          creds: ${{ secrets.AZURELOGINCREDS }}

      - name: DeployARM_StorageAccount
        uses: azure/arm-deploy@v1
        with:
          subscriptionId: "{Subscription GUID}"
          resourceGroupName: "rchabler"
          template: ./StorageAccount.json
```

Note: Your implementation of GitHub Enterprise may not have a ready image for “ubuntu-latest” and require an update or other work-around.

Click the “**Start commit**” button and then the “**Commit new file**” button on the resulting pop-up.

The screenshot shows a GitHub repository page for 'rchapler / adsgr'. The URL in the address bar is <https://github.com/rchapler/adsgr/tree/main/.github/workflows>. The page lists a single file, 'rchapler20220217.yml', which was created by 'rchapler' 13 seconds ago. The commit message is 'Create rchapler20220217.yml'. At the bottom of the page, there is a link to the raw YAML file: <https://github.com/rchapler/adsgr/blob/main/.github/workflows/rchapler20220217.yml>.

On the resulting form, click on the link for the committed YAML file.

The screenshot shows the GitHub Actions page for the 'rchapler / adsgr' repository. The URL in the address bar is <https://github.com/rchapler/adsgr/actions/workflows/rchapler20220217.yml>. The page shows the workflow 'rchapler20220216' with one workflow run. The run is labeled 'Create rchapler20220217.yml' and was triggered by commit 781490d pushed by 'rchapler' 7 minutes ago. The run status is 'In progress'.

Click on the run {i.e., “Create rchapler20220217.YAML”}.

You have successfully requested a re-run of the workflow run.

rchabler / adsgr Private

Code Issues Pull requests Actions Projects Security Insights Settings

Create rchabler20220217.yml rchabler20220216 #1

Cancel workflow Latest #3 ...

Summary

Re-run triggered now
rchabler -> 781490d main

Status Queued Total duration - Artifacts -

Jobs

build

rchabler20220217.yml
on: push

build

[+]

Click on the **build** box.

Update rchabler20220217.yml - rchabler20220217 #2

rchabler / adsgr Private

Code Issues Pull requests Actions Projects Security Insights Settings

Re-run all jobs ...

Summary

build succeeded now in 1m 17s

Search logs

build

Set up job

Run actions/checkout@v2

Azure Login

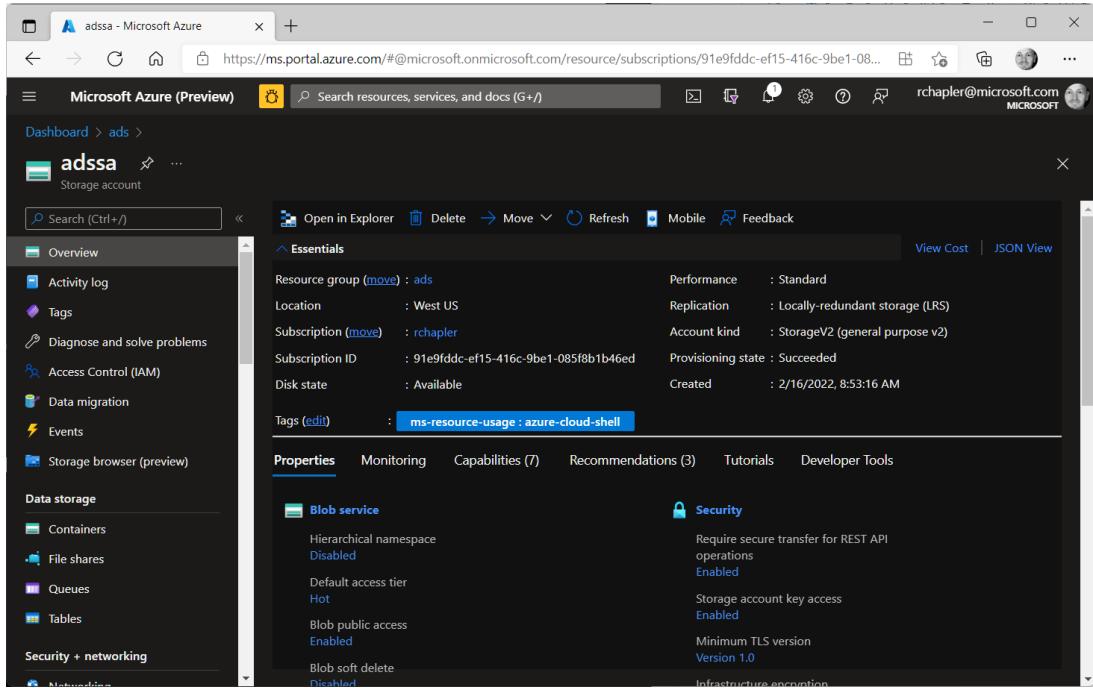
Run ARM deploy

Post Run actions/checkout@v2

Complete job

Confirm Success

We can confirm successful deployment (using either DevOps Pipeline or GitHub Action) simply by navigating to Azure and confirming that the Storage Account was created.



The screenshot shows the Microsoft Azure portal interface. The left sidebar navigation bar includes 'Dashboard', 'Overview', 'Activity log', 'Tags', 'Diagnose and solve problems', 'Access Control (IAM)', 'Data migration', 'Events', 'Storage browser (preview)', 'Data storage' (with 'Containers', 'File shares', 'Queues', 'Tables'), and 'Security + networking' (with 'Networking'). The main content area displays the 'adssa' storage account details under the 'Essentials' tab. Key information shown includes:

Resource group	ads	Performance	Standard
Location	West US	Replication	Locally-redundant storage (LRS)
Subscription	rchapler	Account kind	StorageV2 (general purpose v2)
Subscription ID	91e9fddc-ef15-416c-9be1-085f8b1b46ed	Provisioning state	Succeeded
Disk state	Available	Created	2/16/2022, 8:53:16 AM
Tags	ms-resource-usage : azure-cloud-shell		

The 'Properties' tab is selected, showing the Blob service configuration (Hierarchical namespace: Disabled, Default access tier: Hot, Blob public access: Enabled, Blob soft delete: Disabled) and the Security settings (Require secure transfer for REST API operations: Enabled, Storage account key access: Enabled, Minimum TLS version: Version 1.0, Infrastructure encryption).

Schema Comparison

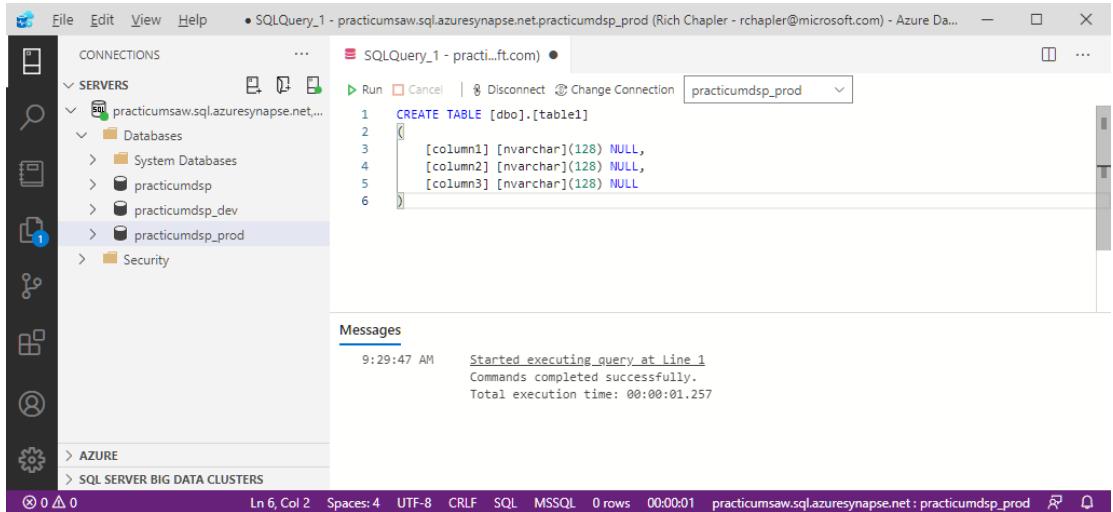
This solution requires the following resources:

- Data Studio
- Synapse (with two SQL pool; e.g., <UseCase>dev and <UseCase>test)

Although this example focuses on Synapse, the same solution applies to SQL.

Stage Resources

Open Azure Data Studio.



The screenshot shows the Azure Data Studio interface. The left sidebar has a 'CONNECTIONS' tree with a node for 'practicumsaw.sql.azuresynapse.net...'. The main area is a query editor titled 'SQLQuery_1 - practicumsaw.sql.azuresynapse.net.practicumdsp_prod'. It contains the following T-SQL code:

```
CREATE TABLE [dbo].[table1]
(
    [column1] [nvarchar](128) NULL,
    [column2] [nvarchar](128) NULL,
    [column3] [nvarchar](128) NULL
)
```

The 'Messages' pane at the bottom shows the execution log:

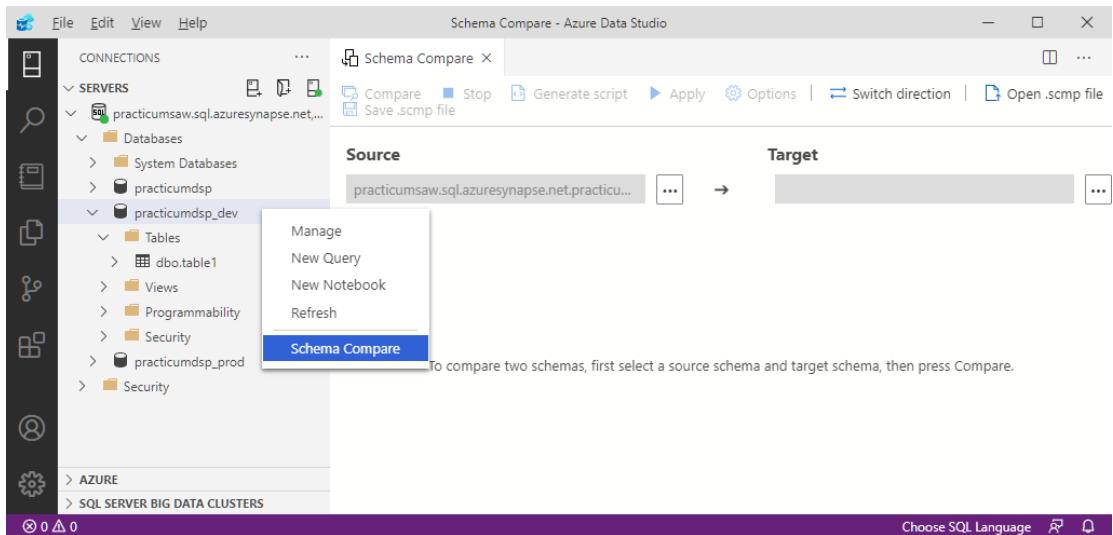
```
9:29:47 AM Started executing query at Line 1
Commands completed successfully.
Total execution time: 00:00:01.257
```

The status bar at the bottom indicates the query was run at Ln 6, Col 2, with 0 rows affected in 00:00:01.

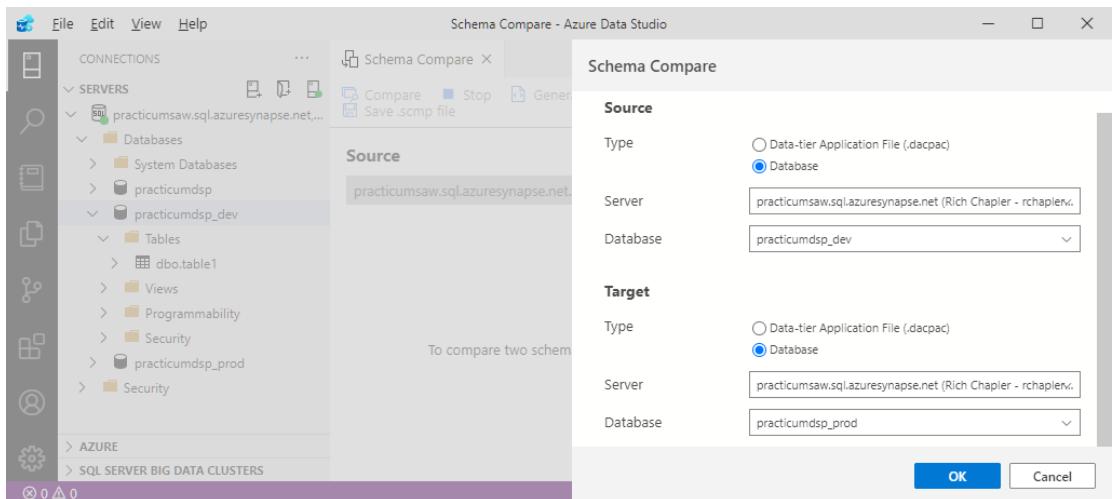
Execute the following T-SQL on <UseCase>dsp_dev:

```
CREATE TABLE [dbo].[table1]
(
    [column1] [nvarchar](128) NULL,
    [column2] [nvarchar](128) NULL,
    [column3] [nvarchar](128) NULL
)
```

Schema Compare



Right-click on the SQL Pool and click “Schema Compare” in the resulting drop-down menu.



Click the ellipses button.

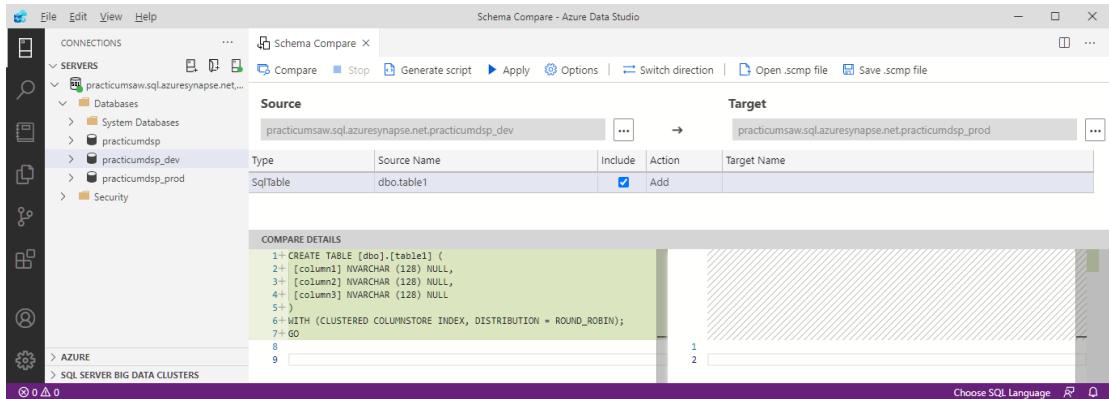
Complete the “Schema Compare” pop-out, including:

Source Type	Select the Database radio button
Source Server and Database	Confirm selection of your Synapse server and the <UseCase> dsp_dev database
Target Type	Select the Database radio button

Target | Server and Database

Confirm selection of your Synapse server and the <UseCase>**dsp_prod** database

Click the **OK** button.

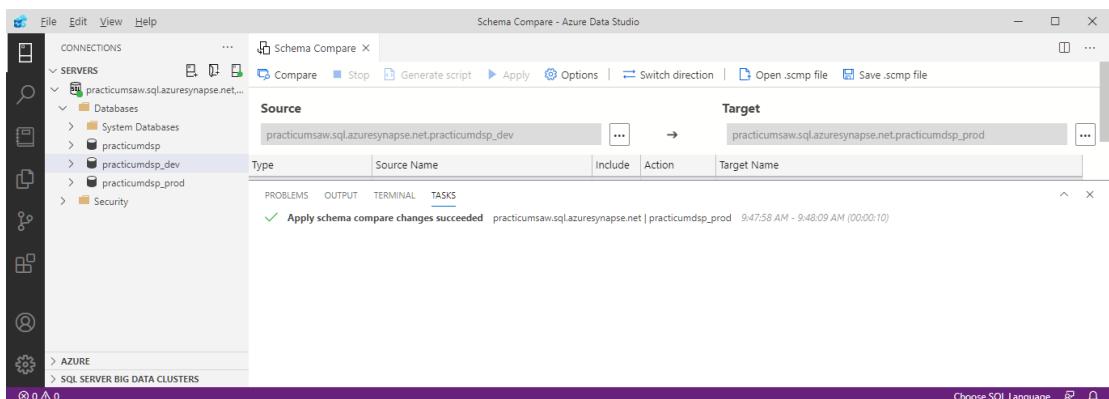


Complete the “Schema Compare” tab, click the **Compare** button.

We only created a table on the development database, so Schema Compare should surface that table as something missing from the production database.

Click on the row to see T-SQL for the identified item.

Click the **Apply** button to update the target.



Solution Deployment

Now that we have a grasp of the basics, we will discuss full data solution deployment.

Use Case

Example notes:

- “...deploy, **in measurable steps**, each of the core elements of our Azure Data Solution to the PROD environment”
- “...use **GitHub Actions** and establish a **repeatable template**”
- “...minimize code included in created templates”
- “...use programmatic rather than fixed values {e.g., the latest API version rather than a specific dated version} wherever possible”
- “...assume that all resources naming will follow the convention of {Resource Group name} + {acronym for resource type}”

Prerequisites

This solution requires the following resources:

- GitHub Enterprise (with repository and AZURELOGINCREDS secret)

Infrastructure

This section describes deployment and configuration of data solution infrastructure {e.g., resource group, data lake, synapse, etc.}. These tasks will be completed infrequently.

Resource Group

We will start by deploying a Resource Group because it is a **prerequisite for all other resources**.

Resource Groups must be deployed at the subscription-level {i.e., scope = “subscription”}.

Reference the detailed instructions in the “**Infrastructure-as-Code**” section, as required.

First, upload the following ARM Template JSON to GitHub (modified, as required).

```
{  
    "$schema": "https://schema.management.azure.com/schemas/2018-05-  
01/subscriptionDeploymentTemplate.json#",  
    "contentVersion": "1.0.0.0",  
    "parameters": { "ResourceGroupName": { "type": "string" } },  
    "resources": [  
        {  
            "type": "Microsoft.Resources/resourceGroups",  
            "apiVersion": "[providers('Microsoft.Resources','resourceGroups').apiVersions[0]]",  
            "name": "[parameters('ResourceGroupName')]",  
            "location": "eastus"  
        }  
    ]  
}
```

Second, create a secret for RESOURCEGROUPNAME_TEST and ..._PROD (where the suffix refers to the related environment). Repeat for SUBSCRIPTIONID_TEST and ...PROD.

Third, create a workflow using the following YAML (modified, as required):

```
name: Deploy Resource Group  
on: workflow_dispatch  
jobs:  
  build:  
    runs-on: self-hosted  
    steps:  
      - uses: actions/checkout@v2  
  
      - name: Login to Azure  
        uses: azure/login@v1.1  
        with:  
          creds: ${{ secrets.AZURELOGINCREDS }}  
  
      - name: Create Resource Group  
        uses: azure/arm-deploy@v1  
        with:  
          scope: subscription  
          subscriptionId: ${{ secrets.SUBSCRIPTIONID_TEST }}  
          region: "eastus"  
          template: ./arm/resourcegroup.json  
          parameters:  
            ResourceGroupName=${{ secrets.RESOURCEGROUPNAME_TEST }}
```

Fourth, run the workflow and monitor for success. When complete, confirm success by navigating to your TEST subscription and confirming that the Resource Group has been created.

Assuming success and when you are ready to proceed, repeat the process for the PROD.

Role Assignment

Repeat the process with the following ARM Template JSON:

```
{
    "$schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",
    "contentVersion": "1.0.0.0",
    "variables": {
        "RoleDefinitionId_Owner": "8e3af657-a8ff-443c-a75c-2fe8c4bcb635"
        , "RoleDefinitionId_Contributor": "b24988ac-6180-42a0-ab88-20f7382dd24c"
        , "ObjectId_GroupOrUser": "29757728-136d-4ab7-96ff-4f902f750975"
    },
    "resources": [
        {
            "type": "Microsoft.Authorization/roleAssignments",
            "apiVersion": "[providers('Microsoft.Authorization','roleAssignments').apiVersions[0]]",
            "name": "[guid(concat(variables('ObjectId_GroupOrUser'),'-',
',variables('RoleDefinitionId_Contributor')))]",
            "properties": {
                "roleDefinitionId": "[resourceId('Microsoft.Authorization/roleDefinitions',
variables('RoleDefinitionId_Contributor'))]",
                "principalId": "[variables('ObjectId_GroupOrUser')]"
            }
        }
    ]
}
```

Notes:

- The *RoleDefinitionId* for built-in role “Owner” is always *8e3af657-a8ff-443c-a75c-2fe8c4bcb635*
 - ...for role “Contributor” the guid is *b24988ac-6180-42a0-ab88-20f7382dd24c*
- This template assumes the existence of and GUID for a group or user {i.e., *PrincipalId*}

...and workflow YAML:

```
name: Deploy Role Assignment
on: workflow_dispatch
jobs:
  build:
    runs-on: self-hosted
    steps:
      - uses: actions/checkout@v2

      - name: Login to Azure
        uses: azure/login@v1.1
        with:
          creds: ${{ secrets.AZURELOGINCREDS }}

      - name: Create Role Assignment
        uses: azure/arm-deploy@v1
        with:
          subscriptionId: ${{ secrets.SUBSCRIPTIONID_TEST }}
          resourceName: ${{ secrets.RESOURCEGROUPNAME_TEST }}
          template: ./arm/roleassignment.json
```

Data Lake

Repeat the process with the following ARM Template JSON:

```
{  
  "$schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",  
  "contentVersion": "1.0.0.0",  
  "resources": [  
    {  
      "type": "Microsoft.Storage/storageAccounts",  
      "apiVersion": "[providers('Microsoft.Storage','storageAccounts').apiVersions[0]]",  
      "name": "[concat(resourceGroup().name,'dl')]",  
      "location": "[resourceGroup().location]",  
      "sku": { "name": "Standard_LRS" },  
      "kind": "StorageV2",  
      "properties": { "isHnsEnabled": true }  
    }  
  ]  
}
```

Notes:

- Create a data lake using the Storage Account template plus the **isHnsEnabled** property
- I am not bothering to consider Gen1 data lakes in my use of “dl” for the suffix

...and workflow YAML:

```
name: Deploy Data Lake  
on: workflow_dispatch  
jobs:  
  build:  
    runs-on: self-hosted  
    steps:  
      - uses: actions/checkout@v2  
  
      - name: Login to Azure  
        uses: azure/login@v1.1  
        with:  
          creds: ${{ secrets.AZURELOGINCREDS }}  
  
      - name: Create Data Lake (pre-requisite for Synapse)  
        uses: azure/arm-deploy@v1  
        with:  
          subscriptionId: ${{ secrets.SUBSCRIPTIONID_TEST }}  
          resourceGroupName: ${{ secrets.RESOURCEGROUPNAME_TEST }}  
          template: ./arm/datalake.json
```

File Share

Repeat the process with the following ARM Template JSON:

```
{  
  "$schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",  
  "contentVersion": "1.0.0.0",  
  "resources": [  
    {  
      "type": "Microsoft.Storage/storageAccounts/fileServices",  
      "apiVersion": "[providers('Microsoft.Storage','storageAccounts').apiVersions[0]]",  
      "name": "[concat(resourceGroup().name,'dl/default')]"  
    },  
    {  
      "type": "Microsoft.Storage/storageAccounts/fileServices/shares",  
      "apiVersion": "[providers('Microsoft.Storage','storageAccounts').apiVersions[0]]",  
      "name": "[concat(resourceGroup().name,'dl/default/',resourceGroup().name,'dlfs')]",  
      "dependsOn": [  
        "[ resourceId('Microsoft.Storage/storageAccounts/fileServices',concat(resourceGroup().name,'dl'),'default')]"  
      ]  
    }  
  ]  
}
```

Note: This template is dependent on pre-instantiation of a Data Lake

...and workflow YAML:

```
name: Deploy File Share  
on: workflow_dispatch  
jobs:  
  build:  
    runs-on: self-hosted  
    steps:  
      - uses: actions/checkout@v2  
  
      - name: Login to Azure  
        uses: azure/login@v1.1  
        with:  
          creds: ${{ secrets.AZURELOGINCREDS }}  
  
      - name: Create Data Lake, File Share (pre-requisite for Synapse)  
        uses: azure/arm-deploy@v1  
        with:  
          subscriptionId: ${{ secrets.SUBSCRIPTIONID_TEST }}  
          resourceGroupName: ${{ secrets.RESOURCEGROUPNAME_TEST }}  
          template: ./arm/datalake_fileshare.json
```

Synapse

Notes:

- This section depends on pre-instantiation of a Data Lake and a code repository (DevOps or GitHub)
- To instantiate Synapse on a new subscription, we must ensure that Subscription > Resource Provider > Microsoft.SQL has been registered... logic for this is baked into the “**All-In YAML Example**” section
- Parameters for “ResourceID_Administrator”, “SynapseSQL_AdministratorName”, and “SynapseSQL_AdministratorPassword”... these values should be passed in from the deployment template (and captured there as secrets)
- “workspaceRepositoryConfiguration” {i.e., DevOps / Git configuration} is **not included** because it is not possible to programmatically set the related permissions {e.g., GitHub Personal Access Token} ... I am treating this entire, particularly key step, as manual-by-necessity
- If you disable “publicNetworkAccess”, you must also provide VNET configuration

```
{  
    "$schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",  
    "contentVersion": "1.0.0.0",  
    "parameters":  
    {  
        "SynapseSQL_AdministratorName": { "type": "string" },  
        "SynapseSQL_AdministratorPassword": { "type": "secureString" }  
    },  
    "resources": [  
        {  
            "type": "Microsoft.Synapse/workspaces",  
            "apiVersion": "[providers('Microsoft.Synapse', 'workspaces').apiVersions[0]]",  
            "name": "[concat(resourceGroup().name,'s')]",  
            "location": "[resourceGroup().location]",  
            "identity": { "type": "SystemAssigned" },  
            "properties": {  
                "defaultDataLakeStorage": {  
                    "resourceId":  
                        "[resourceId('Microsoft.Storage/storageAccounts',concat(resourceGroup().name,'dl'))]",  
                    "createManagedPrivateEndpoint": false,  
                    "accountUrl": "[concat('https://',resourceGroup().name,'dl.dfs.core.windows.net')]",  
                    "filesystem": "concat(resourceGroup().name,'dlfs')"  
                },  
                "connectivityEndpoints": {  
                    "web":  
                        "[concat('https://web.azuresynthesize.net?workspace=%2fsubscriptions%2f',subscription().Id,'%2fresourceGroups%2f',resourceGroup().name,'%2fproviders%2fMicrosoft.Synapse%2fworkspaces%2f',resourceGroup().name,'s')]",  
                    "dev": "[concat('https://',resourceGroup().name,'s','.dev.azuresynthesize.net')]",  
                    "sqlOnDemand": "[concat(resourceGroup().name,'s','-'  
ondemand.sql.azuresynthesize.net')]",  
                    "sql": "[concat(resourceGroup().name,'s','.sql.azuresynthesize.net')]"  
                },  
                "managedResourceGroupName": "[concat(resourceGroup().name,'mrg-s')]",  
                "sqlAdministratorLogin": "[parameters('SynapseSQL_AdministratorName')]",  
                "sqlAdministratorLoginPassword": "[parameters('SynapseSQL_AdministratorPassword')]",  
                "publicNetworkAccess": "Enabled"  
            }  
        }  
    ]  
}
```

Firewall Rules

Use this template to add Firewall Rules to Synapse.

Rules should align with your security policies and network configuration.

