

Foreword

This book is essentially a collection of objectives with step-by-step instructions not appropriately launched with a flowery introduction. That said, I feel like it needs **something**. Do not worry, I intend to be brief...

The title for my role, Cloud Solution Architect, falls short of self-explanation. So, when asked, I explain that I exist in the grey area between various roles:

- **I sell, but I am not a sales person...** I am certainly biased in favor of 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 hands-on solution development, but there is a role-defined limit to how deep I can dig (and after that is where customer resources take over)
- **I support, but I am not a support engineer...** I will dig into just about any issue to help with a challenge, but Support Engineers have the knowledge and resources to go far deeper than I
- **I train, but I am not a training professional...** I work with customers, whenever possible, directly in their subscriptions and environments; as such, I spend more time coaching than coding

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



Many times, customer ask questions or need solutions where I must learn more. This book is an opportunistic capture of the knowledge that I have gained {i.e., a collection of step-by-step instructions describing solution implementation}.

Content is organized into **product- | activity- focused objectives** and **discipline-specific sections**; consider browsing directly to those objectives that align with your focus and interest.

Table of Contents

Foreword.....	1
Prepare.....	3
Naming Convention	3
Infrastructure	4
Source	61
Migration from On-Prem	61
External Data.....	67
Simple Load.....	76
Incremental Load	102
Mount Data Lake.....	154
Sourcing from APIs.....	160
Batch Upsert	164
Synchronize Unstructured Data.....	168
Surface	183
Query from On-Prem	183
Detect Anomalies.....	188
Application+ AI	195
Audit Usage.....	216
Deploy	219
Infrastructure-as-Code.....	219
Source Control	237
Schema Comparison	243
Govern.....	246
Discover Data	246
Classify Data	259
Understand Lineage.....	265
Quick Answers.....	270
Subscription Identifier	270
Tenant Identifier	270

Prepare

Naming Convention

Consider best practice guidance from the Azure Cloud Adoption Framework:

[Define your naming convention - Cloud Adoption Framework | Microsoft Docs](#)

When setting on a naming convention, also remember that:

- Some resources will not allow names with a dash (“-”) or other special characters
- Some resources will allow names with capitals and some only lowercase

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

I do not adhere to the guidance in the Cloud Adoption Framework; rather, I typically use a combination of project name as prefix and resource type acronym as suffix. For example:

- Suffix: “ads” (as in Azure Data Solutions)
- Prefix: “dl” (as in Data Lake)
- Resource Name: “adsdl”

Because this book has been written over time and for various projects, you might see different naming used in different exercises.

Infrastructure

The following sections describe various ways {e.g., Azure Portal, Bash, PowerShell, etc.} to manually instantiate the Azure resources necessary to complete the objectives in this book.

Application Registration (aka Service Principal)

This exercise requires the following resource(s):

- [Key Vault](#)

Consider using a User-Assigned Managed Identity instead of a Service Principal.

Create using Azure Portal

Navigate to “**Azure Active Directory**”, select “**App Registrations**” in the **Manage** group of the navigation pane, and then click the “**+ New Registration**” button.

On the “**Register an application**” page, enter values for the following items:

Name	Enter a meaningful name aligned with standards
-------------	--

No additional changes are required. Review settings on the remaining tabs.

Click the **Register** button.

Key Vault Secrets, Tenant, and Client Identifiers

Open a new tab and navigate to Key Vault.

Click **Secrets** in the **Settings** group of the navigation pane.

Click the “**+ Generate/Import**” button.

On the “**Create a secret**” page, enter values for the following items:

Name	Enter a meaningful name aligned with standards
Value	Paste the previously copied value

No additional changes are required. Review settings on the remaining tabs.

Click the **Create** button.

Repeat for **Tenant Id**.

Client Secret

Navigate to “**Certificates & secrets**” in the **Manage** group in the navigation pane.

Click the “+ New client secret” button, and then in the resulting “**Add a client secret**” popout, click the **Add** button.

Copy **Value**.

Key Vault Secret, Client 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.

On the “**Create a secret**” page, enter values for the following items:

Name	Enter a meaningful name aligned with standards
Value	Paste the previously copied Value

No additional changes are required. Review settings on the remaining tabs.

Click the **Create** button.

Create using Cloud Shell (PowerShell)

Navigate to Cloud Shell and select **PowerShell** from the “**Select Environment**” dropdown.

Update and execute the following command create a Service Principal and give it Contributor access to the subscription:

```
az ad sp create-for-rbac --name "adssp" --role contributor --scopes /subscriptions/{Subscription GUID}
```

Logic explained...

- az ad sp create-for-rbac
...creates a service principal and configures access ([az ad sp | Microsoft Docs](#))
- --name
...self-explanatory
- --role contributor
...role of the service principal
- --scopes /subscriptions/{Subscription GUID}
...replace with your Subscription GUID
...scope the service principal's role assignment applies to

You can expect a response like the following redacted example:

```
{  
  "appId": "{App GUID, aka Client Id}",  
  "displayName": "adssp",  
  "password": "{Password String, aka Client Secret \"rbac\"}",  
  "tenant": "{Tenant GUID}"  
}
```

These values will be useful in many exercises; keep them handy.

Data Lake

Create using Azure Portal

This exercise requires the following resource(s):

- Resource Group
- Storage Account

On the “**Create Storage Account**” page, enter values for the following items:

Subscription	Select your subscription
Resource Group	Select your resource group
Storage Account Name	Enter a meaningful name aligned with standards
Location	Match resource group selection
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)”

Navigate to the **Advanced** tab.

Click on the **Enabled** radio button in the “**Data Lake Storage Gen2**” grouping.

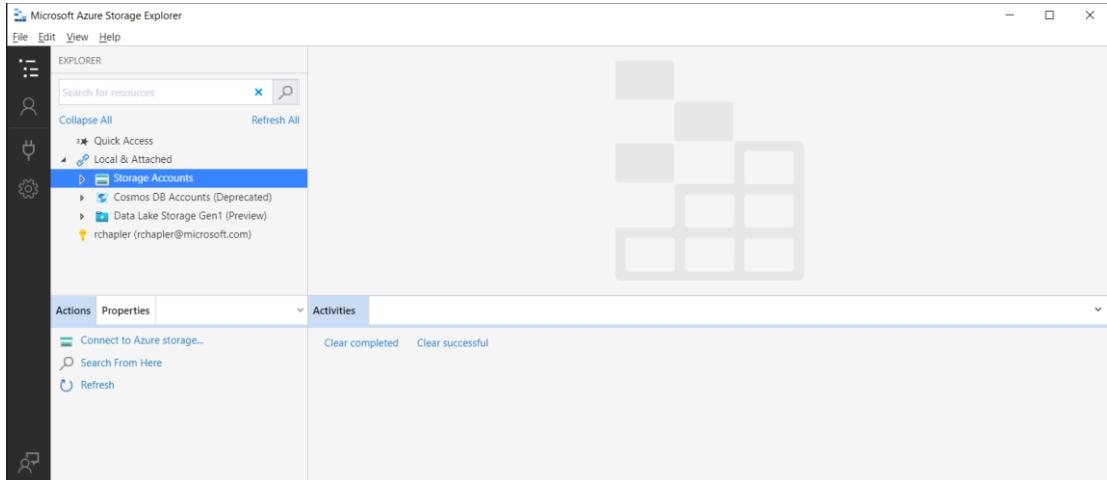
No additional changes are required. Review settings on the remaining tabs.

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

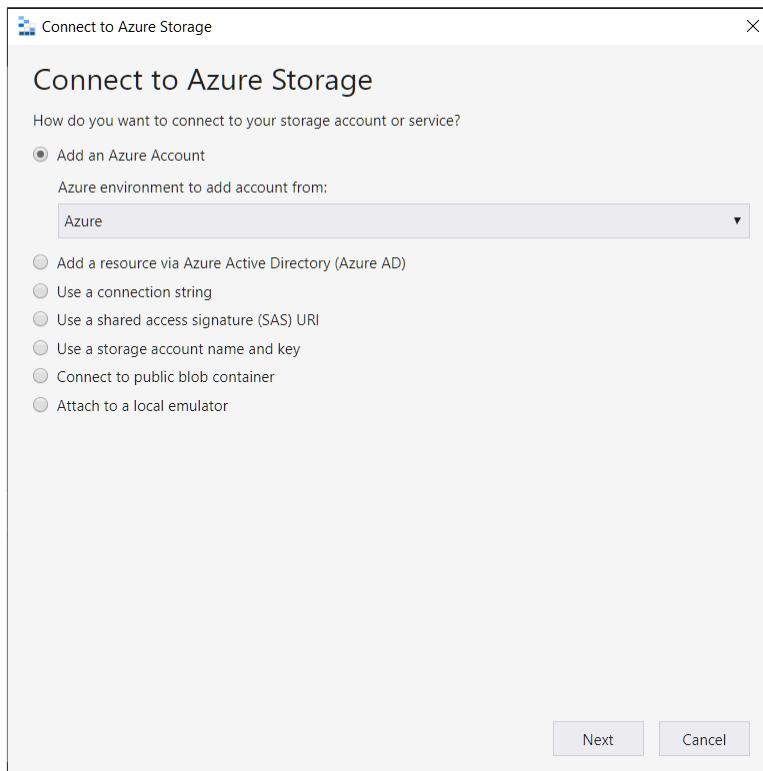
Storage Explorer

Download and install the Microsoft Azure Storage Explorer app ([Azure Storage Explorer – cloud storage management | Microsoft Azure](#)).

Launch the app and provide Azure credentials.



Click the “Connect to Azure storage...” link.

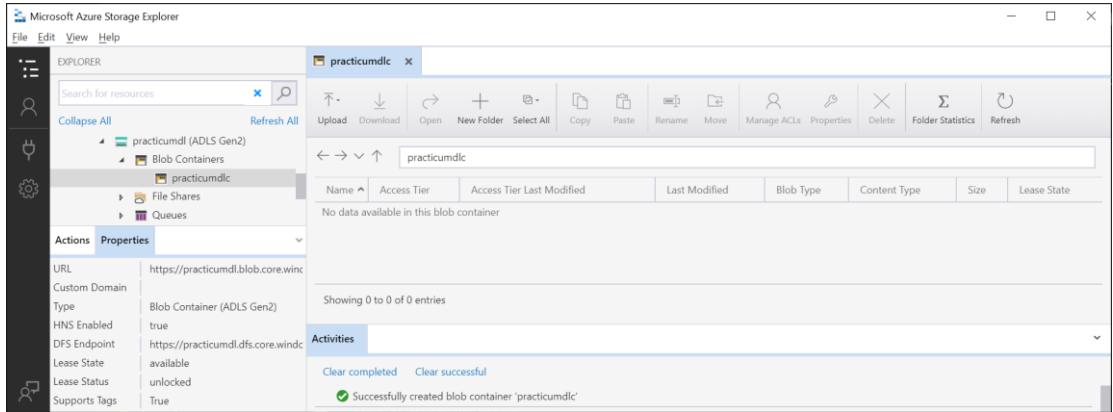


On the resulting popup, click the “**Add an Azure Account**” radio button. Click the **Next** button. Authenticate with Azure credentials.

Container

Continue with the “Microsoft Azure Storage Explorer” app.

Use the search bar to focus on your Storage Account.



Right-click on “**Blob Containers**” and click “**Create Blob Container**” in the resulting popup menu. Enter a meaningful name.

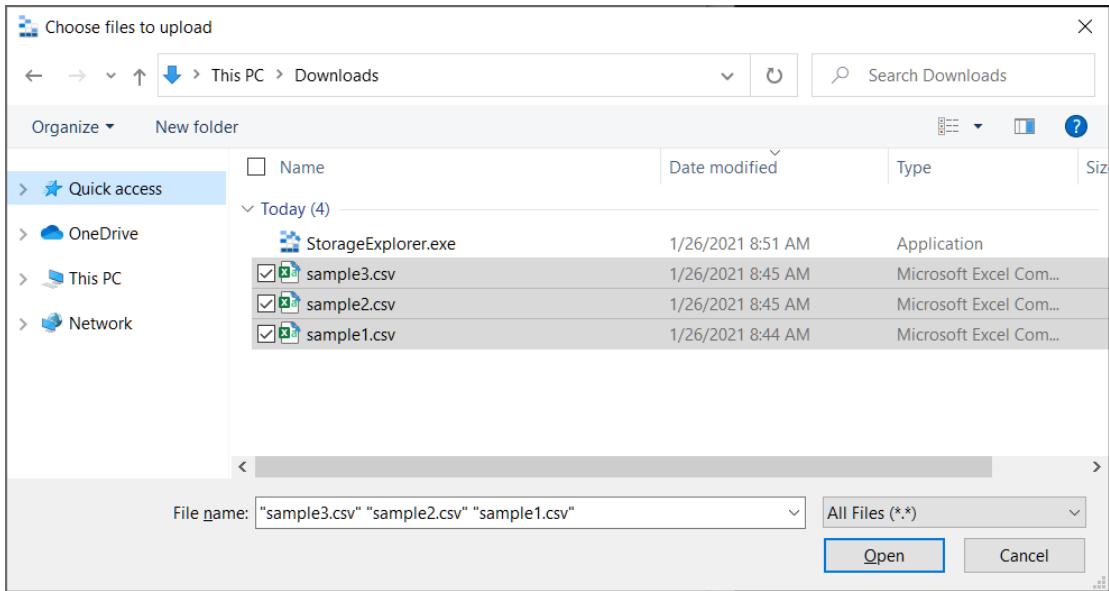
Sample Data

Download sample files from a site like [Sample CSV Files Download - Get Examples Instantly \(filesamples.com\)](http://filesamples.com)

Return to the “Microsoft Azure Storage Explorer” app and navigate to your container. Click the **Upload** button and select “Upload Files” in the resulting popup menu.

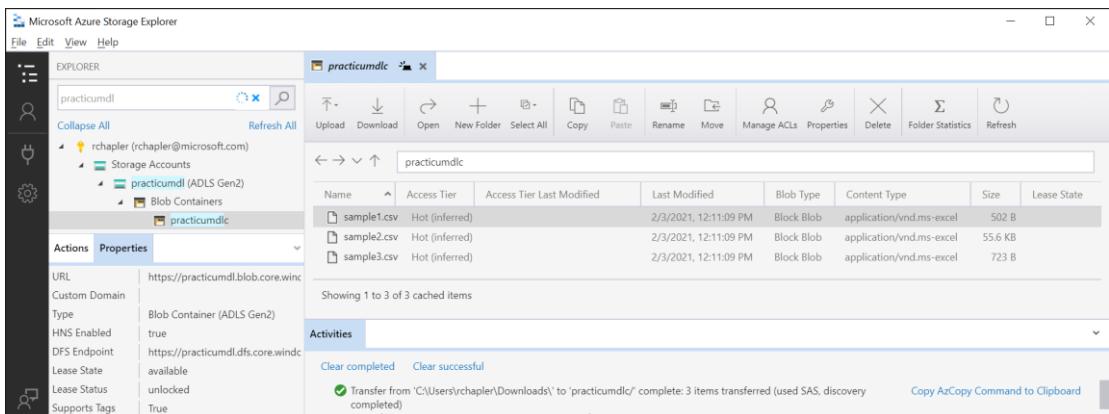
Click the ellipses button to the right of the “**Selected Files**” box.

Navigate to **Downloads** and select your sample data files.



Click the **Open** button.

On the “Upload Files” popup, click the **Upload** button. Allow time for file transfer.



Data Migration Assistant

Browse to [Download Microsoft® Data Migration Assistant v5.5 from Official Microsoft Download Center.](#)

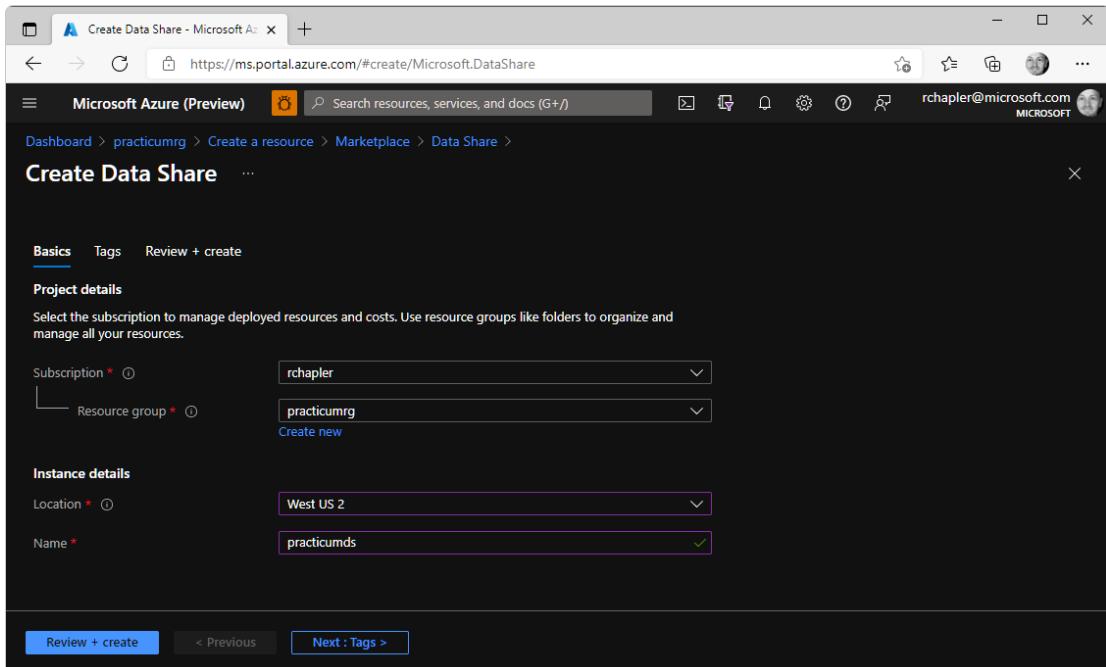
Download, open, and install the “**User Installer**” for your platform.

Data Share

Create using Azure Portal

This exercise requires the following resource(s):

- Data Explorer (2 instances)
- Resource Group



On the “**Create Data Share**” page, enter values for the following items:

Subscription	Select your subscription
Resource Group	Select your resource group
Location	Match resource group selection
Name	Enter a meaningful name aligned with standards

No additional changes are required. Review settings on the remaining tabs.

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

After deployment is complete, click the “**Go to resource**” button.

The screenshot shows the Microsoft Azure portal interface. The title bar says "practicums - Microsoft Azure". The URL in the address bar is <https://ms.portal.azure.com/#@microsoft.onmicrosoft.com/resource/subscriptions>. The top navigation bar includes "Microsoft Azure (Preview)", a search bar, and a user profile for "rchapler@microsoft.com". Below the header, the breadcrumb navigation shows "Dashboard > Microsoft.DataShare-20210915092919 >". The main content area displays the "practicums" Data Share settings. On the left, there's a sidebar with sections like "Overview", "Activity log", "Access control (IAM)", "Tags", "Diagnose and solve problems", "Properties", "Locks", "Data Share", "Sent Shares", and "Received Shares". The main panel has tabs for "Delete", "Refresh", and "Move". Under the "Essentials" tab, it shows the Resource group (practicumrg), Location (West US 2), Subscription (rchapler), and Subscription ID. It also shows Tags (none). At the bottom, there are two buttons: "Start sharing your data" and "View received shares".

Sample Share

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

Click the “Start sharing your data” button.

On the resulting “**Sent Shares**” page, click the “+ Create” button.

The screenshot shows the Microsoft Azure portal interface for creating a Data Share. The left sidebar has a dark theme with white icons and text. The main area has a light background. The top navigation bar shows the URL as https://ms.portal.azure.com/#@microsoft.onmicrosoft.com/resource/subscription/... and the user rchapler@microsoft.com MICROSOFT.

Dashboard > Microsoft.DataShare-20210915092919 > practicumsd

practicumsd | Sent Shares Data Share

1. Details 2. Datasets 3. Recipients 4. Review + Create

Share name *

Share type In-place Learn more

Description Enter description for the share

Terms of use Enter terms of use for the share

Cancel **Continue**

On the “Sent Shares” page, “1. Details” tab, enter values for the following items:

Share Name Enter a meaningful name aligned with standards

Share Type Select “In-place” from the dropdown menu

Click the **Continue** button.

practicumds | Sent Shares

1. Details 2. Datasets 3. Recipients 4. Review + Create

Datasets

Select datasets to be shared. You must have permission to add role assignment to the data store. This permission exists in the Owner role. See [Share Your Data tutorial](#) for details.

Add datasets

On the “Sent Shares” page, “2. Datasets” tab, click the “Add datasets” button.

practicumds | Sent Shares

1. Details 2. Datasets 3. Recipients

Select dataset type

Azure Data Explorer

Cancel Next

On the resulting “Select dataset type” popout, click the “Azure Data Explorer” button, then click the **Next** button.

The screenshot shows the Microsoft Azure portal interface. On the left, a sidebar for the 'practicumds' Data Share lists various options like Overview, Activity log, and Tags. The main area shows the 'Sent Shares' blade with tabs for Details, Datasets, and Recipients. The 'Datasets' tab is active, showing a step-by-step wizard. Step 2, 'Datasets', has three sub-steps: 1. Details, 2. Datasets (which is selected), and 3. Recipients. A 'Previous' and 'Continue' button are at the bottom. An overlay window titled 'Azure Data Explorer' is displayed, prompting the user to select datasets. It includes fields for Subscriptions (set to 'rchapler'), Resource groups (set to 'practicumrg'), and Azure data explorer clusters (set to 'practicumdec2'). Buttons for 'Previous' and 'Next' are also present at the bottom of the overlay.

On the resulting “**Azure Data Explorer**” popout, enter values for the following items:

Subscriptions	Select your subscription
Resource Groups	Select your resource group
Azure Data Explorer Clusters	Select your Data Explorer cluster
Name	Enter a meaningful name aligned with standards

Click the **Next** button.

The screenshot shows the Microsoft Azure portal interface. On the left, there's a sidebar with various options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Settings (Properties, Locks), Data Share, and Sent Shares. The 'Sent Shares' option is currently selected. The main area has tabs for '1. Details', '2. Datasets', and '3. Recipients'. The '2. Datasets' tab is active, showing a modal window titled 'Azure Data Explorer' with the sub-header 'Select datasets'. Inside the modal, there's a search bar, a refresh button, and a list containing 'practicumdec2'. A checkbox next to 'practicumdec2' is checked. Below the list, it says 'Showing 1 to 1 of 1 items'. At the bottom of the modal are 'Previous' and 'Next' buttons.

On the second “...Select datasets” popout, check the box next to your cluster and then click the **Next** button.

This screenshot shows the same Azure portal interface as the previous one, but the '2. Datasets' step has been completed. The modal window now has a different title, 'Azure Data Explorer' with the sub-header 'Rename datasets'. It contains instructions: 'Provide names for your datasets. This is the name that your data consumer will see when they accept the share. Dataset names must be unique.' Below this, there's a table with two columns: 'Dataset name' and 'Path'. There is one row in the table with 'practicumdec2' in the 'Dataset name' column and 'practicumdec2/' in the 'Path' column. At the bottom of the modal are 'Previous' and 'Add datasets' buttons.

On the “...Rename datasets” popout, confirm the automatically generated “Dataset name” value. Click the “Add datasets” button.

practicumds | Sent Shares

1. Details 2. Datasets 3. Recipients 4. Review + Create

Datasets ↑	Type	Path
practicumdec2	Azure Data Explorer Cluster	practicumdec2/

Previous Continue

Back on the “Sent Shares” page, “2. Datasets” tab, click the **Continue** button.

practicumds | Sent Shares

1. Details 2. Datasets 3. Recipients 4. Review + Create

Enter email address of the recipients for the share. Please ensure you are using recipient's Azure login email.

Add recipient Delete All Update expiration for all

Email	Share expiration
rchapler@microsoft.com	<input type="checkbox"/> Fri Oct 15 2021 <input type="button"/> 11:59

Previous Continue

On the “Sent Shares” page, “3. Recipients” tab, click the **Add recipient** button.

In the resulting interface, enter an **Email** value.

Click the **Continue** button.

The screenshot shows the Microsoft Azure portal interface. On the left, there's a sidebar with various options like Overview, Activity log, Access control (IAM), Tags, and Diagnose and solve problems. Under the 'Data Share' section, 'Sent Shares' is selected. The main content area has tabs: 1. Details, 2. Datasets, 3. Recipients, and 4. Review + Create. The '4. Review + Create' tab is active. It displays 'Share Contents' (1 dataset), 'Settings' (Name: practicumdec, Description: NA, Terms of use: NA), and 'Recipients' (1 recipient). At the bottom, there are 'Previous' and 'Create' buttons, with 'Create' being highlighted.

On the “Sent Shares” page, “4. Review + Create” tab, click the **Create** button.

The designated email recipient can expect to receive an email like the one below:

The email message is titled "Azure Data Share invitation from Rich Chapler - Message (HTML)". The header includes standard Office 365 ribbon icons for Delete, Respond, Share to Teams, Quick Steps, Move, Tags, Editing, Immersive, Translate, Zoom, Dynamics 365, Insights, Report Message, Reply with Meeting Poll, and FindTime. Below the header, the message content starts with "Azure Data Share invitation from Rich Chapler". It shows a Microsoft Azure logo and the recipient's name, "Rich Chapler". There are buttons for Reply, Reply All, Forward, and more. The date is listed as "Wed 9/15/2021 11:03 AM". A note says, "If there are problems with how this message is displayed, click here to view it in a web browser." The body of the email reads: "You're invited to access data from Microsoft. You're receiving this email because Rich Chapler from Microsoft wants to share the following data with you. Share name: practicumdec. Description: NA. View invitation >".

Click the “View invitation >” link.

The screenshot shows the Microsoft Azure Data Share Invitations page. At the top, there's a header bar with the Microsoft logo, a search bar, and a user profile. Below the header, the main title is "Data Share Invitations". A "Refresh" button is available. A note says "Pending invitations sent to your Azure login email are listed. See [accept](#) and [receive data tutorial](#) for details." A table lists one invitation:

Invitation	Sender	Company	Status	Received On
practicumdec	Rich Chapter	Microsoft	Pending	9/15/2021 11:02:56 AM

Click the **Invitation** link.

The screenshot shows the "practicumdec" invitation configuration page. At the top, it displays the sender information: Rich Chapter from Microsoft. It also shows the number of datasets (1) and the expiration date. Below this, there's a "Description" section which is currently empty. Under "Terms of use", there is a link that is partially visible. The main section is titled "TARGET DATA SHARE ACCOUNT". It contains four dropdown fields:

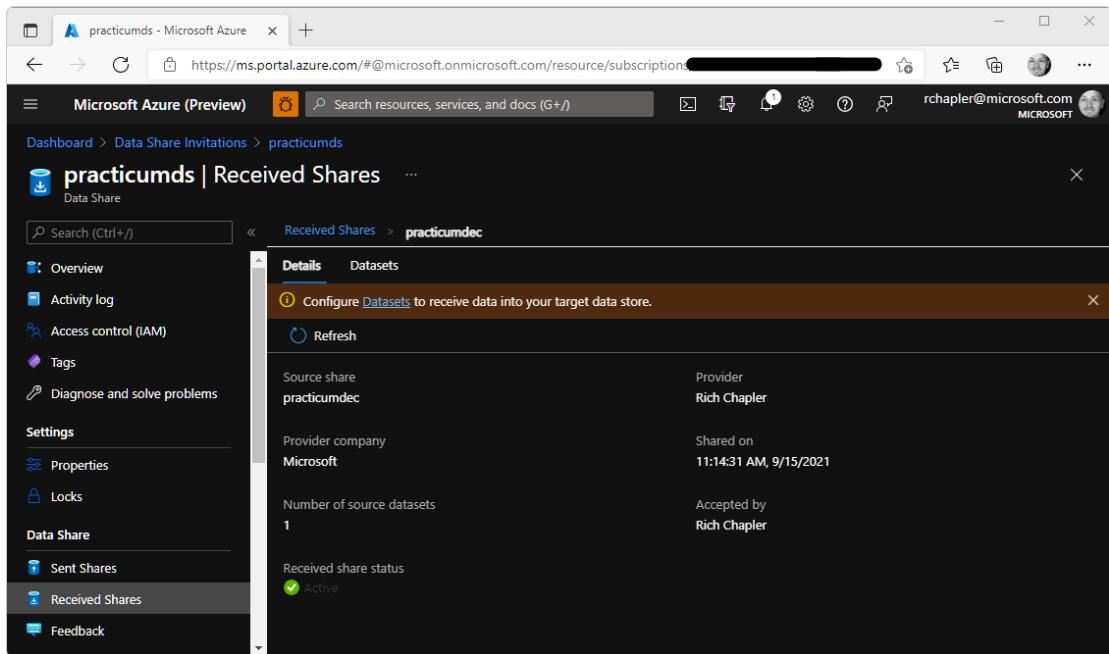
- Subscription *: rchaper
- Resource group *: practicumrg
Create new
- Data share account *: practicums
Create New
- Received share name *: practicumdec

At the bottom, there are two buttons: "Accept and configure" and "Reject".

On the resulting “...Invitation” page, enter values for the following items:

Subscriptions	Select your subscription
Resource Groups	Select your resource group
Data Share Account	Select your Data Share account
Received Share Name	Confirm default value

Click the “Accept and configure” button.



The screenshot shows the Microsoft Azure portal interface. The left sidebar has a dark theme with categories like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Settings, Properties, Locks, Data Share, Sent Shares, Received Shares, and Feedback. The 'Received Shares' item under Data Share is highlighted. The main content area shows a 'practicumds | Received Shares' blade. On the left of this blade is a sidebar with 'Received Shares' and 'practicumdec'. The main area has tabs for 'Details' (selected) and 'Datasets'. Under 'Details', there's a yellow info icon next to 'Configure Datasets to receive data into your target data store.' Below that is a 'Refresh' button. The data listed includes: Source share 'practicumdec', Provider 'Rich Chapler', Shared on '11:14:31 AM, 9/15/2021', Number of source datasets '1', Accepted by 'Rich Chapler', and Received share status 'Active' with a green checkmark. A callout box points to the 'Datasets' tab.

Click the **Datasets** link.

On the resulting “Received Shares...” page, **Datasets** tab, check the box next to your Data Explorer cluster.

Click the “+ Map to target” button.

The screenshot shows the Microsoft Azure portal interface. On the left, there's a sidebar with various navigation options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Properties, Locks, Sent Shares, Received Shares (which is selected), Feedback, Monitoring, Alerts, Metrics, and Diagnostic settings. The main area shows a 'Received Shares' blade for a Data Share named 'practicumdec'. A modal window titled 'Map datasets to target' is open, prompting the user to select datasets to map to target data stores. It includes fields for Subscriptions (set to 'rchapler'), Resource groups (set to 'practicumrg'), and location (set to 'West US 2'). A table lists the mapped dataset 'practicumdec' with its source type as 'Azure Data Explorer Cluster' and source path as 'practicumdec'. At the bottom of the modal are 'Cancel' and 'Map to target' buttons.

On the resulting “Map datasets to target” popout, enter values for the following items:

Subscriptions	Select your subscription
Resource Groups	Select your resource group
Select Kusto clusters	Select your Data Explorer cluster

Click the “Map to target” button.

The screenshot shows the Microsoft Azure portal interface. The left sidebar has a tree view: Dashboard > Data Share Invitations > practicumds. The main content area is titled "practicumds | Received Shares". On the left, there's a sidebar with "Overview", "Activity log", "Access control (IAM)", "Tags", "Diagnose and solve problems", "Properties", and "Locks". The right side shows a table with a single row:

Datasets	Source Type	Source Path	Status
practicumdec	Azure Data Explorer Cluster	practicumdec	Mapped

Data Studio

Browse to [Download and install Azure Data Studio - Azure Data Studio | Microsoft Docs](#)

Download, open, and install the “User Installer” for your platform.

Add Extensions

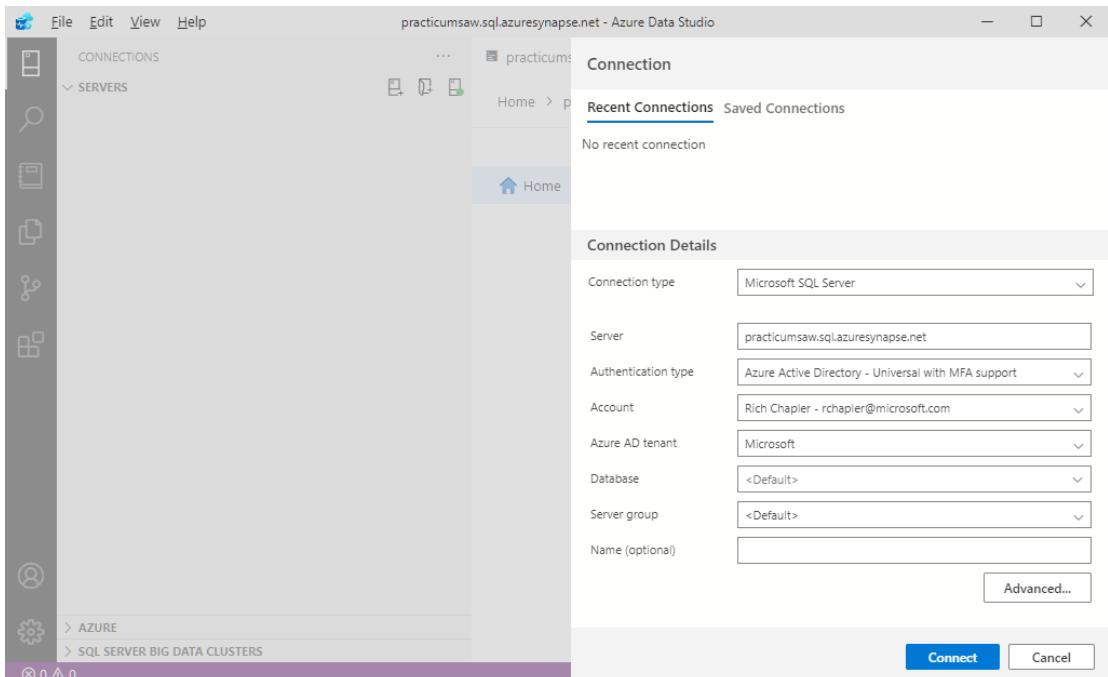
Launch Data Studio and then navigate to **Extensions** in the navigation pane.

The screenshot shows the Azure Data Studio interface. The left sidebar has icons for File, Edit, View, Help, Extensions, and a search bar. The main area shows the "EXTENSIONS: MARKETPLACE" tab with a search bar containing "schema compare". A card for the "SQL Server Schema Compare 1.10.0" extension by Microsoft is displayed, with an "Install" button. To the right, the extension details page is shown with the title "SQL Server Schema Compare" and a description: "Schema Compare tool for dacpac and databases". The "Details" and "Feature Contributions" tabs are visible. At the bottom, there's a note: "This extension is recommended by Azure Data Studio." and a "Choose SQL Language" dropdown.

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.

On the “Connection Details” popout, enter values for the following items:

Connection Type	Confirm default value, “Microsoft SQL Server”
Server	Paste the “ Dedicated SQL endpoint ” value copied in Instantiate Resources Synapse
Authentication Type	Select “Azure Active Directory – Universal with MFA support”
Account	Confirm credentials
Azure AD Tenant	Confirm tenant

Click the **Connect** button.

Databricks

Create using Azure Portal

This exercise requires the following resource(s):

- Resource Group

The screenshot shows the 'Create an Azure Databricks workspace' wizard. The 'Basics' tab is selected. In the 'Project Details' section, the subscription is set to 'rchapler' and the resource group is 'practicumrg'. In the 'Instance Details' section, the workspace name is 'practicumdb', the region is 'West US 2', and the pricing tier is 'Premium (+ Role-based access controls)'. At the bottom, there are buttons for 'Review + create', '< Previous', and 'Next : Networking >'.

On the “Create an Azure Databricks workspace” page, enter values for the following items:

Subscription	Select your subscription
Resource Group	Select your resource group
Workspace Name	Enter a meaningful name aligned with standards
Region	Match resource group selection
Pricing Tier	Select “ Premium... ” to enable features that we will need for this exercise

No additional changes are required. Review settings on the remaining tabs.

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 Overview page in a web browser. The URL is https://adb-2154823451042175.15.azuredatabricks.net/?o=2154823451042175. The page features a navigation bar with Microsoft Azure and Databricks tabs, a user profile, and a search bar. On the left is a sidebar with icons for Home, Workspace, Recents, Data, Clusters, Jobs, Models, and Search. The main content area has a title "Azure Databricks" with a logo. It includes three main sections: "Explore the Quickstart Tutorial" (with a note to spin up a cluster), "Import & Explore Data" (with a file upload interface), and "Create a Blank Notebook" (with a notebook icon). Below these are three tabs: "Common Tasks" (listing New Notebook, Create Table, New Cluster, New Job, New MLflow Experiment, Import Library, and Read Documentation), "Recents" (empty), and "Documentation" (listing 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

Min Workers: 2

Max Workers: 8

This Runtime version supports only Python 3.

Create Cluster

On the “Create Cluster” page, enter values for the following items:

Cluster Name Enter a meaningful name aligned with standards

No additional changes are required. Review settings on the remaining tabs.

Click the “Create Cluster” button.

New Notebook

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

Create Notebook

Name:

Default Language: Python

Cluster: practicumdbc

Cancel Create

Explore the Quickstart Tu

Spin up a cluster, run queries on preloaded data, and display results in 5 minutes.

Quickly import data, preview its schema, create a table, and query it in a notebook.

Common Tasks

- New Notebook
- Create Table
- New Cluster
- New Job
- New MLflow Experiment
- Import Library
- Read Documentation

Recents

Recent files appear here as you work.

Documentation

- Documentation
- Release Notes
- Getting Started

Create a Blank Notebook

Create a notebook to start querying, visualizing, and modeling your data.

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

practicumdb1 (Python)

Detached

Cmd 1

1

Shift+Enter to run shortcuts

Home

Workspace

Recents

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

Secret Scope

Navigate to Key Vault.

Click **Properties** in the **Settings** group of the navigation pane.

Name: practicumkv

Sku (Pricing tier): Standard

Location: westus2

Vault URI: https://practicumkv.vault.azure.net/

Resource ID: /subscriptions/[REDACTED]...resourceGroups/practicumrg...

Subscription ID: [REDACTED]

Subscription Name: rchapler

Directory ID: [REDACTED]

Directory Name: Microsoft

Soft-delete: Soft delete has been enabled on this key vault

Days to retain deleted vaults: 90

Purge protection:

Disable purge protection (allow key vault and objects to be purged during retention period)

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”.

On a new tab, navigate to <https://{{databricksInstance}}#secrets/createScope>.

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 left sidebar includes links for Home, Workspace, Recents, Data, Clusters, Jobs, and Models. The main content area is titled "Create Secret Scope" with a "Create" button. It contains fields for "Scope Name" (practicumdbss), "Manage Principal" (Creator), and "Azure Key Vault" (DNS Name: https://practicumkv.vault.azure.net/ and Resource ID: /46ed/resourceGroups/practicumrg/providers/Microsoft.KeyVault/vaults/practicumkv).

On the “Create Secret Scope” page, enter values for the following items:

Scope Name Enter a meaningful name aligned with standards

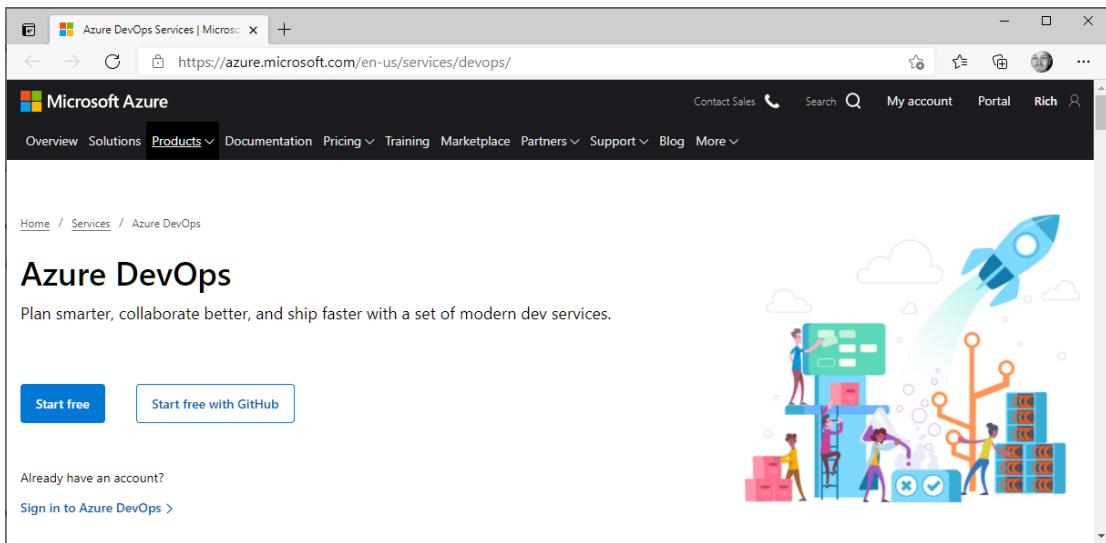
DNS Name Paste the copied “Vault UI” value

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 page, click the “**New organization**” link on the navigation pane.

Click the **Continue** button.

On the “**Almost done...**” page, enter values for the following items:

Name your Azure DevOps organization	Enter a meaningful name aligned with standards
--	--

Example: “adsd0” where “ads” refers to “Azure Data Solutions” and “do” is an acronym for “DevOps Organization”

We will host your projects in	Match resource group selection
--------------------------------------	--------------------------------

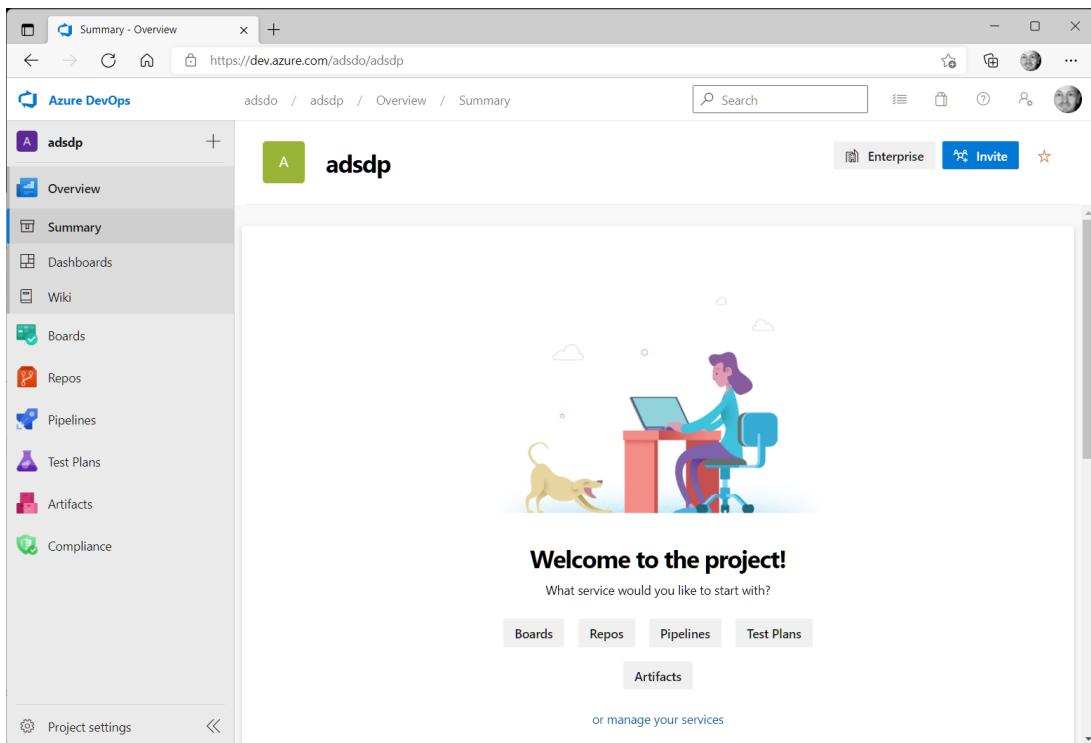
Click the **Continue** button.

Create Project

On the “**Create a project to get started**” page, enter values for the following items:

Project Name	Enter a meaningful name aligned with standards
Example: “adsdp” where “ads” refers to “Azure Data Solutions” and “dp” is an acronym for “DevOps Project”	
Visibility	Select the value that best aligns with your requirements

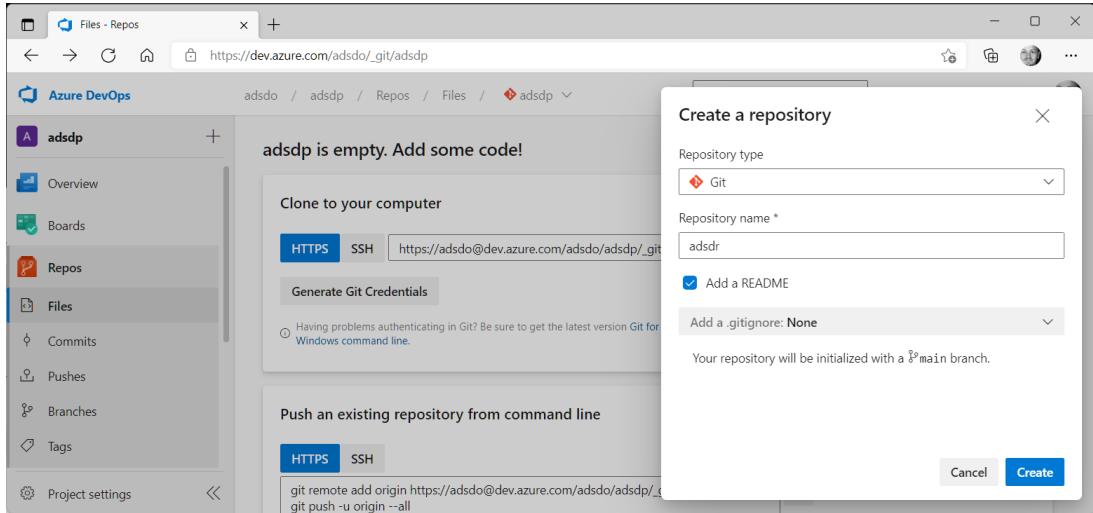
Click the “**+ Create project**” button.



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 dropdown.



On the “**Create a repository**” page, enter values for the following items:

Repository Type Confirm default selection, **Git**

Repository Name Enter a meaningful name aligned with standards

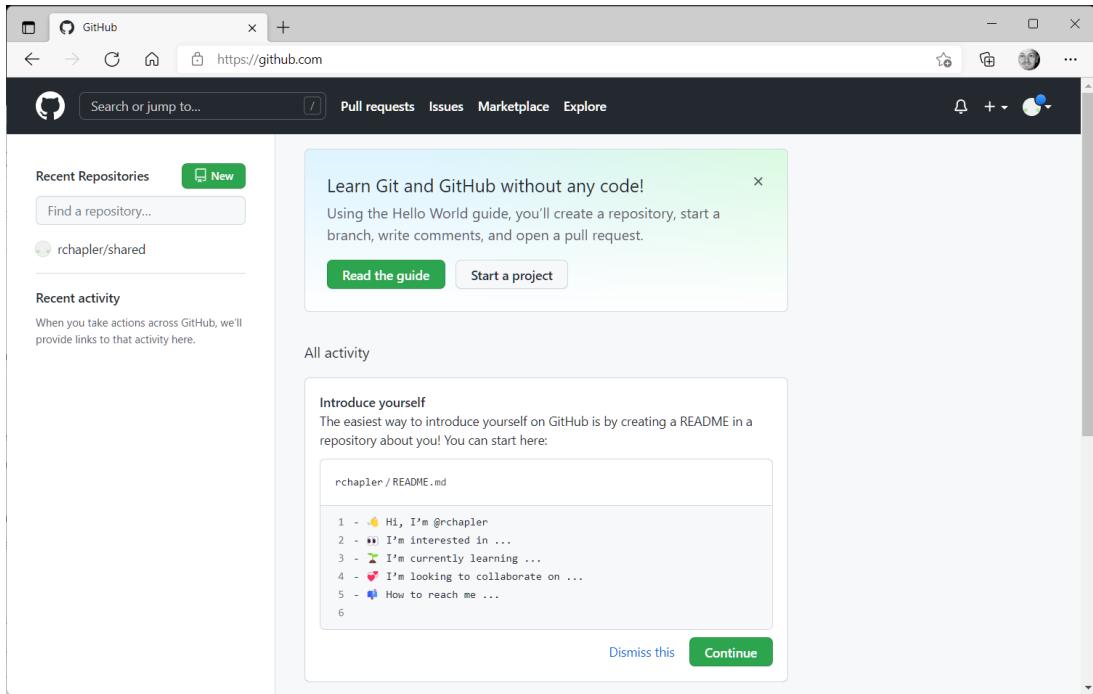
Example: “adsdr” where “ads” refers to “Azure Data Solutions” and “dr” is an acronym for “DevOps Repository”

Click the “**Create**” button.

GitHub Enterprise

Browse to [Sign in to GitHub · GitHub](#) and sign in (or sign up, as needed).

Create Repository



The screenshot shows the GitHub Enterprise homepage. On the left, there's a sidebar with 'Recent Repositories' (containing 'rchapler/shared') and 'Recent activity'. The main area features a green callout box titled 'Learn Git and GitHub without any code!' with a 'Read the guide' button. Below it is a 'All activity' section with a card titled 'Introduce yourself' containing a sample README.md file with five items. At the bottom of this card are 'Dismiss this' and 'Continue' buttons.

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

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner * rchapler / Repository name * adsgr

Great repository names are short and memorable. Need inspiration? How about [cuddly-octo-spork](#)?

Description (optional)

Public
Anyone on the internet can see this repository. You choose who can commit.

Private
You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

Add a README file
This is where you can write a long description for your project. [Learn more.](#)

Add .gitignore
Choose which files not to track from a list of templates. [Learn more.](#)

Choose a license
A license tells others what they can and can't do with your code. [Learn more.](#)

Create repository

© 2022 GitHub, Inc. [Terms](#) [Privacy](#) [Security](#) [Status](#) [Docs](#) [Contact GitHub](#) [Pricing](#) [API](#) [Training](#) [Blog](#) [About](#)

On the “Create a new repository” page, enter values for the following items:

Repository Name	Enter a meaningful name aligned with standards Example: “adsgr” where “ads” refers to “Azure Data Solutions” and “gr” is an acronym for “GitHub Repository”
Public or Private	Choose Private (unless you have a specific reason for using Public)

Click the “Create repository” button.

Key Vault

Create using Azure Portal

On the “**Create Key Vault**” page, enter values for the following items:

Subscription	Select your subscription
Resource Group	Select your resource group
Key Vault Name	Enter a meaningful name aligned with standards {e.g., practicumkv }
Region	Match resource group selection
Pricing Tier	Confirm default selection, “Standard”

No additional changes are required. Review settings on the remaining tabs.

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

Log Analytics

Create using Azure Portal

The screenshot shows the 'Create Log Analytics workspace' page in the Microsoft Azure (Preview) portal. The URL in the address bar is <https://ms.portal.azure.com/#create/Microsoft.LogAnalyticsOMS>. The page has a dark theme. At the top, there's a navigation bar with 'Microsoft Azure (Preview)', a search bar, and a user profile for 'rchapler@microsoft.com'. Below the navigation, the breadcrumb trail shows 'Dashboard > practicumrg > New > Marketplace > Log Analytics Workspace'. The main title is 'Create Log Analytics workspace'. There are tabs for 'Basics', 'Pricing tier', 'Tags', and 'Review + Create'. The 'Basics' tab is selected. A callout box provides information about what a Log Analytics workspace is and what specific considerations should be taken when creating one. The 'Project details' section asks to select a subscription and resource group. The 'Subscription' dropdown is set to 'rchapler' and the 'Resource group' dropdown is set to 'practicumrg'. The 'Instance details' section asks for a name and region. The 'Name' field is filled with 'practicumlaw' and the 'Region' dropdown is set to 'West US 2'. At the bottom, there are buttons for 'Review + Create', '<< Previous', and 'Next : Pricing tier >'.

On the “Create Log Analytics workspace” page, enter values for the following items:

Subscription	Select your subscription
Resource Group	Select your resource group
Name	Enter a meaningful name aligned with standards
Region	Match resource group selection

No additional changes are required. Review settings on the remaining tabs.

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

Metrics Advisor

Create using Azure Portal

The screenshot shows the 'Create Metrics Advisor' wizard in the Microsoft Azure (Preview) portal. The current step is 'Basics'. The page includes the following sections:

- Project details:** Subscription is set to 'rchapler' and Resource group is 'practicumrg'.
- Instance details:** Region is 'West US 2', Name is 'practicumma', and Pricing tier is 'S0'.
- Storage:** 'Bring your own storage' is set to 'No'.
- Disclaimer:** A note states, "The deployment could take up to 60 minutes to complete, although it normally finishes in less than 10 minutes."
- Agreement:** A checkbox for "I confirm I have read and understood the notice below." is checked.
- Service Agreement & Terms:** A link to the service agreement and terms.
- Buttons at the bottom:** 'Review + create' (highlighted in blue), '< Previous', and 'Next : Virtual network >'.

On the “Create Metrics Advisor” page, enter values for the following items:

Subscription	Select your subscription
Resource Group	Select your resource group
Region	Match resource group selection
Name	Enter a meaningful name aligned with standards
Pricing Tier	Select S0

Bring Your Own Storage

Confirm default selection, “No”

I confirm...Check the box

No additional changes are required. Review settings on the remaining tabs.

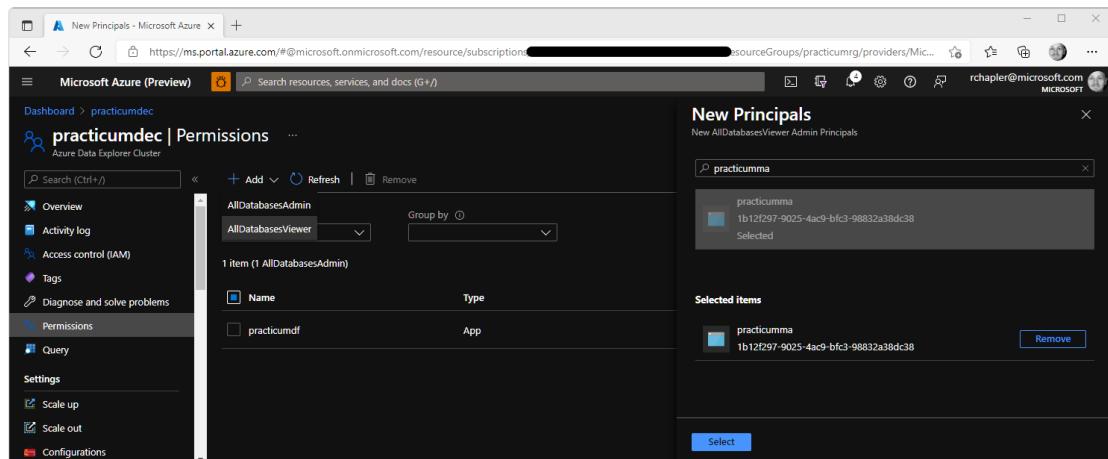
Click the “**Review + create**” button, review configuration, and then click the **Create** button.

Add Permissions to Data Explorer

This exercise requires the following resource(s):

- Data Explorer

Navigate to Data Explorer.



The screenshot shows the Microsoft Azure portal interface. The left sidebar shows the navigation path: Dashboard > practicumdec > Azure Data Explorer Cluster. The main content area is titled "practicumdec | Permissions". On the left, there's a sidebar with various options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Permissions (which is selected), Query, Settings, Scale up, Scale out, and Configurations. The main content area has a search bar with "practicum". Below it, there's a dropdown menu for "Type" set to "AllDatabaseViewer". A table lists one item: "practicumdf" (App type). To the right, a "Selected items" panel shows the same item with a "Select" button. The top of the screen shows the URL https://ms.portal.azure.com/#@microsoft.onmicrosoft.com/resource/subscriptions/[REDACTED]/resourceGroups/practicumrg/providers/Mic... and the user rchaper@microsoft.com MICROSOFT.

Click **Permissions** in the navigation pane.

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

Select **AllDatabasesViewer** from the resulting dropdown.

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

Purview

Create using Azure Portal

This exercise requires the following resource(s):

- [Key Vault](#)
- Resource Group

The screenshot shows the 'Create Purview account' wizard in the Microsoft Azure (Preview) portal. The page is titled 'Create Purview account' and displays the 'Basics' tab. It asks for 'Provide Purview account info'. The 'Project details' section includes a 'Subscription' dropdown set to 'rchapler' and a 'Resource group' dropdown set to 'practicumrg'. The 'Instance details' section includes a 'Purview account name' dropdown set to 'practicump' and a 'Location' dropdown set to 'West US 2'. A note at the bottom of this section states: '1 Capacity unit (CU) = 25 ops/sec and 2 GB of metadata storage. Any new purview account will be provisioned with 1 CU with auto scale capabilities.' The 'Managed resources' section shows a 'Managed resource group name' dropdown set to 'practicumpmrg', with 'Storage account name' and 'Event Hubs namespace name' fields below it, both with placeholder text: 'Name will be auto-generated during account creation.' At the bottom, there are buttons for 'Review + Create', 'Previous', and 'Next: Networking >'.

On the “Create Purview account” page, enter values for the following items:

Subscription	Select your subscription
Resource Group	Select your resource group
Purview Account Name	Enter a meaningful name aligned with standards
Location	Match resource group selection
Managed Resource Group...	Self-explanatory

No additional changes are required. Review settings on the remaining tabs.

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 URL is https://ms.portal.azure.com/#@microsoft.onmicrosoft.com/resource/subscriptions/[REDACTED]/resourceGroups/practicumrg/providers/Microsoft.KeyVault/vaults/practicumkv. The user is signed in as rchabler@microsoft.com. The left sidebar has a 'Key vault' icon and lists 'Overview', 'Activity log', 'Access control (IAM)', 'Tags', 'Diagnose and solve problems', 'Events', and 'Settings' (which includes 'Keys', 'Secrets', 'Certificates', 'Access policies', and 'Networking'). The main content area is titled 'practicumkv' and shows the 'Essentials' tab. It lists the Resource group (practicumrg), Location (West US 2), Subscription (rchabler), and Directory ID ([REDACTED]). Other details shown include Vault URI (https://practicumkv.vault.azure.net/), Sku (Standard), Directory Name (Microsoft), Soft-delete (Enabled), and Purge protection (Disabled). A 'Tags' section is also present.

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

The screenshot shows the 'Access policies' page for the 'practicumkv' key vault. The URL is https://ms.portal.azure.com/#@microsoft.onmicrosoft.com/resource/subscriptions/[REDACTED]/resourceGroups/practicumrg/providers/Microsoft.KeyVault/vaults/practicumkv/accessPolicies. The left sidebar shows 'Access policies' under 'Settings'. The main content area has a 'Save' and 'Discard' button. It includes sections for 'Enable Access to:' (checkboxes for Azure VM deployment, ARM template deployment, and Azure Disk Encryption) and 'Permission model' (radio buttons for 'Vault access policy' and 'Azure role-based access control', with 'Vault access policy' selected). Below these are buttons for '+ Add Access Policy' and 'Current Access Policies'. A table lists the current access policy:

Name	Email	Key Permissions	Secret Permissions	Certificate Permissions	Action
APPLICATION					

Click the “+ Add Access Policy” link.

Principal

Select a principal

practicump

practicump
d0206ad4-fa5d-4923-982a-4cf48bcf3559
Selected

Selected items

practicump
d0206ad4-fa5d-4923-982a-4cf48bcf3559

Remove

Select

On the “Add access policy” page, enter values for the following items:

Secret Permissions Select **Get** and **List** in the dropdown

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

Click the **Add** button.

practicumkv - Microsoft Azure

practicumkv | Access policies

Key vault

Search (Ctrl+ /)

Save Discard Refresh

Please click the 'Save' button to commit your changes.

Enable Access to:

Azure Virtual Machines for deployment

Azure Resource Manager for template deployment

Azure Disk Encryption for volume encryption

Permission model

Vault access policy

Azure role-based access control

+ Add Access Policy

Current Access Policies

Name	Email	Key Permissions	Secret Permissions	Certificate Permissions	Action
APPLICATION	practicump	0 selected	2 selected	0 selected	Delete

Click the **Save** button.

Connect Key Vault

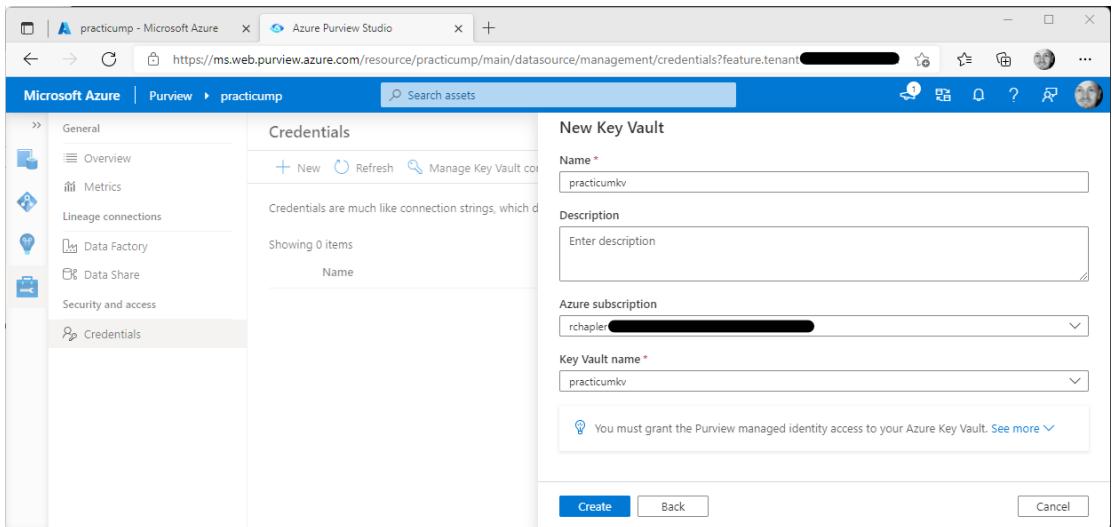
Navigate to Purview and click on the “**Open Purview Studio**” button.

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 popout.



On the “**New Key Vault**” popout, enter values for the following items:

Name	Enter a meaningful name aligned with standards
Subscription	Select your subscription
Key Vault Name	Select your Key Vault

Click the **Create** button.

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

Confirm the addition of Key Vault on the “**Manage Key Vault connections**” popout.

Click the **Close** button.

Add New Credential

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

The screenshot shows the 'Credentials' page in Azure Purview Studio. On the left, there's a navigation sidebar with icons for General, Overview, Metrics, Lineage connections, Data Factory, Data Share, Security and access, and Credentials. The 'Credentials' icon is selected. The main area has a title 'Credentials' with a 'New' button, a 'Refresh' button, and a 'Manage Key Vault connection' link. Below this, it says 'Credentials are much like connection strings, which do not contain sensitive information.' and 'Showing 0 items'. To the right, a 'New credential' form is open. It has fields for 'Name' (containing 'practicumsds'), 'Description' (empty), 'Authentication method' (set to 'SQL authentication'), 'User name' (containing 'rchapler'), 'Password' (with sub-fields for 'Key Vault connection' set to 'practicumkv', 'Secret name' set to 'practicumsds-adminpassword', and 'Secret version' set to 'Use the latest version if left blank'), and two buttons at the bottom: 'Create' (highlighted in blue) and 'Cancel'.

On the resulting “**New Credential**” popout, enter values for the following items:

Name	Enter a meaningful name aligned with standards
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

No additional changes are required. Review settings on the remaining tabs.

Click the **Create** button.

Resource Group

Create using Azure Portal

On the “**Create a Resource Group**” page, enter values for the following items:

Subscription	Select your subscription
Resource Group	Enter a meaningful name aligned with standards {e.g., shared }
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

No additional changes are required. Review settings on the remaining tabs.

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

SQL

Create using Azure Portal

Step 1: Create Resource

The screenshot shows the 'Select SQL deployment option' page in the Microsoft Azure portal. At the top, there are four main options: 'SQL databases', 'SQL managed instances', 'SQL virtual machines', and a grid of three smaller options: 'Single database', 'Elastic pool', and 'Database server'. The 'Single database' option under 'SQL databases' is highlighted with a dark background. Below each option, there is a brief description and a 'Resource type' dropdown. The 'Single database' dropdown is set to 'Single database'. Each option has a 'Create' button and a 'Show details' link.

On the “**Select SQL deployment option**” page, select the deployment option appropriate to your use case. For most examples, we will typically select “**Single database**” in the “**SQL databases**” > “**Resource type**” dropdown.

Click the **Create** button.

On the “**Create SQL Database**” page, **Basics** tab, enter values for the following items:

Resource Group	Select your resource group
Database Name	Enter a meaningful name aligned with standards {e.g., sharedsd }
Server	Click the “ Create new ” link, enter values in the resulting “ New server ” popout and then click the OK button
...use SQL elastic pool?	Confirm default value, No
Compute + Storage	Click the “ Configure database ” link

On the resulting **Configure** page, enter values for the following items:

Service Tier	Confirm default value, “General Purpose...”
Compute Tier	Click the Serverless radio button
Max vCores	Confirm minimum value
Min vCores	Confirm minimum value
Enable Auto-Pause	Confirm default value, checked
Days Hours Minutes	Confirm default values, 0 1 0
Data Max Size (GB)	Set minimum appropriate value

Review the automatically generated “**Cost Summary**” and then click the **Apply** button.

Back on and further down on the “**Create SQL Database**” page, **Basics** tab, enter values for the following items:

Backup Storage Redundancy	Confirm default value, “Locally-redundant backup storage”
----------------------------------	---

If you want to add a sample database, navigate to the “**Additional settings**” tab and enter values for the following items:

Use Existing Data	Select the “Sample” option
--------------------------	----------------------------

No additional changes are required. Review settings on the remaining tabs.

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

Step 2: Prepare Key Vault

Create Secret for Admin Password

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

Click the “**+ Generate/Import**” button.

On the “**Create a secret**” page, enter values for the following items:

Name	Enter a meaningful name aligned with standards {e.g., “ sharedsds-adminpassword ”}
Value	Enter the value used when creating the SQL Database Server

No additional changes are required. Review settings on the remaining tabs.

Click the **Create** button.

Step 3: Configure Firewall

Navigate to your Azure SQL Server and click the “**Show firewall settings**” link on the **Overview** page.

Set “Allow Azure services and resources to access this server” to “Yes”. Click the **Save** button.

Storage Account

Create using Azure Portal

On the “**Create Storage Account**” page, enter values for the following items:

Subscription	Select your subscription
Resource Group	Select your resource group
Storage Account Name	Enter a meaningful name aligned with standards
Location	Match resource group selection
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)”

No additional changes are required. Review settings on the remaining tabs.

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

Synapse

Create using Azure Portal

Step 1: Instantiate Prerequisites

This exercise requires the following resource(s):

- Key Vault
- Resource Group
- SQL

Step 2: Create Resource

On the “**Create Synapse workspace**” page, enter values for the following items:

Subscription	Select your subscription
Resource Group	Select your resource group
Managed Resource Group	This additional Resource Group will hold ancillary resources created specifically for Synapse. Because you are likely to have more than one managed resource group, consider using a consistent naming pattern {e.g., “ sharedmrg-s ” for Synapse and “ sharedmrg-x ” for X}
Workspace Name	Enter a meaningful name aligned with standards {e.g., practicums }
Region	Match resource group selection
Select Data Lake Storage Gen2	Confirm default, “ From subscription ”
Account Name	Select or create a data lake {e.g., shareddl }
File System Name	Select or create a data lake file system {e.g., shareddlfs }
Assign myself...	Check to assign necessary permissions

Click the “**Next: Security >**” button and enter values for “**SQL Server Admin Login**” and “**SQL Password**”.

No additional changes are required. Review settings on the remaining tabs.

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

Step 3: Prepare Key Vault

Create Secret for Admin Password

Navigate to Key Vault.

Click **Secrets** in the **Settings** group of the navigation and then click the “**+ Generate/Import**” button. On the “**Create a secret**” page, enter values for the following items:

Name	Enter a meaningful name aligned with standards {e.g., “ practicums-adminpassword ”}
Value	Enter the value used when creating the Synapse Workspace

No additional changes are required. Review settings on the remaining tabs.

Click the **Create** button.

Add Access Policy to Key Vault

Navigate to Key Vault and then click “**Access Policies**” in the **Settings** group of the navigation pane.

Click the “**+ Add Access Policy**” link and on the resulting “**Add access policy**” page, enter values for the following items:

Secret Permissions	Select Get and List in the dropdown
Select Principal	Search for, and then Select the managed identity for your Synapse Workspace

Click the **Add** button and then click the **Save** button.

Step 4: Add Permissions to Data Explorer

This exercise requires the following resource(s):

- Data Explorer

Navigate to Data Explorer.

Click **Permissions** in the navigation pane.

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

Select **AllDatabasesViewer** for read permissions and **AllDatabasesAdmin** for write, etc. permissions from the resulting dropdown.

Search for and then click to **Select** Synapse.

Step 5: 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**” popout, **Basics** tab, enter values for the following items:

Data Explorer Pool Name	Enter a meaningful name aligned with standards {e.g., shareddep }
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**” popout, “**Additional settings**” tab, and then enter values for the following items:

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

No additional changes are required. Review settings on the remaining tabs.

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

Create Data Explorer 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 dropdown.

On the resulting “**Create Data Explorer database...**” popout, enter values for the following items:

Pool Name	Enter a meaningful name aligned with standards {e.g., shareddep }
Name	Enter a meaningful name aligned with standards {e.g., sharedded }
Default Retention...	Confirm default, 365
Default Cache...	Confirm default, 31

Click the **Create** button.

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., **sharedded**} and select “New KQL script” from the resulting dropdown.

Execute the following query 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.

The screenshot shows the Azure Data Explorer interface. The left sidebar has 'Data', 'Query' (selected), and 'Dashboards (Preview)' options. The main area shows a query editor with the following KQL script:

```
1 .create table StormEvents (StartTime: datetime, EndTime: datetime, EpisodeId: int, EventId: int, State: string, EventType: string)  
2  
3 .ingest into table StormEvents 'https://kustosamplefiles.blob.core.windows.net/sampledata/StormEvents.kusto'  
4  
5 StormEvents | sort by StartTime desc | take 10
```

Below the query editor is a table titled 'Table 1' with columns: StartTime, EndTime, EpisodeId, EventId, State, and EventType. The table contains 10 records of storm event data. The first few rows are:

StartTime	EndTime	EpisodeId	EventId	State	EventType
2007-12-31 23:53:00.0000	2007-12-31 23:53:00.0000	12,037	65,839	CALIFORNIA	High Wind
2007-12-31 23:53:00.0000	2007-12-31 23:53:00.0000	12,037	65,838	CALIFORNIA	High Wind
2007-12-31 22:30:00.0000	2007-12-31 23:59:00.0000	12,950	71,590	MICHIGAN	Winter Storm
2007-12-31 22:30:00.0000	2007-12-31 23:59:00.0000	12,950	71,588	MICHIGAN	Winter Storm
2007-12-31 22:30:00.0000	2007-12-31 23:59:00.0000	12,950	71,589	MICHIGAN	Winter Storm

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 dropdown.