```
{  
  "$schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",  
  "contentVersion": "1.0.0.0",  
  "resources": [  
    {  
      "type": "Microsoft.Synapse/workspaces/firewallRules",  
      "apiVersion": "[providers('Microsoft.Synapse','workspaces').apiVersions[0]]",  
      "name": "[concat(resourceGroup().name,'s','/AllowAllWindowsAzureIps')]",  
      "properties": { "startIpAddress": "0.0.0.0", "endIpAddress": "0.0.0.0" }  
    },  
    {  
      "type": "Microsoft.Synapse/workspaces/firewallRules",  
      "apiVersion": "[providers('Microsoft.Synapse','workspaces').apiVersions[0]]",  
      "name": "[concat(resourceGroup().name,'s','/allowRange')]",  
      "properties": { "startIpAddress": "#.#.0.0", "endIpAddress": "#.#.255.255" }  
    }  
  ]  
}
```

Azure Active Directory >> Set Admin

The screenshot shows the Microsoft Azure (Preview) portal interface. The URL is https://ms.portal.azure.com/#@microsoft.onmicrosoft.com/resource/subscriptions/91e9ffdc-ef15-416c-9be1-085f8b1b46ed/resource... The page title is "rchaplers | Azure Active Directory". On the left, there's a sidebar with "Azure Active Directory" selected under "Settings". The main content area is titled "Azure Active Directory admin" and contains a note about Azure Active Directory authentication. It shows the "Active Directory admin" field set to "rchapler@microsoft.com" and the "Admin Object/App ID" field empty. Below this, there's a section for "Azure Active Directory authentication only" with a note that only Azure Active Directory will be used for authentication. A checkbox labeled "Support only Azure Active Directory authentication for this workspace" is checked. The top right corner shows the user's email "rchapler@microsoft.com" and the Microsoft logo.

Once deployment of a Synapse Analytics Workspace is complete, consider what entity should be set as the Azure Active Directory admin.

Repository

The “how to” is already described in the Preparation > Synapse > Configure Repository section. Inclusion here is intended only to call out two deployment-related considerations:

- **Repository** – each Synapse instantiation {i.e., DEV, TEST, and PROD} must be manually configured after deployment of infrastructure-as-code **programmatic setting of Synapse + GitHub Personal Access Token is not yet an available feature**
- **Branch** – the collaboration branch for each Synapse instance should be configured with the corresponding branch {e.g., “development”, “test”, and “production”}in each repository

Synapse Studio, Role-Based Access Control (RBAC)

A person that creates a Synapse workspace is initialized as a Workspace Administrator, allowing full access to Synapse Studio, and granting the ability to manage role assignments.

When we create a Synapse workspace using an ARM template, however, there is no way to grant Synapse roles via the Studio interface.

And an aside... ARM templates are not a supported method for changing Synapse Studio roles.

So, the Deployment Manager will have to manually update Synapse RBAC settings as below.

Manage Access with Cloud Shell (PowerShell)

Navigate to Cloud Shell and select **PowerShell** from the “**Select Environment**” drop-down menu.

List Role Definitions

Update and execute the following command to detail Synapse Studio roles (in preparation for future commands):

```
az synapse role definition list --workspace-name <synapseWorkspaceName>
```

(Partial) logic explanation:

- --workspace-name <synapseWorkspaceName>
...where <synapseWorkspaceName> refers to the name of your Synapse instance {e.g., <UseCase>s}

The JSON response will include role definition data like the snippet below:

```
{  
    "availabilityStatus": "Available",  
    "description": "Full Synapse access to serverless SQL pools, Apache Spark pools and Integration runtimes. Includes create, read, update and delete access to all published code artifacts. Includes Compute Operator, Linked Data Manager, and Credential User permissions on the workspace system identity credential. Includes granting access. Azure permissions are required to create, delete, or manage compute resources.",  
    "id": "6e4bf58a-b8e1-4cc3-bbf9-d73143322b78",  
    "isBuiltIn": true,  
    "name": "Synapse Administrator",  
    ...  
}
```

Capture the **id** and **name** values for later use.

Create Role Assignment

Update and execute the following command to create a Synapse Studio role assignment:

```
az synapse role assignment create --workspace-name <SynapseWorkspaceName> --role <roleNameOrId> --assignee <aadPrincipalId>
```

Example:

```
az synapse role assignment create --workspace-name rchapters --role "Synapse Administrator" --assignee cbe2948c-5701-4265-811d-633a314673f0
```

(Partial) logic explanation:

- `--workspace-name <synapseWorkspaceName>`
...where `<synapseWorkspaceName>` refers to the name of your Synapse instance {e.g.,
`<UseCase>s`}

```
PowerShell PS /home/rich> az synapse role assignment create --workspace-name rchapters --role "Synapse Administrator" --assignee cbe2948c-5701-4265-811d-633a314673f0
Command group 'synapse' is in preview and under development. Reference and support levels: https://aka.ms/CLI_RefStatus
{
  "id": "84cd291-ef13-481b-b5b9-e1421940e518",
  "principalId": "cbe2948c-5701-4265-811d-633a314673f0",
  "principalType": "User",
  "roleDefinitionId": "6e4bf58a-b8e1-4cc3-bbf9-d73143322b78",
  "scope": "workspaces/rchapters"
}
PS /home/rich> 
```

Note: Considering adding various Synapse RBAC Roles; for example:

- “*Synapse Artifact Publisher*” to allow you to publish
 - “*Synapse Linked Data Manager*” to allow you to create a *Linked Server*

Data Explorer Pool

The Data Explorer Pool template requires a “**workspaceUID**” value (that will help us uniquely identify the Synapse workspace created in the prior step). You can get this value using the REST API call described at the following link: [Workspaces - List - REST API \(Azure Synapse\) | Microsoft Docs](#)

Use the “**Try It**” button and resulting form to personalize and submit the following REST API request:

GET
<https://management.azure.com/subscriptions/{subscriptionId}/providers/Microsoft.Synapse/workspaces?api-version=2021-06-01>

The response Body will include a “**workspaceUID**” value that you can bake into the template below:

```
{  
    "$schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",  
    "contentVersion": "1.0.0.0",  
    "resources": [  
        {  
            "type": "Microsoft.Synapse/workspaces/kustoPools",  
            "apiVersion": "[providers('Microsoft.Synapse', 'workspaces/kustoPools').apiVersions[0]]",  
            "name": "[concat(resourceGroup().name,'/',resourceGroup().name,'dep')]",  
            "location": "[resourceGroup().location]",  
            "sku": {  
                "name": "Compute optimized",  
                "size": "Extra small",  
                "capacity": 2  
            },  
            "properties": {  
                "enableStreamingIngest": false,  
                "enablePurge": false,  
                "workspaceUID": "db162ad0-8a4e-4aa7-a7df-60d6b044999a"  
            }  
        }  
    ]  
}
```

Data Explorer Database

Use this template to add a database to a Data Explorer database. This template can be used with a Data Explorer Pool or Data Explorer Cluster.

```
{  
    "$schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",  
    "contentVersion": "1.0.0.0",  
    "resources": [  
        {  
            "type": "Microsoft.Synapse/workspaces/kustoPools/Databases",  
            "apiVersion":  
                "[providers('Microsoft.Synapse', 'workspaces/kustoPools/Databases').apiVersions[0]]",  
            "name":  
                "[concat(resourceGroup().name,'/',resourceGroup().name,'dep','/',resourceGroup().name,'ded')]",  
            "location": "[resourceGroup().location]",  
            "kind": "ReadWrite",  
            "properties": { "softDeletePeriod": "P365D", "hotCachePeriod": "P31D" }  
        }  
    ]  
}
```

Code

This section describes deployment of data solution “code” {i.e., pipelines, datasets, linked services, etc.}. These tasks will be completed more frequently.

Once Synapse is re-configured, we shift to recurring tasks.

Deployment process can be remarkably simple or quite complex depending on your team and operational requirements. Consider the following activities when designing your process:

1. **Developer begins work on a development task** by creating a new branch {i.e., “Branch X”} based on the collaboration branch
2. **Developer completes development task**, tests changes, and ensures quality of commits in their branch (“Branch X”)
3. **Developer creates a pull request** (“Branch X” >> “Collaboration” branch), and assigns necessary reviewers / approvers (individually or as part of a group)
4. **Peers review**, provide feedback and approve (or reject) the pull request
5. On a regular schedule, **Development Manager creates “Release” branch** including only those commits that should make their way through the deployment process
6. **Development Manager creates a pull request** (“Release” branch >> “TEST” branch), and assigns to Deployment Manager
7. **Deployment Manager reviews “Release” branch** for obvious issues
8. Assuming no issues, **Deployment Manager creates a pull request** (“Release” branch >> “PROD” branch)

A few notes:

- *Before deploying “Release” branch commits {e.g., to a “TEST” branch}, consider replacing the contents of the “TEST” branch with a copy of the contents of the “PROD” branch*
- *When you begin your formal deployment process, and IF you have a previously created “PROD” branch ... consider purging any questionable content to make “PROD” as perfect as possible*

Deploy Schema

Before turning on the pipelines and triggers to move data, we must prepare the destination database (specific to the deployment environment) by creating or updating the schema.

There are many options for semi-automated deployment of database schema elements; for now, we will discuss two: 1) via GitHub Action or 2) via Synapse Pipeline.

GitHub Action

If you are using an Azure Data Explorer Cluster, you can use this YAML in a GitHub Action to execute KQL Logic such as “.create table”

```
name: Azure Data Solution, Synapse Data Explorer, Schema for TableX
on: workflow_dispatch
jobs:
  build:
    runs-on: self-hosted
    steps:
      - uses: actions/checkout@v2
      - uses: elstudio/action-install-azure-cli@main

      - name: Login to Azure
        uses: azure/login@v1.1
        with:
          creds: ${{ secrets.AZURELOGINCREDS_PROD }}

      - name: _TableX
        uses: azure/arm-deploy@v1
        with:
          subscriptionId: ${{ secrets.SUBSCRIPTIONID_PROD }}
          resourceName: ${{ secrets.RESOURCEGROUPNAME_PROD }}
          template: ./templates/dataexplorerscript.json
          parameters:
            kqlScript=".create table _TableX ( Col1: int, Col2: string, _id: string, _dt: datetime, _op: string ) with ( folder = 'Bronze' )"
```

[Trigger an Azure Pipelines run from GitHub Actions - Azure Pipelines | Microsoft Docs](#)

Bonus Materials

All-In YAML Example

The following YAML will deploy ARM templates for each of the resources necessary for a Data Solution team starting use of a new Azure subscription:

```
name: Data Solution Architecture, Deployment
on: workflow_dispatch
jobs:
  build:
    runs-on: self-hosted
    steps:
      - uses: actions/checkout@v2

      - name: Login to Azure
        uses: azure/login@v1.1
        with:
          creds: ${{ secrets.AZURELOGINCREDS }}

      # ##### Only needs to be uncommented for initial deployment
      # - name: Register Resource Provider Microsoft.Sql (pre-requisite for Synapse)
      #   run: az provider register --namespace Microsoft.Sql

      # - name: Pause for completion of Resource Provider registration
      #   run: sleep 5m
      #   shell: bash

      - name: Create Synapse Analytics Workspace
        uses: azure/arm-deploy@v1
        with:
          subscriptionId: ${{ secrets.SUBSCRIPTIONID_TEST }}
          resourceName: ${{ secrets.RESOURCEGROUPNAME }}
          template: ./arm/synapse.json
          parameters:
            SynapseSQL_AdministratorName=${{ secrets.SYNAPSESQL_ADMINISTRATORNAME }}
            SynapseSQL_AdministratorPassword=${{ secrets.SYNAPSESQL_ADMINISTRATORPASSWORD }}

      - name: Create Synapse, Firewall Rules
        uses: azure/arm-deploy@v1
        with:
          subscriptionId: ${{ secrets.SUBSCRIPTIONID_TEST }}
          resourceName: ${{ secrets.RESOURCEGROUPNAME }}
          template: ./arm/synapse_firewallrules.json

      - name: Create Synapse, Data Explorer Pool
        uses: azure/arm-deploy@v1
        with:
          subscriptionId: ${{ secrets.SUBSCRIPTIONID_TEST }}
          resourceName: ${{ secrets.RESOURCEGROUPNAME }}
          template: ./arm/synapse_dataexplorerpool.json

      - name: Create Data Explorer Database
        uses: azure/arm-deploy@v1
        with:
          subscriptionId: ${{ secrets.SUBSCRIPTIONID_TEST }}
          resourceName: ${{ secrets.RESOURCEGROUPNAME }}
          template: ./arm/dataexplorerdatabase.json
```

More Templates

There are some excellent ARM templates at: <https://azure.microsoft.com/en-us/resources/templates>

File Sync (ARM)

(aka Storage Sync Service)

```
{  
  "$schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",  
  "contentVersion": "1.0.0.0",  
  "resources": [  
    {  
      "type": "microsoft.storagesync/storageSyncServices",  
      "apiVersion": "[providers('Microsoft.StorageSync','storageSyncServices').apiVersions[0]]",  
      "name": "[concat(resourceGroup().name,'sss')]",  
      "location": "[resourceGroup().location]",  
      "properties": {  
        "incomingTrafficPolicy": "AllowAllTraffic"  
      }  
    }  
  ]  
}
```

Key Vault (ARM)

```
{  
  "$schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",  
  "contentVersion": "1.0.0.0",  
  "resources": [  
    {  
      "type": "Microsoft.KeyVault/vaults",  
      "apiVersion": "[providers('Microsoft.KeyVault','vaults').apiVersions[0]]",  
      "name": "[concat(resourceGroup().name,'kv')]",  
      "location": "westus",  
      "properties": {  
        "sku": {  
          "family": "A",  
          "name": "Standard"  
        },  
        "tenantId": "[subscription().tenantId]",  
        "accessPolicies": [  
          {  
            "tenantId": "[subscription().tenantId]",  
            "objectId": "{Object ID for Azure Active Directory User}",  
            "permissions": {  
              "keys": [  
                "Get",  
                "List",  
                "Update",  
                "Create",  
                "Delete"  
              ],  
              "secrets": [  
                "Get",  
                "List",  
                "Set",  
                "Delete"  
              ]  
            }  
          }  
        ]  
      }  
    }  
  ]  
}
```

```
        ]
    }



## KQL Script (ARM)



```
{
 "$schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",
 "contentVersion": "1.0.0.0",
 "parameters": {
 "kqlscript": {
 "defaultValue": ".create table _bt_info (nlfdkbnr: int, sbaureihe: string, spnr8: string,
dtinsert:datetime, _id: string, _dt: datetime, _op: string) with (folder = 'Bronze')",
 "type": "String"
 }
 },
 "resources": [
 {
 "type": "Microsoft.Kusto/Clusters/Databases/Scripts",
 "apiVersion": "[providers('Microsoft.Kusto','Clusters/Databases/Scripts').apiVersions[0]]",
 "name": "[concat(resourceGroup().name,'dec/',resourceGroup().name,'ded/','guid(resourceGroup().id,
deployment().name))]",
 "properties": {
 "scriptContent": "[parameters('kqlScript')]",
 "continueOnErrors": false
 }
 }
]
}
```


```

Storage Account (ARM)

```
{  
  "$schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",  
  "contentVersion": "1.0.0.0",  
  "resources": [  
    {  
      "type": "Microsoft.Storage/storageAccounts",  
      "apiVersion": "[providers('Microsoft.Storage','storageAccounts').apiVersions[0]]",  
      "name": "[concat(resourceGroup().name,'sa')]",  
      "location": "[resourceGroup().location]",  
      "sku": { "name": "Standard_LRS" },  
      "kind": "StorageV2"  
    }  
  ]  
}
```

Virtual Machine (ARM)

```
{  
  "$schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",  
  "contentVersion": "1.0.0.0",  
  "resources": [  
    {  
      "type": "Microsoft.Network/publicIPAddresses",  
      "apiVersion": "[providers('Microsoft.Network','publicIPAddresses').apiVersions[0]]",  
      "name": "[concat(resourceGroup().name,'-pia')]",  
      "location": "[resourceGroup().location]",  
      "sku": {  
        "name": "Basic"  
      },  
      "properties": {  
        "publicIPAllocationMethod": "Dynamic",  
        "dnsSettings": {  
          "domainNameLabel": "[toLower(format('{0}-{1}', resourceGroup().name,  
uniqueString(resourceGroup().id, resourceGroup().name)))]"  
        }  
      }  
    },  
    {  
      "type": "Microsoft.Network/networkSecurityGroups",  
      "apiVersion": "[providers('Microsoft.Network','networkSecurityGroups').apiVersions[0]]",  
      "name": "[concat(resourceGroup().name,'-nsg')]",  
      "location": "[resourceGroup().location]",  
      "properties": {  
        "securityRules": [  
          {  
            "name": "default-allow-3389",  
            "properties": {  
              "priority": 1000,  
              "access": "Allow",  
              "direction": "Inbound",  
              "destinationPortRange": "3389",  
              "protocol": "Tcp",  
              "sourcePortRange": "*",  
              "sourceAddressPrefix": "*",  
              "destinationAddressPrefix": "*"  
            }  
          }  
        ]  
      }  
    },  
    {  
      "type": "Microsoft.Network/virtualNetworks",  
      "apiVersion": "[providers('Microsoft.Network','virtualNetworks').apiVersions[0]]",  
      "name": "[concat(resourceGroup().name,'-vn')]",  
      "location": "[resourceGroup().location]",  
      "properties": {  
        "addressSpace": {  
          "addressPrefixes": ["10.0.0.0/16"]  
        },  
        "subnets": [  
          {  
            "name": "Subnet1",  
            "properties": {  
              "addressPrefix": "10.0.1.0/24"  
            }  
          }  
        ]  
      }  
    }  
  ]  
}
```

```

"properties": {
    "addressSpace": {
        "addressPrefixes": [
            "10.0.0.0/16"
        ]
    },
    "subnets": [
        {
            "name": "Subnet",
            "properties": {
                "addressPrefix": "10.0.0.0/24",
                "networkSecurityGroup": {
                    "id": "[resourceId('Microsoft.Network/networkSecurityGroups',
concat(resourceGroup().name,'-nsg'))]"
                }
            }
        }
    ]
},
"dependsOn": [
    "[resourceId('Microsoft.Network/networkSecurityGroups', concat(resourceGroup().name,'-nsg'))]"
]
},
{
    "type": "Microsoft.Network/networkInterfaces",
    "apiVersion": "2021-02-01",
    "name": "[concat(resourceGroup().name,'-ni')]",
    "location": "[resourceGroup().location]",
    "properties": {
        "ipConfigurations": [
            {
                "name": "ipconfig1",
                "properties": {
                    "privateIPAllocationMethod": "Dynamic",
                    "publicIPAddress": {
                        "id": "[resourceId('Microsoft.Network/publicIPAddresses', concat(resourceGroup().name,'-pia'))]"
                    },
                    "subnet": {
                        "id": "[resourceId('Microsoft.Network/virtualNetworks/subnets',
concat(resourceGroup().name,'-vn'), 'Subnet')]"
                    }
                }
            }
        ]
    },
    "dependsOn": [
        "[resourceId('Microsoft.Network/publicIPAddresses', concat(resourceGroup().name,'-pia'))]",
        "[resourceId('Microsoft.Network/virtualNetworks', concat(resourceGroup().name,'-vn'))]"
    ]
},
{
    "type": "Microsoft.Compute/virtualMachines",
    "apiVersion": "[providers('Microsoft.Compute','virtualMachines').apiVersions[0]]",
    "name": "[resourceGroup().name]",
    "location": "[resourceGroup().location]",
    "properties": {
        "hardwareProfile": {
            "vmSize": "Standard_D2_v3"
        },
        "osProfile": {
            "computerName": "[resourceGroup().name]",
            "adminUsername": "Admin123",
            "adminPassword": "Password123!"
        },
        "storageProfile": {
            "imageReference": {
                "publisher": "MicrosoftWindowsServer",
                "offer": "WindowsServer",
                "sku": "WindowsServer2022-Datacenter"
            }
        }
    }
}

```

```

        "sku": "2022-Datacenter",
        "version": "latest"
    },
    "osDisk": {
        "name": "[concat(resourceGroup().name, '-d-os')]",
        "createOption": "FromImage",
        "managedDisk": {
            "storageAccountType": "StandardSSD_LRS"
        }
    },
    "dataDisks": [
        {
            "name": "[concat(resourceGroup().name, '-d-data')]",
            "diskSizeGB": 1023,
            "lun": 0,
            "createOption": "Empty"
        }
    ],
    "networkProfile": {
        "networkInterfaces": [
            {
                "id": "[resourceId('Microsoft.Network/networkInterfaces', concat(resourceGroup().name, '-ni'))]"
            }
        ]
    },
    "dependsOn": [
        "[resourceId('Microsoft.Network/networkInterfaces', concat(resourceGroup().name, '-ni'))]"
    ],
    "outputs": {
        "hostname": {
            "type": "string",
            "value": "[reference(resourceId('Microsoft.Network/publicIPAddresses',
concat(resourceGroup().name, '-pia'))).dnsSettings.fqdn]"
        }
    }
}

```

User Assigned Managed Identity (ARM)