On the “**Create SQL database**” popout, enter values for the following items:

Select SQL Pool Type Confirm default, **Serverless**

Database Enter a meaningful name aligned with standards

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 popout, enter values for the following items:

Apache Spark Pool Name Enter a meaningful name aligned with standards

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.

Step 6: Create Linked Services

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

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

Select “**Linked services**” in the “**External connections**” grouping of the resulting navigation and then click the “**+ New**” button.

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

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

Click the **Continue** button.

On the “**New linked service...**” popout, enter values for the following common items:

Name	Enter a meaningful name aligned with standards
Connect via...	Confirm default selection, “ AutoResolveIntegrationRuntime ”

And enter values for resource specific items...

Data Explorer

Authentication Method	Select “Managed Identity”
Selection Method	Select “Enter manually”
Endpoint	Enter the URI for your Data Explorer Pool {e.g., https://shareddep.practicums.kusto.azuresynapse.net }
Database	Select your Data Explorer Database

Data Lake

Selection Method	Confirm default selection, “ From Azure subscription ”
Subscription	Select your subscription
Authentication Method	Confirm default selection, “ Account key ”
Storage Account Name	Select your Data Lake

Key Vault

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

Subscription	Select your subscription
Azure Key Vault Name	Select your Key Vault

SQL

Selection Method	Confirm default selection, “ From Azure subscription ”
Subscription	Select your subscription
Server Name	Select your Azure SQL Server {e.g., sharedsds }
Database Name	Select your Azure SQL Database {e.g., sharedsd }
Authentication Type	Select “SQL authentication”
User Name	Enter the “ server admin login ” value used during instantiation of the SQL Database Server
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 Azure SQL Server administrator password

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

On the “Create virtual machine” page, **Basics** tab, enter values for the following items:

Subscription	Select your subscription
Resource Group	Select your resource group
Virtual Machine Name	Enter a meaningful name aligned with standards
Region	Match resource group selection
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” page, **Basics** tab, enter values for the following items:

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

On the “Create virtual machine” page, “SQL Server settings” tab, enter values for the following items:

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	

No additional changes are required. Review settings on the remaining tabs.

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.

On the “Internet Explorer Enhanced Security Configuration” popup, click Off under Administrators and Users, and then click the OK button.

Sample Database

Browse to [AdventureWorks sample databases - SQL Server | Microsoft Docs](#) 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 dropdown
- On the “**Restore Database**” popup, enter AdventureWorks in the “**To database**” text box, “**Destination for restore**” interface grouping
- On the “**Restore Database**” popup, click the “**From device**” radio button in the “**Source for restore**” interface grouping
- Click the **Ellipses** button and in the resulting “**Specify Backup**” popup, 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**” popup, check the box next to new item in “**Select the backup sets to restore**” and then click the **OK** button

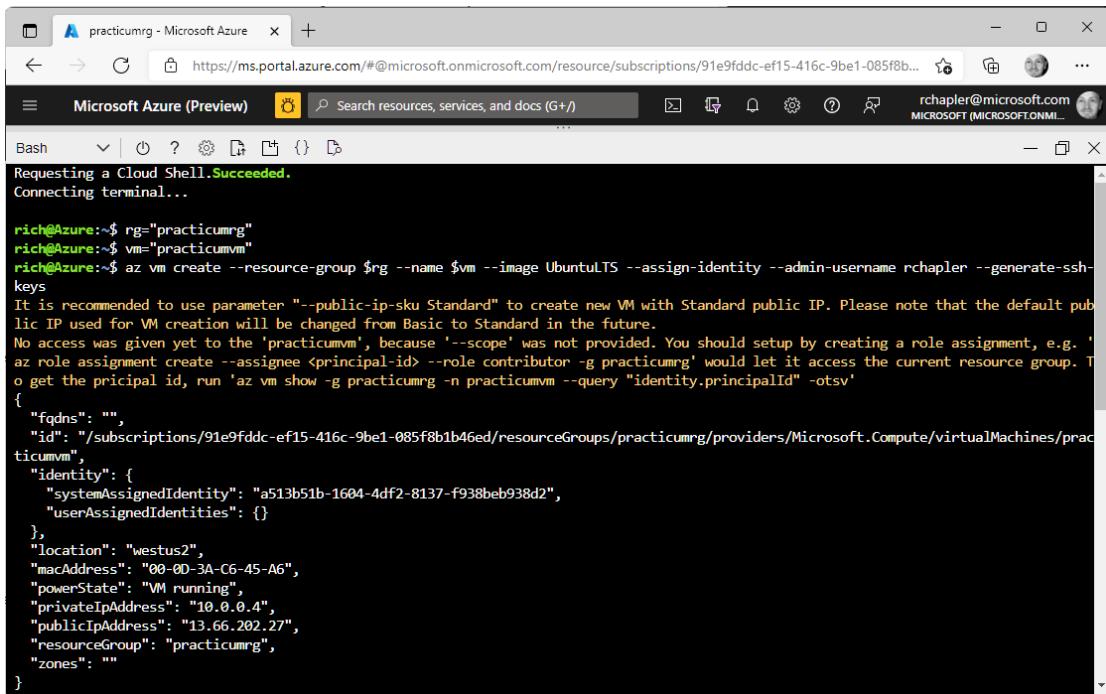
When you have successfully completed these steps, you will see a popup that says “**The restore of database ‘AdventureWorks’ completed successfully.**

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” dropdown.



The screenshot shows a Microsoft Azure Cloud Shell interface. The browser tab is titled "practicumrg - Microsoft Azure". The URL is "https://ms.portal.azure.com/#@microsoft.onmicrosoft.com/resource/subscriptions/91e9fddc-ef15-416c-9be1-085f8b...". The Cloud Shell environment is set to "Bash". The terminal output shows the creation of a resource group and a virtual machine, including setting identities and generating SSH keys. A detailed JSON object representing the VM configuration is displayed at the bottom of the terminal window.

```
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="practicumrg"
```

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

```
vm="practicumvm"
```

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+HgNSdWgb9+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@practicumvm**.

Source



Migration from On-Prem

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

Step 1: Understand Use Case

Work with your customer to understand requirements; 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 in Azure, and the on-prem source will be deprecated over time”

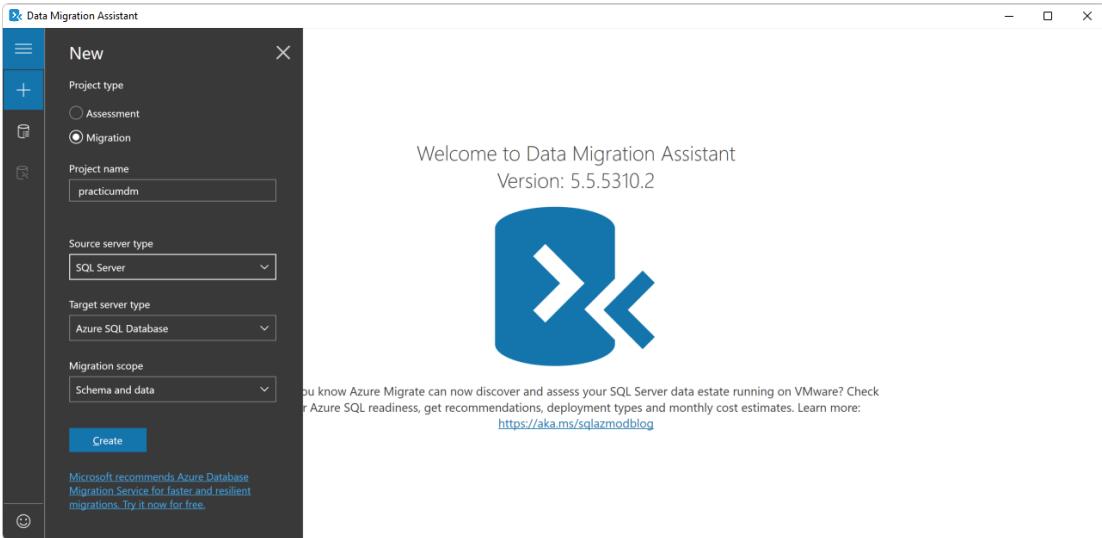
Step 2: Instantiate Prerequisites

This exercise requires the following resource(s):

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

Step 3: Create Migration Project

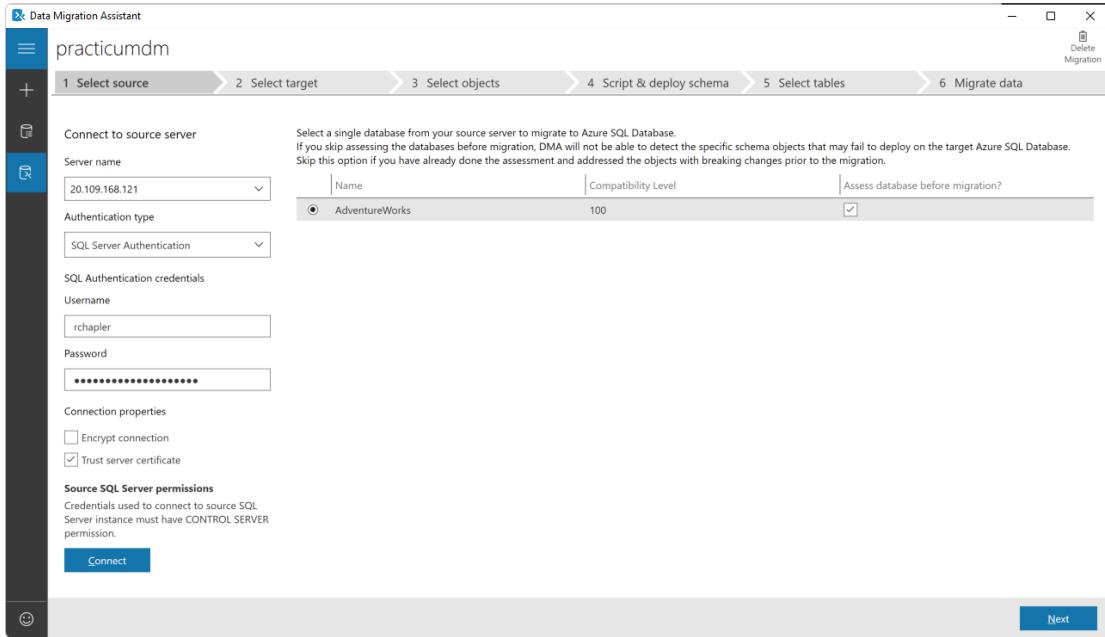
Open **Data Migration Assistant** and click + to create a new project



On the “Create Data Share” popout, enter values for the following items:

Project Type	Select the Migration radio button
Project Name	Enter a meaningful name aligned with standards
Source Server Type	Select “SQL Server”
Target Server Type	Select “Azure SQL Database”
Migration Scope	Select “Schema and data”

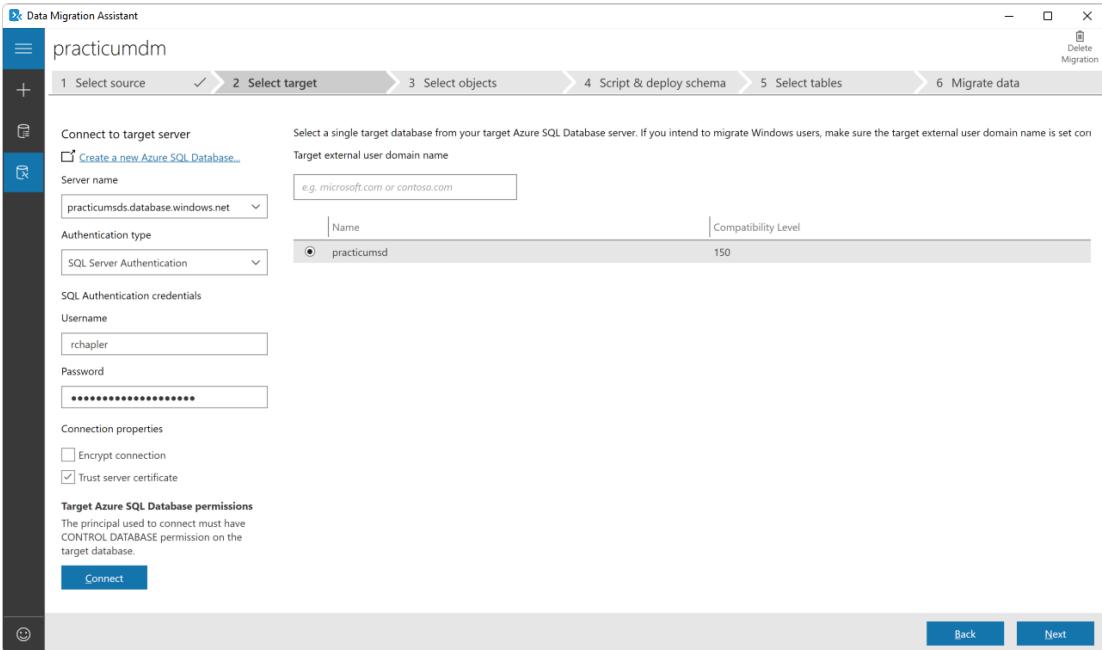
Click the **Create** button.



On the “**1. Select source**” page, enter values for the following items:

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

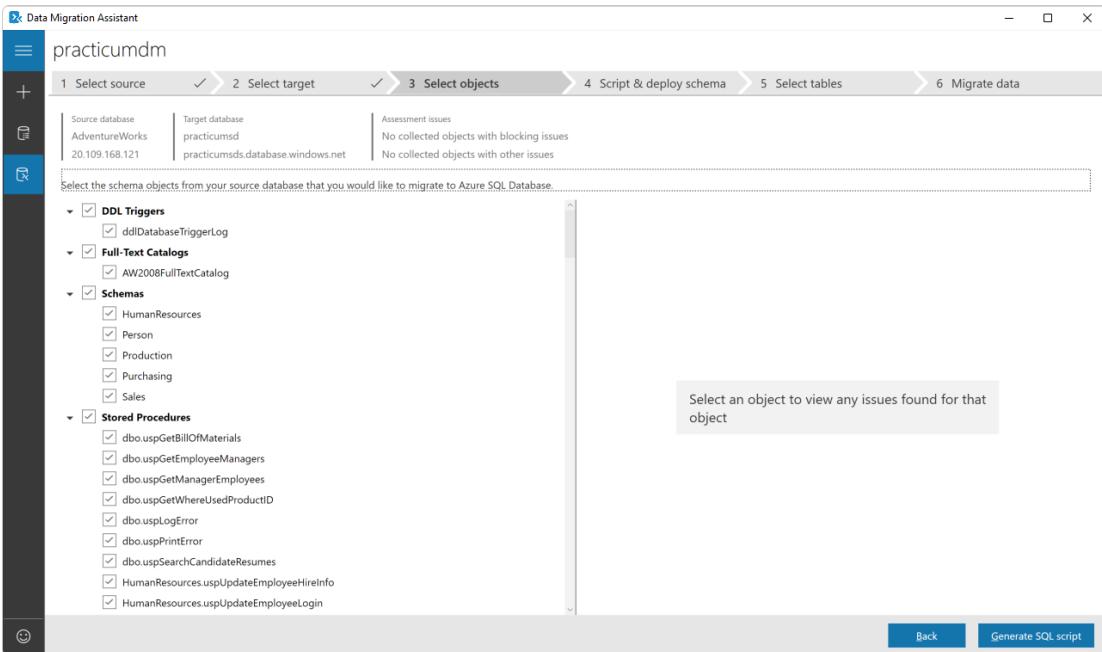
Click the **Connect** button, confirm selection of **AdventureWorks** and then click the **Next** button.



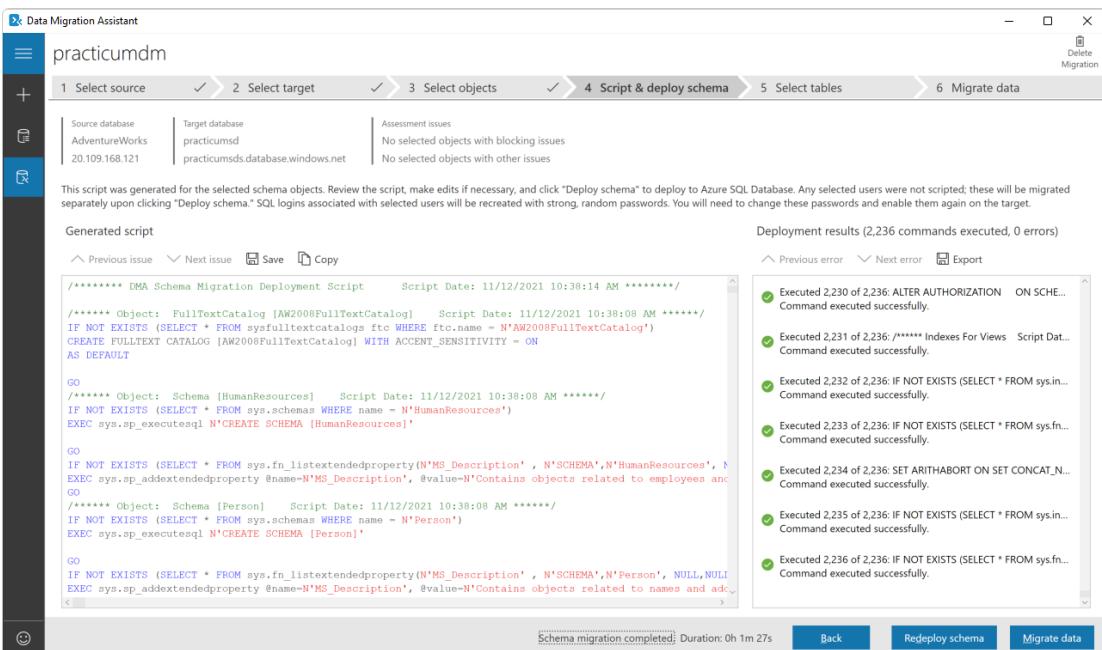
On the “**2. Select target**” page, enter values for the following items:

Server Name	Enter a meaningful name aligned with standards
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.



On the “3. Select objects” page, 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.



On the “4. Script & deploy schema” page, 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.

The screenshot shows the 'Data Migration Assistant' interface on the '5. Select tables' page. The title bar says 'practicumdm'. The top navigation bar has steps 1-6: 'Select source' (checkmark), 'Select target' (checkmark), 'Select objects' (checkmark), 'Script & deploy schema' (checkmark), 'Select tables' (checkmark), and 'Migrate data'. On the left, there's a sidebar with icons for Home, New, Open, Save, and Delete. The main area shows 'Selected tables (65/71)'. A table lists 13 tables with their row counts and status: [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), and [Person].[BusinessEntityAddress] (19,614, OK). At the bottom right are 'Back' and 'Start data migration' buttons.

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

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

The screenshot shows the 'Data Migration Assistant' interface on the '5. Select tables' page. The title bar says 'practicumdm'. The top navigation bar has steps 1-6: 'Select source' (checkmark), 'Select target' (checkmark), 'Select objects' (checkmark), 'Script & deploy schema' (checkmark), 'Select tables' (checkmark), and 'Migrate data' (checkmark). On the left, there's a sidebar with icons for Home, New, Open, Save, and Delete. The main area shows 'Server objects' (65 total, 0 in-progress, 65 successful, 0 warnings, 0 failed). Below it is a 'Tables (65)' section. A table lists 7 tables with their migration details: [Person].[Address] (Migration successful. Duration: 0 hrs 0 mins 2 secs), [Person].[AddressType] (Migration successful. Duration: 0 hrs 0 mins 1 secs), [Person].[BusinessEntity] (Migration successful. Duration: 0 hrs 0 mins 1 secs), [Person].[BusinessEntityAddress] (Migration successful. Duration: 0 hrs 0 mins 1 secs), [Person].[BusinessEntityContact] (Migration successful. Duration: 0 hrs 0 mins 1 secs), [Person].[ContactType] (Migration successful. Duration: 0 hrs 0 mins 0 secs), and [Person].[CountryRegion] (Migration successful. Duration: 0 hrs 0 mins 0 secs).

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

Objective Complete... congratulations!

External Data

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

Step 1: Understand Use Case

Work with your customer to understand requirements; example notes:

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

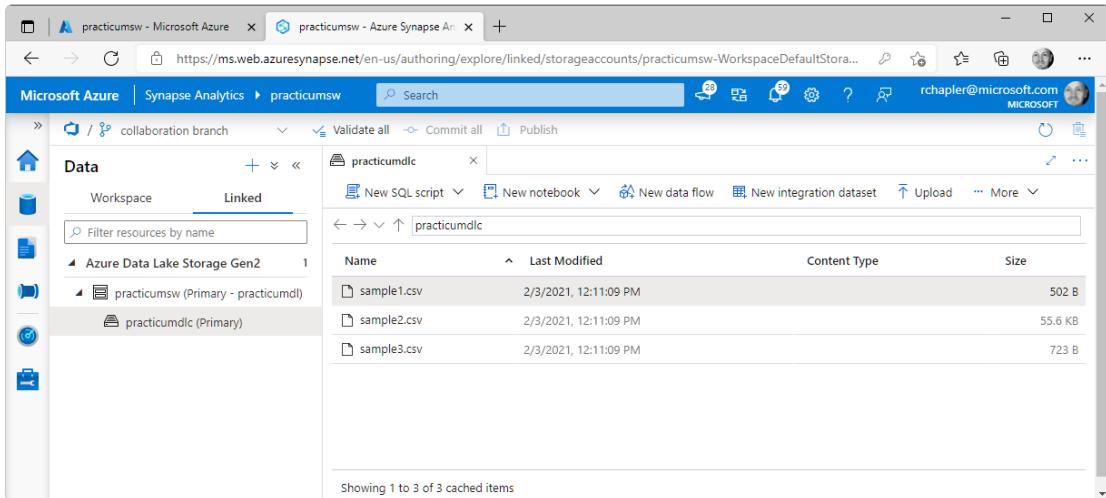
Step 2: Instantiate Prerequisites

This exercise requires the following resource(s):

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

Step 3: Explore 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.



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

You will see that the Azure Data Lake Storage Gen2 specified during Synapse instantiation {i.e., shareddlc} is surfaced with no additional configuration.

Method 1: SQL Script (OPENROWSET)

The screenshot shows the Microsoft Azure Synapse Analytics Data blade. On the left, the 'Data' sidebar is open, showing 'Linked' resources. Under 'Linked', there is a section for 'practicumdlc' which contains three CSV files: 'sample1.csv', 'sample2.csv', and 'sample3.csv'. The 'sample1.csv' file is selected. The main pane displays the contents of 'sample1.csv'.

Click the “New SQL script” button and “SELECT TOP 100 rows” from the resulting dropdown.

The screenshot shows the Microsoft Azure Synapse Analytics Data blade with a new SQL script named 'SQL script 1'. The code generated is:

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

In the 'Messages' pane, the execution log shows:

- 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://practicumdl.dfs.core.windows.net/practicumdlc/sample1.csv'.
- Total execution time: 00:00:01.590

A warning message at the bottom states: 00:00:01 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.

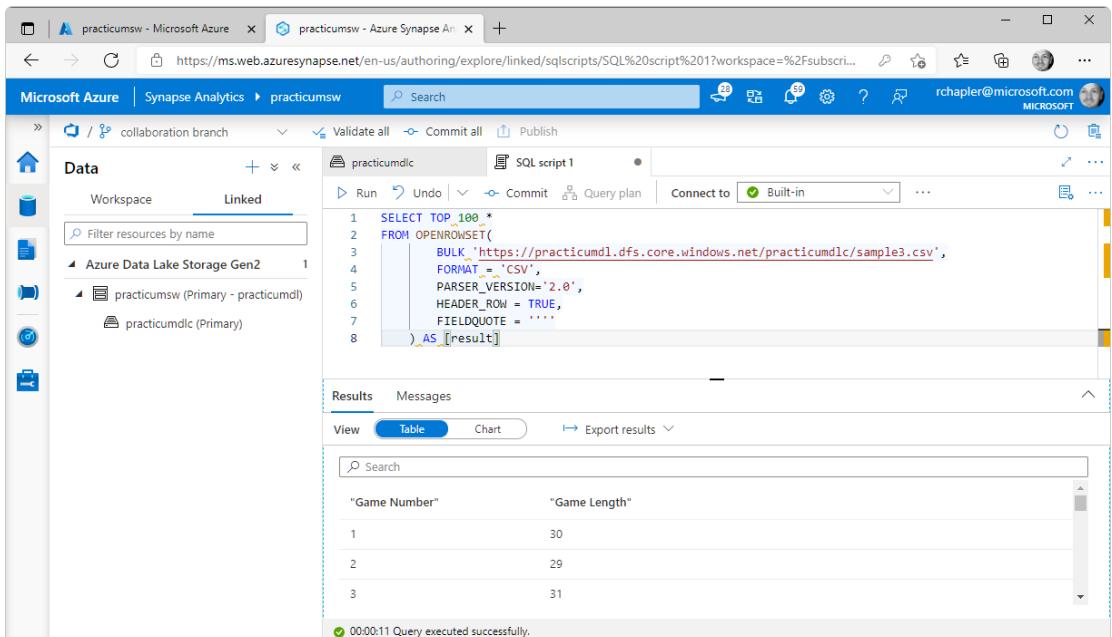
Update the code to:

```

SELECT TOP 100 *
FROM OPENROWSET(
    BULK 'https://shareddl.dfs.core.windows.net/shareddlc/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'. On the left, the 'Data' navigation pane is open, showing 'Workspace' and 'Linked' sections. Under 'Linked', there is a tree view of 'Azure Data Lake Storage Gen2' and its container 'practicumsw (Primary - practicumdl)', which contains a file 'practicumdlc (Primary)'. The main area is a 'SQL script 1' editor window with the following SQL code:

```

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 selected, displaying the query results in a table format:

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

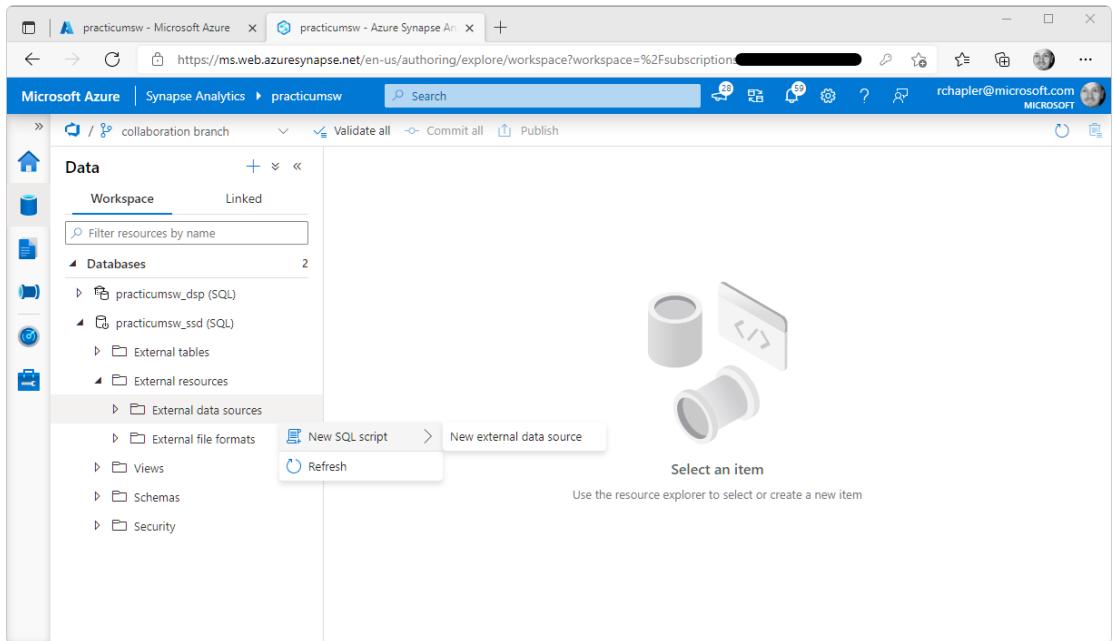
A message at the bottom of the results pane states: "00:00:11 Query executed successfully."

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**”.



Click “**New SQL script**”, and then “**New external data source**” in the dropdown.

A screenshot of the Microsoft Azure Synapse Analytics Data blade. The left navigation pane shows the 'Data' section with 'External resources' and 'External data sources' expanded. In the main area, a 'SQL script 1' tab is active. The script editor contains the following T-SQL code:

```
1 CREATE EXTERNAL DATA SOURCE [ExternalDataSource] WITH
2 (
3     LOCATION = 'https://practicumdl.blob.core.windows.net/practicumdlc'
4 )
```

The 'Results' tab at the bottom 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 Synapse Analytics workspace interface. The left sidebar is titled 'Data' and contains sections for 'Workspace' (selected), 'Linked', 'Databases' (with two items: 'practicumsw_dsp (SQL)' and 'practicumsw_ssd (SQL)'), 'External tables', 'External resources', 'External data sources', 'External file formats' (selected), 'Views', 'Schemas', and 'Security'. A dropdown menu at the bottom right of the sidebar includes 'New SQL script' (selected) and 'New external file format'. The main area displays a large icon of two cylinders and a code editor window with the placeholder text 'Select an item'.

Click “**New SQL script**”, and then “**New external file format**” in the dropdown.

The screenshot shows the Microsoft Azure Synapse Analytics workspace interface with the 'New SQL script' option selected. The main area contains 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
```

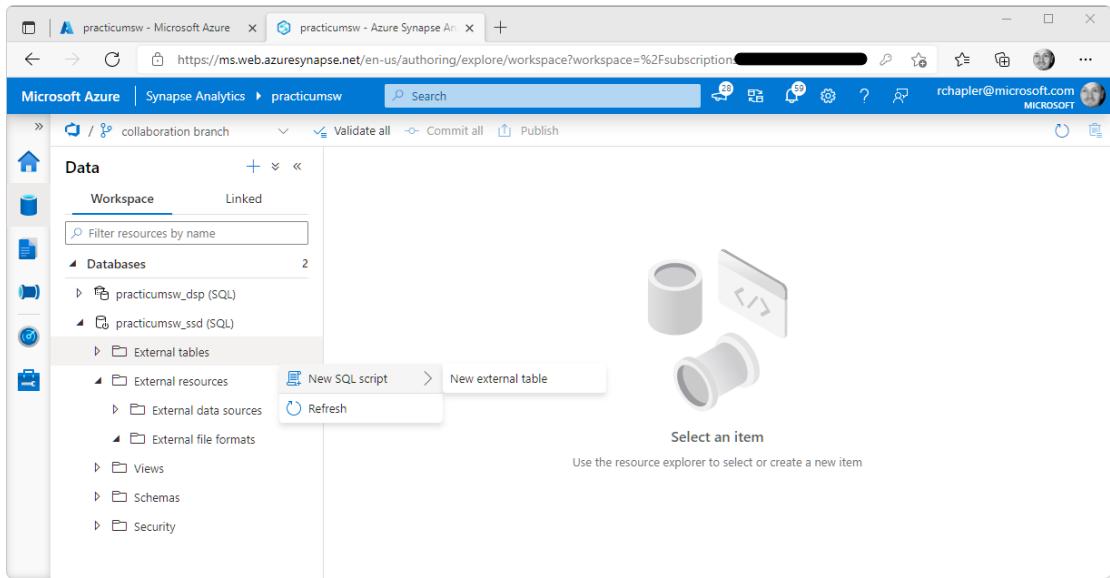
The 'Messages' tab shows:

```
00:00:02 Query executed successfully.
```

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

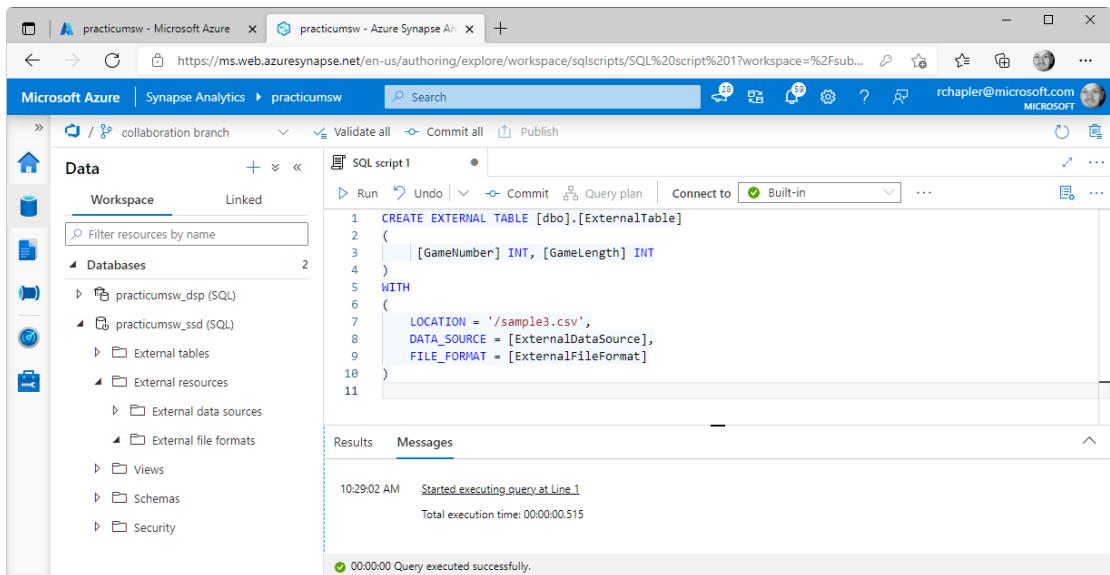
External Table

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



The screenshot shows the Microsoft Azure Synapse Analytics workspace interface. On the left, there's a navigation sidebar with icons for Home, Databases, External resources, and so on. The main area is titled 'Data' and has tabs for 'Workspace' and 'Linked'. Under 'External resources', the 'External tables' section is selected. A dropdown menu is open at the top of this list, showing 'New SQL script' and 'New external table' as options. Below the dropdown, there's a placeholder text '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 dropdown.



The screenshot shows the Microsoft Azure Synapse Analytics workspace interface. The left sidebar is identical to the previous screenshot. The main area now shows a SQL script editor. The title bar says 'SQL script 1'. The editor contains the following T-SQL code:

```
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
```

Below the editor, there are two tabs: 'Results' and 'Messages'. The 'Results' tab shows the execution log:

10:29:02 AM Started executing query at Line 1
Total execution time: 00:00:00.515

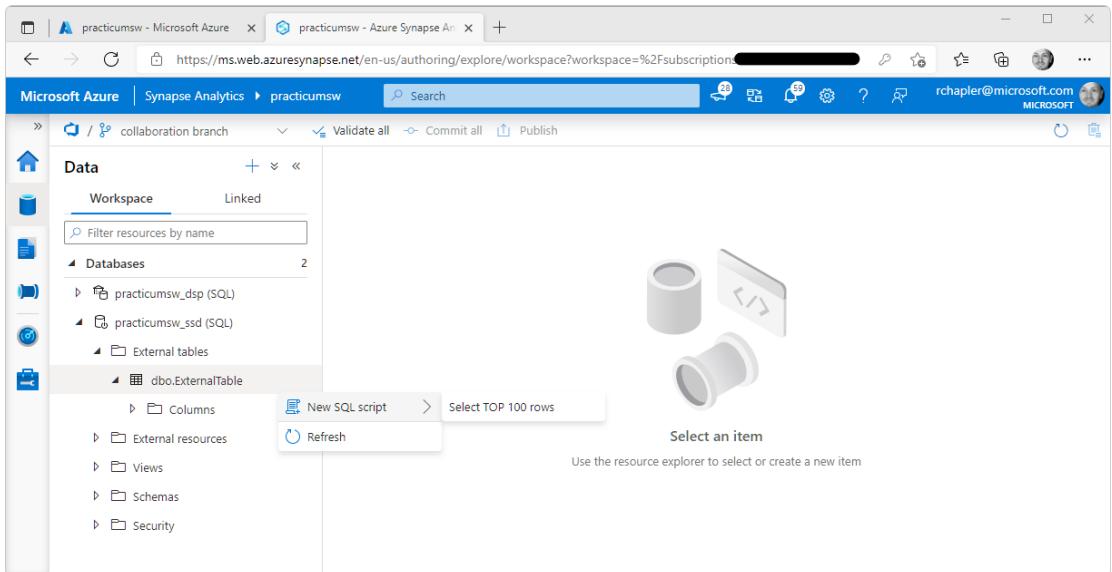
00:00:00 Query executed successfully.

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

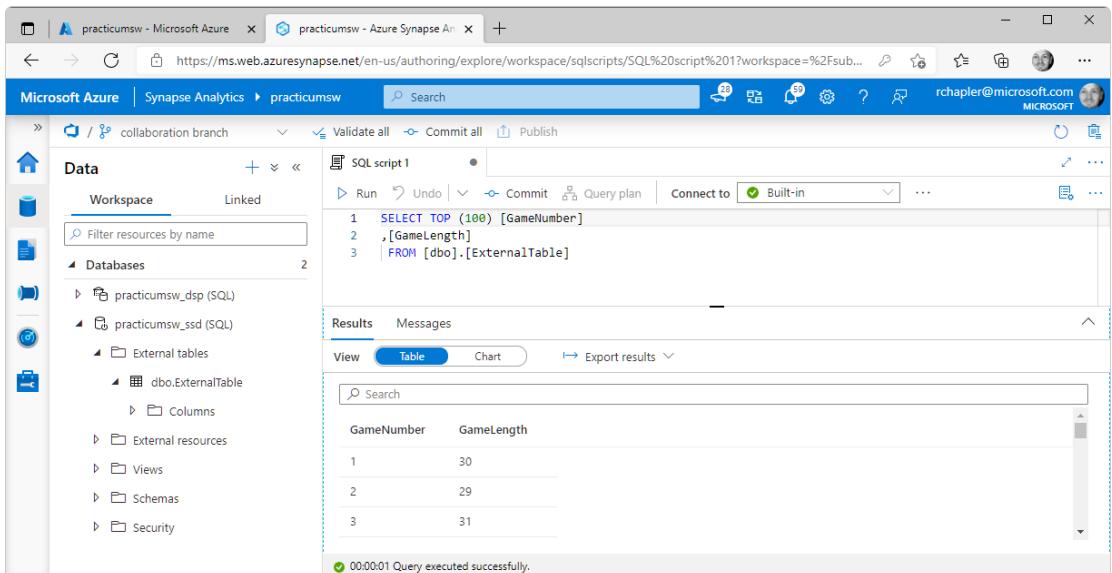
- **Schema Definition** ... replace `[Id]` `INT` with columns matching the external data source
- **LOCATION** ... replace `'/folder/file'` with values matching those in your container
- **DATA_SOURCE** ... replace `[DataSource1]` with the name used in [Create External Data Source](#)
- **FILE_FORMAT**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



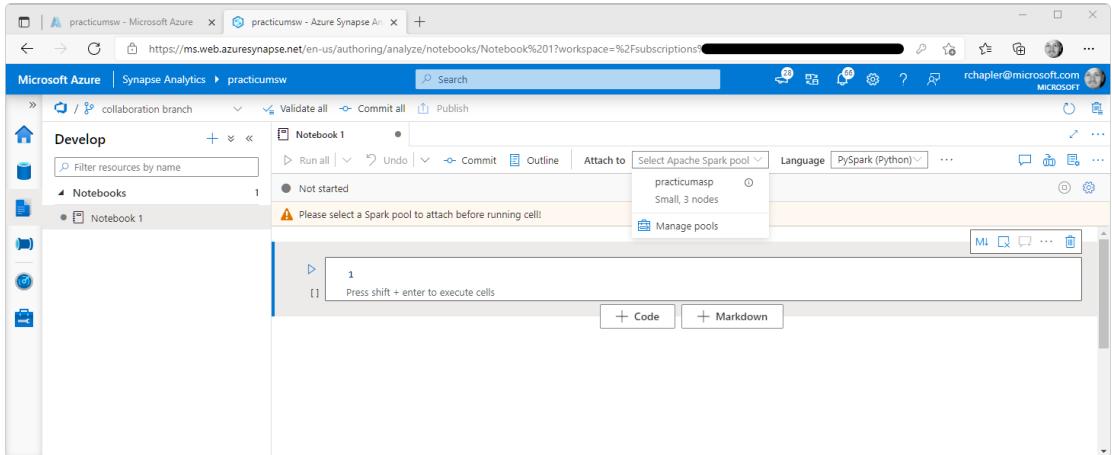
Click Run.

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 dropdown.



Select your Apache Spark Pool from the “**Attach to**” dropdown.

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

```
%pyspark  
theData = spark.read.load('abfss://shareddlc@shareddl.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

collaboration branch Validate all Commit all Publish

Develop + <

Notebooks Notebook 1

Filter resources by name

Notebook 1 Ready

Run all Undo Commit Outline Attach to: practicumsap Language: PySpark (Python) ...

[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

✓ 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 | 2 | 0 | 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" ... "2012" "2013" "2014" "2015"
0 May 0.1 0 0 ... 2 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 1
7 Dec 0.0 1 0 ... 0 0 0 1

[8 rows x 13 columns]

+ Code + Markdown

Objective Complete... congratulations!

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.

Step 1: Understand Use Case

Work with your customer to understand requirements; example notes:

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

Step 2: Instantiate Prerequisites

This exercise requires the following resource(s):

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

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

Step 3: 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 portal. 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.

On the “Source data store” page, enter values for the following items:

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 (successful). At the bottom are "Commit" and "Cancel" buttons.

On the “**New connection (Azure Data Lake Storage Gen2)**” page, enter values for the following items:

Name	Enter a meaningful name aligned with standards
Connect via...	Confirm default selection, AutoResolveIntegrationRuntime
Account Selection Method	Confirm default selection, “ From Azure subscription ”
Azure Subscription	Select your 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**” page, click **Browse** to select container/file {e.g., “shareddlc/sample1.csv”}.

No additional changes are required. Review settings on the remaining tabs.

Click the “**Next >**” button.

Source > Configuration

The screenshot shows the Microsoft Azure Copy Data tool interface. The left sidebar lists five steps: Properties (selected), Source, Dataset, Configuration, Target, Settings, and Review and finish. The main area is titled "File format settings". It includes fields for "File format" (set to "Text format"), "Column delimiter" (set to "Comma (,)"), "Row delimiter" (set to "Line feed (\n)"), and "Compression type" (set to "None"). A checkbox for "First row as header" is checked. Below these are sections for "Advanced" and "Additional columns". At the bottom are "Previous" and "Next >" buttons, and a "Cancel" button on the right.

On the “**File format settings**” page, 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'.

On the “**Destination data store**” page, enter values for the following items:

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", "AutoResolveIntegrationRuntime" is selected. The "Connection string" tab is active, showing "From Azure subscription" selected. The "Azure subscription" dropdown shows "rchaple" (redacted). 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" field shows "rchapler". The "AKV linked service" dropdown shows "AzureKeyVault1". The "Secret name" field shows "practicumsw-adminpassword". The "Secret version" dropdown shows "Use the latest version if left blank". Below these, there are sections for "Additional connection properties" (with a "New" button), "Annotations" (with a "New" button), "Parameters", and "Advanced". At the bottom are "Previous" and "Next" buttons, a "Commit" button, a "Cancel" button, and a "Test connection" link.

On the “New connection (Azure Synapse Analytics)” popout, enter values for the following items:

Name	Enter a meaningful name aligned with standards
Connect via...	Confirm default selection, AutoResolveIntegrationRuntime

Account Selection Method	Confirm default selection, “From Azure subscription”
Azure Subscription	Select your 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 area is titled "Destination data store". It specifies the target type as "Azure Synapse Analytics", the connection as "practicumsa", and the integration runtime as "AutoResolveIntegrationRuntime". Below this, under "Source" and "Target", it shows "Azure Data Lake Storage Gen2 file" mapping to "dbo" and "Sample1" respectively. A checkbox for "Skip column mapping for all tables" is present. At the bottom are "Previous", "Next >", and "Cancel" buttons.

Back on the “**Destination data source**” page, enter values for 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** page, enter values for the following items:

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 blue 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.

Step 3: 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' named 'practicumsw'. The main area displays a pipeline named 'CopyPipeline_8th'. The 'Activities' list on the left includes '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 'Copy data' activity named 'Copy_8th' is highlighted with a green checkmark. Below it, the 'Output' tab shows a table of pipeline runs. The first run is listed with details: Name 'Copy_8th', Type 'Copy data', Run start '2021-09-22T17:38:30.320', Duration '00:00:10', Status 'Succeeded', and 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 Data workspace interface. The left navigation pane is titled 'Data' and includes sections for 'Workspace' and 'Linked'. Under 'Tables', there is a table named 'dbo.Sample1'. A context menu is open over this table, listing options: 'New SQL script', 'New notebook', 'New data flow', 'New integration dataset', 'Refresh', and 'Run'. The 'Run' option is highlighted. The main area displays a SQL script titled 'SQL script 5' with the following code:

```
1 SELECT TOP (100) [Month]
2 , [ "Average"]
3 , [ "2005"]
```

Below the script, the results are shown in a table format:

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

At the bottom of the results area, 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 dropdown menu.

Objective Complete... congratulations!

Pipeline

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

Step 1: Understand Use Case

Work with your customer to understand requirements; example notes:

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

Step 2: Instantiate Prerequisites

This exercise requires the following resource(s):

- 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.

Step 3: 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.

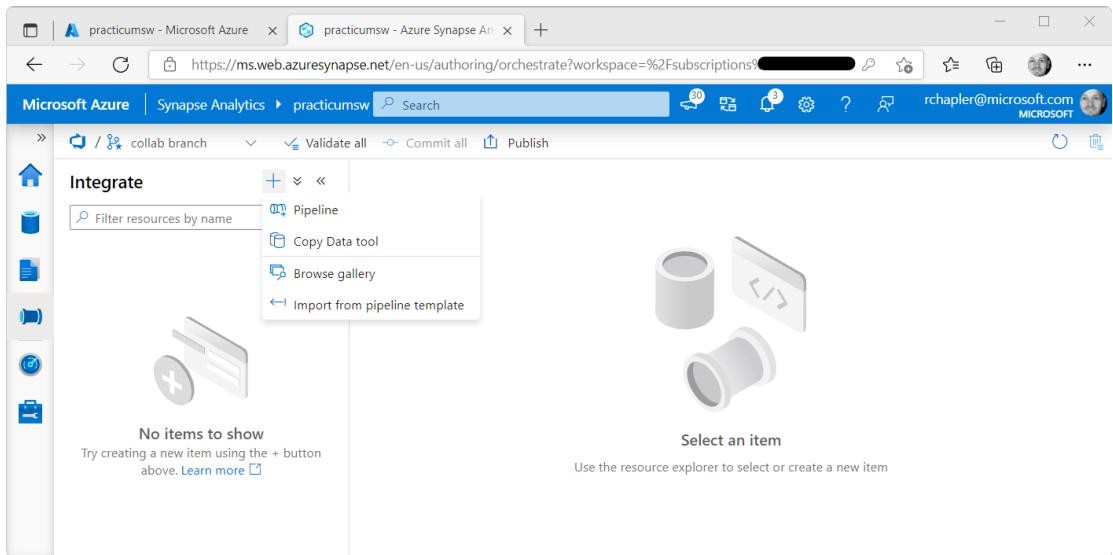
The screenshot shows the Microsoft Azure portal interface for a Synapse Analytics workspace named 'practicumsw'. The left sidebar navigation includes 'Analytics pools', 'External connections', 'Linked services' (which is currently selected), 'Azure Purview (Preview)', 'Integration', 'Triggers', 'Integration runtimes', 'Security', 'Access control', 'Credentials', 'Managed private endpoints', 'Code libraries', 'Workspace packages', 'Source control', and 'Git configuration'. The main content area is titled 'Linked services' and contains a sub-instruction: 'Linked services are much like connection strings, which define the connection information needed for Azure Synapse Analytics to connect to external resources.' Below this, there is a 'New' button, a 'Filter by name' input field, and an 'Annotations : Any' dropdown. A table lists five existing linked services: 'practicumded' (Azure Data Explorer (Kusto)), 'practicumkv' (Azure Key Vault), 'practicumsd' (Azure SQL Database), 'practicumsw-WorkspaceDefaultSqlServer' (Azure Synapse Analytics), and 'practicumsw-WorkspaceDefaultStorage' (Azure Data Lake Storage Gen2). The table has columns for Name, Type, Related, and Annotations.

Name ↑↓	Type ↑↓	Related ↑↓	Annotations ↑↓
practicumded	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., “practicumsw-WorkspaceDefault...”} in addition to instantiated items.

Step 4: 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 dropdown.

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. In the main workspace, a 'Copy data' activity is displayed in a card. Below the card, there are tabs for General, Source (highlighted), Sink, Mapping, Settings, and User properties. The 'Source' tab has a 'Source dataset *' field with a dropdown menu showing 'Select...' and a '+ New' button.

Add a “**Copy data**” activity.

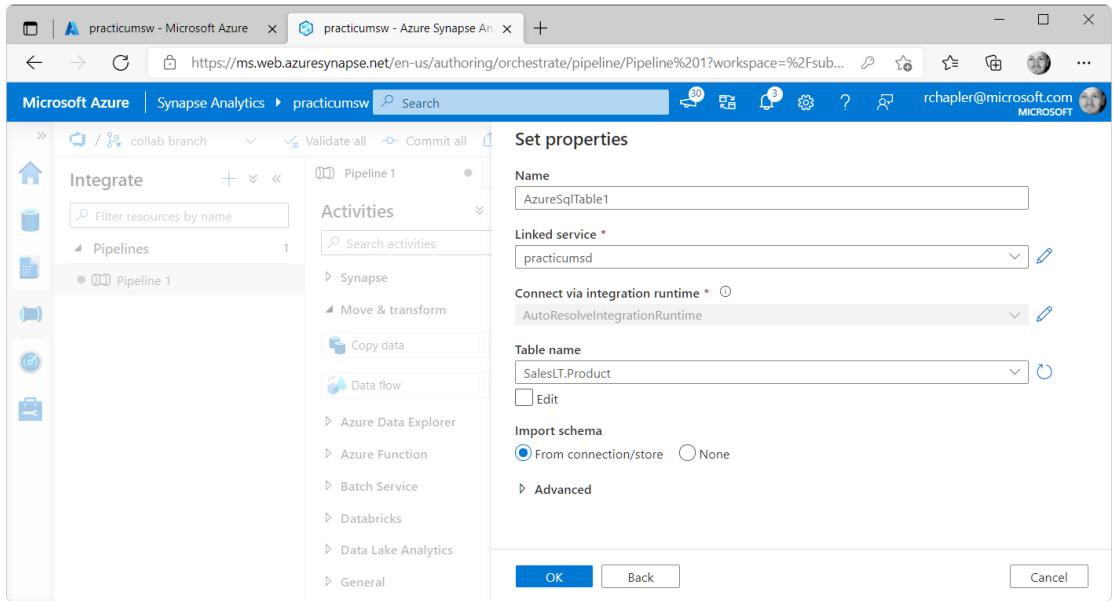
Source

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, there is a note: '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, there are tabs for All, Azure, Database, File, Generic protocol, NoSQL, and Services and apps. Under the 'Database' tab, two options are shown: 'Azure SQL Database' and 'Azure SQL Database Managed Instance'. At the bottom of the dialog are 'Continue' and 'Cancel' buttons.

On the “New integration dataset” popout, search for and then select “Azure SQL Database”.

Click the **Continue** button.



On the “Set properties” popout, enter values for the following items:

Name	Enter a meaningful name aligned with standards
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.

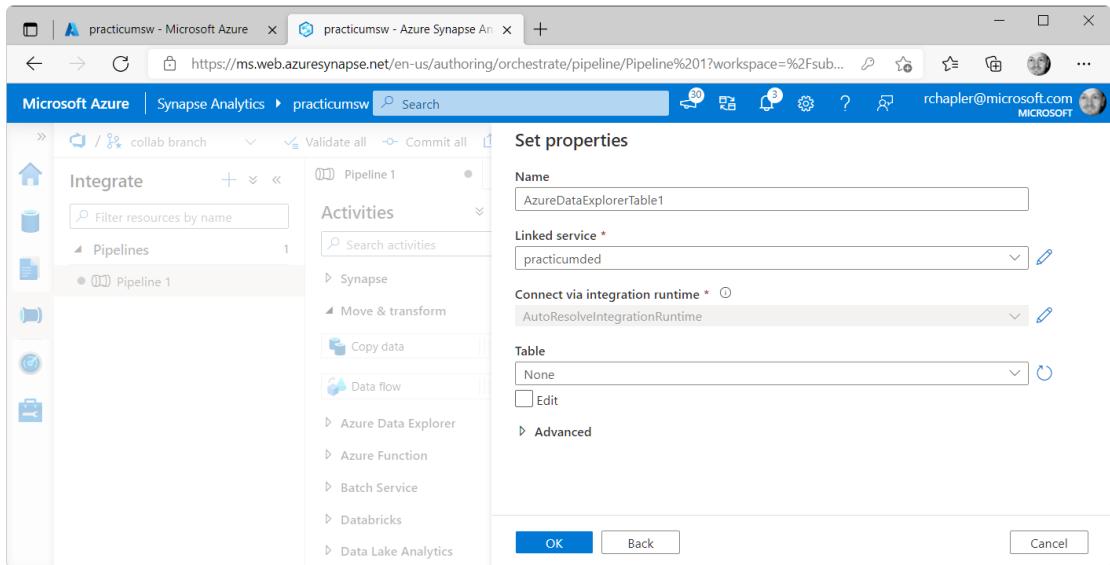
The screenshot shows the Microsoft Azure Synapse Analytics pipeline editor. On the left, the navigation pane is open with 'Integrate' selected. Under 'Pipelines', 'Pipeline 1' is selected. The main area shows a 'Copy data' activity. The 'Source' tab is active, displaying configuration for the source dataset 'AzureSqlTable1'. Other tabs include 'Sink', 'Mapping', 'Settings', and 'User properties'. A note at the bottom of the 'Source' tab 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 'Sink' tab selected. A modal dialog titled 'New integration dataset' is open. It asks to 'Select a data store' and shows a search bar with 'data explorer'. Below it, a list of data stores is shown, with 'Azure Data Explorer (Kusto)' highlighted. At the bottom of the dialog are 'Continue' and 'Cancel' buttons.

On the “New integration dataset” popout, search for and then select “Azure Data Explorer (Kusto)”.
Click the **Continue** button.



On the “Set properties” popout, enter values for the following items:

Name	Enter a meaningful name aligned with standards
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.

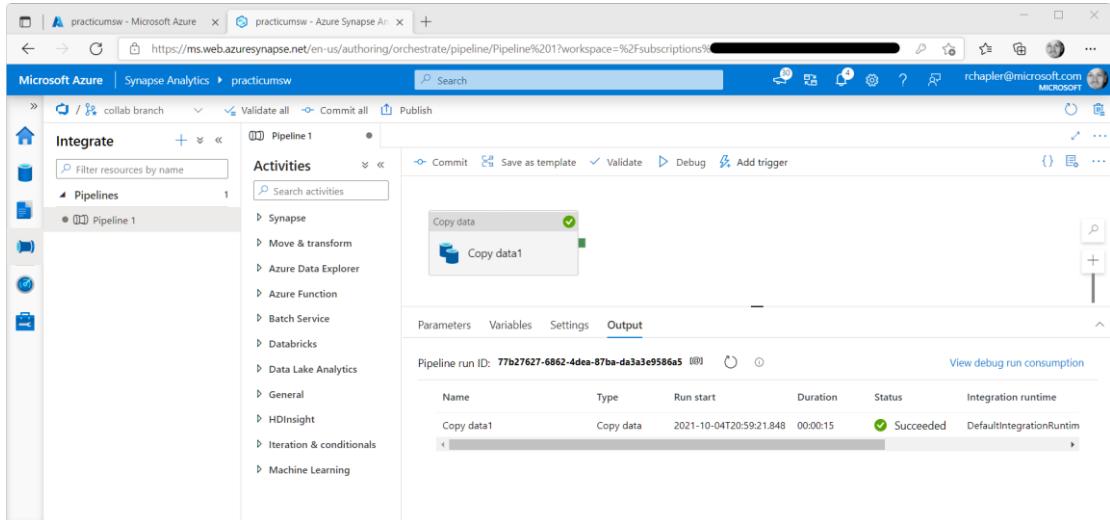
Mapping

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 precision and scale for the destination columns are set to 19 and 4 respectively.

Step 5: 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 sidebar displays 'Integrate' and 'Pipelines'. Under 'Pipelines', 'Pipeline 1' is selected, which contains a single activity named 'Copy data1'. The main workspace shows the activity details with 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 listed: 'Copy data1' (Type: Copy data), Run start: 2021-10-04T20:59:21.848, Duration: 00:00:15, Status: Succeeded (green checkmark), and Integration runtime: DefaultIntegrationRuntime. At the top of the page, there are navigation tabs for 'practicumsw - Microsoft Azure' and 'practicumsw - Azure Synapse AI', along with a search bar and user information for 'rchapler@microsoft.com'.

Objective Complete... congratulations!

Conditional Activity

Follow these instructions to demonstrate conditional logic in a pipeline.

Step 1: Understand Use Case

Work with your customer to understand requirements; example notes:

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

Step 2: Instantiate Prerequisites

This exercise requires the following resource(s):

- Synapse

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

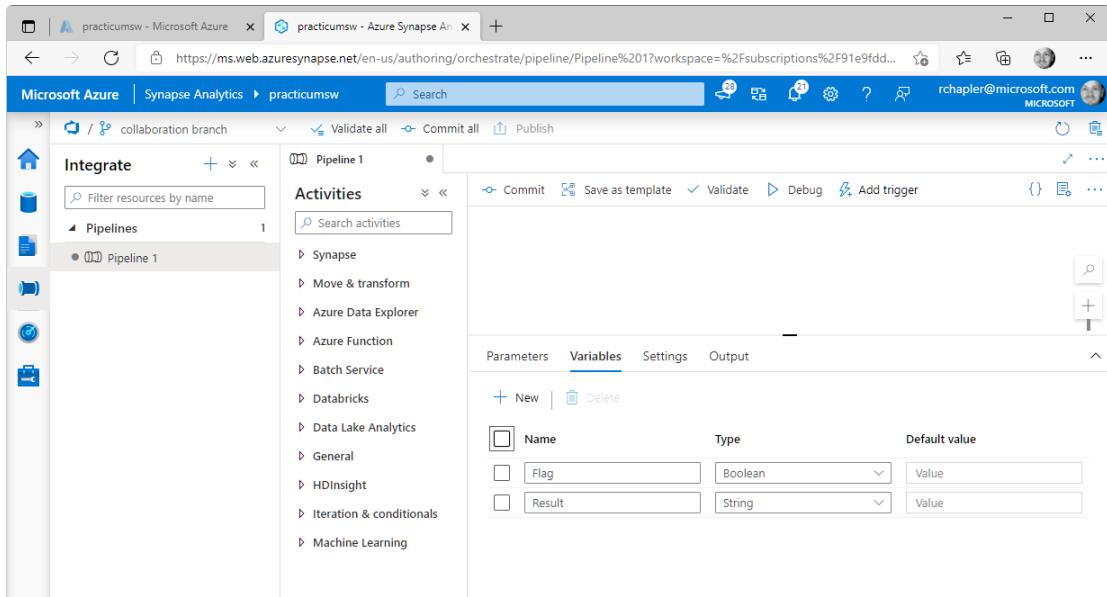
Step 3: Create Pipeline

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 dropdown.



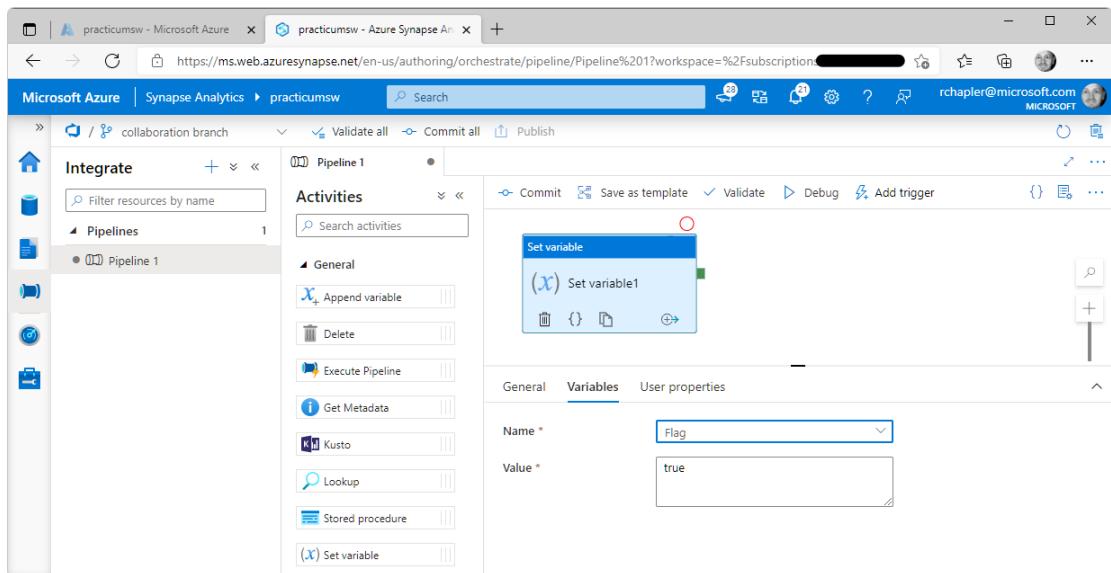
Click on the **Variables** tab.

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, and then drag-and-drop a “**Set variable**” component into the activity window.



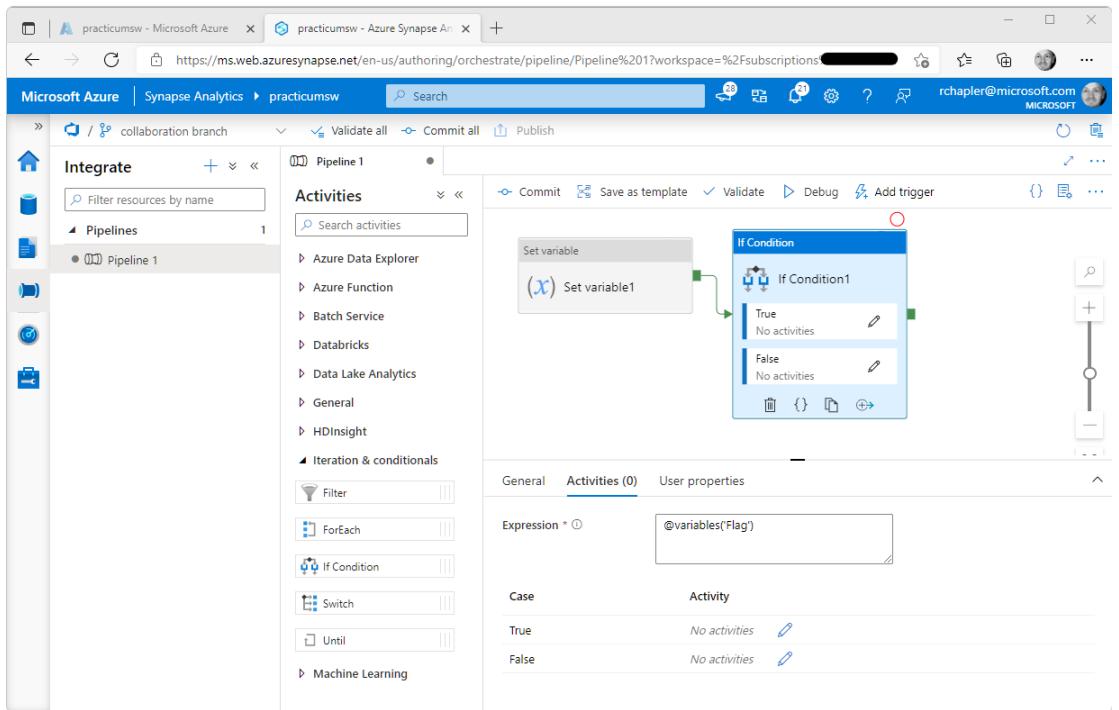
On the **Variables** tab of the “**Set Variable...**” component, enter values for the following items:

Name	Select Flag from the dropdown
Value	Enter true

Activity 2, If Condition

Expand “Iteration & conditionals” in the Activities bar.

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

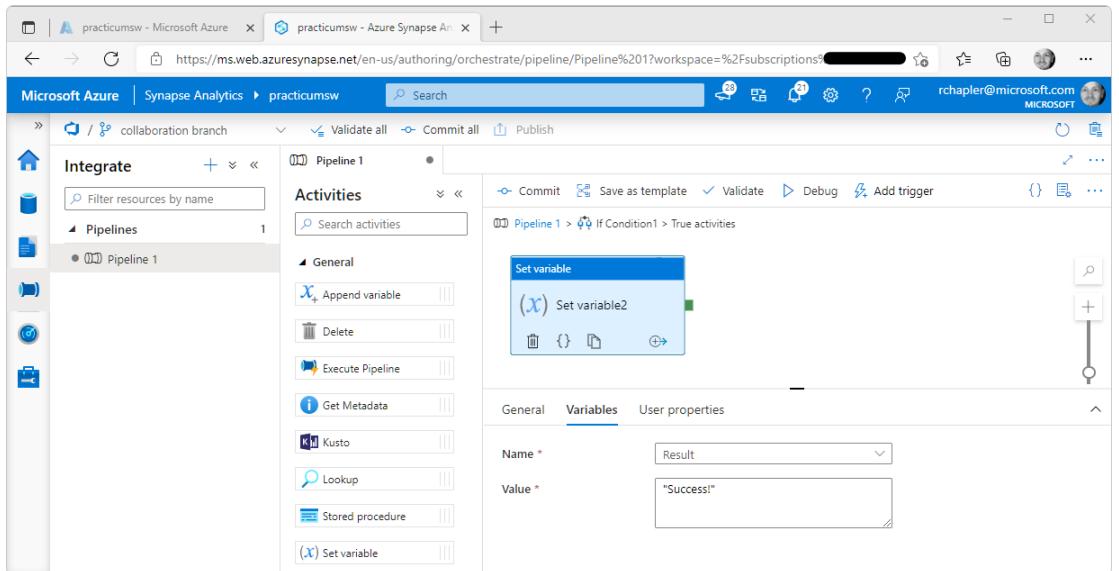


Create a dependency between the “**Set variable**” and “**If Condition**” components.

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.



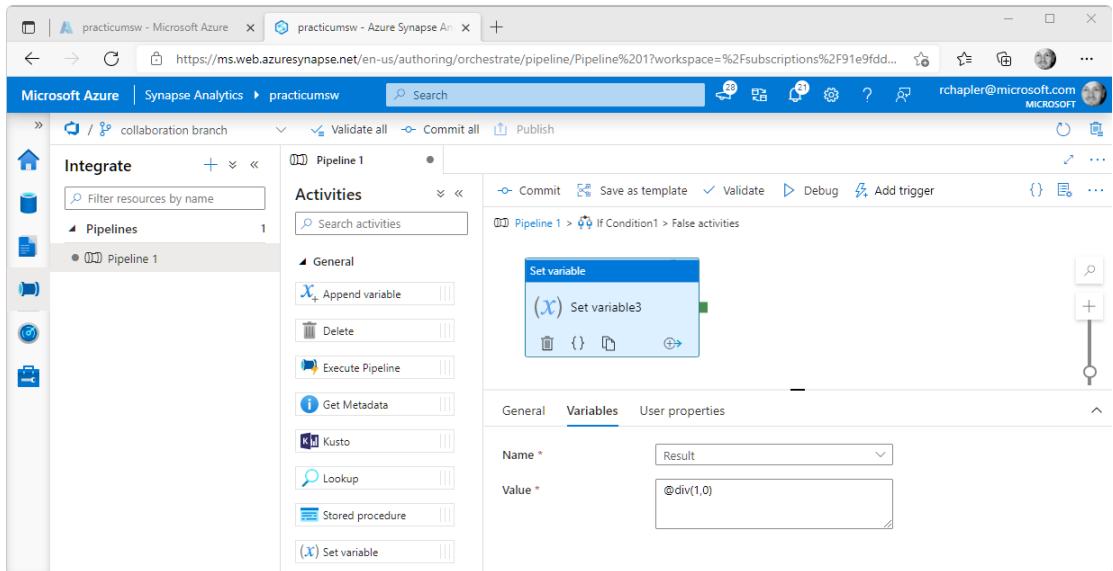
Expand **General** in the **Activities** bar, and then drag-and-drop a “**Set variable**” component into the activity window.

On the **Variables** tab of the “**Set Variable...**” component, enter values for the following items:

Name	Select Result from the dropdown
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, and then drag-and-drop a “**Set variable**” component into the activity window.

On the **Variables** tab of the “**Set Variable...**” component, enter values for the following items:

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

Step 4: Confirm Success

Click **Debug**.

The screenshot shows the Microsoft Azure Synapse Analytics pipeline editor. On the left, the navigation pane shows 'Integrate' selected, with 'Pipelines' expanded and 'Pipeline 1' selected. The main workspace displays a pipeline diagram. The pipeline starts with a 'Set variable' activity named 'Set variable1'. An arrow points from this activity to an 'If Condition' activity. The 'If Condition1' activity has two branches: 'True' leading to '1 activities' and 'False' leading to 'No activities'. Below the pipeline diagram, there are tabs for 'Parameters', 'Variables', 'Settings', and 'Output'. The 'Output' tab is active, showing a table of pipeline runs. The first run, 'Set variable2', is listed with a status of 'Succeeded'. The second run, 'If Condition1', is also 'Succeeded'. The third run, 'Set variable1', is also 'Succeeded'. At the top right of the editor, there are buttons for 'Commit', 'Save as template', 'Validate', 'Debug', and 'Add trigger'. The status bar at the bottom indicates the pipeline ID: 'Pipeline run ID: 08bb1dff-2b6d-4023-8ab7-750cbcff2423'.

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

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

This screenshot shows the same pipeline editor interface as the previous one, but with a different configuration. The 'Set variable1' activity now has a red error circle icon. The 'Variables' tab is selected for this activity, showing a 'Name' field set to 'Flag' and a 'Value' field set to 'false'. The rest of the pipeline diagram and the run history table below remain the same, showing successful runs for the other components.

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.

Objective Complete... congratulations!

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.

Step 1: Understand Use Case

Work with your customer to understand requirements; 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.

Step 2: Instantiate Prerequisites

This exercise requires the following resource(s):

- 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.

Step 3: Prepare Source

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

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

Step 4: Prepare Destination

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

Create Destination Table

Execute the following query:

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

Logic explained...

- 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 query:

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

Logic explained...

- “**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`

Create 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 query:

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

Create _Waterline Table

Execute the following query:

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

Create _Pending Table

Execute the following query:

```
.create table _product_pending ( _id:string, _op:string ) with ( folder = 'Staging' )
```

Name, commit and / or publish your script.

The screenshot shows the Azure Synapse Analytics Data Explorer interface. A script named "Product" is being published. The status bar indicates "Publishing completed" with a green checkmark and "Successfully published". The script itself contains several Kusto queries for creating tables, a materialized view, and a function. The bottom status bar shows "00:00:00 Query executed successfully."