```
{  
    "$schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",  
    "contentVersion": "1.0.0.0",  
    "variables": {},  
    "resources": [  
        {  
            "type": "Microsoft.ManagedIdentity/userAssignedIdentities",  
            "apiVersion":  
                "[providers('Microsoft.ManagedIdentity','userAssignedIdentities').apiVersions[0]]",  
                "name": "[concat(resourceGroup().name,'uami')]",  
                "location": "[resourceGroup().location]"  
        }  
    ]  
}
```

Protect Resources

Use Case

Example notes:

- “We need to prevent the accidental deletion of our production Data Lake”
- “Nothing in the production instance should be deleted without appropriate permissions”

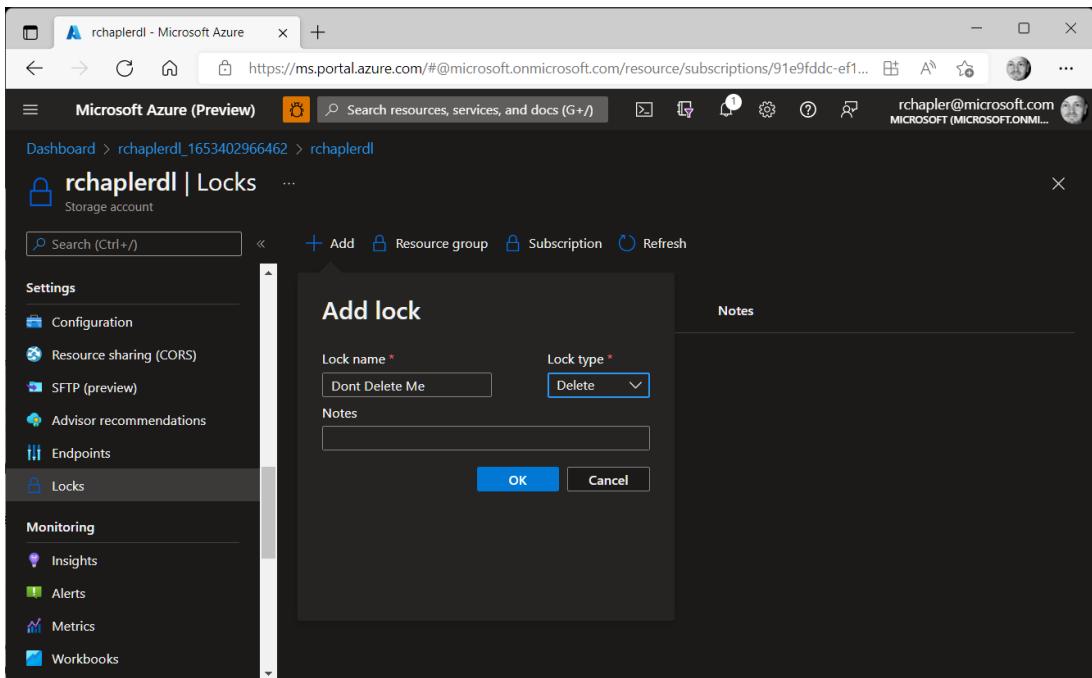
Prerequisites

This solution requires the following resources:

- Data Lake

Lock Storage Account

Navigate to your Data Lake, then click **Locks** in the **Settings** group of the navigation pane.



The screenshot shows the Microsoft Azure portal interface. The user is navigating through the URL: https://ms.portal.azure.com/#@microsoft.onmicrosoft.com/resource/subscriptions/91e9fddc-ef1... . The left sidebar has sections for Settings (Configuration, Resource sharing (CORS), SFTP (preview), Advisor recommendations, Endpoints, Locks), Monitoring (Insights, Alerts, Metrics, Workbooks), and a search bar. The main content area shows a 'rchaplerdl | Locks' page for a Storage account. A modal dialog box titled 'Add lock' is open, prompting for 'Lock name' (containing 'Dont Delete Me') and 'Lock type' (set to 'Delete'). There is also a 'Notes' field and 'OK' and 'Cancel' buttons at the bottom of the dialog.

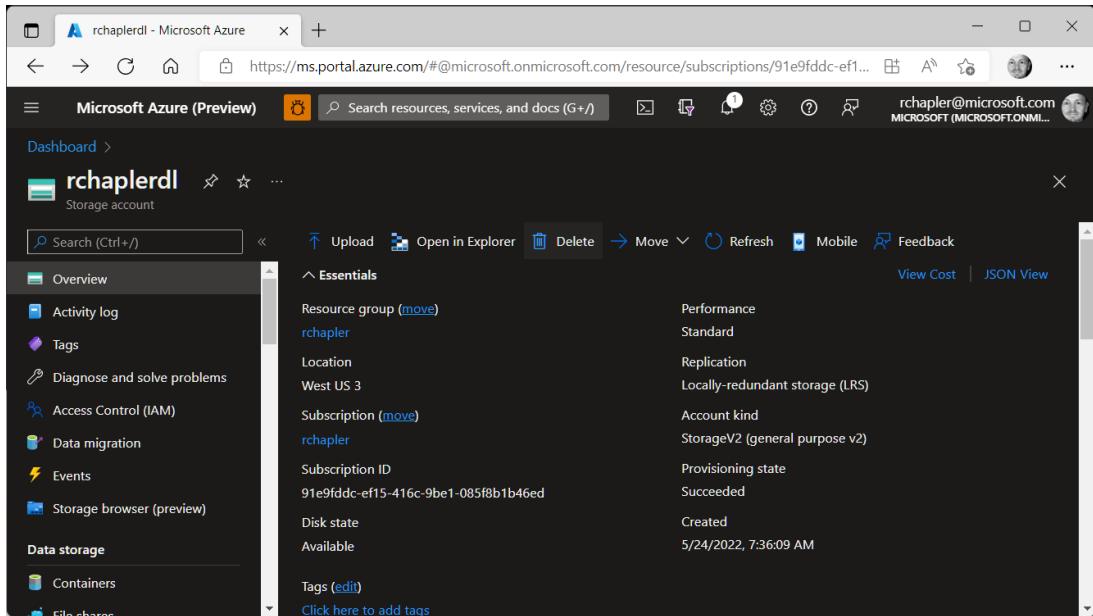
Click the “+ Add” button. On the resulting “Add lock” pop-up, including:

Lock Type	Select Delete
-----------	---------------

Click the **OK** button.

Confirm Success

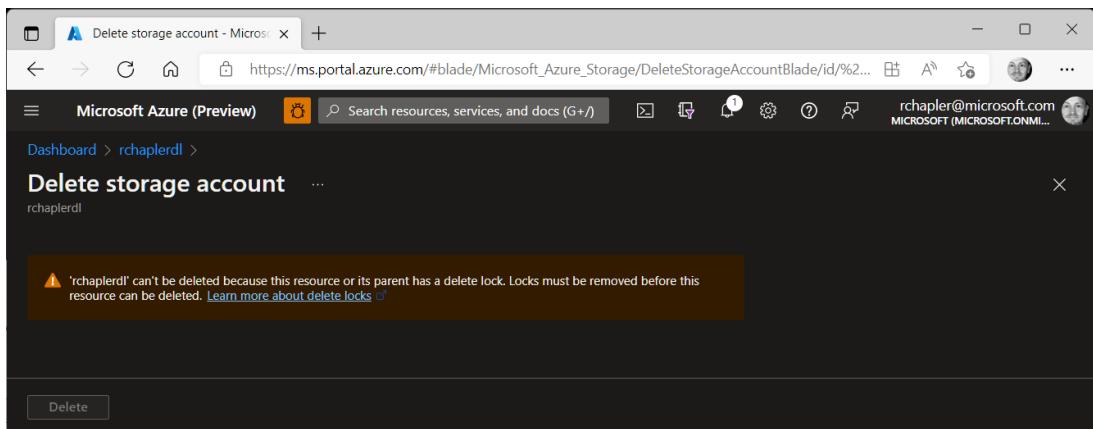
Click Overview at the top of the navigation pane.



The screenshot shows the Microsoft Azure portal interface. The left sidebar has a tree view with 'rchaplerdl' expanded, showing 'Storage account'. The main content area is titled 'rchaplerdl' and shows the 'Overview' tab selected. The 'Essentials' section displays various account details:

Resource group (move)	Performance
rchapler	Standard
Location	Replication
West US 3	Locally-redundant storage (LRS)
Subscription (move)	Account kind
rchapler	StorageV2 (general purpose v2)
Subscription ID	Provisioning state
91e9ffdc-ef15-416c-9be1-085f8b1b46ed	Succeeded
Disk state	Created
Available	5/24/2022, 7:36:09 AM
Tags (edit)	
Click here to add tags	

Click the **Delete** button.



The screenshot shows the 'Delete storage account' blade. At the top, it says 'Delete storage account' and shows the account name 'rchaplerdl'. Below that is a warning message: '⚠️ 'rchaplerdl' can't be deleted because this resource or its parent has a delete lock. Locks must be removed before this resource can be deleted. [Learn more about delete locks](#)'.

You should receive a message indicating that the resource cannot be deleted.

Presentation

Query from On-Prem

Follow these instructions to **query from an on-prem client**.

Prerequisites

This solution requires the following resources:

- Azure CLI
- Data Explorer
- Postman

Get Token

In a Command Prompt window (with [Azure CLI](#) installed), execute the following command: az login

Use the resulting browser window to provide credentials.

In a Command Prompt window (with [Azure CLI](#) installed), execute the following command: az account get-access-token --resource https://<UseCase>dec.westus2.kusto.windows.net --query accessToken --output tsv

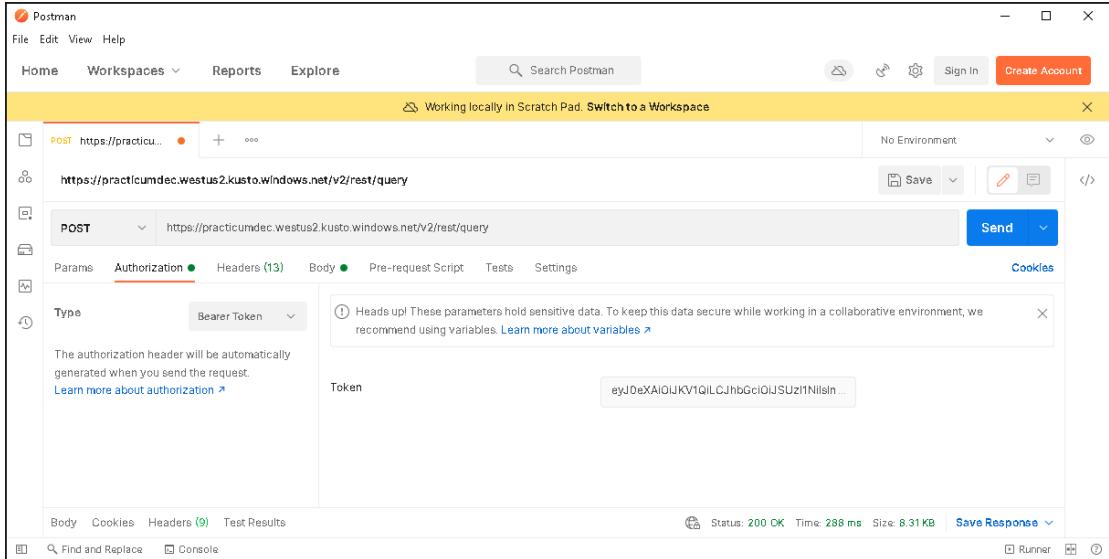
Capture the returned token for use later in this section.



```
Command Prompt
C:\Users\rchapler>az account get-access-token --resource https://practicumdec.westus2.kusto.windows.net --query accessToken --output tsv
eyJ0exA1O1JKV1O1CJhbGcI0JSUz1In1iis1ng1dC16Im5pbzNarH3PRhfSzFgS1doWnNsF3f51hfZyIsImpzC16Im5pbzNarH3PRhfSzFgS1doWnNsF3f51hfZy39...eyJhdWjM
01JodhRwczoV13ByWNN0awN1bwR1Yy53ZXh0dXMyLmt1c3RVlndpbmRvd3MuwmV0IwiXNzTjo1ahR0cIM6ly9zdHMud21uZG93cy5uZXQVnzJm0T4YmYtODZmMS00MFmlTkxYMI
mQ3Y2QwMTFKYj03LyIsIm1hdC16NTyxODk1NjIiNCwibmJmjoNxje40TU2MJu0lC31ehA10jE2MTg5NjAxNTQsImpFjciI6jFe1lC3hawB101JBVVFbdS84VFBUFSY1NLemdTcmvRNm
Frcmh1ly8xU31xzmvXVUtyOEUwVhs51gwJXZ3cGhwSVNOSFhuaxYrc-25ROEp6YTINveWptUW5QOU1JRndt1eUpIMFBWh1M3Q1Zydz091iwiYWiyljpBInJzYSiSm1ySjdLCJhcHBpZCI
6iJA0yJA3Nzk1LThkZGItNDYxyS1i1yMv1LTayZj1IMWmN210Ni1sImFwcGikyWny1joimCsImR1dm1jZWLk1jfe1lC3hawB101JBVVFbdS84VFBUFSY1NLemdTcmvRNm
Zmf-taw5x25hbWu101DagFwbgvYi1w1Z212ZwsfomtZS1611Jp2g1lCjpcGFZh1I0113N14xjhjEuMTk0lJezM1iis1m5hbWU101JsawNo1EnovXbsZX11LCJvalQ01jYmUy0TQ4Y
y01NzAxLTOyNjUt0DExZC02MznHmZeoNjczJ1lLCjvbnyZwlf2c1kIjo1uy0xL TUthMjEtMjEyNzUyMTE4NC0xNjAOADEy0TiwlTE40Dc5Mj1cMjctNDY2NzUyOCIsIm1baQ0101xMD
AzN0ZGRtgwMUFDDYxi1wLkfSb0f2NGo1Y3ZR3iwr1xeTe4MEJY1L11VjnQzVRial1jw3tUNEMzUdfZTBZYUEFVs41LcJzy3A1o1j1c2Vyx21tcGvy29uXRpb241LCJ
zdnW11013sRk32Q0xxkdGf1dHbo0U1c3VyTktx3lqam84ax0452zaM024dHdnv3ZNIliw1dg1kIjo1nZm0Tg4YmYtODZmMS00MFmlTkxYIltMmQ3Y2QwMTFKYj031iwlw55pcXV1X25h
bwU1i01jyY2hhcGxlckBtaWnyb3NvQuNzQy29t1iwidXbUi1jo1cmNoYXbsZXAbWljcm9zB2Z0lmlVbStsInvoaS161KtqdV1hbEdyQ1VhcnfHajNBND83QUE1LCJ2ZXi1o1xLjAifQ.U6
yTE35AdT0hJgUejjUPF6raM90Rf07SzFya4a2bzPzrwKntGRekyjLzqaRSC-c1zPw98dp5Gr1Hw0cpXnZTbr-A-Mv9-g4Gwsst0311lxQXK13BmhfGhqadQv3j1D-N_I1SU1hxJ6F
jgszgfwb1NgCoigt0ndyEWrhFgv031PKwmwTFLu3-huLqR0Dxm1eu519Uf4cxMm00cxSaKeNs2ecodaYAUxcjg02cdj1VlEgb1o92zCCNJRb1AgHJcXHVGAfpBtYl03XL_Xngc-DpZdy
C6MMp1SF4-BEc72S9Nli1JHNpGq-hyBCB2_5bduHwXMGTnflwbq14PQ
C:\Users\rchapler>
```

Post API Request

Open a new request in Postman.

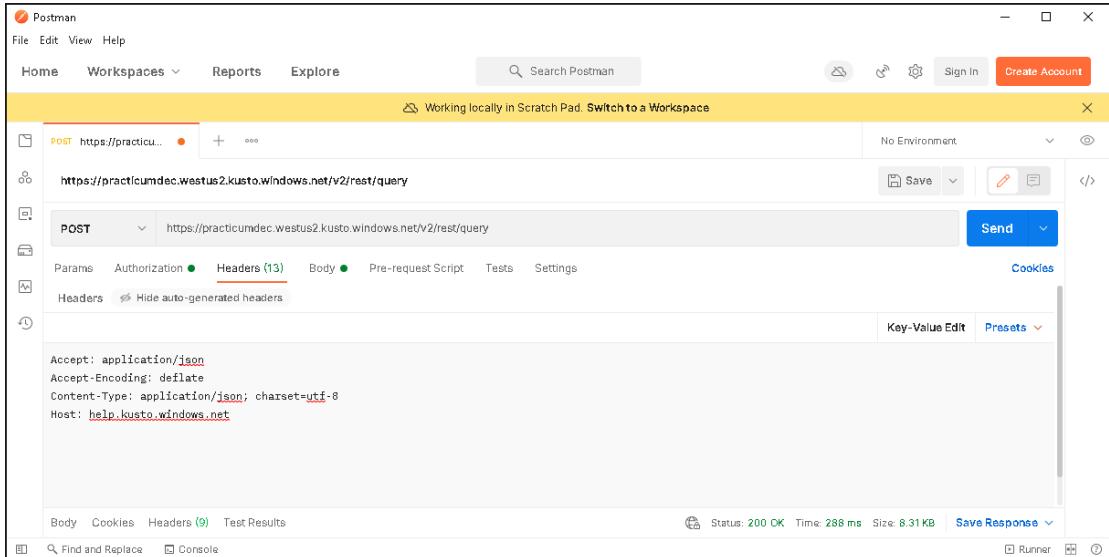


The screenshot shows the Postman application window. At the top, there's a navigation bar with 'File', 'Edit', 'View', 'Help', 'Home', 'Workspaces', 'Reports', 'Explore', and a search bar. On the right side of the header are 'Sign In' and 'Create Account' buttons. Below the header, a yellow banner says 'Working locally in Scratch Pad. Switch to a Workspace'. The main area shows a request configuration for a 'POST' method to 'https://practicumdec.westus2.kusto.windows.net/v2/rest/query'. The 'Authorization' tab is selected, showing a dropdown set to 'Bearer Token'. A note below it says: 'The authorization header will be automatically generated when you send the request.' and 'Learn more about authorization'. To the right, there's a 'Token' input field containing a long JWT token. Other tabs like 'Headers', 'Body', 'Pre-request Script', 'Tests', and 'Settings' are visible. At the bottom, there are buttons for 'Send', 'Cookies', and 'Status: 200 OK Time: 288 ms Size: 8.31 KB Save Response'. The footer has links for 'Find and Replace' and 'Console'.

Select **POST** from the drop-down menu. Enter a hostname (with the name you used for your cluster); example:

`https://<UseCase>dec.westus2.kusto.windows.net/v2/rest/query`

On the **Authorization** tab, select “**Bearer Token**” from the drop-down menu and paste your previously copied token in the resulting interface.



This screenshot shows the same Postman interface as above, but with the 'Headers' tab selected instead of 'Authorization'. The 'Headers' section contains the following entries:
Accept: application/json
Accept-Encoding: deflate
Content-Type: application/json; charset=utf-8
Host: help.kusto.windows.net

On the **Headers** tab, enter the following items:

Accept	application/json
Accept-Encoding	deflate
Content-Type	application/json; charset=utf-8
Host	help.kusto.windows.net

The screenshot shows the Postman application window. At the top, there's a navigation bar with 'File', 'Edit', 'View', 'Help', 'Home', 'Workspaces', 'Reports', 'Explore', and a search bar. On the right of the top bar are 'Sign In' and 'Create Account' buttons. Below the top bar, a yellow header bar says 'Working locally in Scratch Pad. Switch to a Workspace'. The main area has a 'POST https://practicumdec.westus2.kusto.windows.net/v2/rest/query' request in the center. To the left of the URL is a dropdown menu showing 'POST https://practicumdec.westus2.kusto.windows.net/v2/rest/query'. Below the URL are tabs for 'Params', 'Authorization', 'Headers (13)', 'Body', 'Pre-request Script', 'Tests', and 'Settings'. The 'Body' tab is selected and has a dropdown menu with options: 'none', 'form-data', 'x-www-form-urlencoded', 'raw', 'binary', 'GraphQL', and 'Text'. The 'Text' option is selected. The body content is a JSON document:

```
1 {  
2     "db": "practicumdec",  
3     "csl": "StormEvents | take 5"  
4 }
```

At the bottom of the main area, there are tabs for 'Body', 'Cookies', 'Headers (9)', 'Test Results', and status information: 'Status: 200 OK', 'Time: 288 ms', 'Size: 8.31 KB', and 'Save Response'. At the very bottom are 'Find and Replace' and 'Console' buttons.

On the **Body** tab, paste the following:

```
{  
    "db": "<UseCase>ded",  
    "csl": "StormEvents | take 5"  
}
```

Click the **Send** button.

The screenshot shows the Postman application interface. At the top, there's a navigation bar with 'File', 'Edit', 'View', 'Help', 'Home', 'Workspaces', 'Reports', 'Explore', and a search bar. On the right of the top bar are 'Sign In' and 'Create Account' buttons. Below the top bar, a yellow banner says 'Working locally in Scratch Pad. Switch to a Workspace'. The main area has a 'POST https://practicumdec.westus2.kusto.windows.net/v2/rest/query' request in the center. The 'Body' tab is selected, showing a JSON response with line numbers 1 through 24. The response content is as follows:

```
1 {
2     "FrameType": "DataSetHeader",
3     "isProgressive": false,
4     "Version": "v2.0"
5 },
6 {
7     "FrameType": "DataTable",
8     "TableId": 0,
9     "TableKind": "QueryProperties",
10    "TableName": "@ExtendedProperties",
11    "Columns": [
12        {
13            "ColumnName": "TableId",
14            "ColumnType": "int"
15        },
16        {
17            "ColumnName": "Key",
18            "ColumnType": "string"
19        },
20        {
21            "ColumnName": "Value",
22            "ColumnType": "dynamic"
23        }
24    ]
}
```

At the bottom of the interface, there are tabs for 'Pretty', 'Raw', 'Preview', 'Visualize', and 'JSON'. To the right of these tabs is a 'Copy' button. Below the tabs, the status bar shows 'Status: 200 OK Time: 244 ms Size: 8.29 KB' and 'Save Response'.

You should see JSON with both schema and data included.

Detect Anomalies

Use Case: Customer XYZ shared the following requirements:

- Detect data anomalies for Metric 123
- When significant anomalies are detected, the solution should support investigation

Follow these instructions to on-board data feeds and investigate findings.

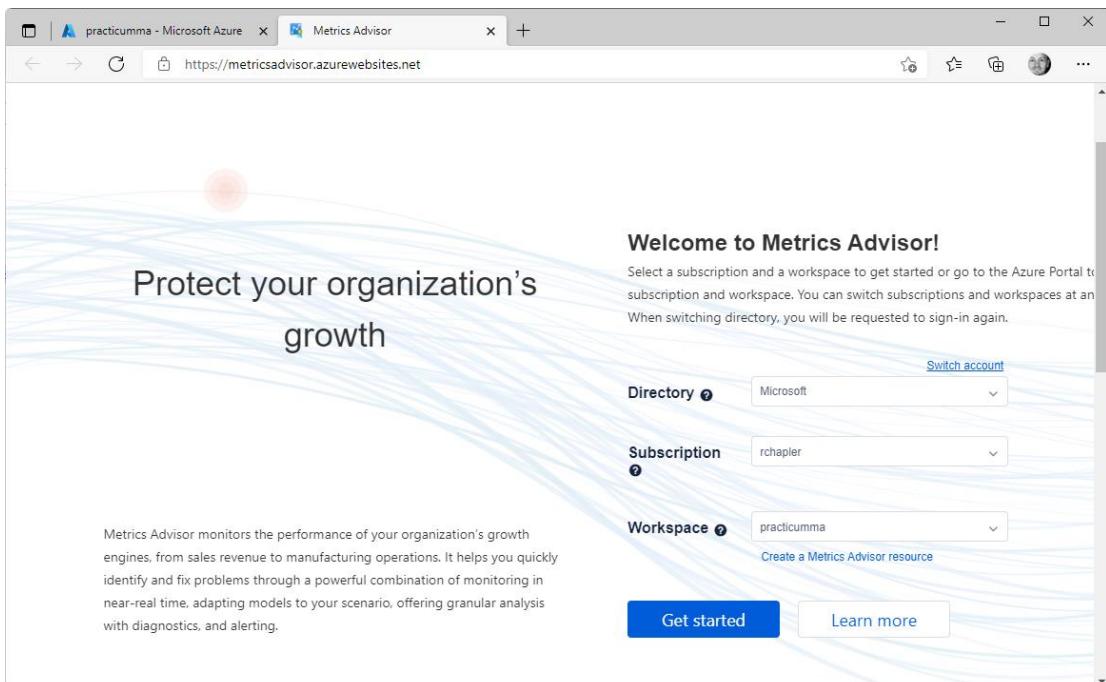
Prerequisites

This solution requires the following resources:

- Data Explorer
- Metrics Advisor

Onboard Data Feed

Navigate to Metrics Advisor and then click the “**Go to your workspace**” link.



Complete the “**Welcome to Metrics Advisor!**” form, including:

Directory	Select your directory
Workspace	Select Metrics Advisor

Click the “Get started” button.

The screenshot shows the Microsoft Azure Metrics Advisor Data feeds Onboarding page. The left sidebar contains navigation links for Onboarding, Add data feed, Monitor & Diagnostic, Data feeds, Incident hub, Metrics graph, Settings, API keys, Hooks, Credential entity, Help, Documentation, and Public community. The main content area is titled "Data feeds" and features three steps: Step 1 (Onboard time-series data), Step 2 (Tune configuration & subscribe anomaly alerts), and Step 3 (Diagnose incidents). Each step has an associated icon and a list of tasks. A blue button at the bottom of the central column says "Onboard my first data feed".

practicumma - Microsoft Azure Metrics Advisor https://metricsadvisor.azurewebsites.net/data-feed

Data feeds

Onboarding Add data feed Monitor & Diagnostic Data feeds Incident hub Metrics graph Settings API keys Hooks Credential entity Help Documentation Public community

Step 1

Onboard time-series data

- Ingest your time-series data from various data sources supported

Step 2

Tune configuration & subscribe anomaly alerts

- Fine tune Detection configurations to better serve real-world scenarios
- Create a hook and subscribe real-time anomaly alerts

Step 3

Diagnose incidents

- Identify key contributors with dimension tree
- Chase down correlations with metrics graph

Onboard my first data feed

Click the “Onboard my first data feed” button.

Complete the “**Add data feed**” form, including:

Source Type	Select “Azure Data Explorer (Kusto)”
Granularity	Confirm default selection, Daily
Ingest Data Since (UTC)	Select a date appropriate for your data set
Authentication Type	Select “Managed Identity”
Connection String	Enter in form:
	<pre>Data Source=https://{{ADX Cluster}}.{{region}}.kusto.windows.net;Initial Catalog={{ADX Database}}</pre>
Query	Create and test a query in your database, then parameterize; example below:
	<pre>StormEvents summarize sum(DamageProperty) by startofday(StartTime), State where StartTime >= todatetime("@IntervalStart") and StartTime < todatetime("@IntervalEnd")</pre>

Click the “**Load Data**” button.

The screenshot shows the Metrics Advisor 'Add Data Feed' interface. On the left, a sidebar includes 'Credential entity', 'Help', 'Documentation', and 'Public community'. The main area has tabs for 'Query' and 'Schema configuration'. The 'Query' tab displays a Kusto-style query:

```
StormEvents  
| summarize sum(DamageProperty) by startofday(StartTime), State  
| where StartTime >= todatetime("@IntervalStart") and StartTime < todatetime("@IntervalEnd")
```

The 'Schema configuration' tab shows sample data and schema mapping. The sample data table has columns: State, StartTime, and sum_DamageProperty. The data rows are:

State	StartTime	sum_DamageProperty
NORTH CAROLINA	2007-01-01T00:00:00Z	0
WISCONSIN	2007-01-01T00:00:00Z	0
NEW YORK	2007-01-01T00:00:00Z	20000
ALASKA	2007-01-01T00:00:00Z	0
DELAWARE	2007-01-01T00:00:00Z	0
OKLAHOMA	2007-01-01T00:00:00Z	775000
INDIANA	2007-01-01T00:00:00Z	110000

The schema mapping table maps columns from the sample data to dimensions and measures. The columns are: Column name, Display name, Column type, Select: Timestamp, Dimension, Measure.

Column name	Display name	Column type	Select: Timestamp	Dimension	Measure
State	State	String	<input type="radio"/> Timestamp <input checked="" type="radio"/> Dimensions <input type="radio"/> Measure		
StartTime	Event Date	String	<input checked="" type="radio"/> Timestamp <input type="radio"/> Dimensions <input type="radio"/> Measure		
sum_DamageProperty	Count of Events	String	<input type="radio"/> Timestamp <input type="radio"/> Dimensions <input checked="" type="radio"/> Measure		

A blue 'Verify schema' button is located at the bottom right of the schema configuration section.

Click the “Verify Schema” button.

practicumma - Microsoft Azure Metrics Advisor https://metricsadvisor.azurewebsites.net/add-data-feed

Advanced settings Help ↗

Automatic roll-up settings Help ↗

My data has already rolled up and the dimension value is represented by .

I need the service to roll up my data by calculating and represent it by . Set roll-up columns

I do not need to perform root cause analysis into dimensions for my metrics.

Ingestion options Help ↗

Ingestion time offset hours Enter Max concurrency... Stop retrying after hours Minimum retrying interval hours

Data feed not available alert Help ↗

Alert hooks Grace period hours Snooze hours consecutive "data feed not available" alerts

By applying hooks, you can receive alerts.
You can [create and manage hooks](#).

Misc

Missing points filling for anomaly detection model Help ↗

Smart filling Fill previous Fill custom value: No filling

Action link template Help ↗

Tips: You can use these placeholders in the URL template: %datafeed%, %metric%, %timestamp%, %detect_config%, %tagset%.

Data feed name

Data feed name

Submit

No changes are required to default values in “**Advanced Settings**.”

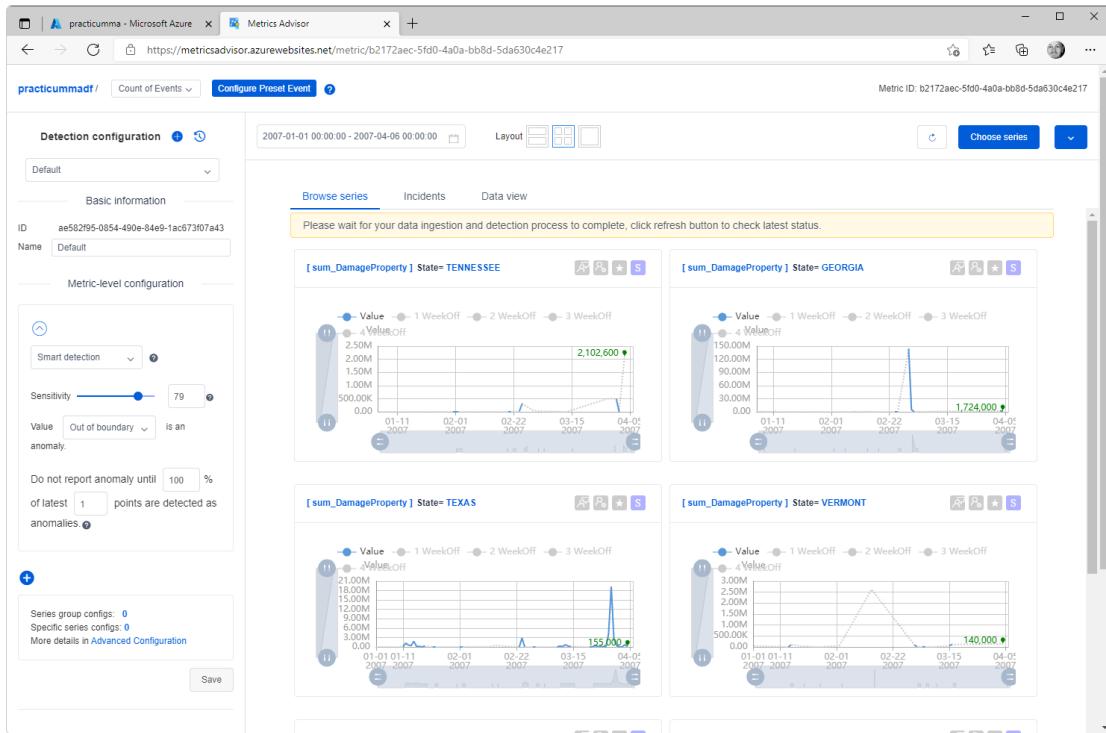
Enter a “**Data Feed Name**” and then click the Submit button.

You will receive a “**Congratulations...**” message.

Click the “**Visit data feed...**” button.

Monitor progress on the data feed page.

Click into the Metric Name link to see analysis.



Application+ AI

Use Case: Company XYZ wants to produce a customer-facing app that can be used to capture images and metadata for an insurance claim {e.g., a vehicle with damage to a headlight}.

Follow these instructions to build an application that employs artificial intelligence with minimal code.

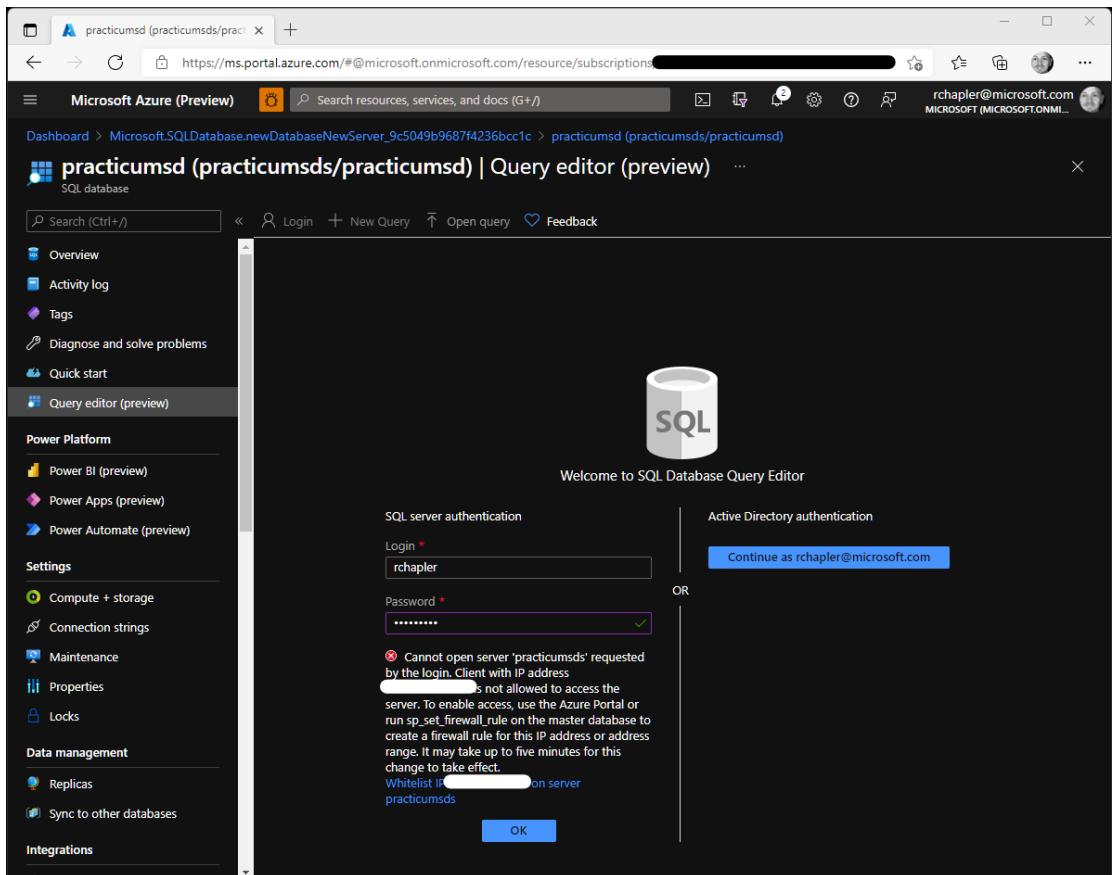
Prerequisites

This solution requires the following resources:

- Azure SQL
- PowerApps

Create Target Table

Navigate to SQL.



Navigate to “Query editor...” and login. Whitelist your IP address as appropriate.

The screenshot shows the Microsoft Azure portal (Preview) interface. The left sidebar has a 'Query editor (preview)' section selected. The main area shows a 'practicumsd (rchapler)' database with a 'Tables' folder containing 'dbo.myTable'. A query editor window titled 'Query 1' is open, displaying the following T-SQL code and its execution results:

```
1 CREATE TABLE dbo.myTable( Id INT NOT NULL IDENTITY(1,1) PRIMARY KEY, Name VARCHAR(64), Picture IMAGE )
```

The 'Messages' tab shows the message: 'Query succeeded: Affected rows: 0'. The status bar at the bottom also indicates 'Query succeeded | 0s'.

Execute the following T-SQL:

```
CREATE TABLE dbo.myTable( Id INT NOT NULL IDENTITY(1,1) PRIMARY KEY, Name VARCHAR(64), Picture IMAGE )
```

Create Canvas App

Navigate to Power Apps (<https://make.preview.Power Apps.com/>).

Click on **Apps** in the navigation pane.

At the top of the “**Apps**” form, click “+ New app” and select **Canvas** from the drop-down menu.

The screenshot shows the Power Apps 'Canvas app from blank' creation dialog. On the left, the navigation pane is visible with 'Apps' selected. The main dialog has the following fields:

- App name ***: rchapler
- Format**: Phone (radio button selected)
- Type**: Canvas (selected from a dropdown list)

A preview area shows a simple canvas with a pencil icon. Below it, there's a note: 'Design the app you want, and connect it to hundreds of data sources.' A 'Canvas app' button is available for further configuration. At the bottom right are 'Create' and 'Cancel' buttons.

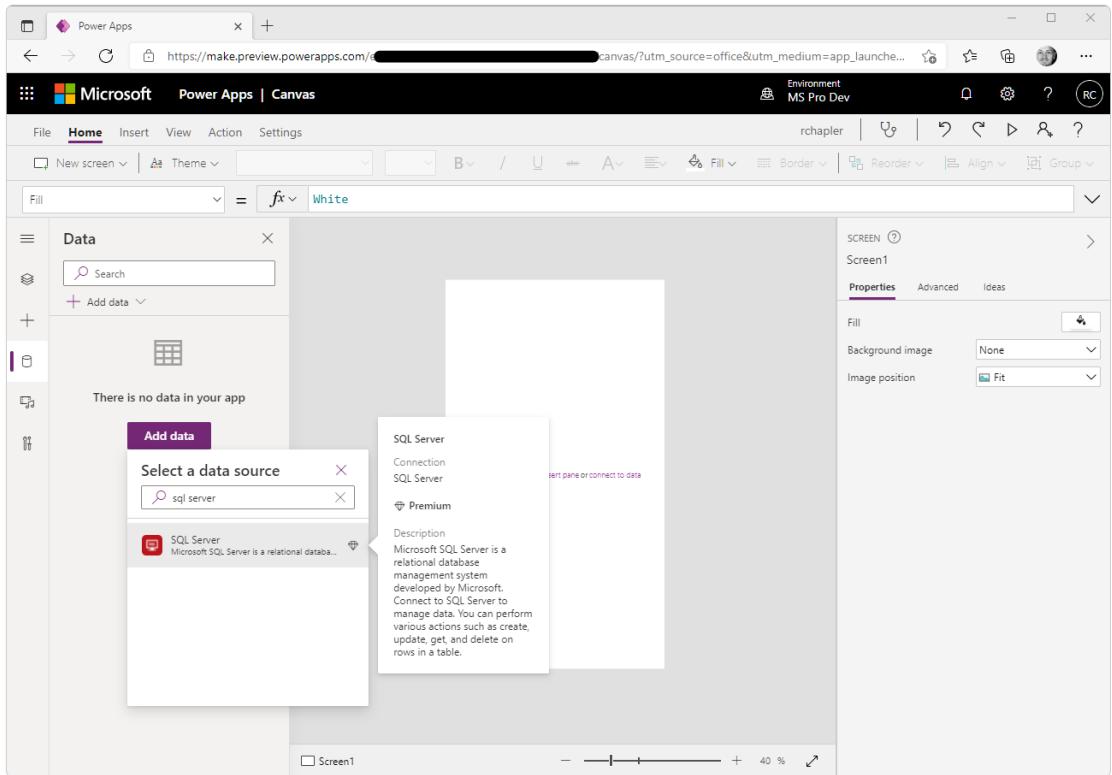
Complete the “**Canvas app from blank**” pop-up, enter the following items:

Format Select the **Phone** radio button

Click the **Create** button.

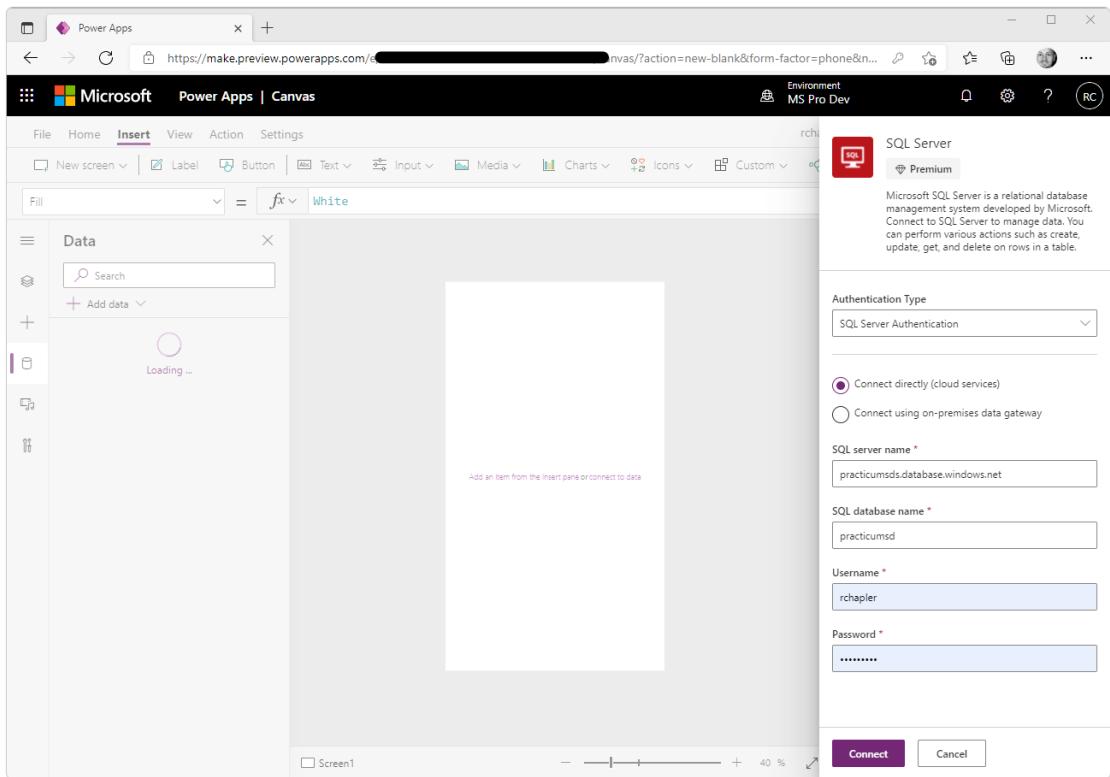
Add Data

Click on **Data** in the navigation pane. Click on the “**Add data**” button.



In “**Select a data source**”, enter “**sql server**” in the search input and select “**SQL Server**” from the results.

Click “**+ Add a connection**” on the resulting drop-down menu.



Complete the “**SQL Server**” pop-out, enter the following items:

Authentication Type SQL Server Authentication

SQL Server Name Enter the values employed during database server creation

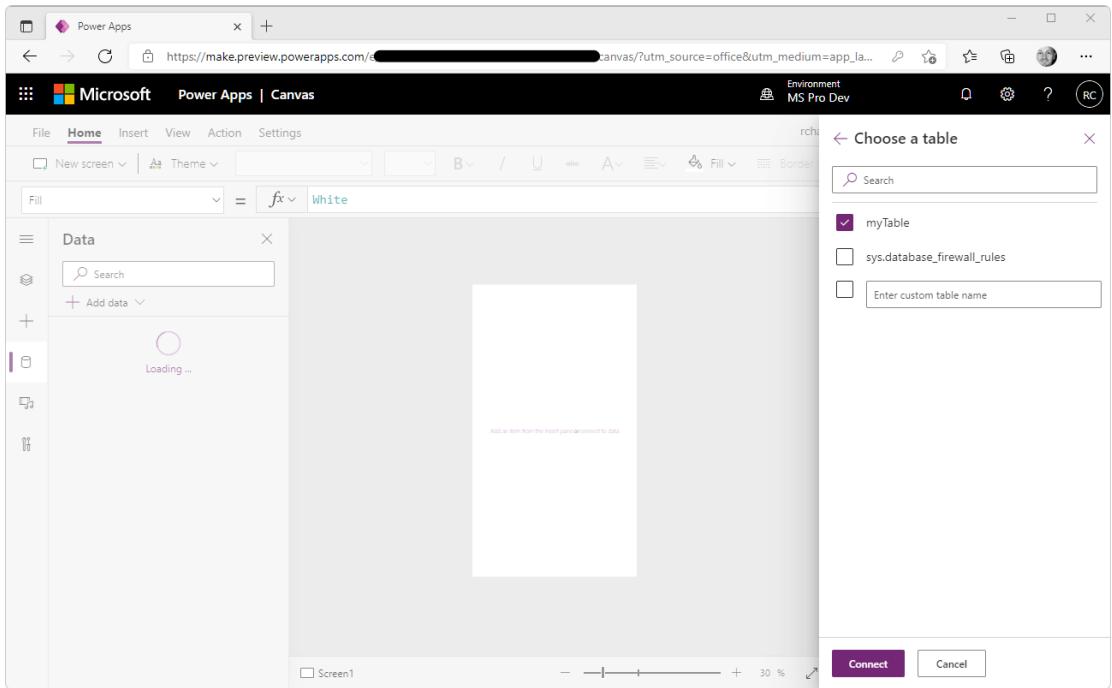
SQL Database Name

Username

Password

Click the **Connect** button.

Note: Resolve error messages “We weren’t able to add this connection...” and “Client with IP address #.#.#.# is not allowed...”, by adding your IP address to the Azure SQL Server firewall.



Complete the “**Choose a table**” pop-out, check the table created in [Create Target Table](#).

Click the **Connect** button.

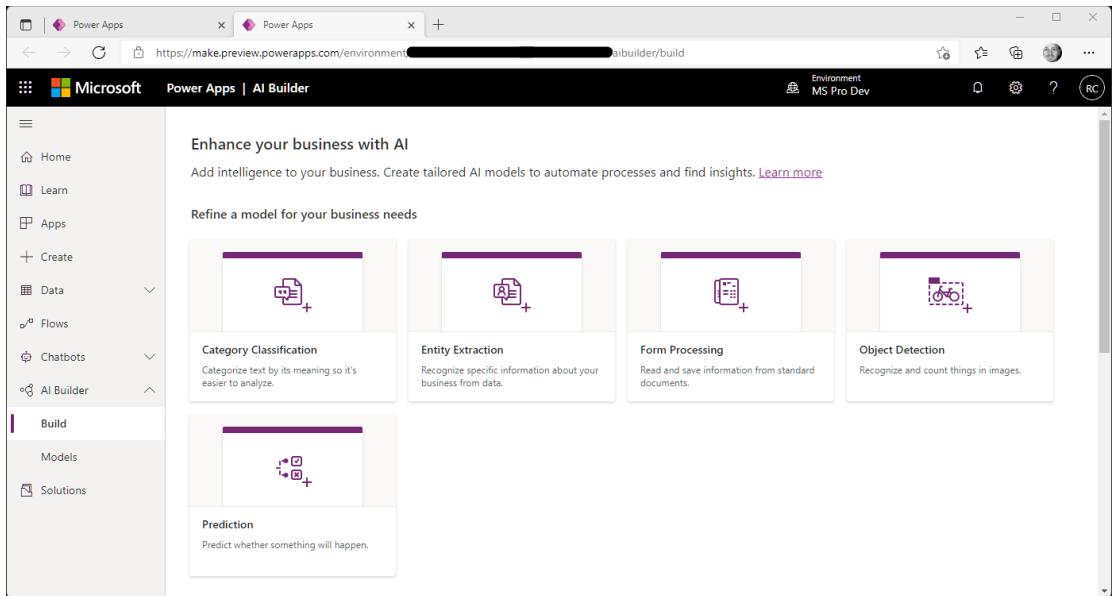
Insert Object Detector

Click the “Add an item from the Insert pane” link.

On the resulting **Insert** menu, expand “**AI Builder**” and select “**Object detector**.”

On the resulting pop-out, click “**+ New model**.”

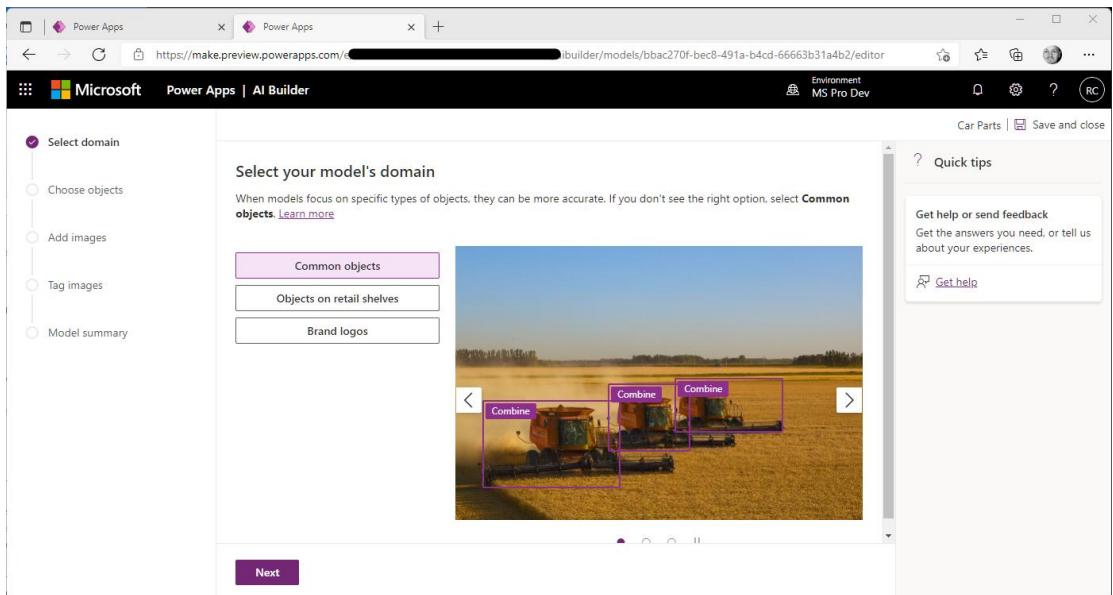
Create AI Model



The screenshot shows the Microsoft Power Apps | AI Builder interface. On the left, there's a navigation sidebar with options like Home, Learn, Apps, Create, Data, Flows, Chatbots, AI Builder, Build, Models, and Solutions. The main area has a heading "Enhance your business with AI" and sub-sections for "Refine a model for your business needs". It lists four categories: "Category Classification" (categorize text by its meaning), "Entity Extraction" (recognize specific information from data), "Form Processing" (read and save info from standard documents), and "Object Detection" (recognize and count things in images). Below these is another section for "Prediction" (predict whether something will happen).

On the new “Power Apps | AI Builder”, “Enhance your business with AI” form, click the “Object Detection” button.

Name the AI model and then click the **Create** button.



The screenshot shows the "Select your model's domain" step in the AI Builder wizard. On the left, there's a sidebar with steps: "Select domain" (which is checked), "Choose objects", "Add images", "Tag images", and "Model summary". The main area has a heading "Select your model's domain" with a sub-note about focusing on specific object types for accuracy. It shows three options: "Common objects" (selected), "Objects on retail shelves", and "Brand logos". To the right is a preview image of a field with three combine harvesters, each highlighted with a purple bounding box and the word "Combine". There are "Next" and "Back" buttons at the bottom. A "Quick tips" sidebar on the right provides help and feedback.

Complete the “Select your model’s domain” form, click the “Common objects” and then the “Next” button.

Screenshot of the Microsoft Power Apps AI Builder interface showing the "Choose objects for your model to detect" step. The left sidebar shows steps: "Select domain" (done), "Choose objects" (selected), "Add images", "Tag images", and "Model summary". The main area has a title "Choose objects for your model to detect" with a note "You can add them manually or select from your database." A text input field contains "Headlights". A button "+ Add new object" is below it. To the right, a "Quick tips" panel suggests "Select from database instead" and "Get help or send feedback". Buttons at the bottom are "Back", "Next", and "1 object name selected".

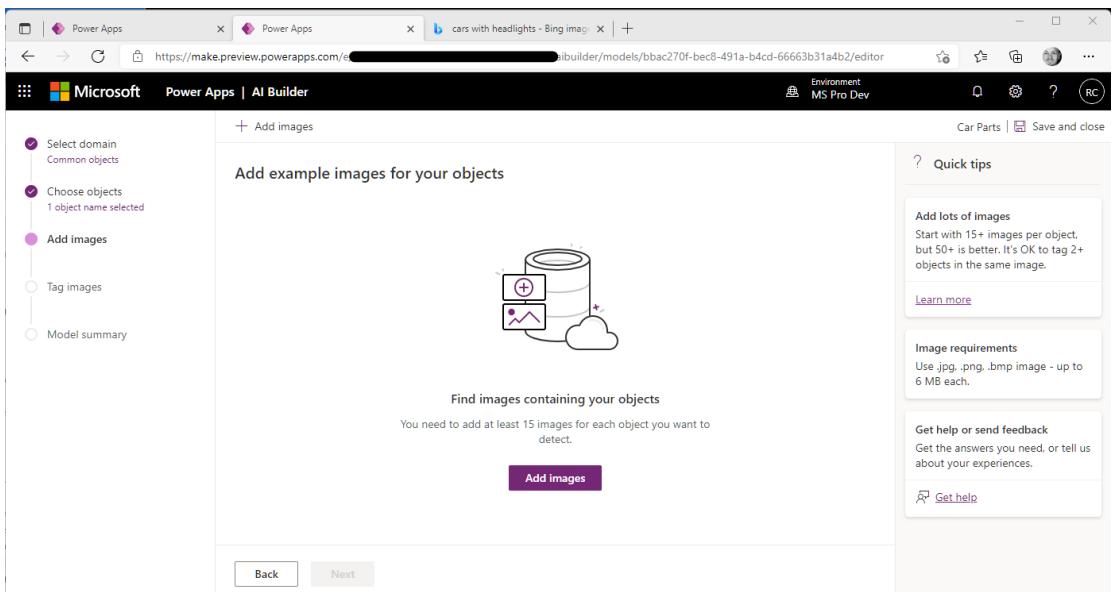
Complete the “**Choose objects for your model to detect**” form, click “**+ Add new object**”, enter an object name and then click the “**Next**” button.

Complete the “**Add example images for your objects**” form, we will add image files that will be used to train the AI model.

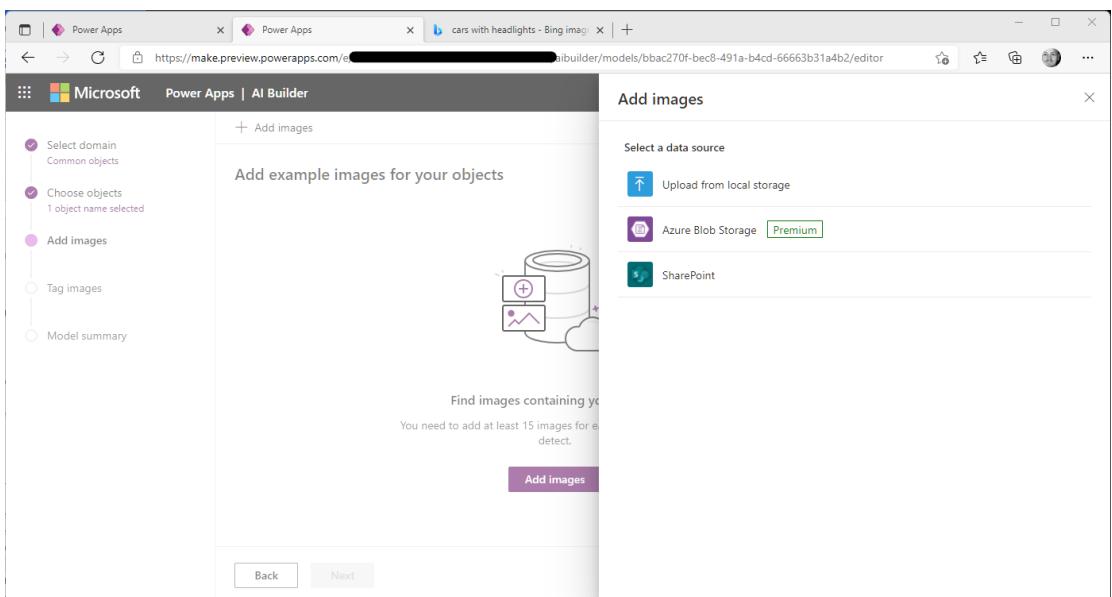
For this exercise, we can pull images of cars (with headlights) from a search engine.

Screenshot of a web browser showing search results for "cars with headlights" on Bing Images. The search bar shows the query. Below it are filter buttons for "Round", "LED Lights", "Night", "Parts", "Replacement", "Front", and "Vintage". The main area displays a grid of 15 car images, mostly featuring headlights. A "Feedback" button is visible in the bottom right corner of the image grid.

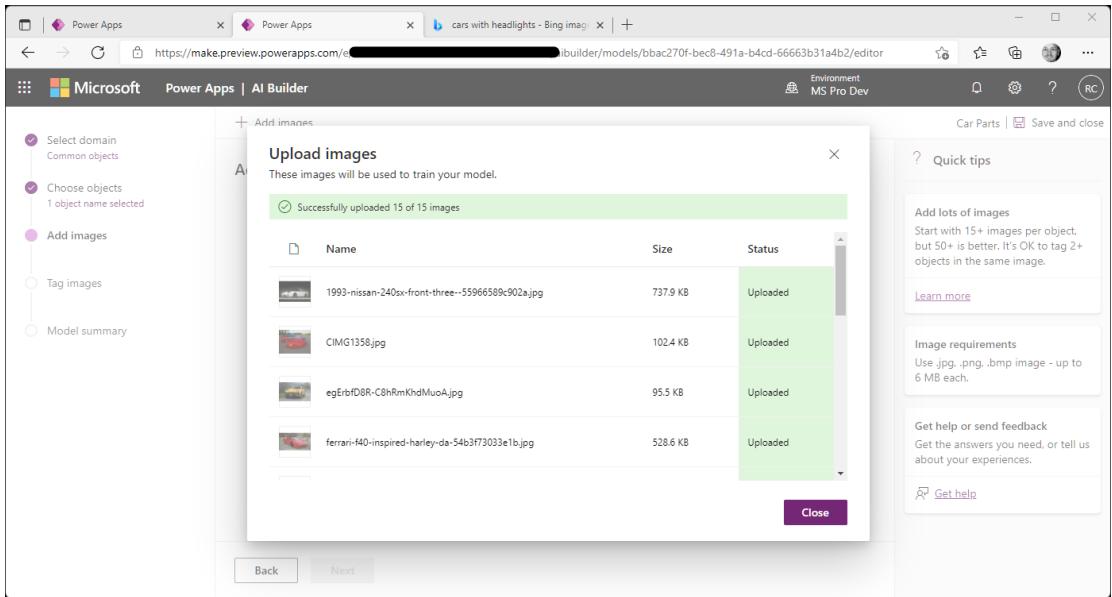
For each of the required 15 images, download an image file to a temporary folder on the Windows desktop.



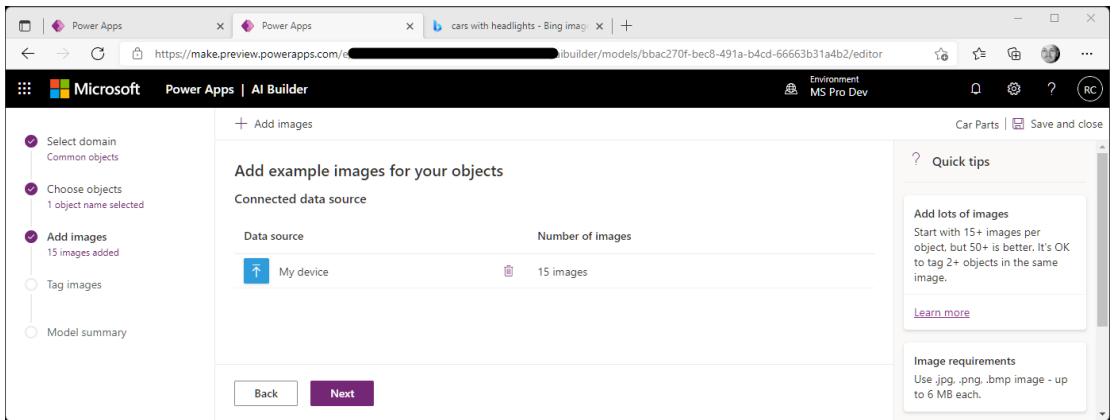
Complete the “Add example images for your objects” form, click the “Add images” button.



On the resulting “**Add images**” pop-out, click “**Upload from local storage**.” Select the Example Images downloaded to your desktop.



Review the items in the “Upload images” pop-up, then click the “Upload 15 images” button. Click **Close**.



Back on the “Add example images for your desktop” form, confirm “Number of images” shows 15 or more images, then click **Next**.

Select domain
Common objects

Choose objects
1 object name selected

Add images
15 images added

Tag images

Model summary

You will need to tag at least 15 images per object to continue.

Tag the objects in your images

All

Tagging requirements
You must tag at least 15 images for each object. Tagging more than 50 images for each object could yield better results. You can have multiple tags per image.

0 Headlights

Want to add more images?
Adding more images may increase model performance.

+ Add images

Get help or send feedback
Get the answers you need, or tell us about your experiences.

Get help

Back Next

Select the first image on the “Tag the objects in your images” page.

Don't use image Clear Discard changes Details ✓ Done tagging

Choose object

Search

Headlights

Tagging progress

Image 1 of 15

You must tag at least 15 images for each object. Tagging more than 50 images for each object could yield better results. You can have multiple tags per image.

Tags applied

0 Headlights

Drag a rectangle around headlights in the image, then click the Headlights button on the “Choose object” pop-up.

Repeat for all taggable items, click > to move to through images, and then click the “Done tagging” button in the upper-right.

Tag the objects in your images

All

15 Headlights

Want to add more images? Adding more images may increase model performance.

+ Add images

Get help or send feedback Get the answers you need, or tell us about your experiences.

Get help

Click the **Next** button.

Model summary

Review your model's details below. If everything looks good, select Train. [Learn more about training](#)

Overview

Model type Object Detection	Owner Rich Chapter	Object type Common objects
--------------------------------	-----------------------	-------------------------------

Image sources

Data source My device	Number of images 15 images
--------------------------	-------------------------------

Information to extract

Object Headlights	Tags 15
----------------------	------------

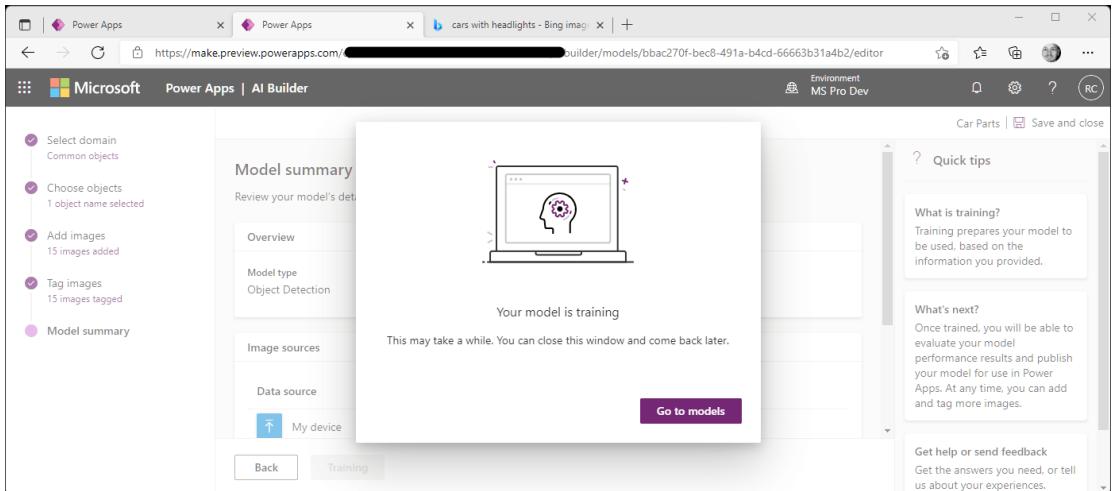
What is training? Training prepares your model to be used, based on the information you provided.

What's next? Once trained, you will be able to evaluate your model performance results and publish your model for use in Power Apps. At any time, you can add and tag more images.

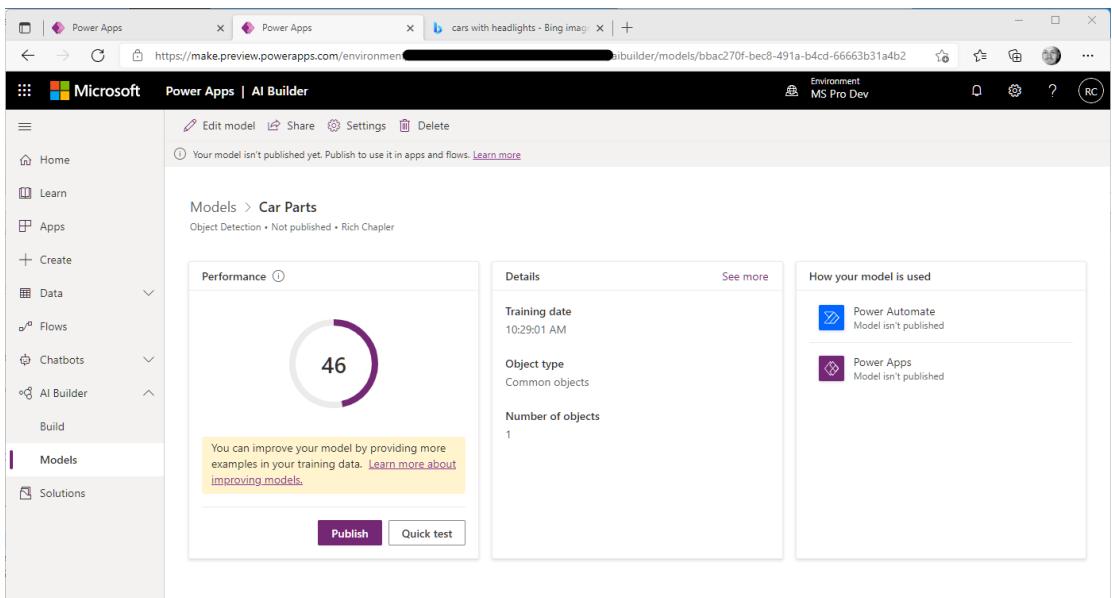
Get help or send feedback Get the answers you need, or tell us about your experiences.

Get help

Review the “**Model Summary**” and then click the **Train** button.



Click the “Go to models” button. After Status changes to **Trained**, navigate to the model.

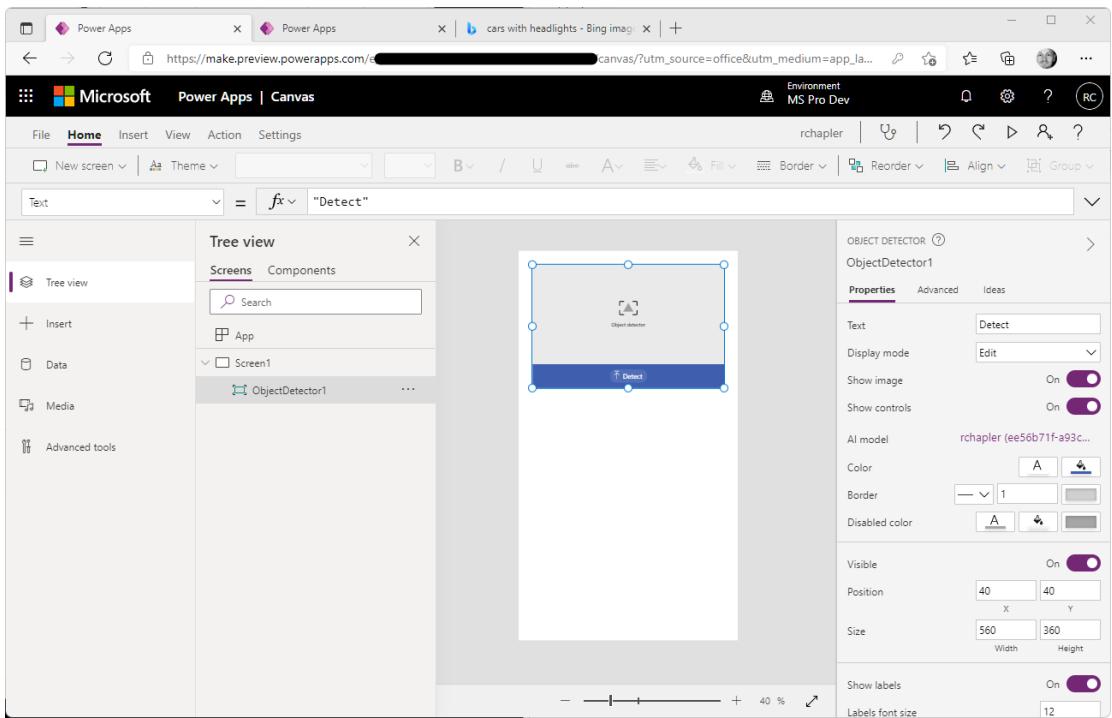


In the snip above, we see that **Performance** {i.e., model performance expectation %} of 46 for the example images included in training.

In the future, you can improve model quality by including more images and re-training the model.

Click the **Publish** button.

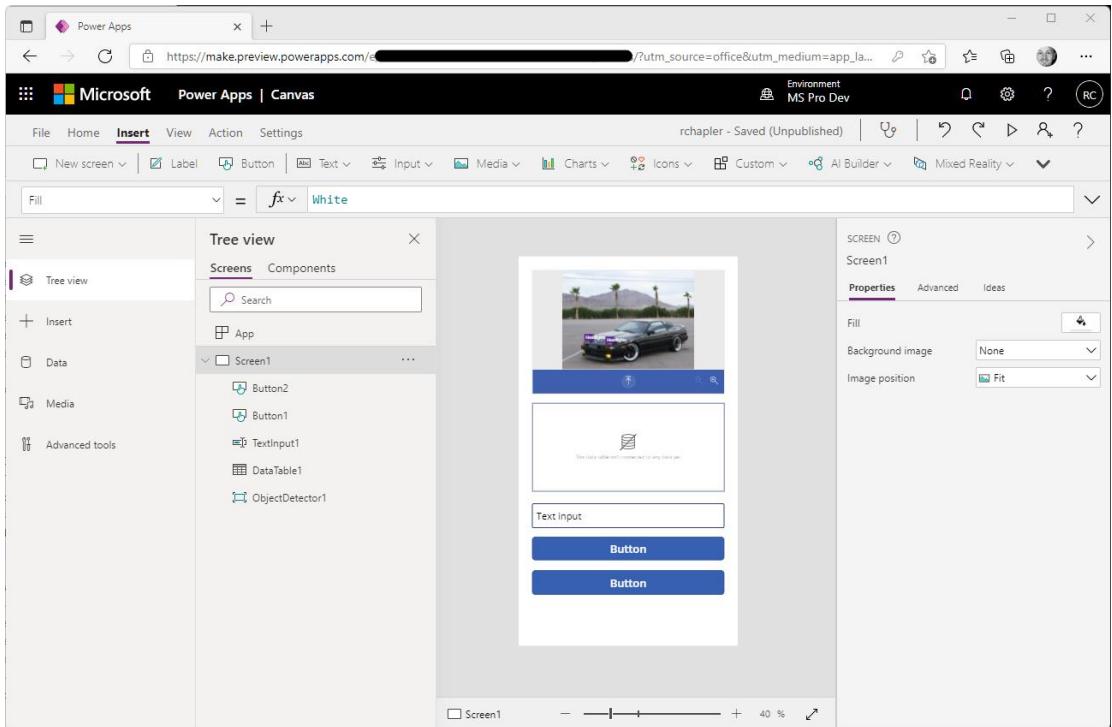
Return to Power Apps and complete the **ObjectDetector1** component by selecting your newly created model.



Note: Consider resizing the component to improve visibility.

Insert / Configure Controls

Return by closing Preview Mode.

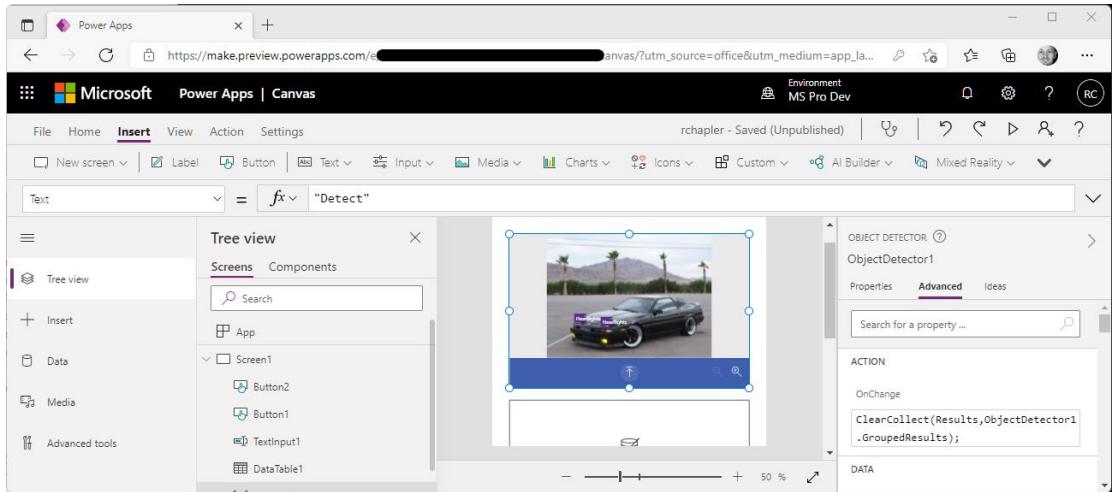


Insert the following controls:

- **Data Table** ... users will use this control to view the results generated by AI Builder
- **Text Input** ... users will use this control to enter comments about the analyzed image
- “Submit” **Button** ... users will use this control to save the image and comment to Azure SQL
- “Reset” **Button** ... users will use this control to clear previously entered values

You might note that I employed formatting {e.g., control width, border size, color, etc.} to enhance usability.

ObjectDetector1

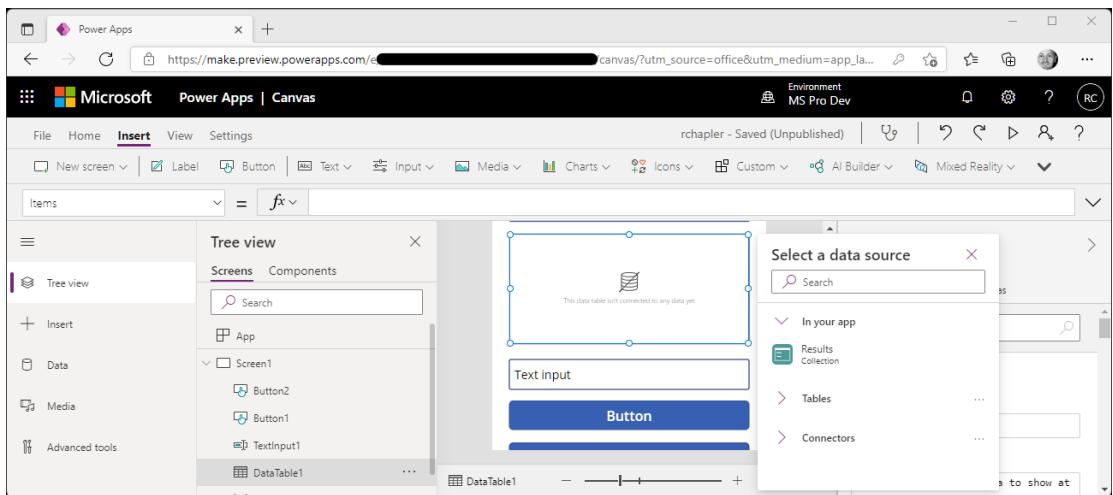


Click on **ObjectDetector1** in the “Tree view” pane. On the resulting pop-out, click the **Advanced** tab.

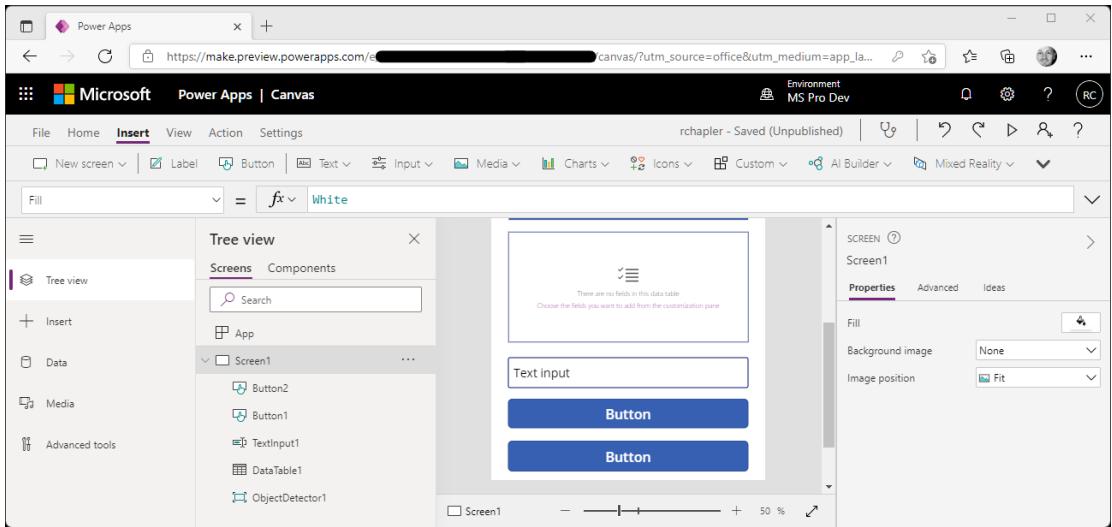
Paste the following logic into **Action | OnChange**:

```
ClearCollect(Results, ObjectDetector1.GroupedResults);
```

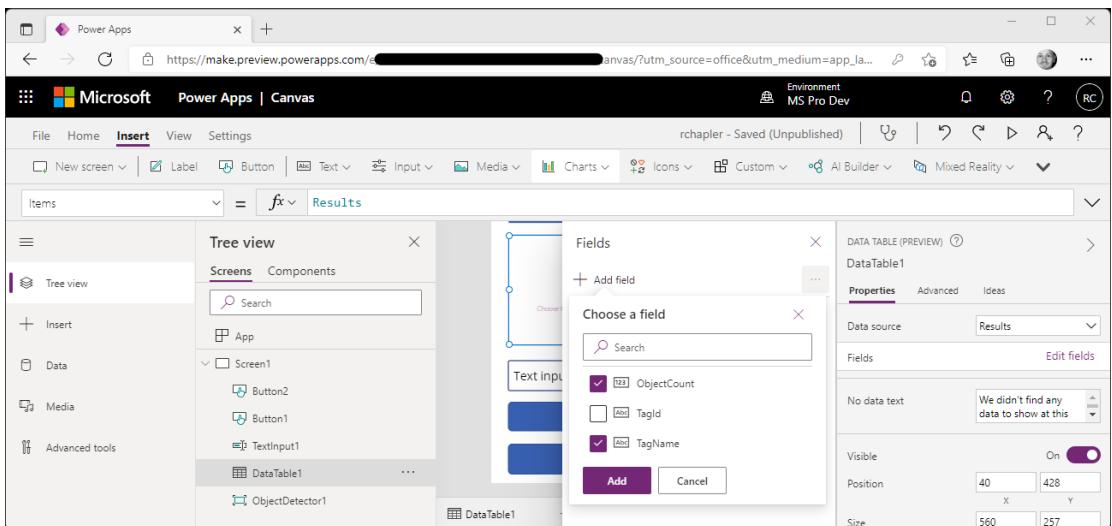
DataTable1



Click on **DataTable1** in the “Tree view” pane. On the resulting pop-up, select “**In your app** | **Results**”.



On the updated control, click the “Choose the fields you want to add...” link.

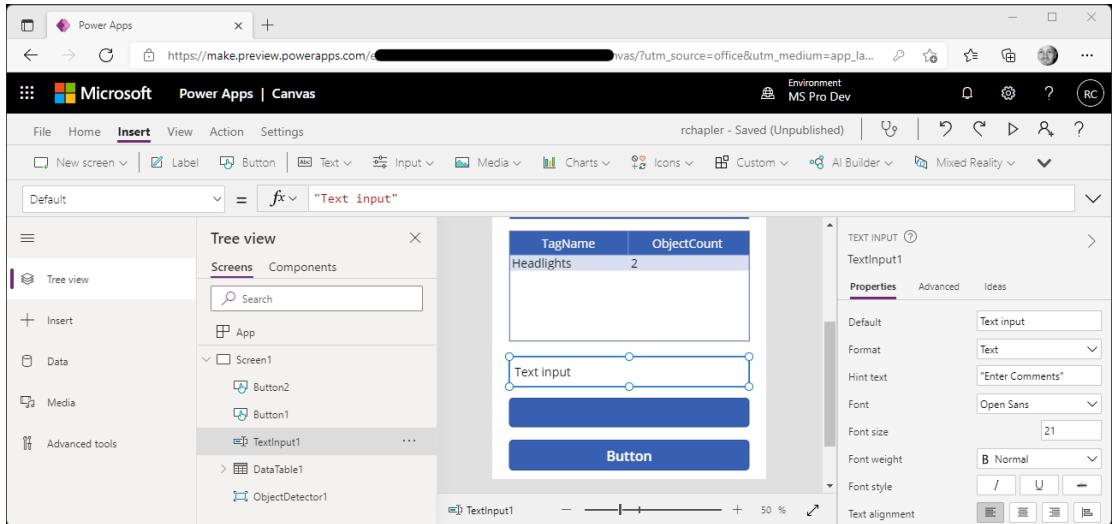


On the resulting **Fields** pop-out, click “+ Add field.”

In the resulting “Choose a field” pop-up, check **TagName** and then **ObjectCount**.

Click the **Add** button.

TextInput1

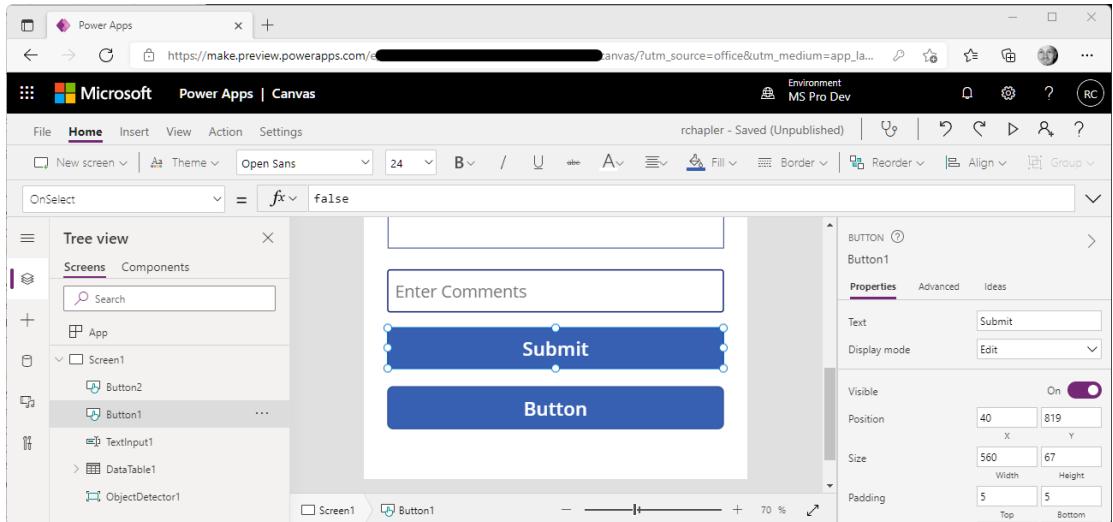


Click on **TextInput1** in the “Tree view” pane.

On the resulting “TEXT INPUT” pop-out, **Advanced** tab, enter the following items:

Default	""
Hint Text	“Enter Comments”

Button1



Click on **Button1** in the “Tree view” pane.

On the resulting pop-out, click the **Properties** tab, and enter the word “Submit” in the **Text** input.

On the resulting pop-out, click the **Advanced** tab,

Paste the following logic into **Action | OnSelect**:

```
Patch(myTable, Defaults(myTable), {Name:TextInput1.Text, Picture:ObjectDetector1.OriginalImage})
```

Button2

The screenshot shows the Microsoft Power Apps Canvas editor interface. In the center, there is a canvas with a text input field labeled "Enter Comments" and two blue buttons below it: "Submit" and "Reset". The "Reset" button is selected. In the top navigation bar, "Insert" is highlighted. The ribbon menu shows "File", "Home", "Insert", "View", "Action", and "Settings". The "Action" tab is active, showing the logic "OnSelect = fx Reset(TextInput1);". The "Properties" tab is also visible. On the left, the "Tree view" pane shows the app structure with "Screen1" expanded, containing "Button2", "Button1", and "TextInput1". The "Advanced" tab is selected in the properties pane on the right, where the "OnSelect" action is defined as "Reset(TextInput1);".

Click on **Button2** in the “**Tree view**” pane.

On the resulting pop-out, click the **Properties** tab, and enter the word “Reset” in the **Text** input.

On the resulting pop-out, click the **Advanced** tab,

Paste the following logic into **Action | OnSelect**:

```
Reset(TextInput1);
```

Confirm Success

Click **File** in the menu bar, then “**Save As**”, confirm the app name, and finally, click the **Save** button in the bottom-right.

Return to the main form, click the “**Preview the app**” button {i.e., Play icon} in the upper-right.

Power Apps

Microsoft Power Apps | Canvas

File Home Insert View Action Settings

New screen Label Button Text Input Media Charts Icons Custom AI Builder

Fill = White

Tree view

Screens Components

Search

App

Screen1

Button2

Button1

TextInput1

DataTable1

ObjectDetector1

Preview the app (F5)

Preview the app to see how it works.

SCREEN Screen1

Properties Advanced Ideas

Search for a property ...

ACTION

OnVisible

OnHidden

DATA

BackgroundImage

ContentLanguage

...

DESIGN

Fill

white

ImagePosition

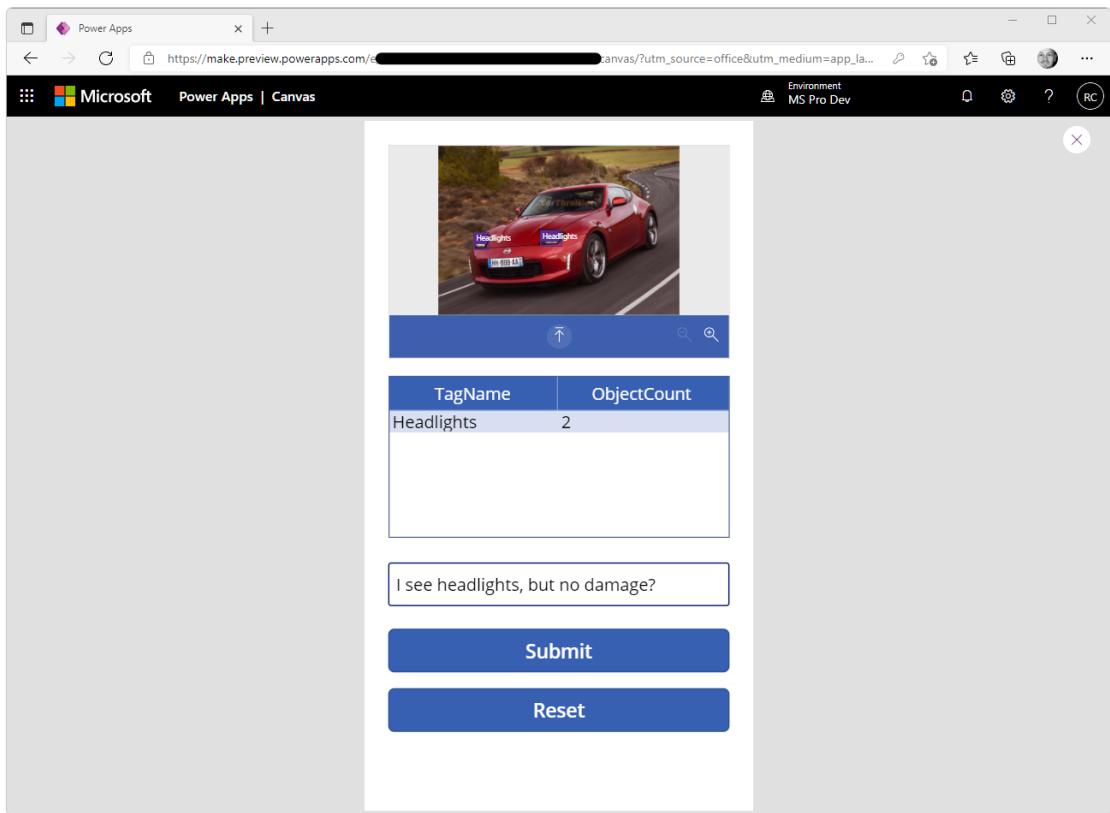
ImagePosition.Fit

The screenshot shows the Microsoft Power Apps Canvas interface. On the left, the 'Tree view' pane lists components: App, Screen1 (selected), Button2, Button1, TextInput1, DataTable1, and ObjectDetector1. The main canvas area displays a red sports car on a road. Below the image is a table with two rows:

TagName	ObjectCount
Headlights	2

Below the table are two buttons: 'Enter Comments' and 'Submit'. A 'Reset' button is also present. The right side of the interface shows the 'Screen1' properties pane under the 'Advanced' tab, with sections for ACTION (OnVisible, OnHidden), DATA (BackgroundImage, ContentLanguage, ...), and DESIGN (Fill, ImagePosition, ImagePosition.Fit). The 'Fill' field is set to 'white' and the 'ImagePosition' field is set to 'ImagePosition.Fit'.

Your app will be presented in a phone simulation.



Click the **Detect** button and select an image file to test objection detection.

Review the resulting values in the data table.

Enter a comment in the text input and click the **Submit** button.

practicumsd (practicumsd/practicumsd) | Query editor (preview)

Dashboard > practicumsd > practicumsd (practicumsd/practicumsd)

practicumsd (rchapler) | Query editor (preview)

Showing limited object explorer here. For full capability please open SSDT.

Tables

dbo.myTable

Views

Stored Procedures

Query 1 × Query 2 ×

Run Cancel query Save query Export data as Show only Editor

1 SELECT TOP (1000) * FROM [dbo].[myTable]

Id	Name	Picture
1	Nice car, but no apparent quality d...	/9j/4AAQSkZJRgABAQEAAAAAAA...
2	I see headlights, but no damage?	/9j/2wBDAAYEBQYFBAYGBQYHbwY...

Results Messages

Query succeeded | 0s

Navigate to **Query Editor** in the Azure SQL Database and confirm that your comment has been added as a new row.

Consider publishing your app and loading it on your phone as one additional confirmation of success.

Audit Usage

Follow these instructions to support regular audit of and alerting on key data products.

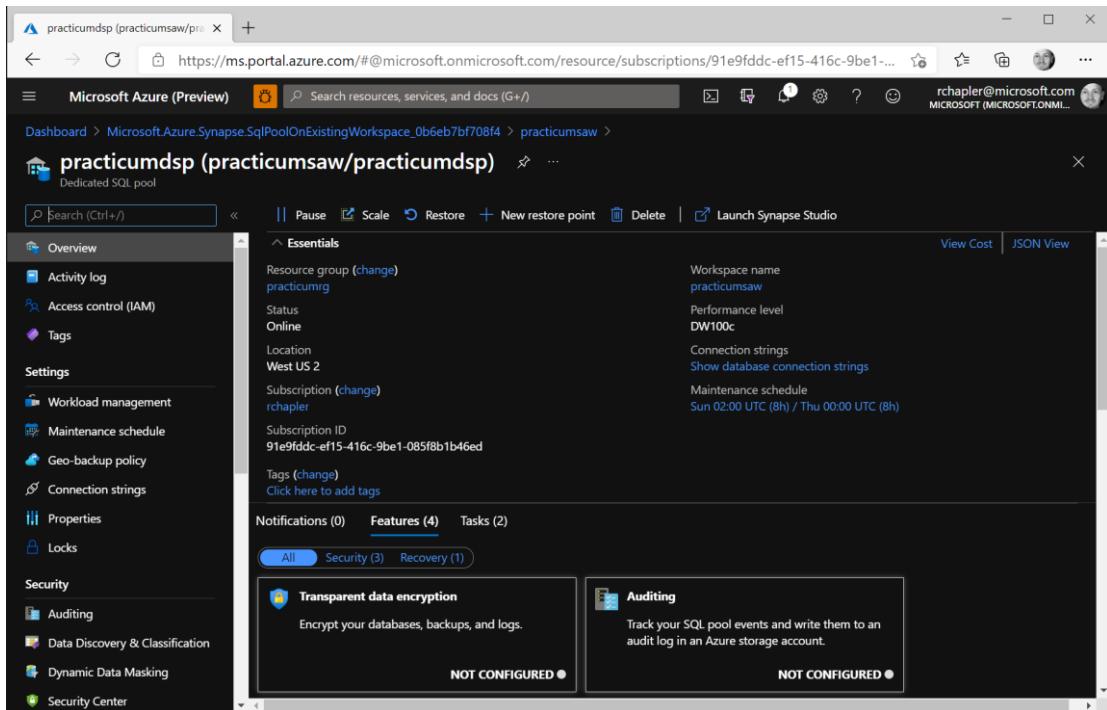
Prerequisites

This solution requires the following resources:

- Synapse

Configure Auditing

Navigate to the **Overview** page in Synapse, Dedicated SQL Pool.



The screenshot shows the Microsoft Azure portal with the URL <https://ms.portal.azure.com/#@microsoft.onmicrosoft.com/resource/subscriptions/91e9fddc-ef15-416c-9be1-085f8b1b46ed/resourceGroups/practicumsg/providers/Microsoft.Synapse/workspaces/practicum dsp>. The page displays the 'practicum dsp (practicum saw/practicum dsp)' Dedicated SQL pool. On the left, a navigation menu includes 'Overview', 'Activity log', 'Access control (IAM)', 'Tags', 'Settings', 'Workload management', 'Maintenance schedule', 'Geo-backup policy', 'Connection strings', 'Properties', 'Locks', 'Security' (with 'Auditing' selected), 'Data Discovery & Classification', 'Dynamic Data Masking', and 'Security Center'. The main content area shows 'Essentials' information: Resource group (change) practicumsg, Workspace name practicum saw, Status Online, Performance level DW100c, Location West US 2, Subscription (change) rchapler, Subscription ID 91e9fddc-ef15-416c-9be1-085f8b1b46ed, and Tags (change). Below this are sections for 'Notifications (0)', 'Features (4)', and 'Tasks (2)'. Under 'Features (4)', there are two cards: 'Transparent data encryption' (NOT CONFIGURED) and 'Auditing' (NOT CONFIGURED). The 'Auditing' card has a description: 'Track your SQL pool events and write them to an audit log in an Azure storage account.'

Click the **Auditing** button.

practicumdsp (practicumsaw/practicumdsp) | Auditing

Search (Ctrl+ /)

Save Discard View audit logs Feedback

If Azure SQL Auditing is enabled on the workspace, it will always apply to the SQL Pool, regardless of the SQL Pool settings.

View workspace settings

Workspace-level Auditing: **Disabled**

Azure SQL Auditing

Azure SQL Auditing tracks SQL Pool events and writes them to an audit log in your Azure storage account. [Learn more about Azure SQL Auditing](#)

Enable Azure SQL Auditing

ON OFF

Storage details >

practicumsa

Flip “**Enable Azure SQL Auditing**” switch to **ON** position. Click to configure “**Storage details**.” Save changes and then click “**View audit logs**.”

Audit records - Microsoft Azure

Refresh Filter Log Analytics View dashboard

Click here to learn more about methods for viewing & analyzing audit records.

Audit source: **Workspace audit** **SQL Pool audit**

Showing audit records up to Wed, 03 Feb 2021 20:58:37 UTC.

Event time (UTC)	Principal name	Event type	Action status
2/3/2021 8:58:30 PM	##MS_InstanceCertificate##	BATCH COMPLETED	Succeeded
2/3/2021 8:56:26 PM	##MS_InstanceCertificate##	BATCH COMPLETED	Succeeded

Click into one of “**BATCH COMPLETED**” audit records.

Audit record - Microsoft Azure

Microsoft Azure (Preview)

Search resources, services, and docs (G+/)

Dashboard > practicumdsp (practicumsaw/practicumdsp) > Audit records > Audit record

Event time (UTC)
2/8/2021 7:32:50 PM

Event type
BATCH COMPLETED

Server name
practicumsaw

Database name
practicumdsp

Application name
SynapseSqlEditor

Principal name
rchapler@microsoft.com

Client IP
76.121.194.132

Status
Succeeded

STATEMENT