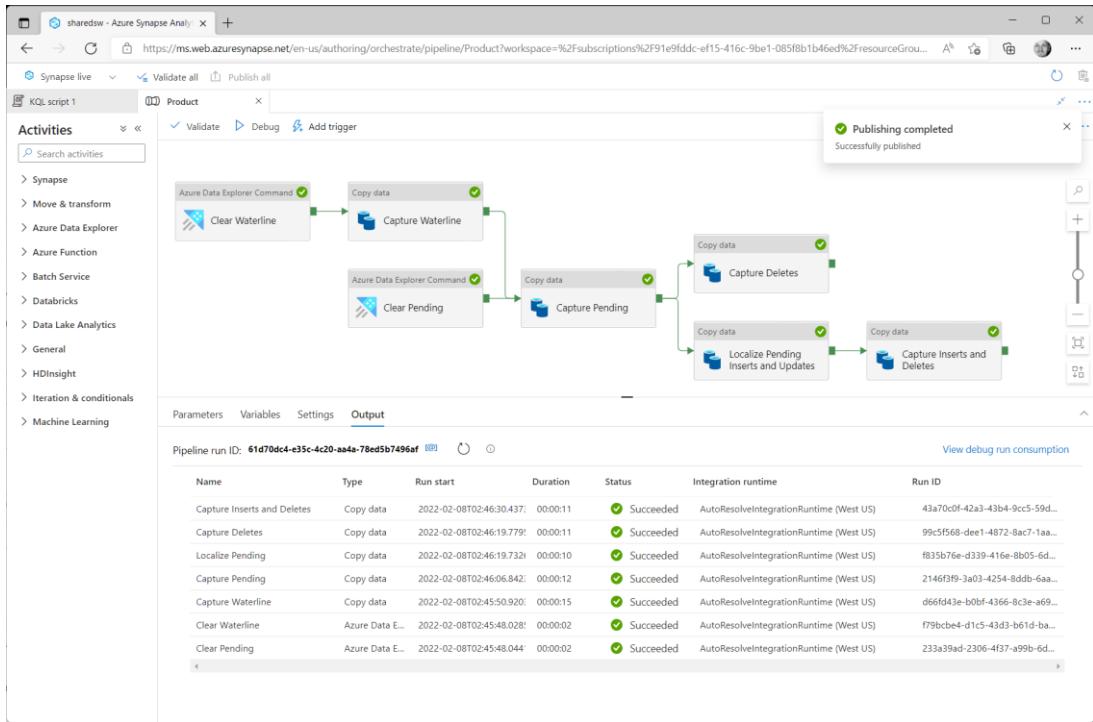
```

1 .create table _product (
2     ProductId:int
3     , ProductNumber:string
4     , Name:string
5     , ListPrice:decimal
6     , ModifiedDate:datetime
7     , _id:string
8     , _op:string
9     , _dt:datetime
10    ) with ( folder = 'Bronze' )
11
12 .create materialized-view with ( backfill = true ) _product_latest on table _product {
13     _product
14     | extend _opdt = ingestion_time()
15     | summarize arg_max(_opdt, *) by _id
16 }
17
18 .create-or-alter function product( excludeDeletes: bool = true )
19     { _product_latest | where _op != iif( excludeDeletes, 'delete', '' ) }
20
21 .create table _product_waterline ( _id:string, _dt:datetime ) with ( folder = 'Staging' )
22
23 .create table _product_pending ( _id:string, _op:string ) with ( folder = 'Staging' )
24

```

Step 5: 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 dropdown.

Activity 1: Clear Waterline

This activity will clear any previously created waterline data.

Expand “**Azure Data Explorer**” in the **Activities** bar, and 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 dropdown.

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, enter values for the following items:

Source Dataset	Create a new “ Azure SQL Database ” dataset using the SQL linked service and the “ None ” table {e.g., “ sharedsd_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 changes are required on this tab.

On the **Sink** tab, enter values for the following items:

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

No additional changes are 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: Clear Pending

This activity will clear any previously created pending data.

Expand “**Azure Data Explorer**” in the **Activities** bar, and 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 dropdown.

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

```
.clear table _product_pending data;
```

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

Activity 4: Capture Pending

This activity will capture a dataset to the destination database that will be used to characterize pending delete operations.

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 Waterline**” and “**Clear Pending**” components.

On the **Source** tab, enter values for the following items:

Source Dataset	Select sharedded_product_waterline
Use Query	Select the Query radio button
Query	Paste the following KQL: <pre>_product_latest project _product_latest_dt = _dt, _product_latest_id = _id, _product_latest_op = _op join kind = fullouter (_product_waterline project _product_waterline_id = _id, _product_waterline_dt = _dt) on \$left._product_latest_id == \$right._product_waterline_id project _id = case(isnull(_product_waterline_dt) and _product_latest_op != 'delete', _product_latest_id ,_product_waterline_id) , _op = case(isnull(_product_waterline_dt) and _product_latest_op != 'delete', 'delete' , isnull(_product_latest_dt), 'insert' , _product_latest_dt < _product_waterline_dt, 'update' ,)) where _op in ('delete', 'insert', 'update')</pre>

No additional changes are required on this tab.

On the **Sink** tab, enter values for the following items:

Sink Dataset	Create an “Azure Data Explorer (Kusto)” dataset using the Data Explorer linked service and the “_product_pending” table {e.g., “sharedded_product_pending”}
---------------------	---

Database and Table	Confirm values
---------------------------	----------------

No additional changes are 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 Deletes (from destination)

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, enter values for the following items:

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

No additional changes are required on this tab.

On the **Sink** tab, enter values for the following items:

Sink Dataset	Create an “ Azure Data Explorer (Kusto) ” dataset using the Data Explorer linked service and the “ _product ” table {e.g., “ sharedded_product ”}
---------------------	---

Database and Table	Confirm values
---------------------------	----------------

No additional changes are required on this tab.

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

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

Activity 6: 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, enter values for the following items:

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

No additional changes are required on this tab.

On the **Sink** tab, enter values for the following items:

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

No additional changes are required on this tab.

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

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

Activity 7: Capture Inserts and Updates (from source)

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, enter values for the following items:

Source Dataset	Select sharedsd_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 changes are required on this tab.

On the **Sink** tab, enter values for the following items:

Sink Dataset	Select shareddd_Product
Database and Table	Confirm values

No additional changes are required on this tab.

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

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

Step 6: 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., **sharedded**} and select “**New KQL script**” from the resulting dropdown.

Paste and Run the following query 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 query:

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

Use the returned value to execute the following query:

```
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 query to confirm that the “delete” record is included in the Product table:

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

Execute the following query 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', 'Workspace' is selected, showing 'shareddep (shareddep)' which contains 'shareded' (External tables), 'Functions' (with 'product' listed), 'Materialized Views', and 'Tables'. The main area has tabs for 'KQL script 1' and 'Publish all'. The 'KQL script 1' tab contains the query '1 _product | where _id == 706'. Below the query, there are 'Results' and 'Messages' tabs, with 'Table' selected under 'View'. The results table shows two rows for ProductId 706. The final message at the bottom says '00:00:00 Query executed successfully.'

Confirmation 3, Insert Operation

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

```
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 query 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 query 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 query to confirm that the updated record is included in the Product table:

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

Step 6: 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 dropdown.

On the resulting popout, click the “Choose trigger...” dropdown and select “+ New”.

On the “New trigger” popout, enter values for the following items:

Name	Enter a meaningful name aligned with standards
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 changes are required. Review remaining tabs and then click the **Commit** button.

Objective Complete... congratulations!

Data Flow

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

Step 1: Instantiate Prerequisites

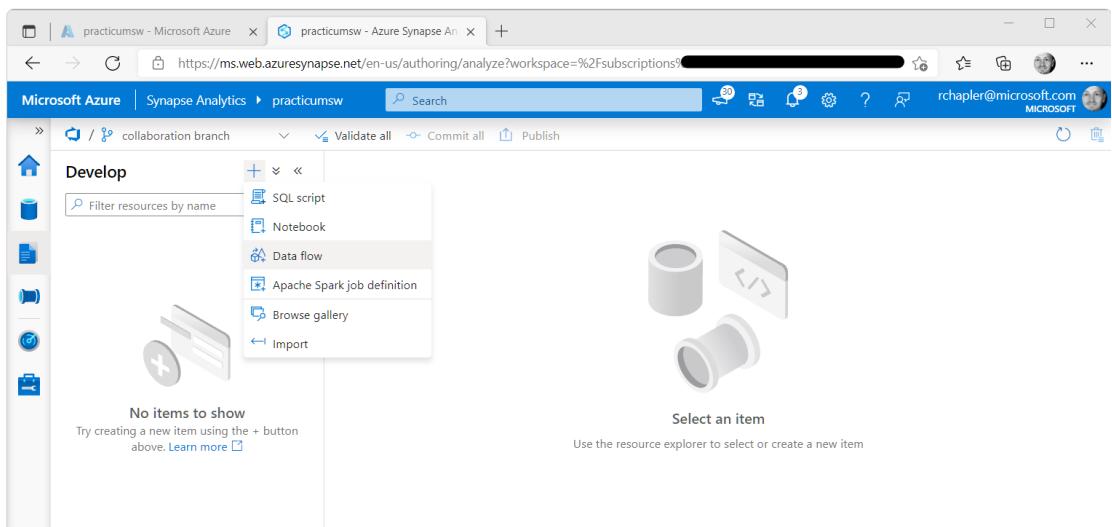
This exercise requires the following resource(s):

- 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.

Step 2: Initial Load

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



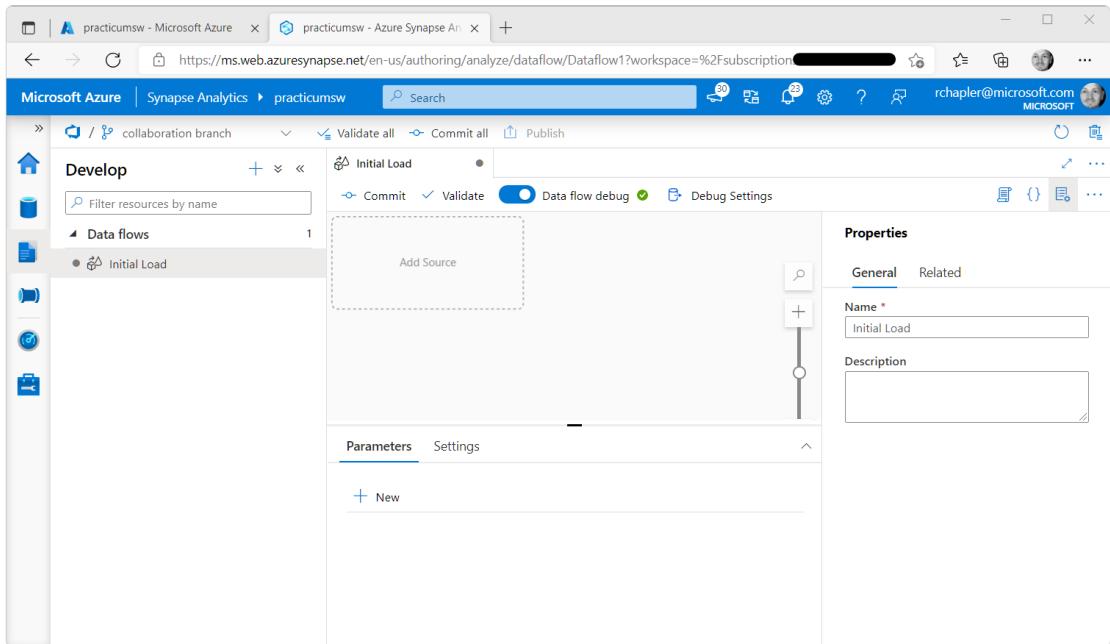
Navigate to your Synapse Analytics workspace and then click the **Open** link on the “**Open Synapse Studio**” rectangle.

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 dropdown.



On the **Properties** popout, enter values for the following items:

Name	Enter a meaningful name aligned with standards
-------------	--

Consider activating "**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. In the left sidebar, under 'Develop', 'Data flows' is selected, showing one item: 'Initial Load'. The main area displays the 'Initial Load' pipeline, which consists of a single step named 'practicumsd'. Below the pipeline, the 'Source settings' tab is active. The configuration includes:

- Output stream name:** practicumsd
- Source type:** Integration dataset (selected)
- Inline dataset type:** Azure SQL Database
- Linked service:** practicumsd (selected)
- Sampling:** Disable (selected)

A 'Test connection' button is present and shows a green checkmark indicating success.

On the “**Source settings**” tab, enter values for the following items:

Output Stream Name Enter a meaningful name aligned with standards

Source Type Select **Inline**

Inline Dataset Type Select “**Azure SQL Database**”

Linked Service Select **sharedsd**

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, the 'Develop' sidebar shows a 'Data flows' section with one item named 'Initial Load'. The main area displays the 'Initial Load' configuration. At the top, there are buttons for 'Commit', 'Validate', 'Data flow debug' (which is turned on), and 'Debug Settings'. Below this, a preview pane shows a single data source named 'practicumsd' with '0 total' columns. The 'Source options' tab is selected, showing settings for 'Input' (set to 'Table'), 'Schema name' (set to 'SalesLT'), 'Table name' (set to 'Product'), and 'Isolation level' (set to 'Read uncommitted').

On the “**Source options**” tab, enter values for the following items:

Input Confirm default setting, **Table**

Schema Name Enter SalesLT

Table Name Enter Product

The screenshot shows the Microsoft Azure Synapse Analytics Dataflow blade with the 'Projection' tab selected. The preview pane now shows '17 total' columns. The 'Projection' tab includes buttons for 'Import schema', 'Clear schema', and 'Schema options'. Below these buttons, a table lists three columns: 'Column name' (ProductID, Name, ProductNumber), 'Type' (integer, string, string), and a third column represented by a downward arrow.

Column name	Type	
ProductID	integer	
Name	string	
ProductNumber	string	

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

The screenshot shows the Microsoft Azure Synapse Analytics Dataflow interface. In the left sidebar, under 'Develop', there is a 'Data flows' section with one item: 'Initial Load'. The main area displays a flow diagram with a blue Data Lake source icon connected to a light blue sink icon labeled 'practicumsd'. Below the diagram, the 'Data preview' tab is selected, showing a table with 295 rows. The columns are ProductID, Name, ProductNumber, and Color. The data includes entries like 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), and ProductID 708 (Name: Sport-100 Helmet, Black, ProductNumber: HL-U509, Color: Black). At the bottom of the preview table, there is a 'Refresh' button.

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

Click the **Commit** button.

Sink (delta lake)

This screenshot is identical to the one above, but the 'Destination' dropdown menu in the top right of the preview panel is open, showing a list with 'Sink' selected. The rest of the interface and data preview are the same as in the previous screenshot.

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

The screenshot shows the Microsoft Azure Synapse Analytics Data Flow blade. On the left, there's a sidebar with icons for Home, Datasets, Pipelines, and Data Flows. Under 'Data flows', 'Initial Load' is selected. The main area displays a pipeline named 'Initial Load'. It consists of two stages: 'practicumsd' (Add source dataset) and 'practicumdl' (Sink). A tooltip indicates there are 17 total columns. Below the pipeline, the 'Sink' tab is active, showing configuration options:

- Output stream name:** practicumdl
- Incoming stream:** practicumsd
- Sink type:** Integration dataset (selected)
- Inline dataset type:** Delta
- Linked service:** practicumsw-WorkspaceDefaultStorage (with 'Test connection' and 'Edit' buttons)
- Options:** Allow schema drift (checked), Validate schema (unchecked)

On the **Sink** tab, enter values for the following items:

Output Stream Name	Enter a meaningful name aligned with standards
Incoming Stream	Confirm default, sharedsd
Sink Type	Select Inline
Linked Service	Select practicumsw-WorkspaceDefaultStorage
Options	Confirm that " Allow schema drift " is checked

The screenshot shows the Microsoft Azure Synapse Analytics Dataflow settings page. On the left, there's a navigation pane with icons for Home, Data flows, and Initial Load. The main area displays a data flow diagram titled 'Initial Load' with two stages: 'practicumsd' and 'practicumdl'. Below the diagram, the 'Settings' tab is selected, showing configuration options for the sink:

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

On the **Settings** tab, enter values for the following items:

File System	Enter shareddlc
Folder Path	Enter "bronze/northwind/product"
Compression Type	Select snappy
Compression Level	Select Fastest
Auto Compact	Checked
Optimize Write	Checked

No additional changes are required. Review settings on the remaining tabs.

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 dropdown.

On the **Properties** popout, 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.

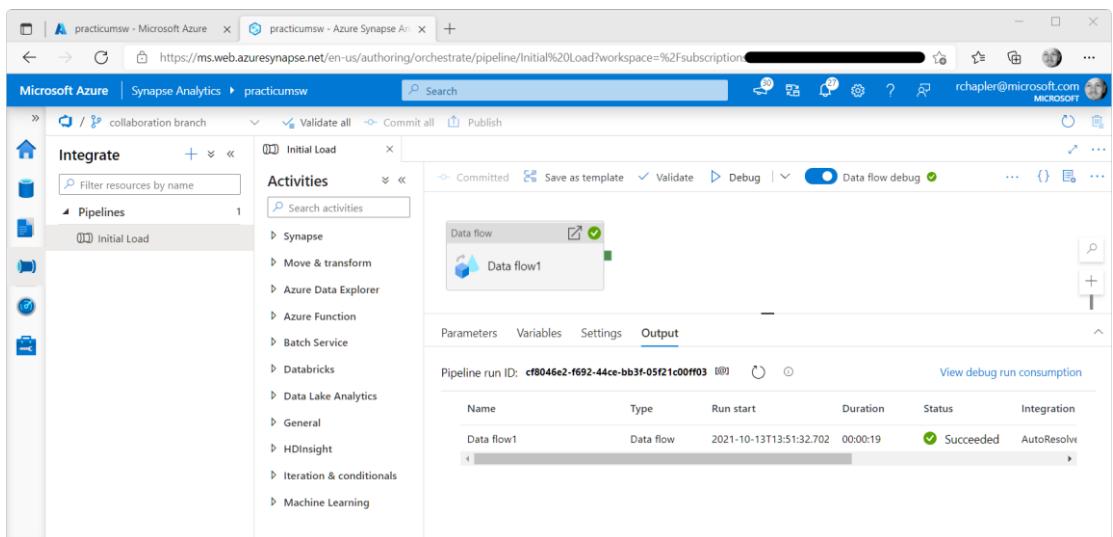
No additional changes are required. Review settings on the remaining tabs.

Click the **Commit** button.

Confirm Success

Debug Pipeline / Data Flow

Click **Debug** and confirm successful execution.



The screenshot shows the Microsoft Azure Synapse Analytics pipeline editor interface. On the left, there's a navigation pane with icons for collaboration branch, Integrate, Pipelines, and Initial Load. The Pipelines section is selected, showing one pipeline named "Initial Load". In the main workspace, there's a "Data flow" component with the name "Data flow1" and a green checkmark icon. Below the component, there are tabs for Parameters, Variables, Settings, and Output. The Output tab is active, displaying a table with one row. The table has columns for Name, Type, Run start, Duration, Status, and Integration. The single row shows "Data flow1" as the Name, "Data flow" as the Type, "2021-10-13T13:51:32.702" as the Run start, "00:00:19" as the Duration, a green checkmark in the Status column indicating "Succeeded", and "AutoResolve" in the Integration column. At the top of the workspace, there are buttons for Validate all, Commit all, Publish, Save as template, Validate, Debug, and Data flow debug. The "Data flow debug" button is currently selected, indicated by a green checkmark. The status bar at the bottom right shows the user's email address: rchaper@microsoft.com.

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 dropdown.

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

The screenshot shows the Microsoft Azure Synapse Analytics interface. On the left, there's a sidebar with 'Develop' selected, showing 'Notebooks' (1) and 'Data flows'. The main area is titled 'Notebook 1' and contains a code cell with 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
```

Below the code, a message indicates a successful job execution: 'Job execution Succeeded Spark 1 executors 4 cores'. There are also links to 'View in monitoring' and 'Open Spark UI'.

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

```
%%pyspark
df =
spark.read.format('delta').load('abfss://shareddlc@shareddl.dfs.core.windows.net/bronze/northwind/product')
print(df.count())
```

The screenshot shows the Microsoft Azure Synapse Analytics interface. The notebook has run two cells. The first cell's output was a count of records, and the second cell's output is a table view of three product records. The table has columns: ProductID, Name, ProductNumber, Color, and Status. The data is as follows:

ProductID	Name	ProductNumber	Color	Status
680	HL Road Frame - Black, 58	FR-R92B-58	Black	10+
707	Sport-100 Helmet, Red	HL-US09-R	Red	13.
708	Sport-100 Helmet, Black	HL-US09	Black	13.

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

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

Step 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 dropdown.

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 shows 'Develop' selected under 'Data flows'. A 'Data flows' item is highlighted with a blue outline. The main workspace shows a data flow named 'Incremental Load'. On the right, the 'Source settings' tab is active, containing the following configuration:

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

On the “**Source settings**” tab, enter values for the following items:

Output Stream Name	Enter a meaningful name aligned with standards
Source Type	Select Inline
Inline Dataset Type	Select Delta
Linked Service	Select practicumsw-WorkspaceDefaultStorage

Sampling

Confirm default setting, Disable

The screenshot shows the Microsoft Azure Synapse Analytics Dataflow interface. On the left, the navigation pane shows 'Develop' with 'Data flows' selected, and a list of data flows including 'Incremental Load' (selected), 'Initial Load', and 'Notebooks'. The main area displays the 'Incremental Load' data flow, which has a single step named 'practicumdlc' with '0 total' columns. Below the data flow, the 'Source options' tab is active. The configuration includes:

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

On the “**Source options**” tab, enter values for the following items:

File System Enter shareddlc

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 popout.

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. On the left, the 'Develop' sidebar shows 'Notebooks' (1), 'Data flows' (2), and 'Incremental Load'. The main area displays an 'Incremental Load' pipeline with two stages: 'practicumdl' (source) and 'practicumsd' (target). 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 'Test connection' button showing 'Connection successful')
- Sampling ***: Disable (radio button selected)

On the “**Source settings**” tab, enter values for the following items:

Output Stream Name Enter a meaningful name aligned with standards

Source Type Select **Inline**

Inline Dataset Type Select “**Azure SQL Database**”

Linked Service Select **sharedsd**

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, the navigation pane is visible with 'Develop' selected. Under 'Data flows', 'Incremental Load' is selected. The main area displays a data flow diagram titled 'Incremental Load'. The diagram consists of two main components: 'practicumdl' (Add source dataset) and 'practicumsd' (Add sink dataset). A connection line connects them. Below the diagram, the 'Source options' tab is selected in the ribbon. The 'Input' section is configured with 'Schema name *' set to 'SalesLT', 'Table name *' set to 'Product', and 'Isolation level' set to 'Read uncommitted'. Other tabs in the ribbon include 'Source settings', 'Projection', 'Optimize', 'Inspect', 'Data preview', 'Description', and '...'.

On the “**Source options**” tab, enter values for the following items:

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 sidebar with icons for Home, Notebooks, Data flows, and Pipelines. The main area displays a pipeline named 'Incremental Load'. The pipeline consists of two main stages: 'practicumdl' (source dataset) and 'practicumsd' (target dataset). The 'practicumsd' stage is highlighted with a blue border and shows '17 total' columns. Below the pipeline diagram, the 'Projection' tab is selected. Under 'Import schema', there are four columns listed: ProductID (integer), Name (string), ProductNumber (string), and Color (string). At the bottom of the projection section, there are buttons for 'Import schema' and 'Import'.

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

This screenshot shows the same Microsoft Azure Synapse Analytics Data Flow blade, but the 'Data preview' tab is now selected. At the top of the preview area, it says 'Number of rows' with counts for INSERT (100), UPDATE (0), DELETE (0), UPSERT (0), LOOKUP (0), and TOTAL (295). Below this, there are buttons for Refresh, Typecast, Modify, Map drifted, Statistics, and Remove. The main preview area displays a table with four columns: ProductID, Name, ProductNumber, and Color. Two rows are visible: one for a 'HL Road Frame - Black, 58' frame and another for a 'Sport-100 Helmet, Red' helmet.

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

Click the Commit button.

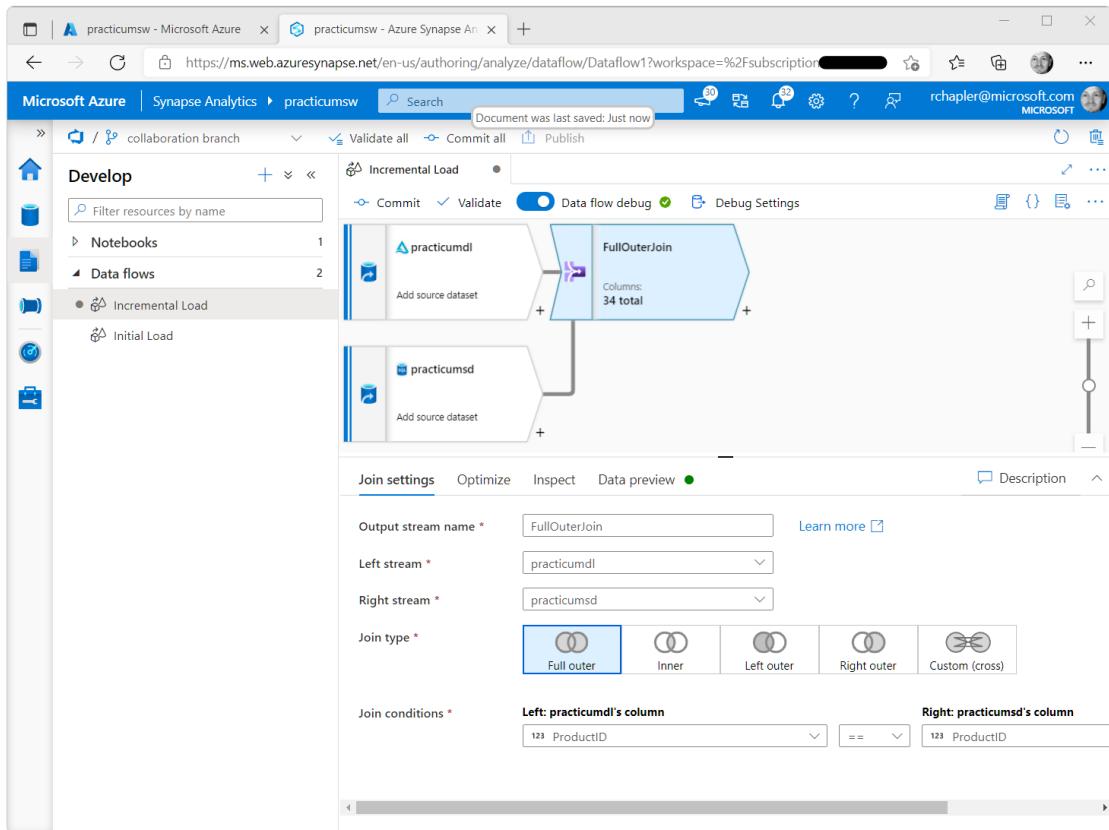
Join Sources

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 data flow, 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

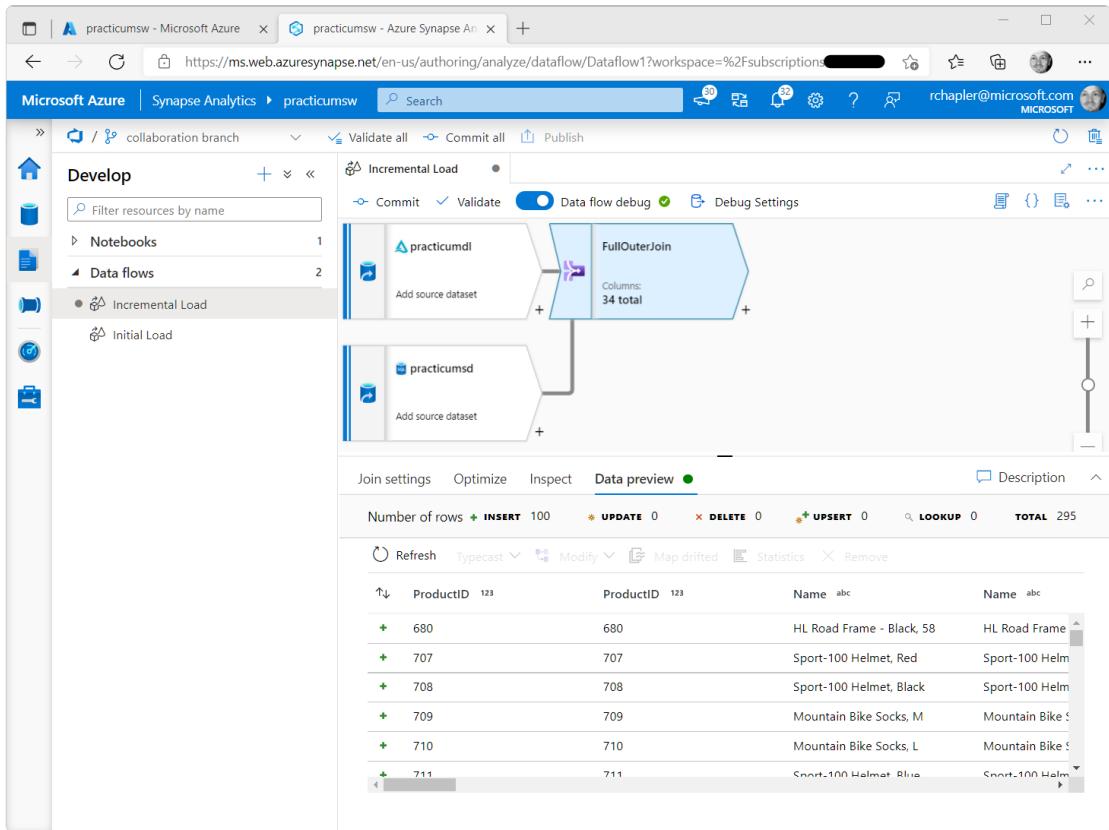
Click the + in the bottom right of the SQL source, and then select **Join** from the resulting popup list.



On the “Join settings” tab, enter values for the following items:

Output Stream Name	Enter a meaningful name aligned with standards
Left Stream	Select sharedsd
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.

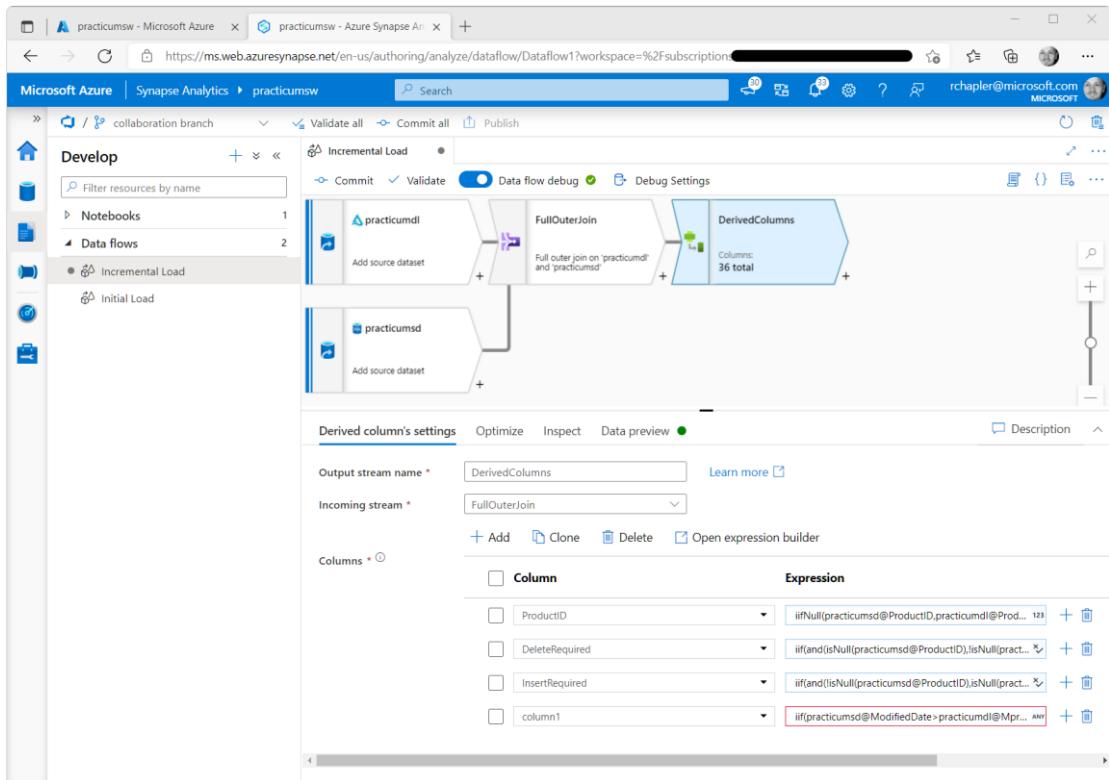
Derive Columns

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, 58	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

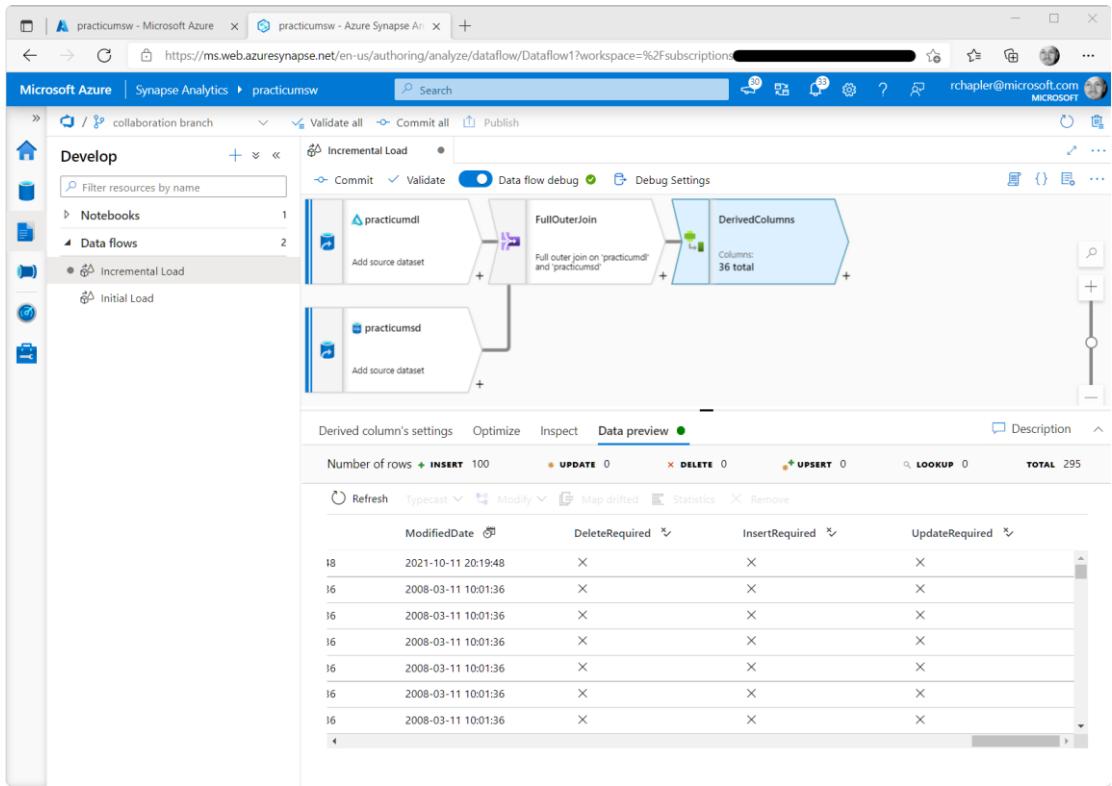
Click the + in the bottom right of the SQL source, and then select **Derived Column** from the resulting popup list.



On the “**Derived column’s settings**” tab, enter values for the following items:

Output Stream Name	Enter a meaningful name aligned with standards
Incoming Stream	Confirm default selection
Columns	<p>ProductID <code>iifNull(sharedsd@ProductID,shareddl@ProductID)</code></p> <p>DeleteRequired <code>iif(and(isNull(sharedsd@ProductID),!isNull(shareddl@ProductID)),true(),false())</code></p> <p>InsertRequired <code>iif(and(!isNull(sharedsd@ProductID),isNull(shareddl@ProductID)),true(),false())</code></p> <p>UpdateRequired <code>iif(sharedsd@ModifiedDate>shareddl@ModifiedDate,true(),false())</code></p>

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

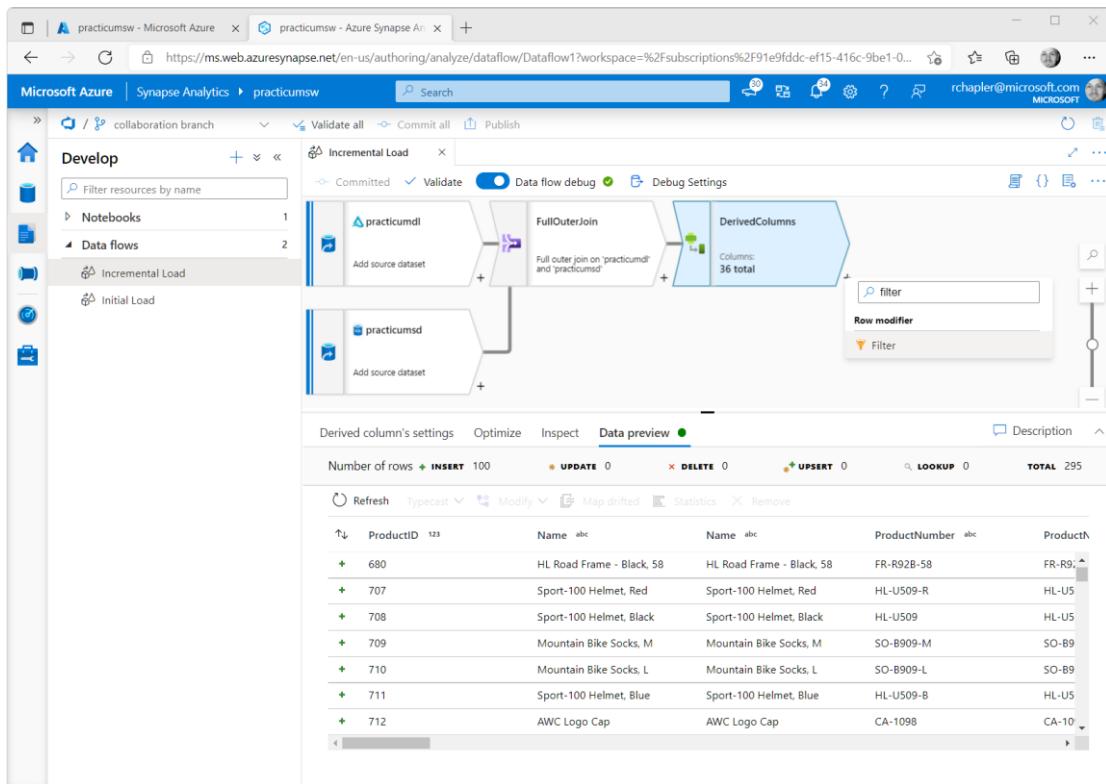


Click the **Commit** button.

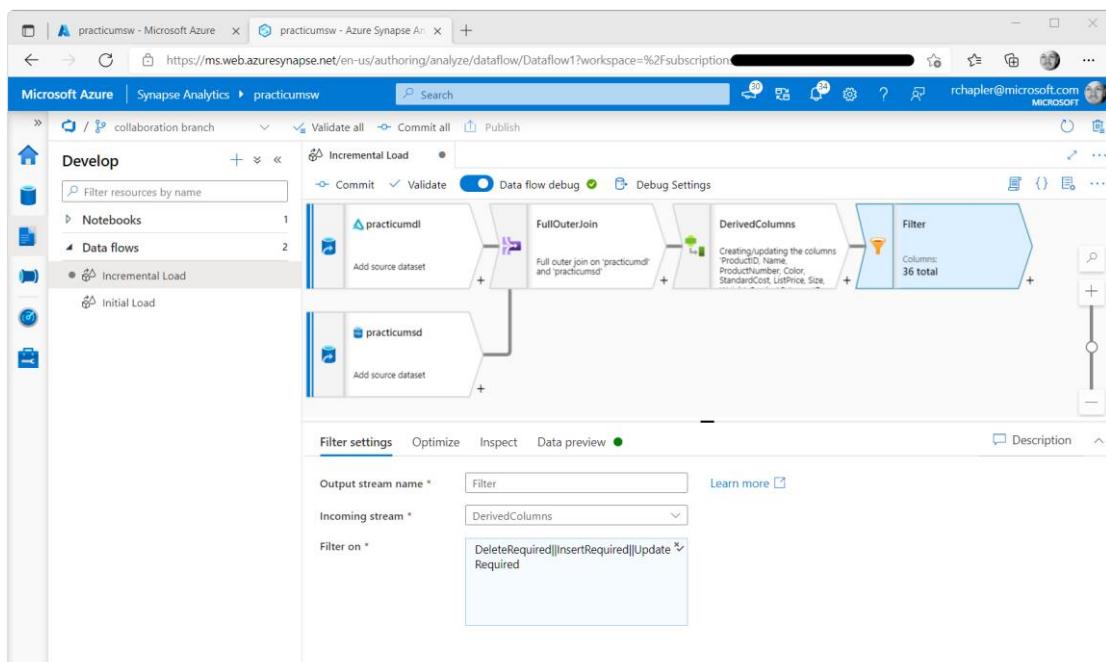
Note: I included this step to provide an easy way to preview and provide for the delete, insert and update of data... arguably, this step could be consolidated into Filter or AlterRows.

Filter Non-Actionable

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



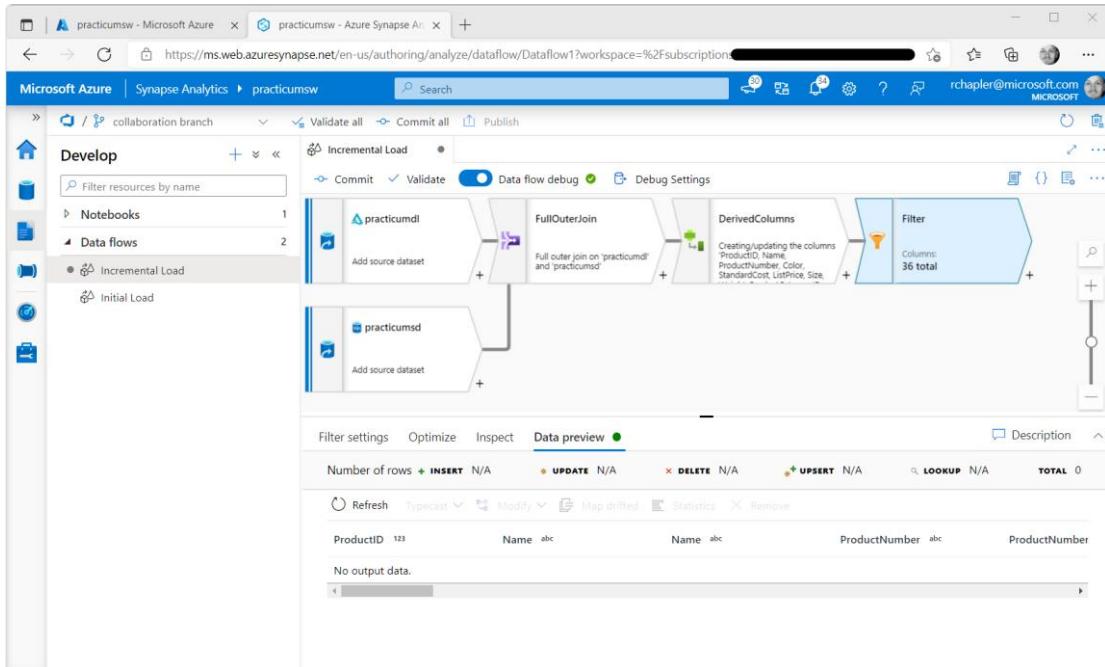
Click the + in the bottom right of the SQL source, and then select **Filter** from the resulting popup list.



On the “**Filter settings**” tab, enter values for the following items:

Output Stream Name	Enter a meaningful name aligned with standards
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 titled "practicumsd (practicumsds/practicumsd)" and includes sections for Overview, Activity log, Tags, Diagnose and solve problems, Quick start, and Query editor (preview). The "Query editor (preview)" section is currently selected. The main area is titled "practicumsd (rchap...)" and contains a "Query 1" tab. The query editor window has tabs for Run, Save query, Export data as, and Show only Editor. The query itself is:

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

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:

```
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 is open, showing navigation options like Dashboard, Resource groups, and the current resource group 'practicumsg'. The main area is titled 'practicumsd (practicumsds/practicumsd) | Query editor (preview)'. A message box in the center says 'Showing limited object explorer here. For full capability please open SSDT.' Below it, the 'Tables' node is expanded. The 'Query 1' tab is active, containing the following T-SQL code:

```
1 INSERT INTO [SalesLT].[Product]
2 ([Name],[ProductNumber],[Color],[StandardCost],[ListPrice],[SellStartDate],[rowguid],[ModifiedDate])
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'. At the bottom, a green status bar says '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 (rchapler@...) | Query editor (preview)' window. The 'Query 1' tab contains the following T-SQL code:

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

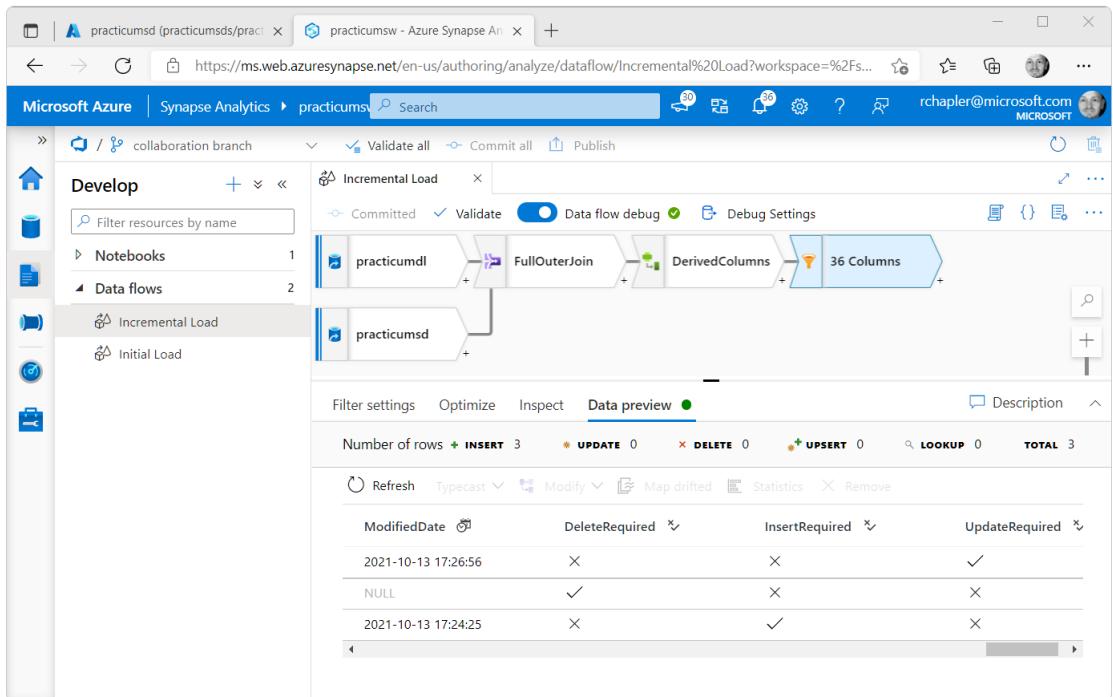
The 'Messages' tab shows the result: "Query succeeded: Affected rows: 1".

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.

The screenshot shows the Microsoft Azure Synapse Analytics Data Flow blade. On the left, the navigation pane shows 'Develop' selected, with 'Data flows' expanded, showing 'Incremental Load' and 'Initial Load'. The main area displays a data flow named 'Incremental Load'. The flow starts with a 'FullOuterJoin' step, followed by a 'DerivedColumns' step, which then branches into two paths. The top path leads to a '36 Columns' output. The bottom path leads to a 'Row modifier' step, which further branches into 'alter row' and 'Alter Row' options. The 'Data preview' tab is selected, showing a table with the following data:

	ModifiedDate	DeleteRequired	InsertRequired	UpdateRequired
1	2021-10-13 17:26:56	X	X	✓
2	NULL	✓	X	X
3	2021-10-13 17:24:25	X	✓	X

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

The screenshot shows the Microsoft Azure Synapse Analytics Data Flow blade with the 'Alter row settings' tab selected. The 'Output stream name' is set to 'AlterRow'. The 'Incoming stream' is set to 'Filter'. The 'Alter row conditions' section contains three rules:

- Delete if DeleteRequired
- Insert if InsertRequired
- Update if UpdateRequired

On the “Alter row settings” tab, enter values for the following items:

Output Stream Name Enter a meaningful name aligned with standards

Incoming Stream	Confirm default selection
Alter Row Conditions	<p>“Delete if” DeleteRequired</p> <p>“Insert if” InsertRequired</p> <p>“Update if” UpdateRequired</p>

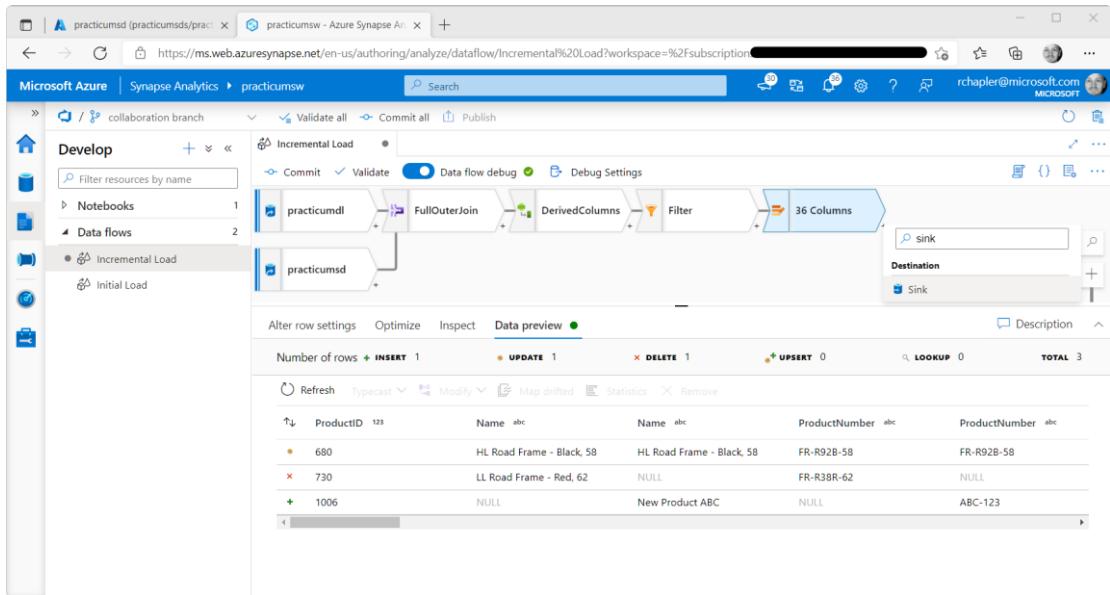
Navigate to the “Data preview” tab and then click Refresh.

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

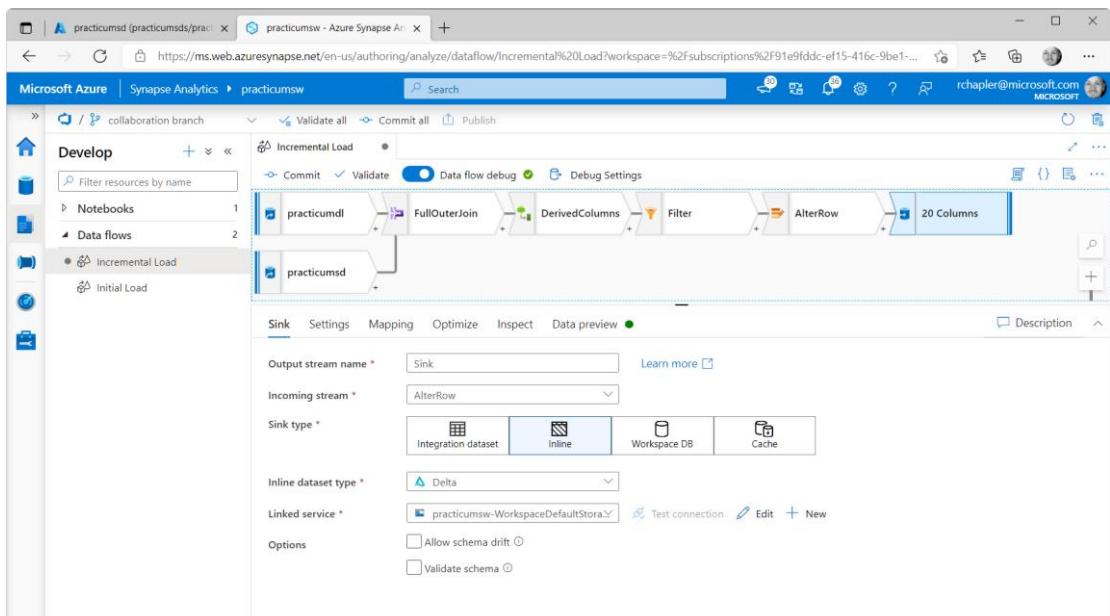
Click the **Commit** button.

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 popup list.



On the **Sink** tab, enter values for the following items:

Output Stream Name	Enter a meaningful name aligned with standards
---------------------------	--

Sink Type	Select Inline
------------------	----------------------

Inline Dataset Type	Select Delta
----------------------------	---------------------

Linked Service

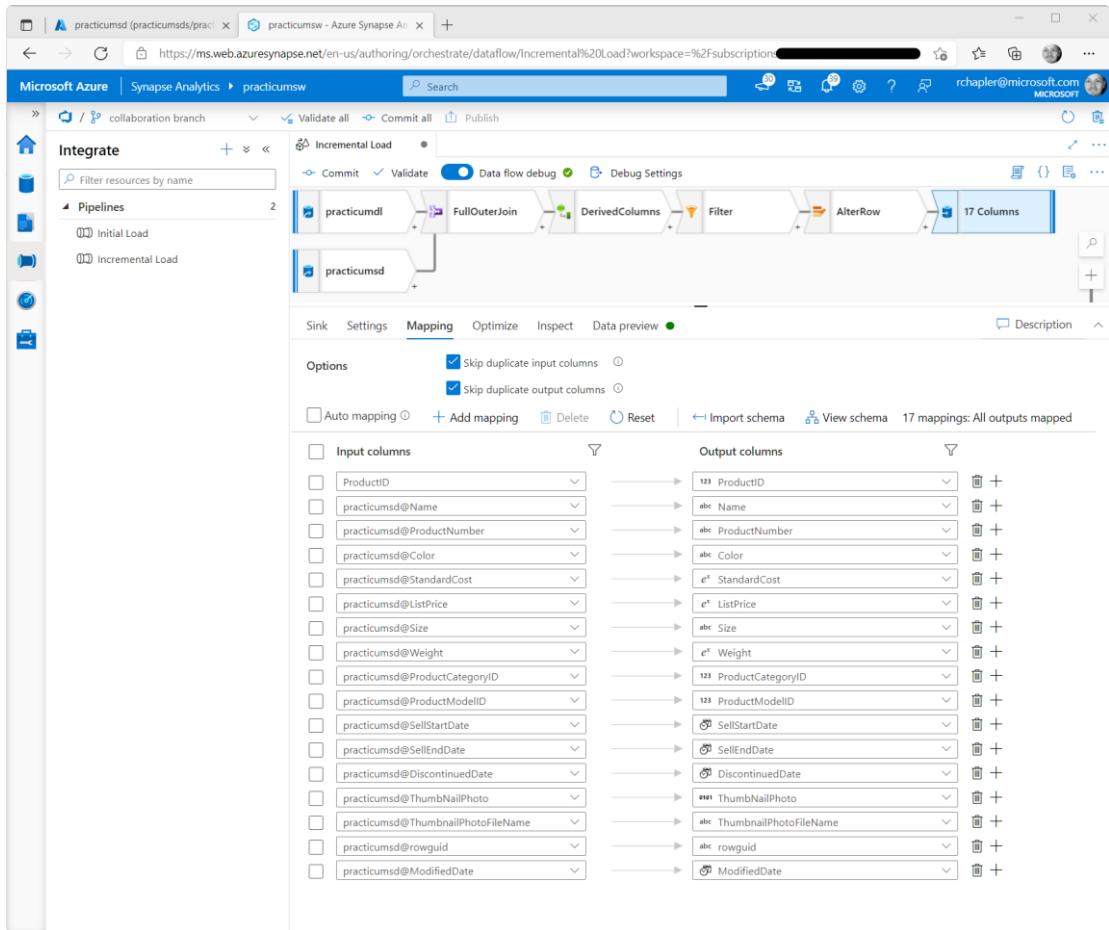
Select practicumsw-WorkspaceDefaultStorage

The screenshot shows the Microsoft Azure Synapse Analytics workspace. In the top navigation bar, there are tabs for 'practicumsd (practicumsds/pract)' and 'practicumsw - Azure Synapse Analytics'. The main area displays a data flow pipeline named 'practicumdl'. The pipeline consists of several stages: 'Initial Load' (practicumsd) followed by 'FullOuterJoin', 'DerivedColumns', 'Filter', 'AlterRow', and finally '20 Columns'. Below the pipeline, the 'Sink' tab is selected. The 'Settings' tab is active, showing the following configuration:

- Folder path:** practicumdl / bronze/northwind/prc
- Compression type:** snappy
- Compression level:** Fastest
- Table action:** None
- Update method:** Allow insert, Allow delete, Allow upsert, Allow update
- Key columns:** List of columns (ProductID)
- Delta options:** Merge schema (unchecked), Auto compact (checked), Optimize write (checked)

On the **Settings** tab, enter values for the following items:

File System	Enter shareddlc
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

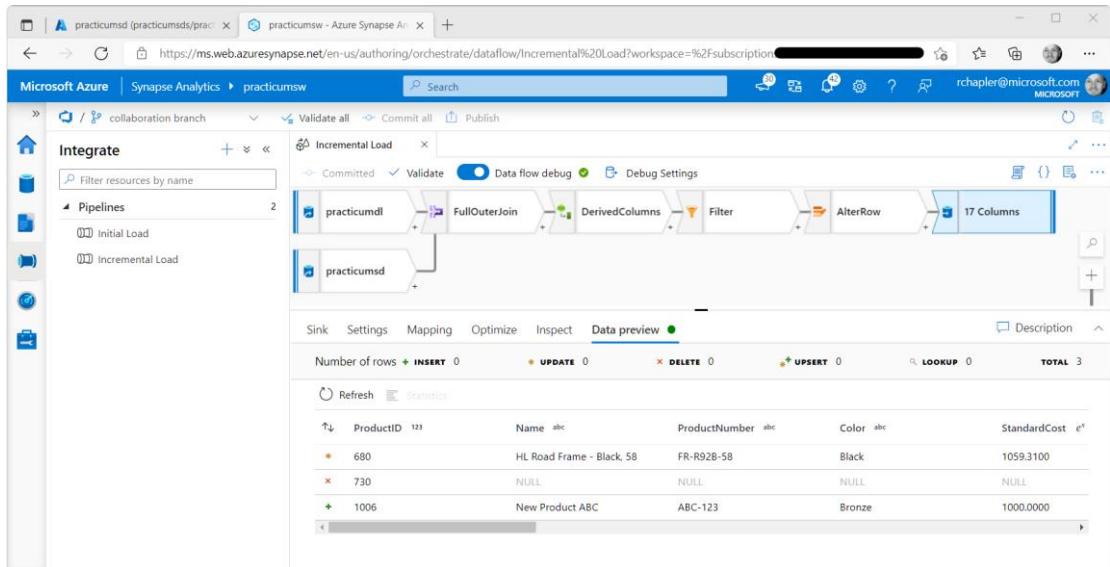


On the **Mapping** tab, un-check “Auto mapping”, click “**Import schema**” and then **Import** on the resulting popover.

Review the auto-generated list of columns and remove:

- Rows where “**Input columns**” begin with “**shareddl...**” (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.

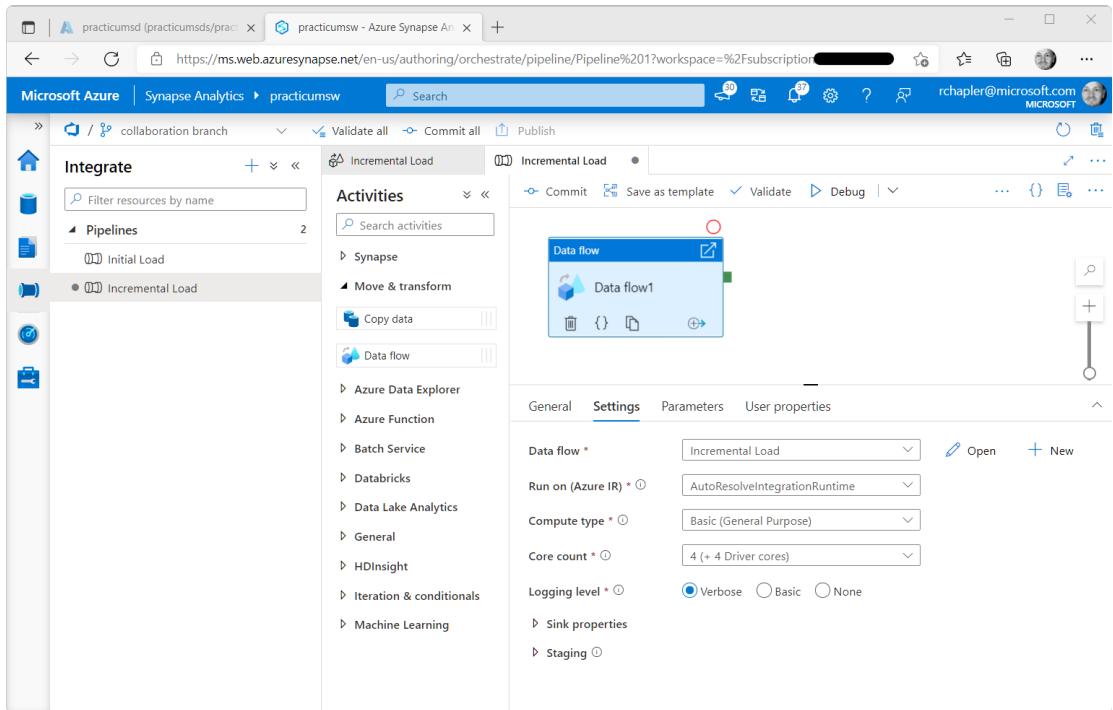
Create Pipeline

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 dropdown.

On the **Properties** popout, 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.

No additional changes are required. Review settings on the remaining tabs.

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. 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. The pipeline details pane shows a single Data flow named 'Data flow1'. The pipeline run table lists one run with the following details:

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 **practicumsw** to **shareddlc**.

Click to navigate to “**shareddlc > bronze > northwind > product**”

The screenshot shows the Microsoft Azure Synapse Analytics workspace. In the left navigation pane, under the 'Data' section, the 'Linked' tab is selected. The main area shows a file list for the 'product' folder under 'shareddlc > bronze > northwind'. The table lists the following files:

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.

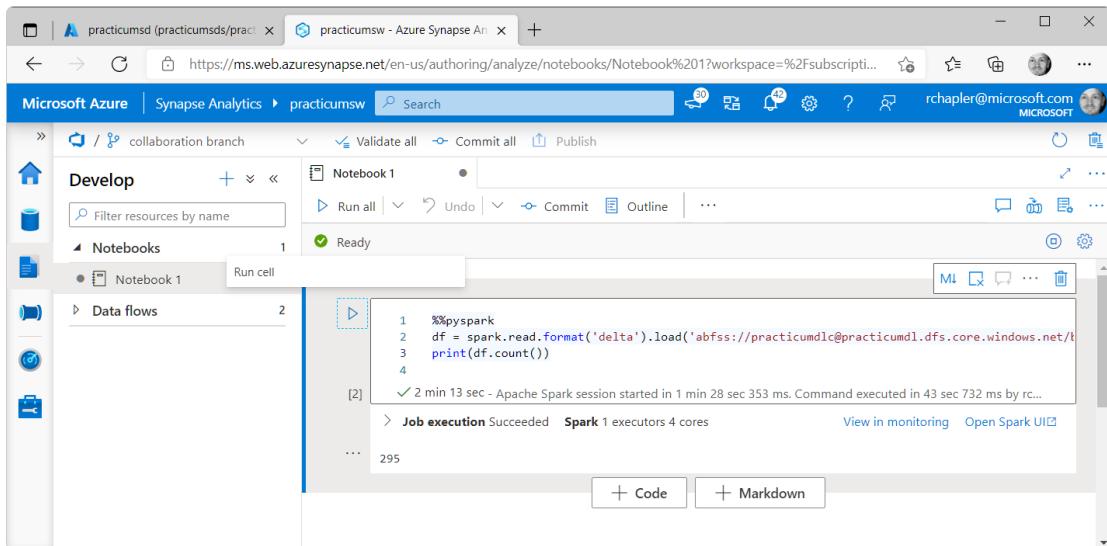
Query Record Count

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 dropdown.

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



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

```
%pyspark
df =
spark.read.format('delta').load('abfss://shareddlc@shareddl.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. In the left sidebar under 'Develop', 'Notebooks' is selected, showing 'Notebook 1'. The main area displays a code cell with the following content:

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

Below the code cell, a message indicates a successful execution: "11 sec - Command executed in 11 sec 23 ms by rchabler on 2:01:43 PM, 10/13/21". It also shows "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://shareddlc@shareddl.dfs.core.windows.net/bronze/northwind/product')
display(df.where("ProductID == 730"))
```

Confirm Insert Example

The screenshot shows the Microsoft Azure Synapse Analytics interface. In the left sidebar under 'Develop', 'Notebooks' is selected, showing 'Notebook 1'. The main area displays a code cell with the following content:

```
1 %%pyspark
2 df = spark.read.format('delta').load('abfss://practicumdlc@practicumdl.dfs.core.windows.net/bronze/northwind/product')
3 display(df.where("ProductID == 1006"))
4
```

Below the code cell, a message indicates a successful execution: "10 sec - Command executed in 10 sec 513 ms by rchabler on 1:59:18 PM, 10/13/21". It shows "Job execution Succeeded" with "Spark 1 executors 4 cores". There are buttons for "View in monitoring" and "Open Spark UI". Below the code cell, there are buttons for "View", "Table" (which is selected), "Chart", and "Export results". A table is displayed with the following data:

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://shareddlc@shareddl.dfs.core.windows.net/bronze/northwind/product')  
display(df.where("ProductID == 1006"))
```

Confirm Update Example

The screenshot shows the Microsoft Azure Synapse Analytics interface. On the left, the navigation pane is visible with sections like 'Develop', 'Notebooks' (containing 'Notebook 1'), and 'Data flows'. The main area is titled 'Notebook 1' and contains the following PySpark code:

```
[3] 1 %%pyspark  
2 df = spark.read.format('delta').load('abfss://practicumdlc@practicumdl.dfs.core.windows.net/northwind/product')  
3 display(df.where("ProductID == 680"))  
4
```

Below the code, a green checkmark indicates the command was executed successfully in 11 seconds by 'rchapler' at 1:54:40 PM on 10/13/21. The 'Job execution' status is 'Succeeded' with 'Spark 1 executors 4 cores'. 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://shareddlc@shareddl.dfs.core.windows.net/bronze/northwind/product')  
display(df.where("ProductID == 680"))
```

Objective Complete... congratulations!

Mount Data Lake

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

Step 1: Instantiate Prerequisites

This exercise requires the following resource(s):

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

Step 2: Prepare Logic

Navigate to Databricks and the **practicumdbn** 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="practicumdbss", key="practicumar-clientid" ),
           "fs.azure.account.oauth2.client.secret": dbutils.secrets.get( scope="practicumdbss", key="practicumar-clientsecret" ),
           "fs.azure.account.oauth2.client.endpoint": "https://login.microsoftonline.com/" +
           dbutils.secrets.get( scope="practicumdbss", key="practicumar-tenantid" ) + "/oauth2/token"}

adlsAccount = "shareddl"
adlsContainer = "shareddlc"
adlsFolder = ""
mountPoint = "/mnt/practicum"

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 )
```

Some callouts...

- **practicumdbss** ... 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”
- **shareddl** ... refers to your Data Lake
- **shareddlc** ... 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 left sidebar contains various icons for file operations like copy, move, delete, and search. The main area has a toolbar with "practicumdbc" selected, and a status bar at the bottom right shows "rchapler@microsoft.com". A code cell labeled "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 )
15
16 (1) Spark Jobs
```

Below the code cell, a message indicates the command took 26.98 seconds. A note at the bottom 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/practicum/myDatabase"
```

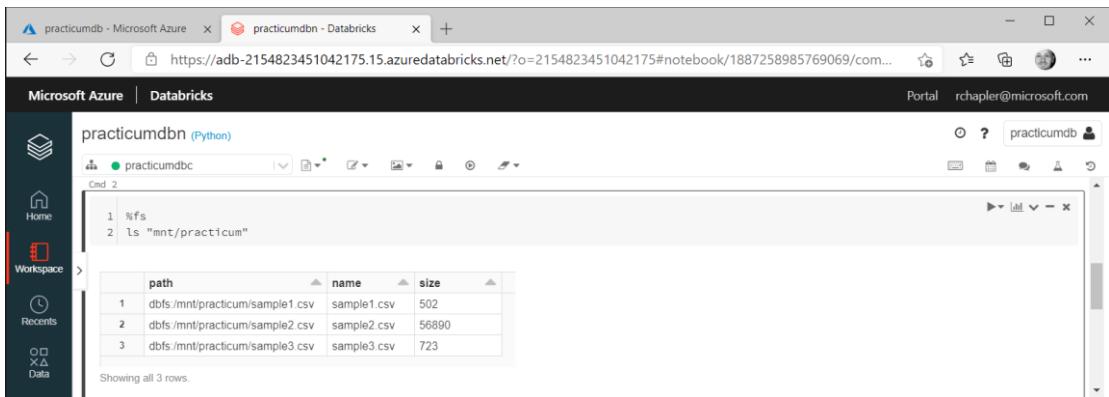
Step 3: Confirm Success

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/practicum"
```



The screenshot shows a Microsoft Azure Databricks notebook interface. On the left, there's a sidebar with icons for Home, Workspace (which is selected), Recents, and Data. The main area has a title bar "practicumdb - Microsoft Azure" and "practicumdb - Databricks". Below the title bar, it says "practicumdb (Python)" and "Cmd 2". The command cell contains the following Python code:

```
1 %fs
2 ls "/mnt/practicum"
```

Below the code, the output is a table showing three CSV files in the mount point:

	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

Showing all 3 rows.

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/practicum/sample1.csv")
df.printSchema()
df.show()
```

The screenshot shows a Microsoft Azure Databricks notebook interface. On the left is a sidebar with icons for Home, Workspace, Recents, Data, Clusters, Jobs, and Models. The main area has tabs for 'practicumdb - Microsoft Azure' and 'practicumdb - Databricks'. The 'practicumdb - Databricks' tab is active, showing a notebook titled 'practicumdb (Python)'. The code in the notebook is:

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

Output:

```
(2) Spark Jobs
> df  pyspark.sql.dataframe.DataFrame = [c0: string, c1: string ... 11 more fields]
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 |
```

Command took 5.57 seconds -- by rchaphler@microsoft.com at 1/26/2021, 9:48:04 AM on practicumdb

Shift+Enter to run [shortcuts](#)

Some 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.

Objective Complete... congratulations!

Localize Mounted Data

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

Step 1: Instantiate Prerequisites

This exercise requires the following resource(s):

- Data Lake
- Databricks

Step 2: 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 **practicumdbn** notebook. Add a new cell and paste the following code:

```
import uuid
myUUID = 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 +
"/'" )
```

practicumdb - Microsoft Azure | prakticumdb - Databricks

practicumdb (Python)

```

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 practicumdbc

Shift+Enter to run shortcuts

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

practicumdb - Microsoft Azure | mydeltatable_fromadls - Databricks

Data

Databases default

Tables mydeltatable_fromadls

Table: mydeltatable_fromadls

mydeltatable_fromadls | Refresh

practicumdbc

Details History

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
Month	"Average"	"2005"	"2006"	"2007"	"2008"
May	0.1	0	0	1	1
Jun	0.5	2	1	1	0
..

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

Objective Complete... congratulations!

Sourcing from APIs

Follow these instructions to **source data from a REST API**.

Step 1: Instantiate Prerequisites

This exercise requires the following resource(s):

- Databricks
- Key Vault

Step 2: 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, there's a search bar labeled "Weather in your city". The main content area has three main sections:

- Current Weather Data**: Includes a "Subscribe" button and a list of bullet points:
 - 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
- Hourly Forecast 4 days**: Includes a "Subscribe" button and a list of bullet points:
 - 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
- One Call API**: Includes a "Subscribe" button and a list of bullet points:
 - 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

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/price>.

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 content is a JSON object representing current weather data for Redmond, Washington:

```
{"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}
```

Step 3: Add Secret to Key Vault

Add your API Key to Key Vault.

The screenshot shows the Microsoft Azure portal interface for creating a secret. 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 has the following fields:

- Upload options:** Manual
- Name ***: myOpenWeatherAPIKey
- Value ***: (redacted)
- Content type (optional):** (empty)
- Set activation date?**: (unchecked)
- Set expiration date?**: (unchecked)
- Enabled?**: Yes (selected)

At the bottom left is a blue 'Create' button.

Step 4: Prepare Logic

Navigate to Databricks and the **practicumdbn** 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="practicumdbss", key="myOpenWeatherAPIKey" ) )

df = spark.read.option( 'multiline', "true" ).json( sc.parallelize( [response.text] ) )
```

Run the cell.

The screenshot shows the Microsoft Azure Databricks workspace. On the left, there's a sidebar with icons for Home, Workspace (which is selected), Recent, Data, Clusters, Jobs, Models, and Search. The main area is titled 'practicumdbn (Python)' and contains a code editor with the following Python script:

```
1 import requests
2 response = requests.get( 'http://api.openweathermap.org/data/2.5/weather?q=redmond&appid=' + dbutils.secrets.get(
3   scope="practicumdbss", key="myOpenWeatherAPIKey" ) )
4 df = spark.read.option( 'multiline', "true" ).json( sc.parallelize( [response.text] ) )

▶ (1) Spark Jobs
  ▶ df: pyspark.sql.dataframe.DataFrame
    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()
```

Objective Complete... congratulations!

Batch Upsert

Follow these instructions to provide for recurring upsert of data.

Step 1: Instantiate Prerequisites

This exercise requires the following resource(s):

- Data Lake
- Databricks

Step 2: CREATE TABLE

Navigate to Databricks and the **practicumdbn** 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 interface. On the left is a sidebar with icons for Home, Workspace (selected), Recents, Data, and Clusters. The main area has a title bar 'practicumdbn - Microsoft Azure' and 'practicumdbn - Databricks'. Below the title bar is a navigation bar with 'Portal' and an email address 'rchapler@microsoft.com'. The main workspace contains a Python notebook titled 'practicumdbn (Python)'. A code editor window shows 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 + "/"")
```

Below the code editor, a message indicates '(4) Spark Jobs' and 'Out[4]: DataFrame[]'. At the bottom, a note says 'Command took 15.87 seconds -- by rchapler@microsoft.com at 2/1/2021, 12:50:40 PM on practicumdbc'. A status bar at the bottom of the interface says 'Shift+Enter to run' and 'shortcuts'.

Step 2: MERGE INTO

Navigate to Databricks and the **practicumdbn** 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="practicumdbss", 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 a Microsoft Azure Databricks notebook interface. The left sidebar contains navigation links: Home, Workspace (selected), Recents, Data, Clusters, Jobs, Models, and Search. The main area displays a Python notebook titled "practicumdbn (Python)". The code cell contains the following Python and SQL code:

```
import requests
import uuid
response = requests.get('http://api.openweathermap.org/data/2.5/weather?q=redmond&appid=' + dbutils.secrets.get(scope="practicumdbss", 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)
```

The output of the cell shows the generated SQL command:

```
▶ (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[]
```

Below the output, a note indicates the command took 16.50 seconds.

Step 3: Confirm Success

Navigate to Databricks and the **practicumdbn** 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. On the left, there's a sidebar with icons for Home, Workspace (selected), Recents, Data, Clusters, Jobs, Models, and Search. The main area has two tabs: "practicumdbn - Microsoft Azure" and "practicumdbn - Databricks". The "practicumdbn - Databricks" tab is active, displaying a Python notebook cell and a command line interface (CLI) cell.

Python Notebook Cell:

```
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

CLI Cell:

```
1 %sql
2 SELECT * FROM myOpenWeatherData
```

▶ (3) Spark Jobs

	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}

Showing all 2 rows.

Command took 1.18 seconds -- by rchapler@microsoft.com at 2/1/2021, 1:22:13 PM on practicumdbc

Shift+Enter to run [shortcuts](#)

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.

Objective Complete... congratulations!

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**

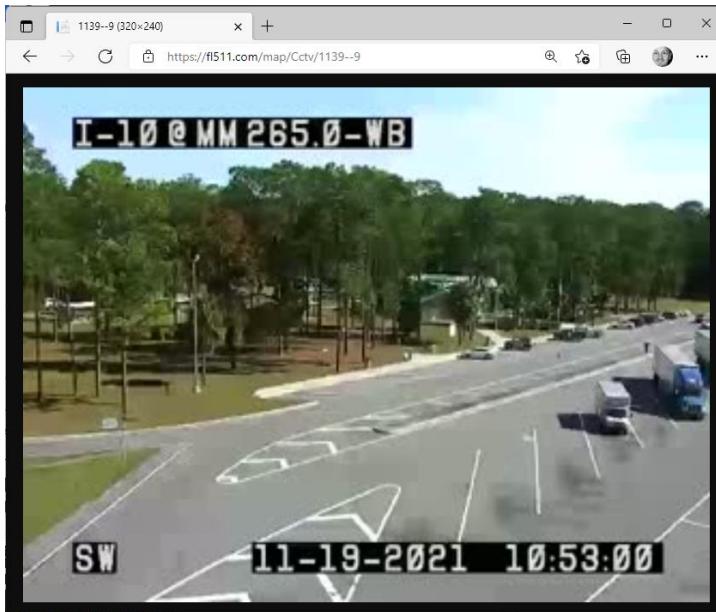
Step 1: Instantiate Prerequisites

This exercise requires the following resource(s):

- 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

Step 2: Stage Sample Data

In this section, we will load three sample images from an online traffic cam source ([1139--9 \(320x240\) \(fl511.com\)](#)) onto our Linux virtual machine.



Navigate to Azure Cloud Shell.

Step 1: 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}
```

Step 2: Create Images Directory

Execute the following command to create a directory for our sample images:

```
mkdir images
```

Step 3: 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
```

Step 3: 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://practicumsa.blob.core.windows.net/images?sp=rw&st=2021-11-23T20:23:01Z&se=2022-10-
01T03:23:01Z&spr=https&sv=2020-08-04&sr=c&sig=U3o98QJ1NJ8wAldkjBBjv8Eken4N3Rlrr50gLxsq6PE%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 Azure Storage Container blade for the 'images' container. The left sidebar lists navigation options: Overview, Diagnose and solve problems, Access Control (IAM), Settings, Shared access tokens, Access policy, Properties, Metadata, and Editor (preview). The main area displays the container's properties: Authentication method (Access key) and Location (images). A search bar allows filtering blobs by prefix. Below is a table listing the 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

Step 4: Prepare Script

In this section, we will prepare a script and confirm success.

Create Script with Vim

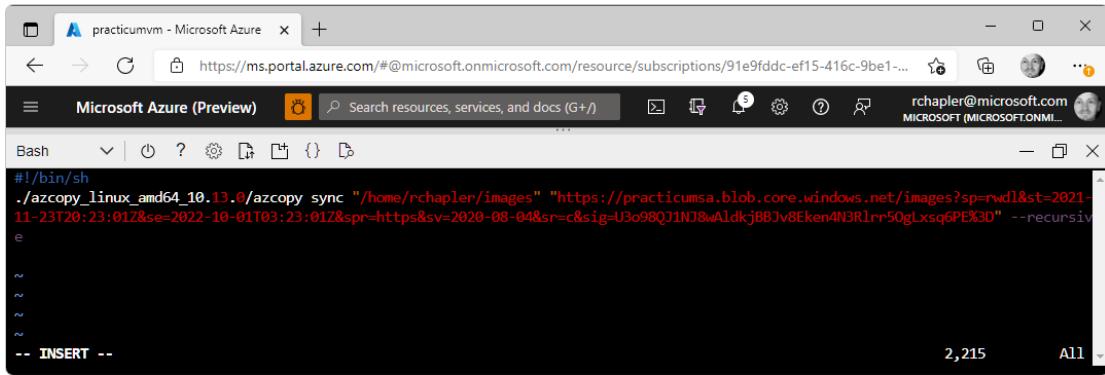
Execute the following command to open a text editor:

```
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://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
```



```
#!/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
~
~
~
~
-- INSERT --
```

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

Step 5: 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
```

Logic explained...

- **Schedule** ... “* * * * *” (in this case, “every minute”)
- **Command** ... “/home/rchapler/script.sh”

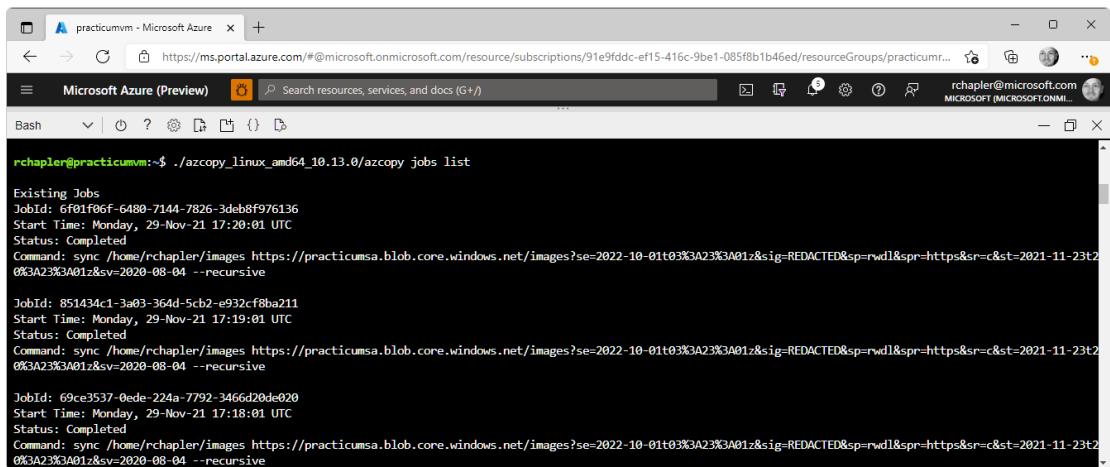
Press CTRL-X on the keyboard to exit crontab; save when prompted.

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 commands include parameters like `--recursive` and `--sync`.

```
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=2021-11-23t20%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=2021-11-23t20%3A23%3A01z&sv=2020-08-04 --recursive

JobId: 69ce3537-0ede-224a-7792-3466d20de020
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=2021-11-23t20%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}.

Step 6: 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
```

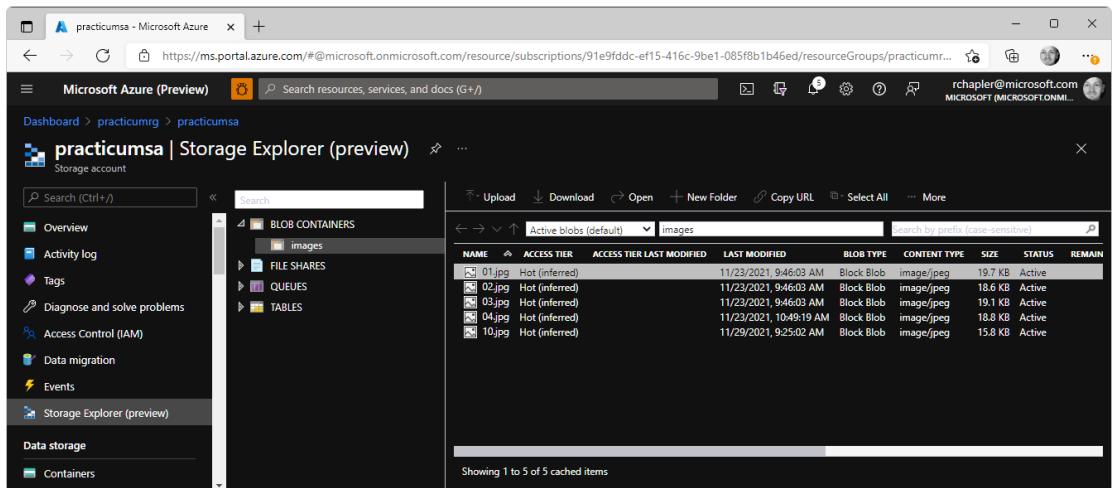
You can expect a response like:

```
--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, and Storage Explorer (preview). The main area shows a tree view of storage resources under 'practicumsa'. Under 'BLOB CONTAINERS', there's a single folder named 'images'. Inside 'images', there are five files: '01.jpg', '02.jpg', '03.jpg', '04.jpg', and '10.jpg'. A table below the list provides detailed information for each file, including Name, Access Tier, Last Modified, Blob Type, Content Type, Size, Status, and Remain. The table shows the following data:

NAME	ACCESS TIER	LAST MODIFIED	BLOB TYPE	CONTENT TYPE	SIZE	STATUS	REMAIN
01.jpg	Hot (inferred)	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	Block Blob	image/jpeg	18.6 KB	Active	
03.jpg	Hot (inferred)	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	Block Blob	image/jpeg	18.8 KB	Active	
10.jpg	Hot (inferred)	11/29/2021, 9:25:02 AM	Block Blob	image/jpeg	15.8 KB	Active	

You should see the new file {i.e., “10.jpg”} included in the **images** container.

Objective Complete... congratulations!

On-Prem to On-Prem

Follow these instructions to migrate files using Azure File Sync and a storage account.

Step 1: Understand 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}

Step 2: Instantiate Prerequisites

This exercise requires the following resource(s):

- 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 [Deploy Infrastructure-as-Code \(feat. DevOps + ARM Templates\)](#) instructions 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: practicum  
        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: practicum  
        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: practicumvm01'
    inputs:
      ConnectedServiceName: a775d3ec-e4f1-46ed-905b-ddd26a59356c
      subscriptionName: 91e9fddc-ef15-416c-9be1-085f8b1b46ed
      resourceGroupName: practicumvm01
      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: practicumvm02'
    inputs:
      ConnectedServiceName: a775d3ec-e4f1-46ed-905b-ddd26a59356c
      subscriptionName: 91e9fddc-ef15-416c-9be1-085f8b1b46ed
      resourceGroupName: practicumvm02
      location: West US
      csmFile: ARM Templates/VirtualMachine.json

```

Pipelines - Run 20211222.3

https://dev.azure.com/practicumadop/practicumadop/_build/results?buildId=348&view=results

practicumadop

- Overview
- Boards
- Repos
- Pipelines**
- Environments
- Releases
- Library
- Task groups
- Deployment groups
- Test Plans
- Artifacts
- Project settings

#20211222.3 Update Objective - Synchronize Unstructured Data, On-Prem to On-Prem

Manually run by Rich Chapler

View 2 changes

Repository and version	Time started and elapsed	Related	Tests and coverage
infrastructure main ↗ 067bd41e	Today at 5:32 AM 6m 53s	0 work items 0 artifacts	Get started

Jobs

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

Step 3: Install Agent

Navigate to the first virtual machine, click the **Connect** button, and select **RDP** from the resulting dropdown.

On the resulting “**Connect with RDP**” page, click the “**Download RDP File**” button.

Click the “**Open file**” link under your RDP file in the **Downloads** dropdown.

Click the **Connect** button on the resulting “**Remote Desktop Connection**” popup.

In the “**Windows Security**” popup, 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**” popup, enter credentials included in the YAML template.

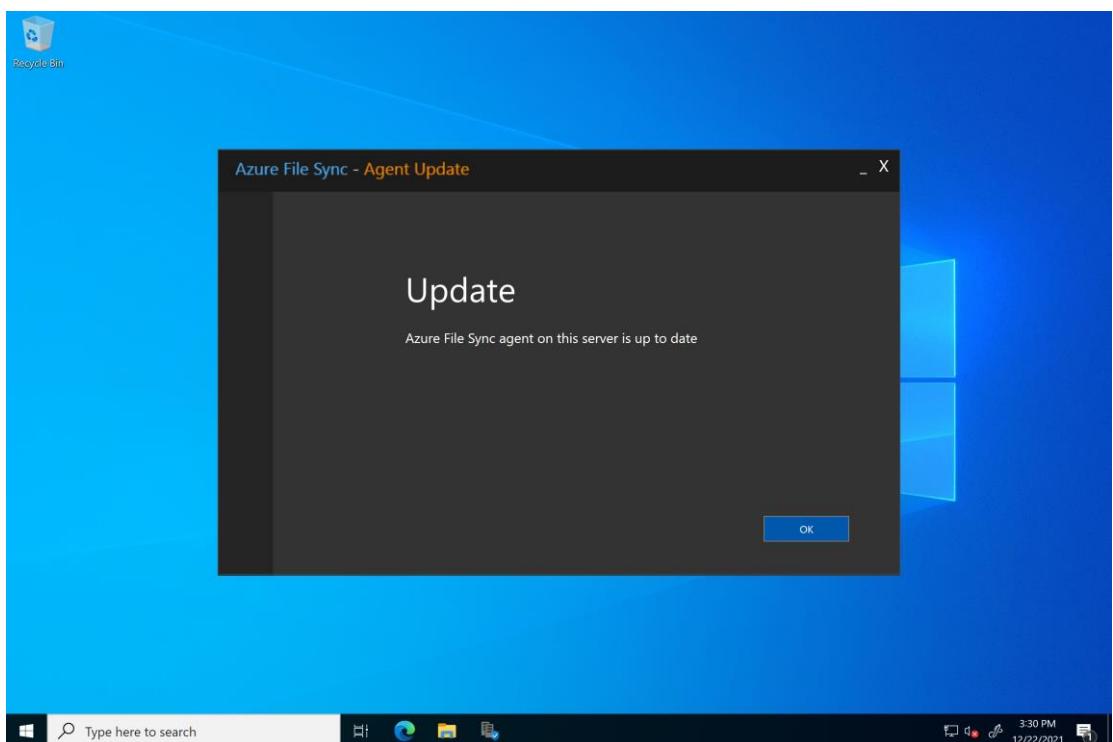
Note: “*User name*” should include the name of the server {e.g., “practicumvm01\Admin123”}.

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.,

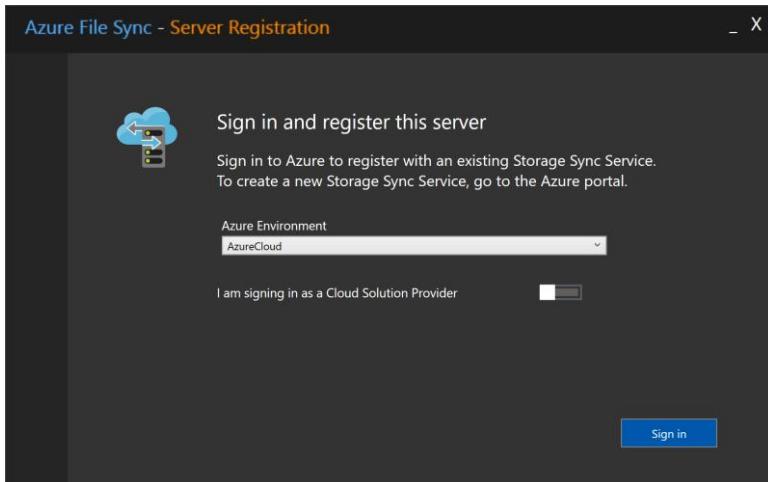
StorageSyncAgent_WS2022.msi}} and click the **Next** button.

Open the downloaded file and complete the installation.



The Azure File Sync, server registration utility will open.

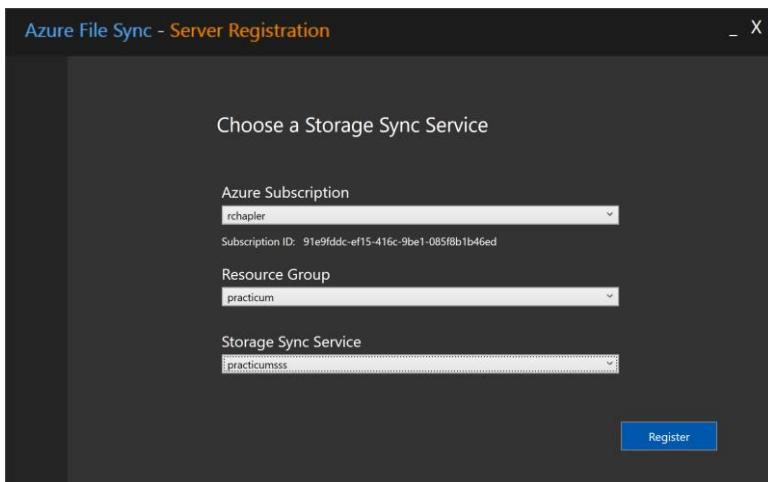
Click the **OK** button on the Update page.



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.



On the “**Choose a Storage Sync Service**” page, enter values for the following items:

Azure Subscription Select your subscription

Resource Group Select your resource group

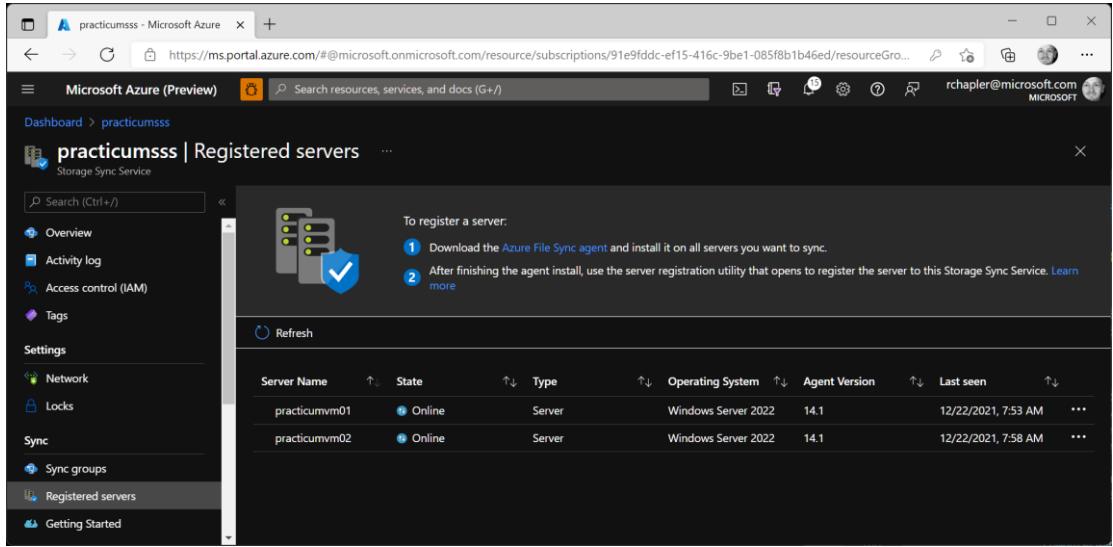
Storage Sync Service Select your storage sync service

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 URL is https://ms.portal.azure.com/#@microsoft.onmicrosoft.com/resource/subscriptions/91e9ffdc-ef15-416c-9be1-085f8b1b46ed/resourceGro... . The page title is "practicumsss | Registered servers". On the left, there's a navigation bar with "Overview", "Activity log", "Access control (IAM)", "Tags", "Settings" (Network, Locks), "Sync" (Sync groups, Registered servers, Getting Started), and a search bar. The main content area shows a "To register a server:" section with two steps: 1. Download the Azure File Sync agent and install it on all servers you want to sync. 2. After finishing the agent install, use the server registration utility that opens to register the server to this Storage Sync Service. Below this is a "Refresh" button and 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	...

Navigate to Storage Sync Service, then “**Registered Servers**” in the **Sync** group of the navigation bar.

You should see your two registered servers.

Step 4: Create File Share

Open your Storage Account, then “**File Shares**” in the “**Data storage**” group of the navigation bar.

Click the “**+ File share**” button and enter a meaningful **Name** value in the resulting popout.

Click the **Create** button.

Step 5: Create Sync Group

Navigate to Storage Sync Service, then “**Registered Servers**” in the **Sync** group of the navigation bar.

Click the “**+ Sync group**” button.

On the “**Choose a Storage Sync Service**” page, enter values for the following items:

Sync Group Name	Enter a meaningful name aligned with standards {e.g., practicumssssg}
Subscription	Select your subscription
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.

Step 6: 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**” popout, enter values for the following items:

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.

Step 7: 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 dropdowns.

Take note of the metadata {e.g., **Created** datetime}.

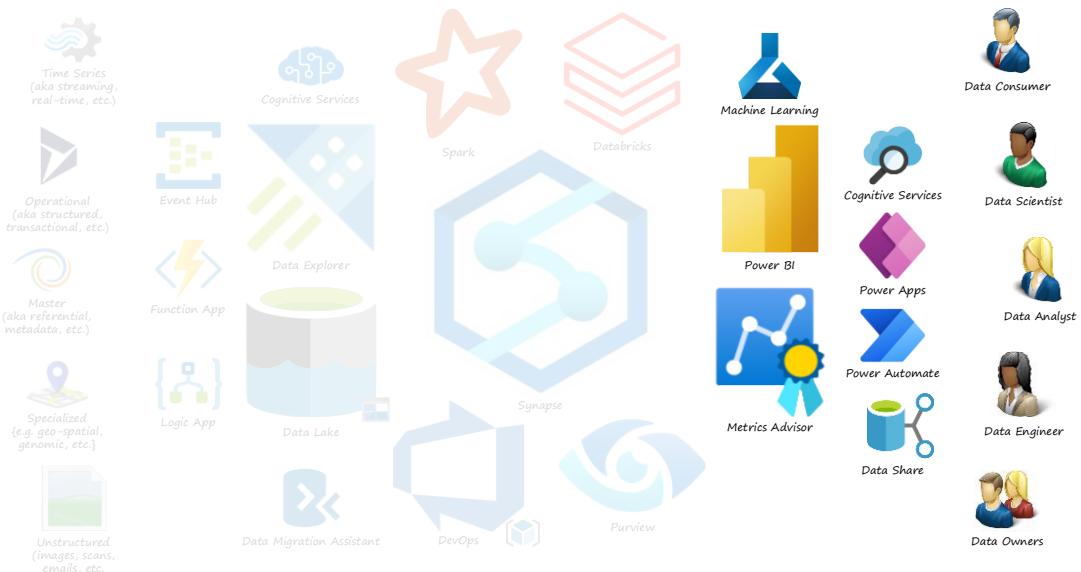
Connect to your second virtual machine and then open File Explorer.

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.

Objective Complete... congratulations!

Surface



Query from On-Prem

Follow these instructions to **query from an on-prem client**.

Step 1: Instantiate Prerequisites

This exercise requires the following resource(s):

- Azure CLI
- Data Explorer
- Postman

Step 2: 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://practicumdec.westus2.kusto.windows.net --query accessToken --output tsv`

Capture the returned token for use later in this section.

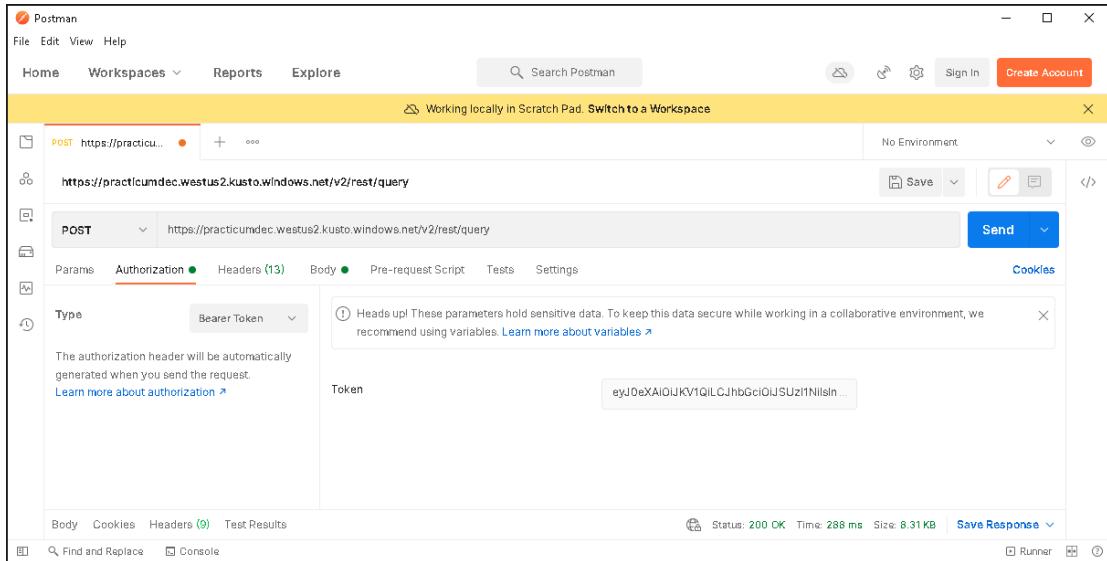
Command Prompt

```
C:\Users\rchapler>az account get-access-token --resource https://practicumdec.westus2.kusto.windows.net --query accessToken --output tsv
eyJ0eXAiOiJKV101CJhbGciOiJSUzI1NiisInIg1dC16Im5PbzNaRHJPRFhfSzFqS1d0MHNsFJfS1hFZyIsImtpZC16Im5PbzNaRHJPRFhfSzFqS1d0MHNsFJfS1hFZyJ9.eJhdWQ
01JodhRwczoVl3byYWNOawN1bwRYy53ZXN0dXMyLmt1c3RVLndpbmRvd3MubmV0IwiAxNztjoiashRoChIMGLy9zdHMud21uZG93cy5uZXQvNzJmOTg4YmYtODZmMS00MfmLTkxYWI
mQ3Y2QwMTFkYjQ3LyIiIm1hdCT6MTx0dk1NjI1NCwibmJmIjoxNjE40TU2MjU0lC31ehA0jE2Mtg5nJAxNTQsImFjciT61je1lCjhaW81O1JBVVF8s84VEFBQUF5Y1NLemtCmVRNm
Frcmh1ly8xU3lxmXvUty0EuwVvh51gwX3cGhwsVNSfhuaxYrc25ROEP6YTNEwlpTUM5QOU1JRmdteUpIMFBWw1M3Q1zydz091iw1yjpbInj2YSiS1m1myS1d1CjHcBpZCI
61JA0yJA3NzklLthZG1tNDYxS11YmV1LTAyZj1MwjmN210N1sImfwcG1kyWny1jo1MCtsImR1dm1jZMlk1joiMjc3MzggZMWEtntQNS00N2NHLTh10D1tDg5NW1LMGN1MWV11iwi
ZmftaWx5x25hbWU101jDaGrFwbvYi1w1Z21Z2W5fbmftZS161j1pY2g1lCjpcGfkZH101i13N14xhjEuMTk0ljeZM1s1m5hbwU101jsaWno1EnovXbsZX11LCjvaQ101jyMuyOTQ4
y01NzaxLTOyNjUt0DExC02MznHze0NjczZjA1lCJvbnyZM1fc21kIjoiUy0xLTUthjtEtMjEyNzUyMTE4NC0xNjAOADEv0tiwLTe40Dc5MjciMjctNDY2nzUyOCisInBiaQ101ixMD
AzN0ZGR7gvMUFDDYxiw1cmg101TwLkf5b0f2Ng01YzHR31Wr1xeTf4MEJY1I1VJNzQVRial1JwR3U7NEMcZldfzTBZYUFEVS41lCjzy3A101j1c2Vyx21tcGVyc29uXRpbz41lCj
zdw11013sRk32Q9xxdgF1dHb0j01c3VyTktoX3lqam84aXo452ZaM024dHdnV3ZNIiwidg1kijoiNz3m0t4YmYtODZm500MfmLThxYIItMnQ3Y20wMTFkYj031iwiw5pcXV1x25h
bWU101jyY2hcg1ck8taWnyb3NvZnQu29t11w1dX0uijoiChmNoYXBsZXJabw1lcm9z220lmNvbSiInV0s161ktqdV1hbdYQ1VrncndHaJNBND3QUE1LCj22X1101i1xLA1fQ.U6
yTE35Adt0hJgUEjjUPFn6ra90Rf07szFya4aabPzrwKntGbrekyjLzaqRSC-c1zPw9c8dp5Gn-g1Hw0cpXnZbrA-Mv9-g4Gws to311lxqXkh3BmhGhqDvQ3j1D-N15U1whxj6f
jsgsgfwb1NgCo1gt0ndyEWhrfgv031PkmwNTfLu3-huqRQDxm1eu519Uf4cxMm00cxSaKegNs2eoco0yAUxCjg02cdjVIEgb1oL9ZdCCNJRb1AgHhJcxHVGafPBTyL03XL_Xngc-DpZdy
C6MMp1SF4-BEc7259N1i141JHNpGq-hy8DCB2_5bduHwXMGOTnf1wbq14PQ
```

C:\Users\rchapler>

Step 3: Post API Request

Open a new request in Postman.

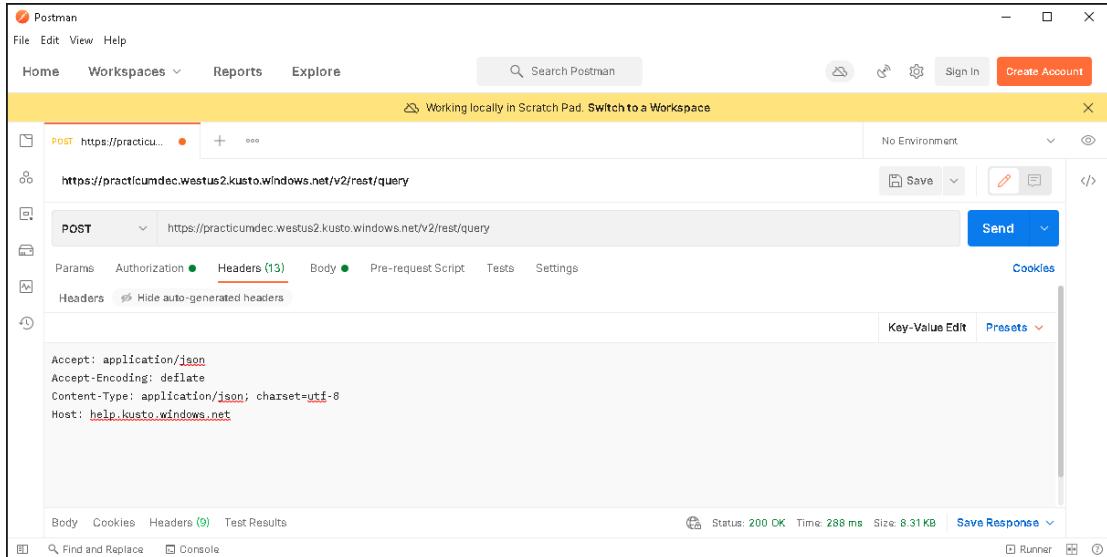


The screenshot shows the Postman application window. A POST request is being prepared to the URL `https://practicumdec.westus2.kusto.windows.net/v2/rest/query`. The **Authorization** tab is active, set to **Bearer Token**, with a token value displayed as a long string of characters. Other tabs like Headers, Body, and Cookies are visible but not selected. The status bar at the bottom indicates a successful `200 OK` response.

Select **POST** from the dropdown. Enter a hostname (with the name you used for your cluster); example:

`https://practicumdec.westus2.kusto.windows.net/v2/rest/query`

On the **Authorization** tab, select “**Bearer Token**” from the dropdown and paste your previously copied token in the resulting interface.



The screenshot shows the Postman application window with a POST request to `https://practicumdec.westus2.kusto.windows.net/v2/rest/query`. The **Headers** tab is active, containing the following custom headers:
`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 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 shows a POST request to 'https://practicumdec.westus2.kusto.windows.net/v2/rest/query'. The 'Body' tab is selected, showing a JSON payload:

```
1 {
2     "db": "practicumdec",
3     "csl": "StormEvents | take 5"
4 }
```

Below the body, the status bar shows 'Status: 200 OK' and other details like time and size.

On the **Body** tab, paste the following:

```
{
  "db": "sharedded",
  "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, displaying a JSON response with line numbers 1 through 24. The response 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 buttons for 'Find and Replace' and 'Console', and on the far right, a 'Runner' button.