```
SELECT
    s.name AS [schema_name],
    o.name AS [object_name], o.type AS [object_type], o.type_desc AS
    [object_type_desc],
    c.name AS [column_name], TYPE_NAME(c.system_type_id) AS
    [column_type]
FROM sys.schemas s
LEFT OUTER JOIN sys.all_objects o
    ON s.schema_id = o.schema_id
LEFT OUTER JOIN sys.all_columns c
    ON c.object_id = o.object_id
```

Expand statement

Governance

Discover Data

Data discovery is “an organic process based on communal knowledge”, characterized by the following questions:

- **Search** ... “does the data that I need exist?”
- **Location** ... “how do I connect to Data Product X?”
- **Related** ... “are there related data products that I might find valuable?”
- **Contacts** ... “who are the owners and experts for Data Product X?”

In this section, we will configure Purview and successfully respond to these questions.

Prerequisites

This solution requires the following resources:

- Purview
- SQL

Register Source

Navigate to Purview in the Azure portal.

Click the **“Open Purview Studio”** button.

Click the **Data Map** icon on the navigation pane.

Click the **Register** button on the resulting page.

Click the **Azure** tab on the **“Register sources”** pop-out.

Click to check the **“Azure SQL Database”** option and then click the **Continue** button.

The screenshot shows the Microsoft Azure Purview Studio interface. On the left, there's a sidebar with icons for Sources, Collections, Source management, Scan rule sets, Integration runtimes, Annotation management, Classifications, and Classification rules. The main area has a blue header bar with 'Microsoft Azure' and 'practicump' navigation. Below the header is a search bar labeled 'Search assets'. The central part of the screen displays a 'Sources' list with a 'Register' button and a 'Map view' button. A 'Filter by name' input field is present. To the right, a 'Register sources (Azure SQL Database)' dialog box is open. It contains fields for 'Name' (set to 'practicumsd'), 'Azure subscription' (set to 'rchapler (91e9ffdc-ef15-416c-9be1-085f8b1b46ed)'), 'Server name' (set to 'practicumsd'), 'Endpoint' (set to 'practicumsd.database.windows.net'), and 'Select a collection' (set to 'practicump'). A note below the collection dropdown says 'All assets under this source will belong to the collection you select.' At the bottom of the dialog are 'Register', 'Back', and 'Cancel' buttons.

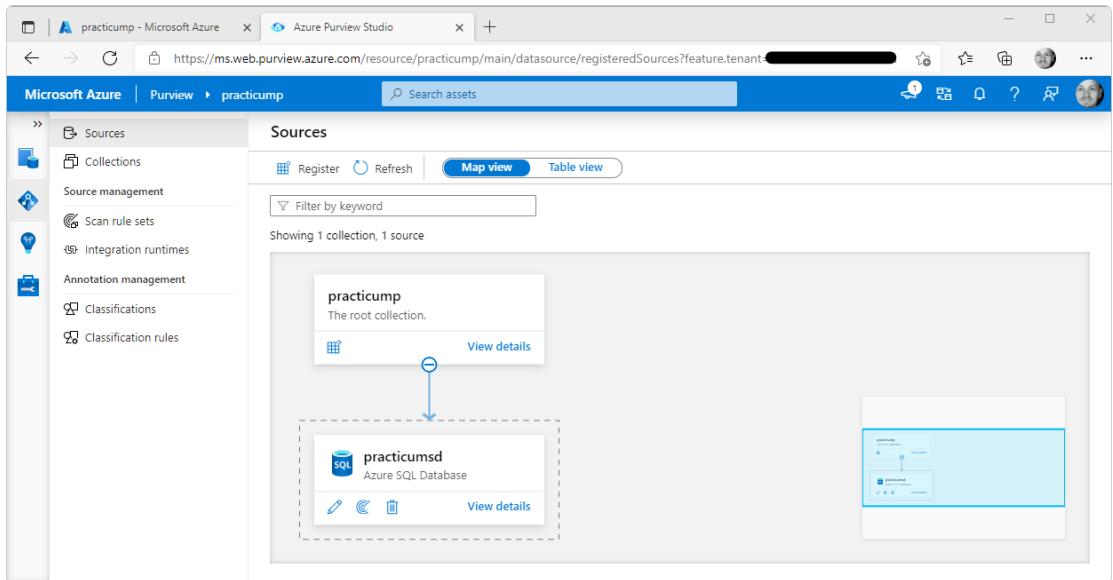
Complete the “**Register sources (Azure SQL Database)**” pop-out, including:

Server Name	Enter the name of your SQL Database Server
Endpoint	Confirm the populated value
Select a Collection	Leave default value

Click the **Register** button.

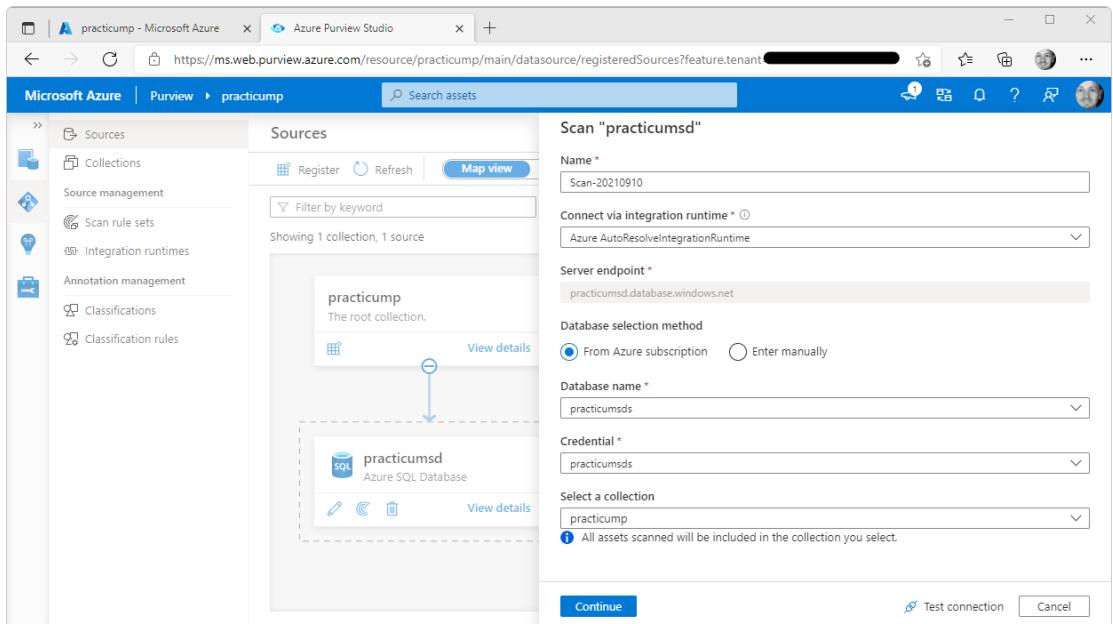
Scan Data

Navigate to Sources, “Map view.”



The screenshot shows the Azure Purview Studio interface. The left sidebar has a 'Sources' section selected. The main area is titled 'Sources' with tabs for 'Map view' (which is active) and 'Table view'. Below is a search bar and a filter box. The results show one collection and one source: 'practicump' (The root collection) and 'practicumsd' (Azure SQL Database). Each item has a 'View details' button.

Click the “New scan” icon on the newly created item.



The screenshot shows the 'Scan "practicumsd"' configuration dialog. The 'From Azure subscription' radio button is selected under 'Database selection method'. Other fields include:

- Name: Scan-20210910
- Connect via integration runtime: Azure AutoResolveIntegrationRuntime
- Server endpoint: practicumsd.database.windows.net
- Database name: practicumsds
- Credential: practicumsds
- Select a collection: practicump

At the bottom are 'Continue', 'Test connection', and 'Cancel' buttons.

Complete the “Scan...” pop-out, including:

Database Selection Method Select the “From Azure subscription” radio button

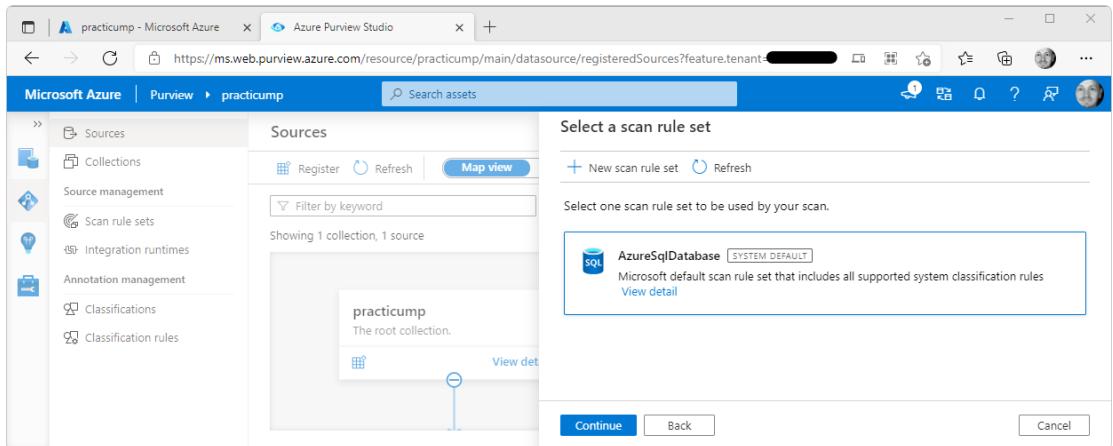
Database Name Select your Azure SQL database

Credential Select your “SQL Authentication” credential

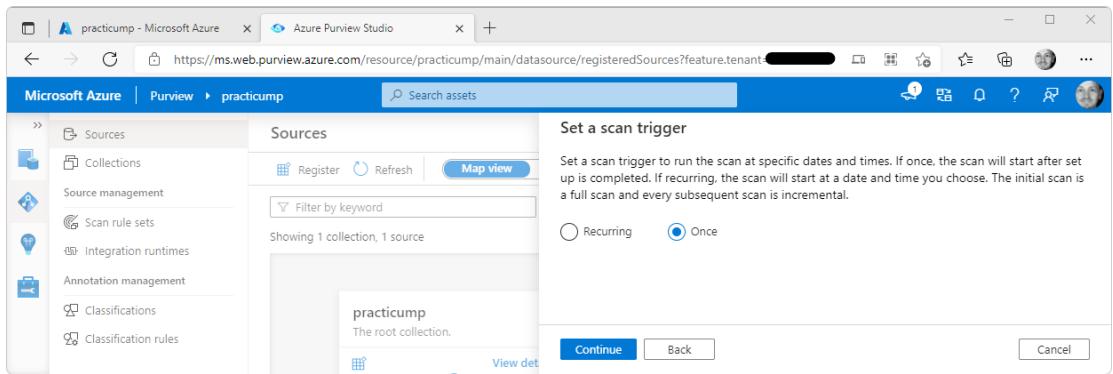
Click “**Test connection**” to confirm successful configuration and then click the **Continue** button.

The screenshot shows the Microsoft Purview Studio interface. On the left, there's a sidebar with various management options like Sources, Collections, Scan rule sets, and Integration runtimes. The main area is titled 'Sources' and shows a collection named 'practicump' which contains one source named 'practicumsd'. This source is identified as an 'Azure SQL Database'. To the right of the source list is a 'Scope your scan' dialog. It includes a 'Search' bar and a list of data products under the 'practicumsd' database. Most items in the list have a checked checkbox next to them. The list includes: SalesLT.Customer, SalesLT.ProductModel, SalesLT.ProductDescription, SalesLT.Product, SalesLT.ProductModelProductDescription, SalesLT.ProductCategory, dbo.BuildVersion, dbo.ErrorLog, SalesLT.Address, SalesLT.CustomerAddress, SalesLT.SalesOrderDetail, SalesLT.SalesOrderHeader, SalesLT.vProductModelCatalogDescription, SalesLT.vProductAndDescription, and SalesLT.vGetAllCategories. At the bottom of the dialog are 'Continue', 'Back', and 'Cancel' buttons.

Review and select desired data products from those listed in the “**Scope your scan**” pop-out and then click the **Continue** button.



Confirm selection of the default Scan Rule Set on the “**Select a scan rule set**” pop-out and then click the **Continue** button.



Click the **Once** radio button on the “**Set a scan trigger**” pop-out and then click the **Continue** button.

The screenshot shows the Azure Purview Studio interface. On the left, a sidebar menu includes Sources, Collections, Source management (Scan rule sets, Integration runtimes), Annotation management (Classifications, Classification rules), and Purview Home.

The main area is titled "Sources" and displays a tree view of collections. At the top right of the Sources section are "Register" and "Refresh" buttons, and a "Map view" button.

The "Review your scan" section on the right contains the following configuration details:

- Basics**: Name is Scan-20210910, Collection is practicum.
- Credential**: Type is SQL authentication, Name is practicumsds.
- Scan Scope**: Scope is Full.
- Scan Rule Set**: Name is AzureSqlDatabase, Type is System.
- Scan Trigger**: Start at Immediately, Recurrence is Once.

At the bottom right of the review section are "Save and Run" (highlighted in blue), "Back", and "Cancel" buttons.

Review scan settings and then click the “Save and Run” button.

Confirm Scan

Navigate to Sources, “Table view.”

The screenshot shows the Microsoft Azure Purview Studio interface. The left sidebar has a 'Sources' section under 'practicump'. The main area is titled 'Sources' with tabs for 'Register', 'Refresh', 'Map view', and 'Table view' (which is selected). A search bar says 'Search assets'. Below it is a filter box 'Filter by name'. A table lists one source: 'AzureSqlDatabase-P3k' (Azure SQL Database), registered on 09/10/21 at 01:59 PM. The 'Scans' column shows a value of 1.

Note that the **Scans** column now shows 1.

Click on the **Name** link to drill through to details. Allow time for processing.

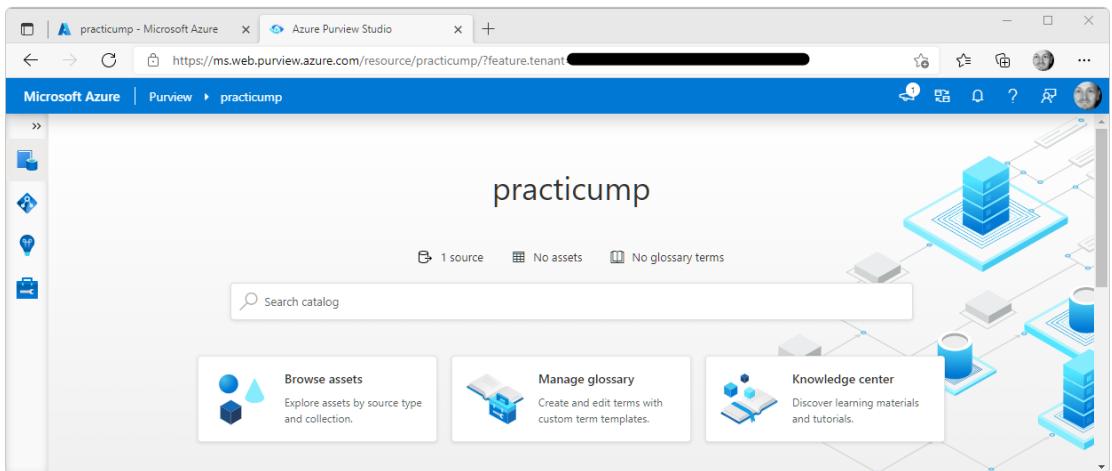
The screenshot shows the details for the 'practicumsd' source. The left sidebar is the same as the previous screen. The main area shows the source ID 'practicumsd.database.windows.net' and the 'Scans' tab. It displays 1 scan, 17 scanned assets, and 15 classified assets. To the right, it shows the 'Registered on' date (09/11/2021 03:18:28 AM), the 'Collection path' ('practicump'), and the 'Source hierarchy' ('rchapter', 'Subscription', 'practicumrg'). Below this are sections for 'Recent scans' (one entry: 'Scan-20210910' in progress for 'AzureSqlDatabase' on 09/11/21 at 03:25 AM) and 'Recent failed scans' (empty). A note at the bottom says 'No recent failed scan runs'.

Browse Assets

After scanning, it takes time for **Asset Insights** to reflect new data products.

We do not have to wait to browse data products, however.

Click the **Data Catalog** icon on the navigation pane.



Click on the “**Browse assets**” button.

A screenshot of the 'Browse assets' page. The left sidebar shows the 'Data catalog' section with a 'Browse assets' link. The main area has a 'Refresh' button and a 'By collection' dropdown set to 'By source type'. A 'Filter by keyword' input field is present. Below this, a table lists one collection: 'practicump' (The root collection., 4 assets, Collection admin: Rich Chapler).

Click on the Root Collection link {i.e., name}.

The screenshot shows the Microsoft Azure Purview Studio interface. The left sidebar has a tree view with 'practicump' selected under 'Data catalog > Browse assets'. Below it are sections for 'Sub collection(s)', 'Related', and 'Narrow results by:' with dropdowns for 'Classification', 'Contact', 'Label', and 'Glossary term'. The main pane displays a list of assets with the following details:

Asset Type	Name	Description
sql	Address	Azure SQL Table mssql://practicumsd.database.windows.net/practicumsds/SalesLT/Address
sql	BuildVersion	Azure SQL Table mssql://practicumsd.database.windows.net/practicumsds/dbo/BuildVersion
sql	Customer	Azure SQL Table mssql://practicumsd.database.windows.net/practicumsds/SalesLT/Customer
sql	CustomerAddress	Azure SQL Table mssql://practicumsd.database.windows.net/practicumsds/SalesLT/CustomerAddress
sql	dbo	Azure SQL Schema mssql://practicumsd.database.windows.net/practicumsds/dbo

At the bottom, there are navigation buttons: < Previous, Page 1 of 1, and Next >.

Explore results.

Click on the first item, **Address**.

The screenshot shows the Azure Purview Studio interface for managing assets. The main page displays the 'Address' asset, which is an 'Azure SQL Table'. The asset is categorized under the 'practicump' collection path. The 'Hierarchy' section shows the database structure: 'practicumsd.database.windows.net' (Azure SQL Server) contains the 'practicumsd' (Azure SQL Database), which includes the 'SalesLT' (Azure SQL Schema) and 'Address' (Azure SQL Table). The 'Overview' tab is selected, showing the asset was last updated on September 11, 2021, at 10:29 AM UTC by an automated scan. Other tabs include 'Properties', 'Schema', 'Lineage', 'Contacts', and 'Related'. The 'Asset description' and 'Classifications' sections both indicate no descriptions or classifications are available. The 'Schema classifications' section lists a single entry: 'World Cities'. The 'Fully qualified name' is listed as 'mssql://practicumsd.database.windows.net/practicumsd/SalesLT/Address'. The 'Glossary terms' section notes that no glossary terms are present for this asset.

practicump - Microsoft Azure

Azure Purview Studio

Data catalog > Browse assets >

Address

SQL Azure SQL Table

Edit Refresh Delete

Open in Power BI Desktop

Overview Properties Schema Lineage Contacts Related

Updated on September 11, 2021 10:29 AM UTC by automated scan

Asset description

No description for this asset.

Classifications

No classifications for this asset.

Schema classifications (1)

World Cities

Fully qualified name

mssql://practicumsd.database.windows.net/practicumsd/SalesLT/Address

Collection path

practicump

Hierarchy

- practicumsd.database.windows.net
 - Azure SQL Server
- practicumsd
 - Azure SQL Database
- SalesLT
 - Azure SQL Schema
- Address
 - Azure SQL Table

Glossary terms

No glossary terms for this asset.

Notice the automatic “**World Cities**” classification.

Drill-through the tabs {e.g., Properties, Schema, Lineage, etc.}.

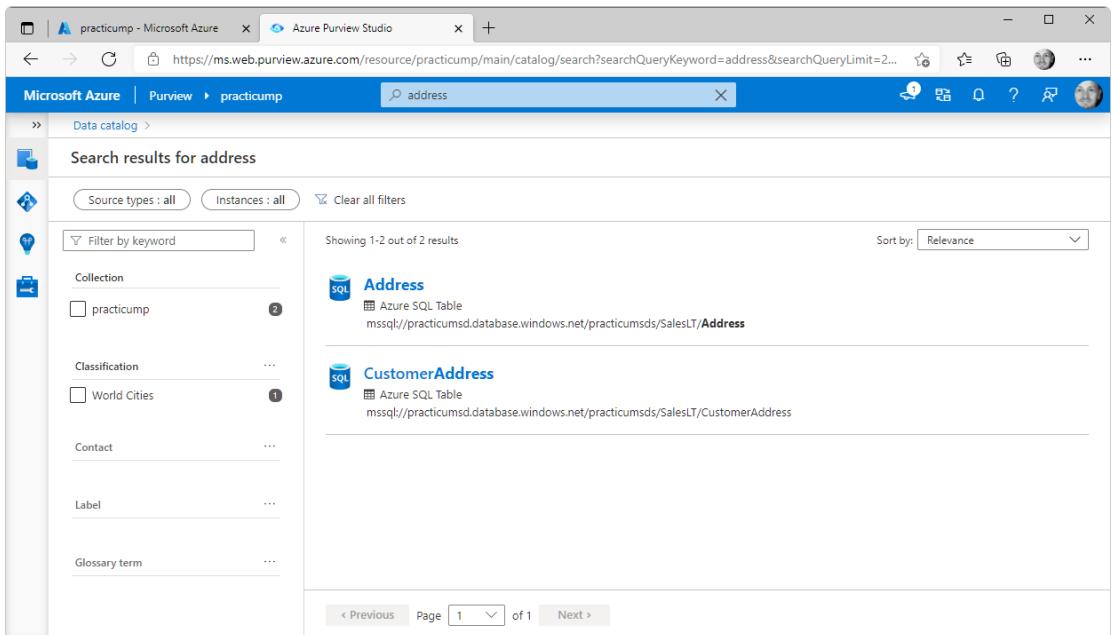
Confirm Success

Now that we have implemented the basics in Purview, can we answer the questions previously posed?

Search

Question: "Does the data that I need exist?"

Enter a phrase into the search box at the top center of the window {e.g., "address"}.



The screenshot shows the Azure Purview Studio interface with a search query of "address". The results page displays two entries:

- Address**: Azure SQL Table, mssql://practicumsd.database.windows.net/practicumsds/SalesLT/Address
- CustomerAddress**: Azure SQL Table, mssql://practicumsd.database.windows.net/practicumsds/SalesLT/CustomerAddress

The sidebar on the left includes filters for Collection, Classification, Contact, Label, and Glossary term. The search bar at the top contains the query "address".

Returned search results will include all data products that include the requested phrase.

Location

Question: "how do I connect to Data Product X?"

Drill into the first search result, Address.

The screenshot shows the Azure Purview Studio interface with the URL <https://ms.web.purview.azure.com/resource/practicump/main/catalog/entity?guid=81afae99-9619-4559-aa95-b5f6f6f60000§i...>. The search bar at the top contains the text "address". The main content area shows a search result for "Address" (Azure SQL Table). The "Overview" tab is selected, displaying the following information:

- Asset description:** No description for this asset.
- Classifications:** No classifications for this asset.
- Schema classifications (1):** World Cities
- Fully qualified name:** mssql://practicumsd.database.windows.net/practicumsds/SalesLT/Address
- Collection path:** practicump
- Hierarchy:**
 - practicumsd.database.windows.net (Azure SQL Server)
 - practicumsds (Azure SQL Database)
 - SalesLT (Azure SQL Schema)
 - Address (Azure SQL Table)
- Glossary terms:** No glossary terms for this asset.

On the **Overview** tab, you will see **Hierarchy** information which includes the Fully Qualified Domain Name of the Azure SQL Server.

Related

Question: “are there related data products that I might find valuable?”

Continue in the search result for phrase “Address.”

The screenshot shows the Microsoft Azure Purview Studio interface. The URL in the address bar is <https://ms.web.purview.azure.com/resource/practicump/main/catalog/entity?guid=81afae99-9619-4559-aa95-b5f6f60000§i...>. The search bar at the top contains the text "address". The main content area displays a search result for the word "Address". On the left, there is a sidebar with icons for Data Catalog, Properties, Schema, Lineage, Contacts, and Related. The "Related" tab is selected. The main pane shows a list of 13 related items under the heading "Showing 1 to 13 of 13 items". The columns in the table are Name, Type, Owner, and Action. The items listed are:

Name	Type	Owner	Action
Address	Azure SQL Table	-	...
Customer	Azure SQL Table	-	...
CustomerAddress	Azure SQL Table	-	...
Product	Azure SQL Table	-	...
ProductCategory	Azure SQL Table	-	...
ProductDescription	Azure SQL Table	-	...
ProductModel	Azure SQL Table	-	...
ProductModelProductDescription	Azure SQL Table	-	...
SalesOrderDetail	Azure SQL Table	-	...
SalesOrderHeader	Azure SQL Table	-	...

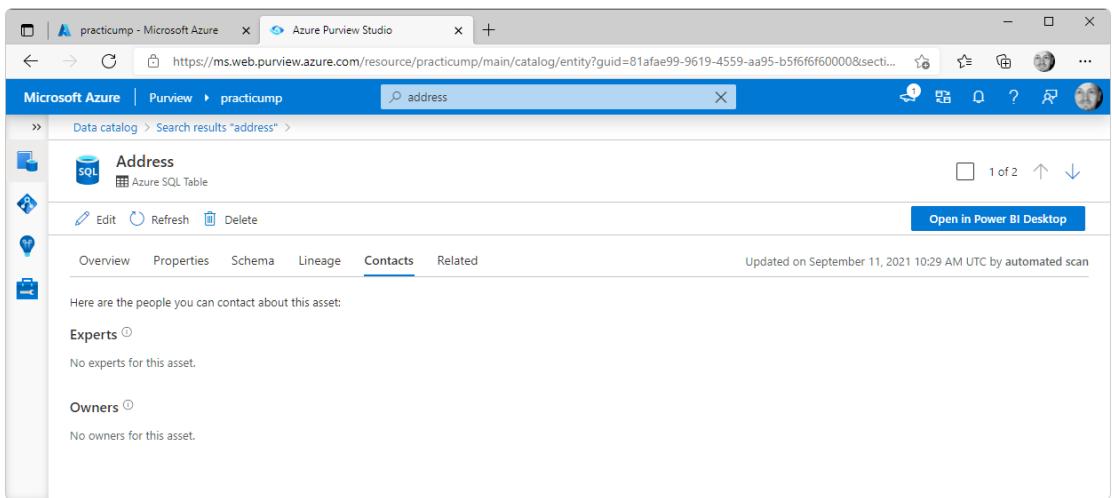
At the bottom right of the main pane, there is a button labeled "Open in Power BI Desktop".

On the **Related** tab, you will see a list of related data products (as identified by Purview).

Contacts

Question: “who are the owners and experts for Data Product X?”

Continue in the search result for phrase “Address.”



The screenshot shows a Microsoft Azure browser window with the URL <https://ms.web.purview.azure.com/resource/practicump/main/catalog/entity?guid=81afae99-9619-4559-aa95-b5f6f6f60000§i...>. The page displays a search result for "address". The main content area shows a single item named "Address" which is an "Azure SQL Table". Below the item, there are buttons for "Edit", "Refresh", and "Delete", along with a "Open in Power BI Desktop" button. The "Contacts" tab is selected, showing two sections: "Experts" and "Owners". Both sections indicate "No experts for this asset" and "No owners for this asset". The status bar at the bottom right shows the last update was on September 11, 2021, at 10:29 AM UTC by an automated scan.

On the **Contacts** tab, you will see lists of **Experts** and **Owners** you can contact about the data product (as identified by Purview).

If, as in the snippet above, there are no Contacts and you can add information not already captured, click the **Edit** button.

Edit "Address"

Overview **Schema** **Contacts**

Select contacts to add to this asset.

Experts
Experts are often business process or subject matter experts in different business areas or departments.

Select user or group
Search by name or email address, and select a user or group.

Name	Info
Rich Chapter Richard.Chapter@microsoft.com	Curator

Owners
Owners are often senior executives or business area owners that define governance or business processes over certain data areas.

Select user or group
Search by name or email address, and select a user or group.

Save **Cancel**

In either the **Experts** or the **Owners** drop-down menu, search for the user or group in your directory. Click **Save**.

Classify Data

Data classification is a process of annotating data products to help organizations answer the following questions:

- **Definition** ... “how does data map to a business context?” and “what is the intended use of Data Product X?”
- **Documentation** ... “where is the documentation of Data Product X?”
- **Vocabulary** ... “what else {e.g., synonyms, acronyms, etc.} might help me understand Data Product X?”
- **Standardization** ... “how do I classify all data products in a standard way?”

In this section, we will configure Purview and successfully respond to these questions.

Prerequisites

This solution requires the following resources:

- Purview
- SQL

Create Term

Navigate to Purview in the Azure portal.

Click the “**Open Purview Studio**” button.

Click the “**Manage Glossary**” button.

Click the “**+ New term**” button on the resulting page.

On the resulting pop-out, confirm selection of “**System default**” and then click the **Continue** button.

The screenshot shows the Microsoft Purview Studio interface for creating a new term. The URL is https://ms.web.purview.azure.com/resource/practicump/main/catalog/term/new?feature.tenant=[REDACTED]. The page title is "practicump - Microsoft Azure" and the sub-page title is "Azure Purview Studio". The main content area is titled "New term" and contains the following fields:

- Term template:** System default
- Status:** Draft
- Overview:** Product Number (Formal name: Product Number)
- Definition:** The stock-keeping unit (SKU) that uniquely identifies a product
- Parent:** Select...
- Acronym:** SKU
- Resources:** Wikipedia Definition: https://en.wikipedia.org/wiki/Stock_keeping_unit

At the bottom, there are "Create" and "Cancel" buttons.

Complete the “**New term**” form, including:

Parent	Select a parent term, if applicable
Acronym	Capture abbreviation, if applicable
Resources	Click “+ Add a resource” to capture hyperlinks with descriptive content

Click the **Create** button.

Use Term

Search for “product” to identify data products that use terms like Product Number.

The screenshot shows the Microsoft Azure Purview Studio interface. The top navigation bar includes 'practicump - Microsoft Azure', 'Azure Purview Studio', and a search bar with the query 'product'. The main content area is titled 'Search results for product'. On the left, there's a sidebar with icons for Data catalog, Source types (all), Instances (all), and Clear all filters. Below these are filters for Collection (practicump), Classification, Contact, Label, and Glossary term. The main pane displays seven results, sorted by Relevance. The first result is 'Product' (Azure SQL Table, mssql://practicumsd.database.windows.net/practicumsds/SalesLT/Product). The second is 'ProductModelProductDescription' (Azure SQL Table, mssql://practicumsd.database.windows.net/practicumsds/SalesLT/ProductModelProductDescription). The third is 'ProductCategory' (Azure SQL Table, mssql://practicumsd.database.windows.net/practicumsds/SalesLT/ProductCategory). The fourth is 'ProductDescription' (Azure SQL Table, mssql://practicumsd.database.windows.net/practicumsds/SalesLT/ProductDescription). At the bottom, there are navigation buttons for < Previous, Page 1, of 1, and Next >.

Click **Product**.

The screenshot shows the 'Schema' tab for the 'Product' table. The top navigation bar is identical to the previous screenshot. The main content area shows the 'Schema' tab selected. There are tabs for Overview, Properties, Schema, Lineage, Contacts, and Related. The Schema tab displays a table of columns with their properties. The columns are: Column name (Color, DiscontinuedDate, ListPrice, ModifiedDate, Name, ProductCategoryID, ProductID, ProductModelID, ProductNumber, rowguid), Classifications (empty), Sensitivity label (empty), Glossary terms (empty), Data type (nvarchar, datetime, money, datetime, nvarchar, int, int, int, nvarchar, uniqueidentifier), Asset description (empty), and Description (empty). A status message at the bottom right says 'Updated on September 11, 2021 10:29 AM UTC by automated scan'. A blue button 'Open in Power BI Desktop' is visible on the right.

Click the **Schema** tab.

Click on the **ProductNumber** link.

A screenshot of the Azure Purview Studio interface. The top navigation bar shows 'practicump - Microsoft Azure' and 'Azure Purview Studio'. The main content area displays the 'product' asset details for 'ProductNumber'. The asset is identified as an 'Azure SQL Column'. Below the asset name are buttons for 'Edit', 'Refresh', and 'Delete'. A horizontal menu bar includes 'Overview', 'Properties', 'Lineage', 'Contacts', and 'Related'. The 'Overview' tab is selected. On the right side, the status is 'Updated on September 11, 2021 10:29 AM UTC'. Under 'Asset description', it says 'No description for this asset.' Under 'Collection path', it shows 'practicump'. Under 'Description', there is a short text entry field containing a single dash. Under 'Hierarchy', there is a link labeled 'Hierarchy'. Under 'Classifications', it says 'No classifications for this asset.' Under 'Glossary terms', it says 'No glossary terms for this asset.' Under 'Fully qualified name', it shows the URL 'mssql://practicumsd.database.windows.net/practicumsds/SalesLT/Product#ProductNumber'.

Click the **Edit** button.

A screenshot of the Azure Purview Studio interface, showing the 'Edit "ProductNumber"' screen. The top navigation bar shows 'practicump - Microsoft Azure' and 'Azure Purview Studio'. The left sidebar has icons for Overview, Properties, Lineage, Contacts, and Related. The 'Overview' tab is selected. The main form contains fields for 'Name' (set to 'ProductNumber'), 'Asset description' (an empty text area), 'Classifications' (a dropdown menu with 'Select...'), and 'Glossary terms' (a dropdown menu with 'Product Number' selected). At the bottom are 'Save' and 'Cancel' buttons.

Select the new term, “**Product Number**” from the “**Glossary terms**” drop-down menu.

Click the **Save** button.

Confirm Success

Now that we have implemented the basics in Purview, can we answer the questions previously posed?

Definition

Questions: “how does data map to a business context?” and “what is the intended use of Data Product X?”

The screenshot shows the Azure Purview Studio interface with the URL <https://ms.web.purview.azure.com/resource/practicump/main/catalog/term?section=overview&termGuid=52836041-66f9-43d5-8...>. The page displays the 'Product Number' term details. The term is marked as 'System default' and is currently in 'Draft' status. It has three tabs: 'Overview' (selected), 'Related', and 'Contacts'. The 'Overview' tab shows the formal name 'Product Number' and its parent status ('No parent term for this term'). The 'Definition' section contains the text: 'The stock-keeping unit (SKU) that uniquely identifies a product.' The 'Last updated' field shows '09/13/2021 07:37 by Rich Chapler'. The 'Acronym' section lists 'SKU' as an acronym associated with 'Catalog assets' (1 asset associated with "Product Number"). The 'Resources' section includes a link to 'Wikipedia Definition'.

This also demonstrates response to questions:

- **Documentation** ... “where is the documentation of Data Product X?”
- Links to documentation can be included in **Resources**
- **Vocabulary** ... “what synonyms, acronyms, etc. might help me understand Data Product X?”

Standardization

Question: “how do I classify all data products in a standard way?”

The screenshot shows the Azure Purview Studio interface with the URL <https://ms.web.purview.azure.com/resource/practicump/main/catalog/entity/edit?guid=81afae99-9619-4559-aa95-b5f6f6f60000&...>. The page displays the 'Edit "Address"' screen for the 'Address' entity. The 'Schema' tab is selected. A note states: 'Making a manual update to the schema will prevent future scans on this asset from updating it.' The schema table has columns: 'Column name', 'Glossary terms', 'Data type', and 'Asset description'. The 'Column name' column lists fields: AddressID, AddressLine1, AddressLine2, City, and CountryRegion. The 'Glossary terms' column shows dropdown menus for each field. The 'Data type' column shows types: int, nvarchar, nvarchar, nvarchar, and nvarchar respectively. The 'Asset description' column contains empty text boxes.

Use “**Column level classification**” options to characterize all country- / region-type schema elements as a standard classification.

Understand Lineage

Data lineage describes provenance {i.e., where data comes from, where it ends up and what happens to it along the way}.

Understanding data lineage enables stakeholders to answer the following question:

- **Accountability** (aka Impact Analysis) ... “who and what would be affected by a change to or problem with Data Product X?”

In this section, we will configure Purview and successfully respond to this question.

Prerequisites

This solution requires the following resources:

- Data Factory
- Data Share
- Purview
- SQL

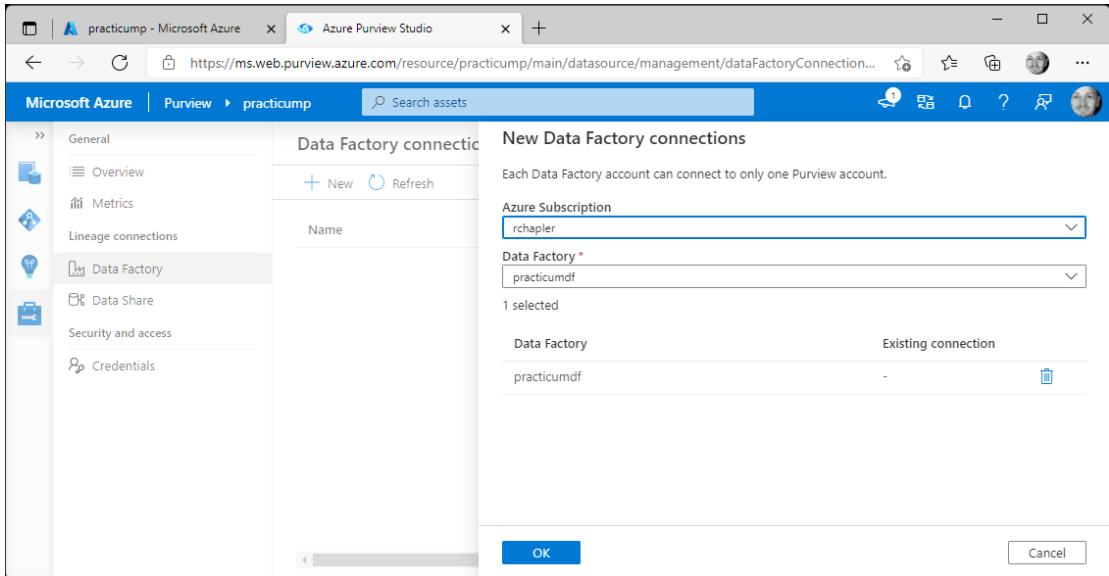
Configure Data Factory

Navigate to Purview in the Azure portal.

Click the “**Open Purview Studio**” button.

Click the **Management** icon on the navigation and then “**Data Factory**” from the resulting menu.

Click the “**+ New**” button on the resulting page.

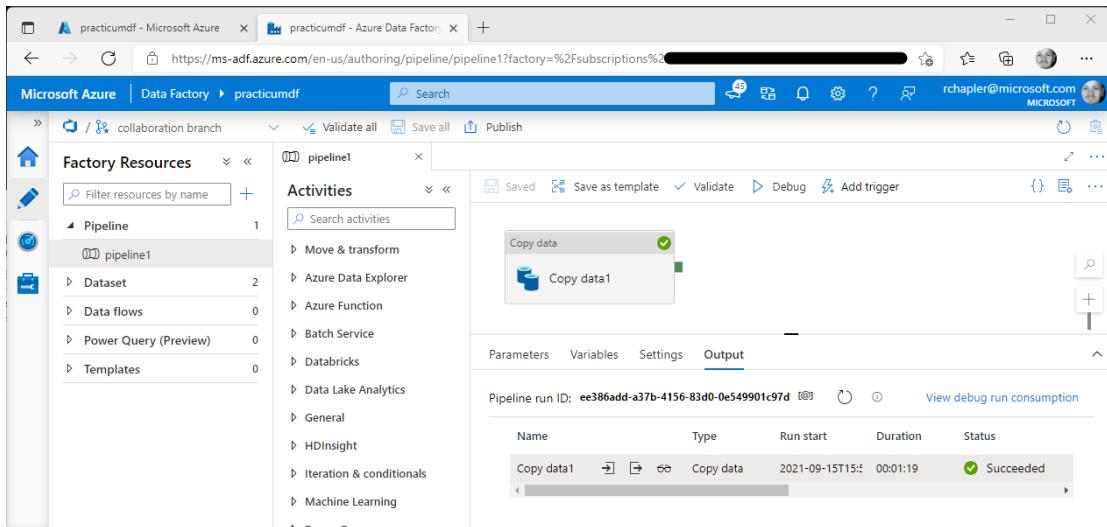


Complete the “**New Data Factory connections**” pop-out, select your Data Factory and then click the **OK** button. Confirm connection.

Confirm Success

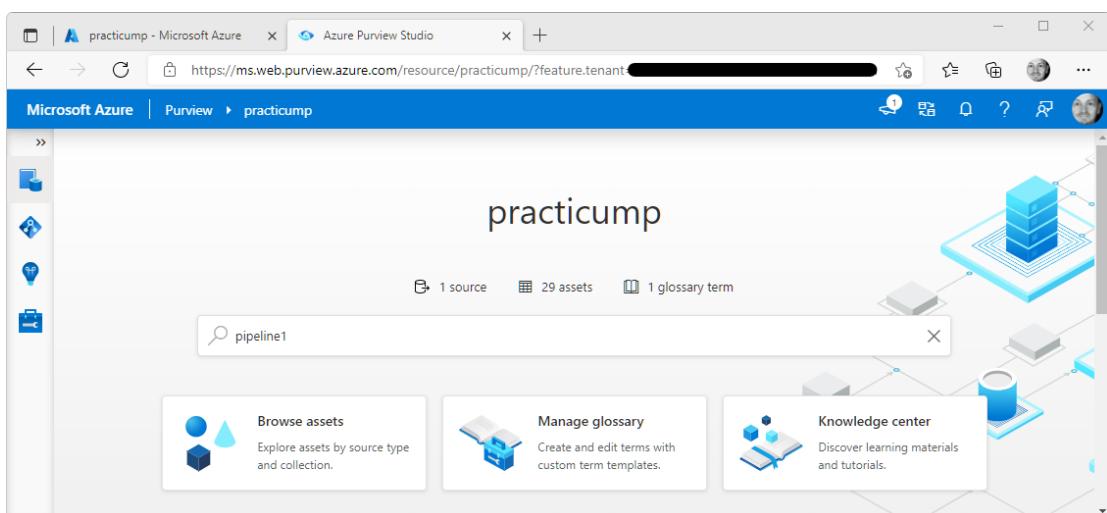
Purview will automatically detect data movement and capture data about pipelines executing in a connected instance of Synapse or Data Factory.

Navigate to Data Factory and execute your sample pipeline.



The screenshot shows the Microsoft Azure Data Factory interface. On the left, the 'Factory Resources' sidebar is open, showing a 'Pipeline' section with 'pipeline1'. The main workspace displays the 'Activities' pane for 'pipeline1', which contains a single activity named 'Copy data1'. Below the activities, the 'Output' tab is selected, showing a table of the pipeline run. The table has columns: Name, Type, Run start, Duration, and Status. One row is visible: 'Copy data1', 'Copy data', '2021-09-15T15:00:00Z', '00:01:19', and 'Succeeded'. The status is indicated by a green checkmark icon.

After successful execution, navigate to Purview.



The screenshot shows the Azure Purview Studio dashboard for the resource 'practicump'. The top navigation bar includes 'Microsoft Azure', 'Purview', and the resource name 'practicump'. The main area features a 3D visualization of data assets, with the text 'practicump' prominently displayed. Below the visualization, there is a search bar with the placeholder 'pipeline1'. To the left, there is a vertical toolbar with icons for 'Browse assets', 'Manage glossary', and 'Knowledge center'. Below the search bar, three cards provide quick access to these features: 'Browse assets' (Explore assets by source type and collection), 'Manage glossary' (Create and edit terms with custom term templates), and 'Knowledge center' (Discover learning materials and tutorials).

Search for “pipeline1.”

The screenshot shows the Microsoft Azure Purview Studio interface. The top navigation bar includes 'practicump - Microsoft Azure', 'Azure Purview Studio', and a search bar with the query 'pipeline1'. The main content area is titled 'Search results for pipeline1' and displays two items:

- Copy data1**: Azure Data Factory Copy Activity. This item is highlighted with a blue border.
- pipeline1**: Azure Data Factory Pipeline.

On the left sidebar, there are filters for 'Collection' (practicump), 'Classification', 'Contact', and 'Content type'. The bottom of the page shows pagination controls: '< Previous', 'Page 1 of 1', and 'Next >'.

Click on the “**Copy data1**” result.

Click on the **Lineage** tab.

The screenshot shows the 'Lineage' tab for the 'Copy data1' activity. The top navigation bar and search bar are identical to the previous screenshot. The main content area shows the lineage diagram for the 'Copy data1' activity, which is part of the 'pipeline1' pipeline.

The lineage diagram consists of three components connected by arrows:

- A source dataset labeled 'Product' (represented by a blue icon).
- An Azure Data Factory Copy Activity labeled 'Copy data1' (represented by a blue box).
- A target dataset labeled 'Product' (represented by a blue icon).

Below the diagram, the 'Copy data1' activity card provides detailed information:

- Factory: practicumpdf
- Pipeline: pipeline1

Buttons at the bottom right of the card include 'See details' and 'Open in Azure Data Factory'.

The left sidebar contains tabs for 'Overview', 'Properties', 'Lineage', 'Contacts', and 'Related'. The 'Lineage' tab is currently selected. A search bar is also present on the left.

Continue to drill-through elements to better understand the capture of Lineage data.

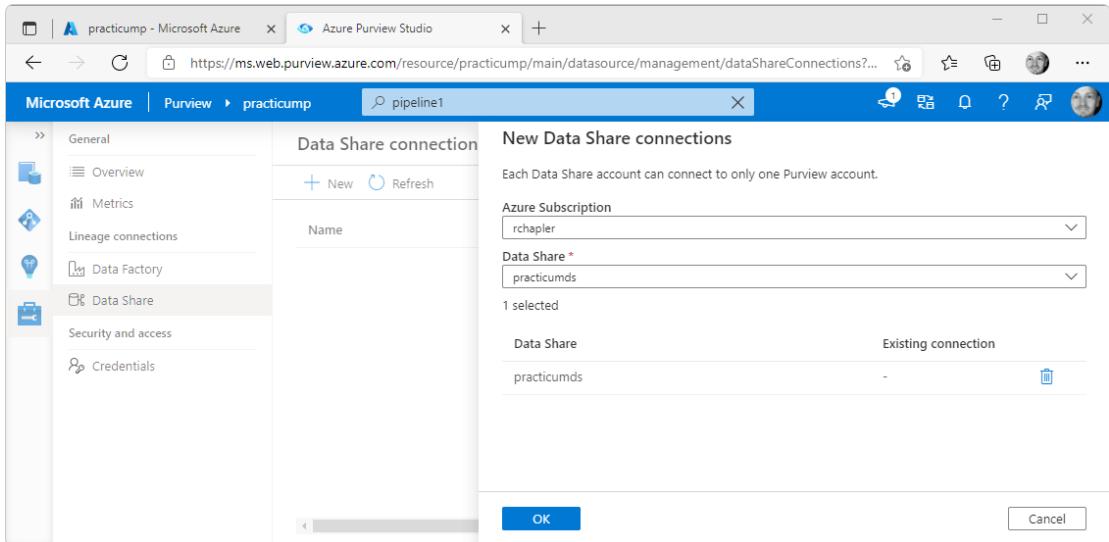
Configure Data Share

Navigate to Purview in the Azure portal.

Click the “**Open Purview Studio**” button.

Click the **Management** icon on the navigation and then “**Data Share**” from the resulting menu.

Click the “**+ New**” button on the resulting page.



Complete the “**New Data Factory connections**” pop-out, select your Data Share and then click **OK**.

Article: <https://docs.microsoft.com/en-us/azure/purview/how-to-link-azure-data-share>

Note from this article... “For Data Share products to show in Purview, a snapshot job must be run after you connect your Data Share to Purview”