You should see JSON with both schema and data included.

Objective Complete... congratulations!

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.

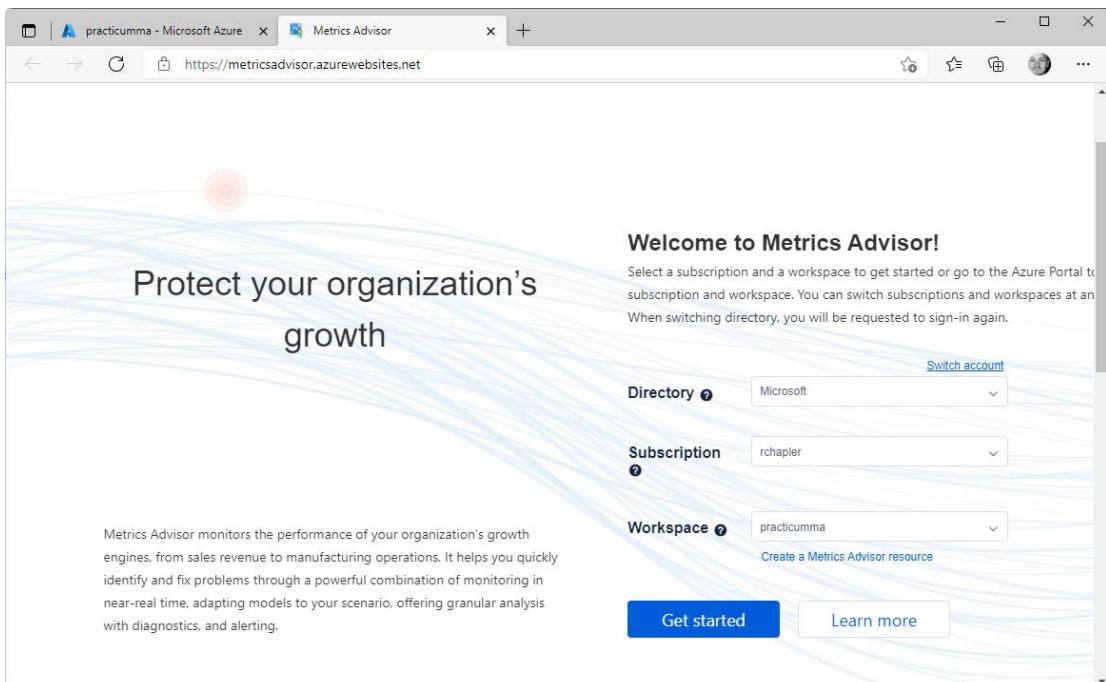
Step 1: Instantiate Prerequisites

This exercise requires the following resource(s):

- Data Explorer
- Metrics Advisor

Step 2: Onboard Data Feed

Navigate to Metrics Advisor and then click the “**Go to your workspace**” link.



On the “**Welcome to Metrics Advisor!**” page, enter values for the following items:

Directory	Select your directory
Subscription	Select your subscription
Workspace	Select Metrics Advisor

Click the “Get started” button.

The screenshot shows the Microsoft Azure Metrics Advisor Data feeds Onboarding page. The left sidebar has a blue background with white text, listing options like Onboarding, Add data feed, Monitor & Diagnostic, Data feeds (which is selected and highlighted in dark blue), Incident hub, Metrics graph, Settings, API keys, Hooks, Credential entity, Documentation, and Public community. The main content area has a white background with a header "Data feeds" and filters for "Filter by name", "Filter source type", "Filter granularity", "Filter status", "Search by owner", "Show 'data feed not available' alerts", and "Start a tour". Below the header are three steps: Step 1 (Onboard time-series data) with an icon of a stack of databases and a gear, Step 2 (Tune configuration & subscribe anomaly alerts) with an icon of a chart and a gear, and Step 3 (Diagnose incidents) with an icon of a document and a magnifying glass. Each step has a list of tasks. At the bottom is a blue button labeled "Onboard my first data feed".

practicumma - Microsoft Azure Metrics Advisor https://metricsadvisor.azurewebsites.net/data-feed

practicumma Data feeds

Onboarding

Add data feed

Monitor & Diagnostic

Data feeds

Incident hub

Metrics graph

Settings

API keys

Hooks

Credential entity

Documentation

Public community

Collage menu

Filter by name

Filter source type

Filter granularity

Filter status

Search by owner

Show "data feed not available" alerts

Start a tour

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.

On the “Add data feed” page, enter values for the following items:

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:

```
Data Source=https://{{ADX Cluster}}.{{region}}.kusto.windows.net;Initial Catalog={{ADX Database}}
```

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.

practicummadf

Congratulations, you have onboarded the data feed. Historical data and initial model building is in progress. Depending on data volume, it will take several minutes or hours to complete. You can leave this page now and let ingestion run in background.

You can also adjust ingestion parameters based on the conditions of your database. Check [Ingestion Best Practices](#) for more information.

[Visit data feed: practicummadf](#) [Check all data feeds](#) [Refresh progress](#)

Ingestion: Running 0, Succeeded 0, Error(s) 0, 0 To be retried. [details](#)

Enrichment initialization
Count of Events: In progress...

Data feed: practicummadf

You will receive a “**Congratulations...**” message.

Click the “**Visit data feed...**” button.

practicummadf

Ingestion Progress: Running 0, Succeeded 50, Error(s) 0, 0 To be retried. [Edit](#) [Backfill](#) [Refresh Progress](#) [Delete](#) [Export](#) [Start a tour](#)

Metrics: It might take 5-8 minutes to see all the metrics here after the data feed is created.

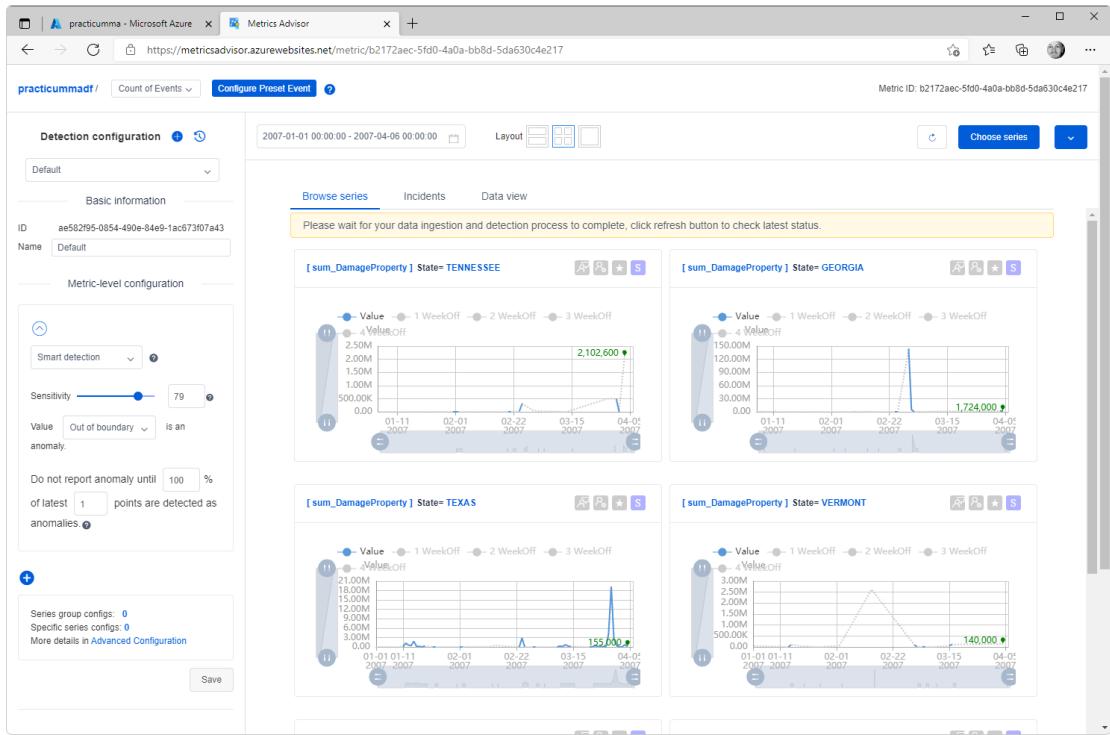
Metric name	Metric ID	Metric description	Series count	Start time	Actions
Count of Events	b2172aec-5fd0-4a0a-bb8d-5da630c4e217	Edit	0	2007-01-01T00:00:00Z	Actions

Add a new metric which is calculated by current metrics.

Name	practicummadf
ID	dbf98b41-f6f3-4fc-8fd-328b6be1a311
Created time	2021-09-21T15:06:28Z
Dimensions	State

Monitor progress on the data feed page.

Click into the Metric Name link to see analysis.



Objective Complete... congratulations!

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.

Step 1: Instantiate Prerequisites

This exercise requires the following resource(s):

- Azure SQL
- PowerApps

Step 2: Create Target Table

Navigate to SQL.

The screenshot shows a Microsoft Azure portal window with a dark theme. The left sidebar lists various service categories like Overview, Activity log, Tags, and Query editor (preview). The main area is titled "practicumsd (practicumsds/practicumsd) | Query editor (preview)" and features a large "SQL" icon. Below it, the text "Welcome to SQL Database Query Editor" is displayed. A "SQL server authentication" section contains fields for "Login" (rchapler) and "Password". To the right, an "Active Directory authentication" section has a "Continue as rchapler@microsoft.com" button. A message box at the bottom left states: "Cannot open server 'practicumsd' requested by the login. Client with IP address [REDACTED] is not allowed to access the server. To enable access, use the Azure Portal or run sp_set_firewall_rule on the master database to create a firewall rule for this IP address or address range. It may take up to five minutes for this change to take effect." There is also a "Whitelist it [REDACTED] on server practicumsd" link. At the bottom center is an "OK" button.

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 displays a 'Query 1' window with the following content:

```
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".

Execute the following T-SQL:

```
CREATE TABLE dbo.myTable( Id INT NOT NULL IDENTITY(1,1) PRIMARY KEY, Name VARCHAR(64), Picture IMAGE )
```

Step 3: 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**” page, click “+ New app” and select **Canvas** from the dropdown.

The screenshot shows the Power Apps 'Canvas app from blank' creation dialog. The left sidebar has 'Apps' selected. The main dialog includes the following fields:

- App name ***: rchabler
- Format**: Phone (radio button selected)
- Type**: Canvas (selected from a dropdown menu)

Below the dialog, there is a preview area with a pencil icon and the text: "Design the app you want, and connect it to hundreds of data sources." A 'Canvas app' button is also present.

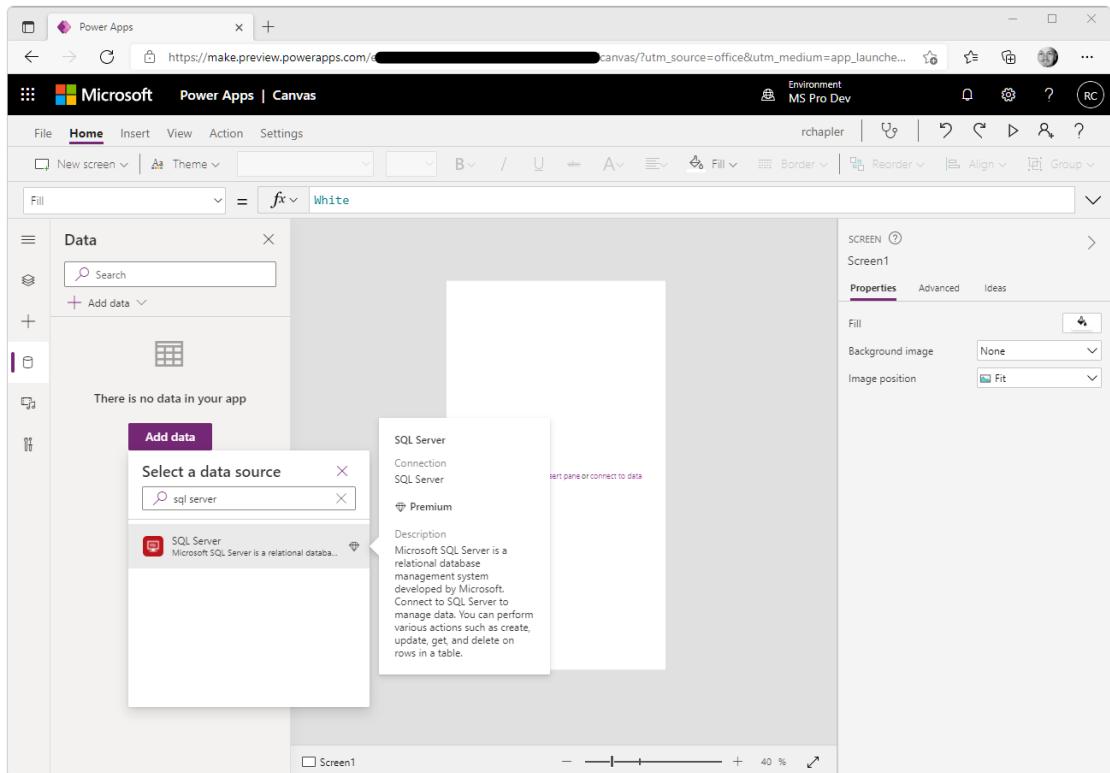
On the “**Canvas app from blank**” popup, enter the following items:

App Name	Enter a meaningful name aligned with standards
Format	Select the Phone radio button

Click the **Create** button.

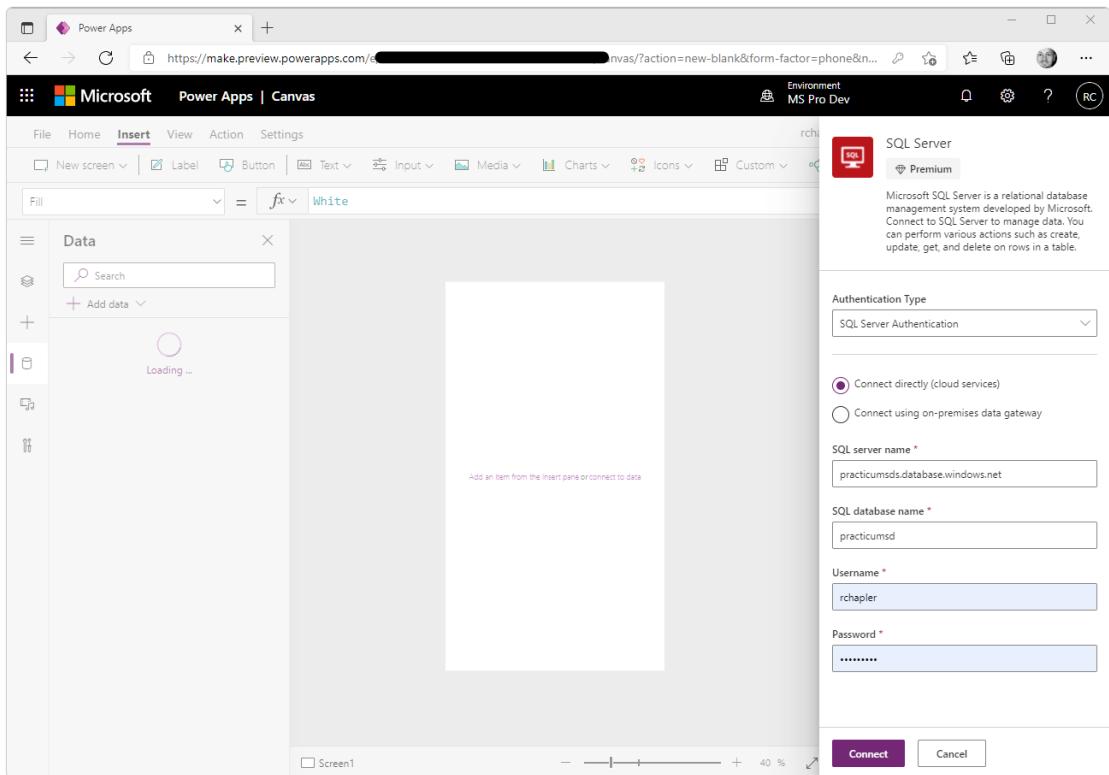
Step 4: Add Data

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 dropdown.



On the “SQL Server” popout, enter the following items:

Authentication Type SQL Server Authentication

SQL Server Name Enter the values employed during database server creation

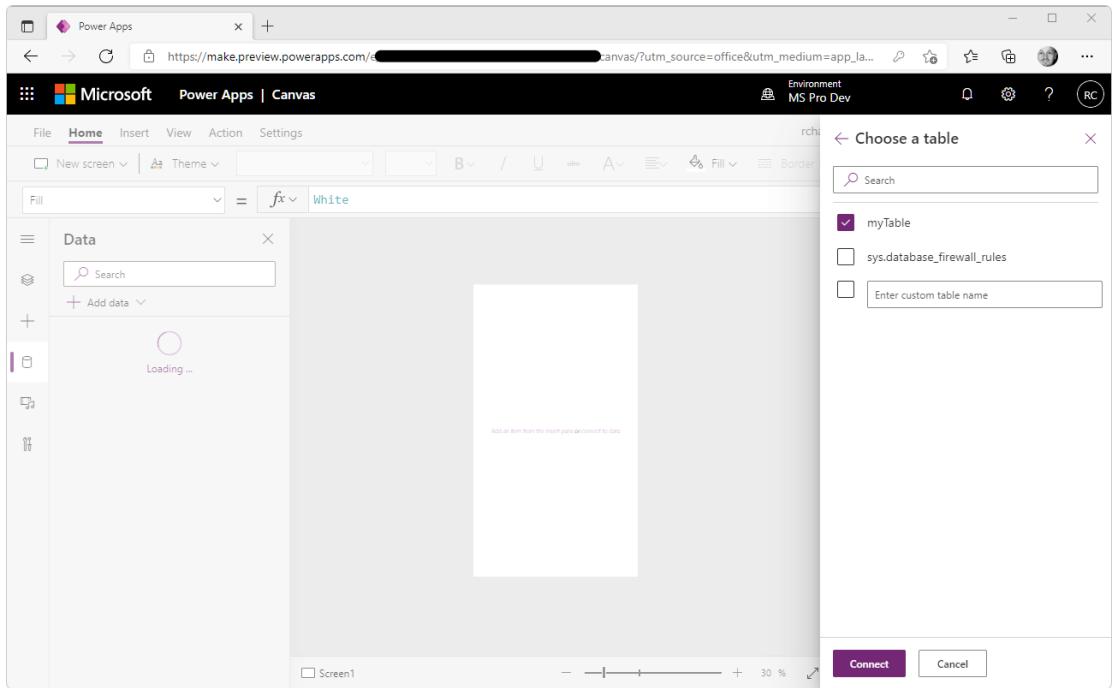
SQL Database Name

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.*



On the “Choose a table” popout, check the table created in [Create Target Table](#).

Click the **Connect** button.

Step 6: Insert Object Detector

Click the “Add an item from the Insert pane” link.

On the resulting, left-hand **Insert** menu, expand “AI Builder” and select “Object detector”.

On the resulting, right-hand popout, click “+ New model”.

Step 7: Create AI Model

The screenshot shows the Microsoft Power Apps | AI Builder interface. On the left, there is a navigation sidebar with options like Home, Learn, Apps, Create, Data, Flows, Chatbots, AI Builder, Build, Models, and Solutions. The main content area has a heading "Enhance your business with AI" and a sub-section "Refine a model for your business needs". It displays five cards: Category Classification, Entity Extraction, Form Processing, Object Detection, and Prediction. The "Object Detection" card is highlighted with a purple border.

On the new “Power Apps | AI Builder”, “Enhance your business with AI” page, click the “Object Detection” button.

Name the AI model and then click the **Create** button.

The screenshot shows the "Select your model's domain" page. On the left, there is a sidebar with options: Select domain (which is checked), Choose objects, Add images, Tag images, and Model summary. The main content area has a heading "Select your model's domain" and a sub-section explaining that models focus on specific types of objects. It shows three buttons: Common objects (which is selected and highlighted in purple), Objects on retail shelves, and Brand logos. Below these buttons is a preview image of a field with two yellow combines. A purple rectangular box highlights the "Common objects" button. To the right, there is a "Quick tips" section with a "Get help or send feedback" link.

On the “Select your model's domain” page, click the “Common objects” and then the “Next” button.

The screenshot shows the Microsoft Power Apps AI Builder interface. On the left, a sidebar lists steps: 'Select domain' (checked), 'Common objects', '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 sub-instruction 'You can add them manually or select from your database. [Learn more](#)'. It features a text input 'Object names' containing 'Headlights' and a button '+ Add new object'. A progress bar at the bottom indicates 'Back' and 'Next' buttons, with '1 object name selected'. To the right, there's a 'Car Parts' section, a 'Save and close' button, 'Quick tips' (about selecting from database instead), and help links ('Select from database instead', 'Switch to database', 'Get help', 'Get help feedback').

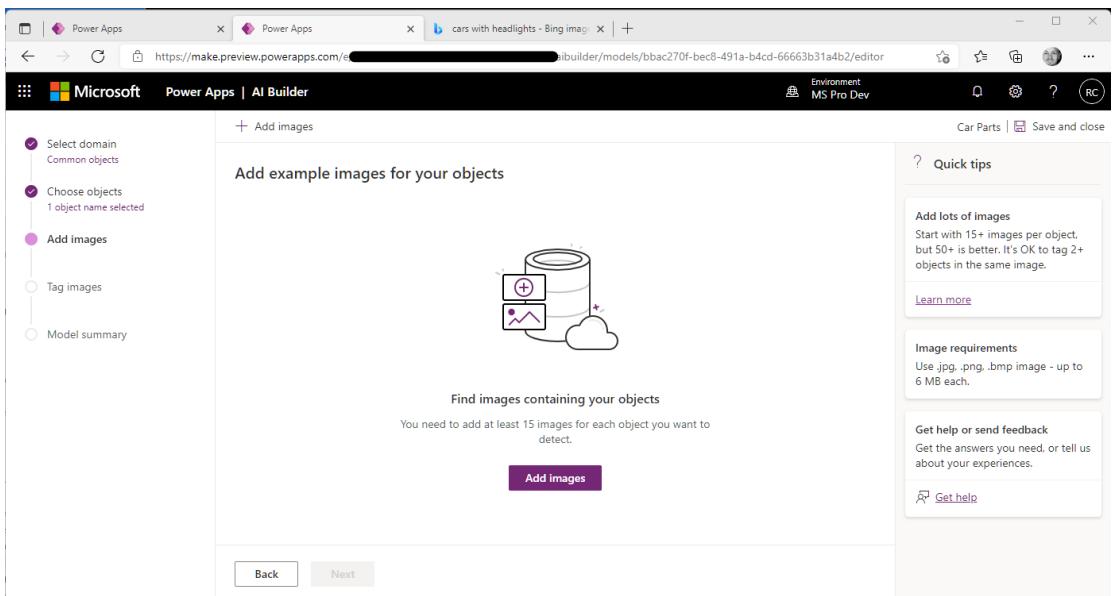
On the “Choose objects for your model to detect” page, click “+ Add new object”, enter an object name and then click the “Next” button.

On the “Add example images for your objects” page, we will add image files that will be used to train the AI model.

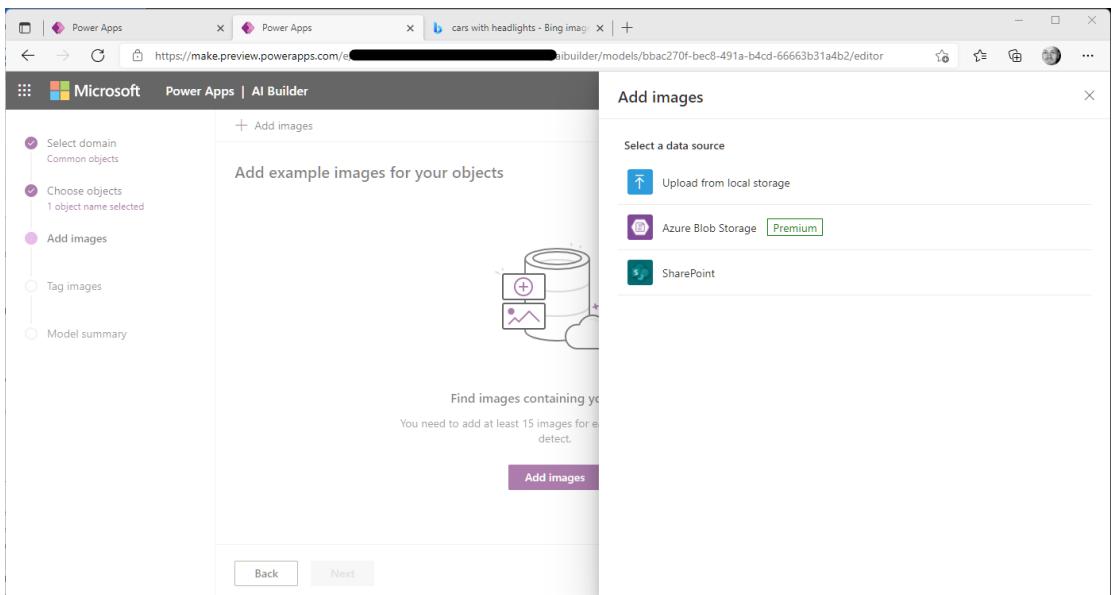
For this exercise, we can pull images of cars (with headlights) from a search engine.

The screenshot shows a Bing image search results page with the query 'cars with headlights'. The search bar shows 'cars with headlights'. Below it are several filter buttons: Round, LED Lights, Night, Parts, Replacement, Front, and Vintage. The main area displays a grid of 15 car images. The first row contains five images: a green classic sports car, a silver sedan, a yellow classic sports car, a yellow hatchback, and a close-up of a white car's headlight. The second row contains five images: a white car with headlights on, a grey car with headlights on, a blue Audi R8, a yellow hatchback, and a black vintage sports car. The third row contains five images: a white car with headlights on, a red sports car, three cars side-by-side with headlights on, a red car, a white car, and a close-up of a red car's headlight. A 'Feedback' button is visible in the bottom right corner of the image grid.

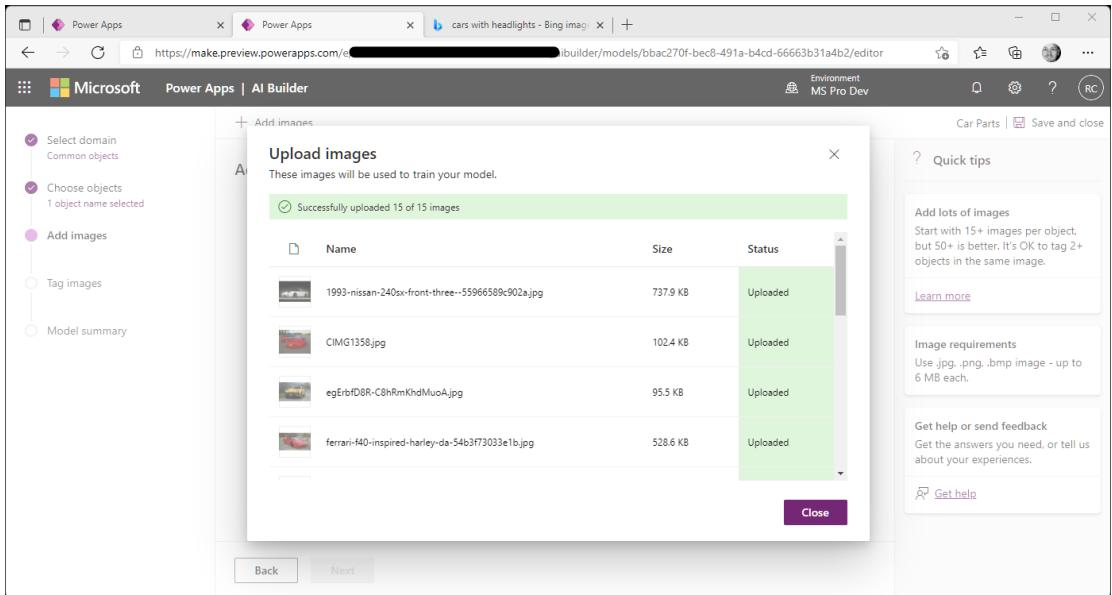
For each of the required 15 images, download an image file to a temporary folder on the Windows desktop.



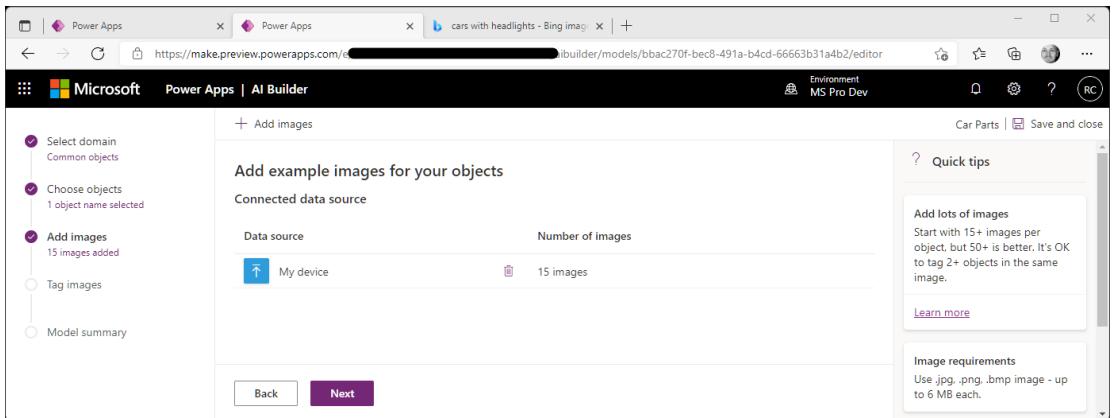
On the “Add example images for your objects” page, click the “Add images” button.



On the resulting “**Add images**” popout, click “**Upload from local storage**”. Select the Example Images downloaded to your desktop.



Review the items in the “Upload images” popup, then click the “Upload 15 images” button. Click **Close**.



Back on the “Add example images for your desktop” page, confirm “Number of images” shows 15 or more images, then click **Next**.

Select the first image on the “Tag the objects in your images” page.

Drag a rectangle around headlights in the image, then click the Headlights button on the “Choose object” popup.

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.

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	Number of images
My device	15 images

Information to extract

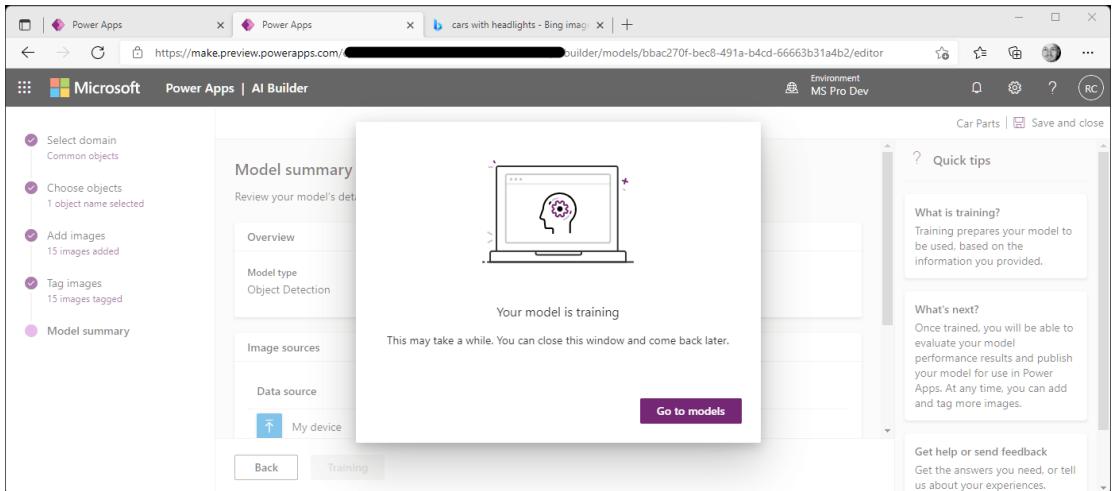
Object	Tags
Headlights	15

What is training? Training prepares your model to be used, based on the information you provided.

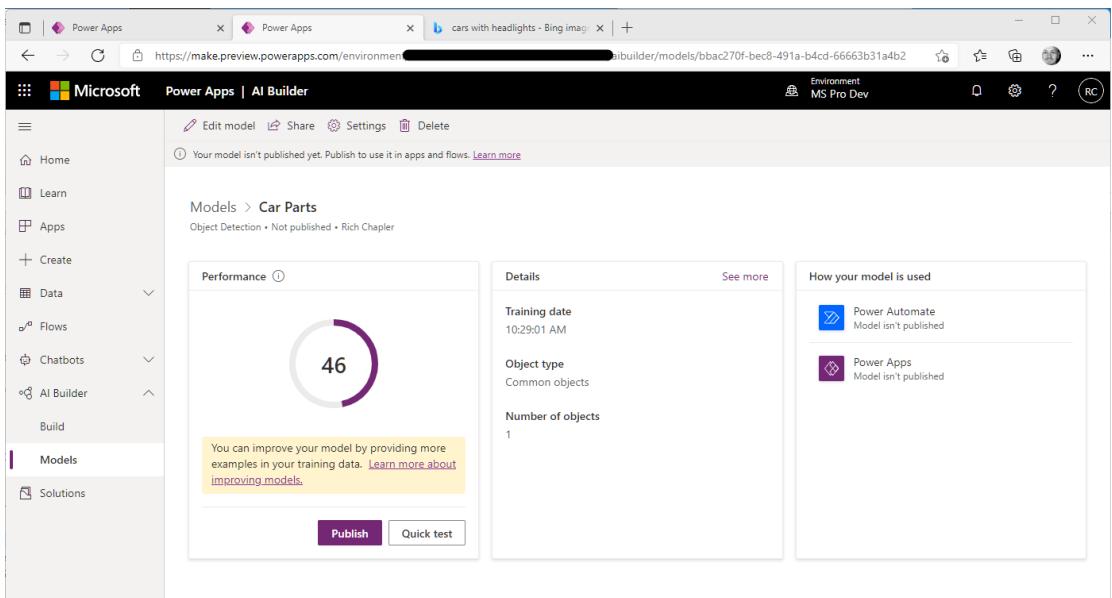
What's next? Once trained, you will be able to evaluate your model performance results and publish your model for use in Power Apps. At any time, you can add and tag more images.

Get help or send feedback Get the answers you need, or tell us about your experiences.

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.

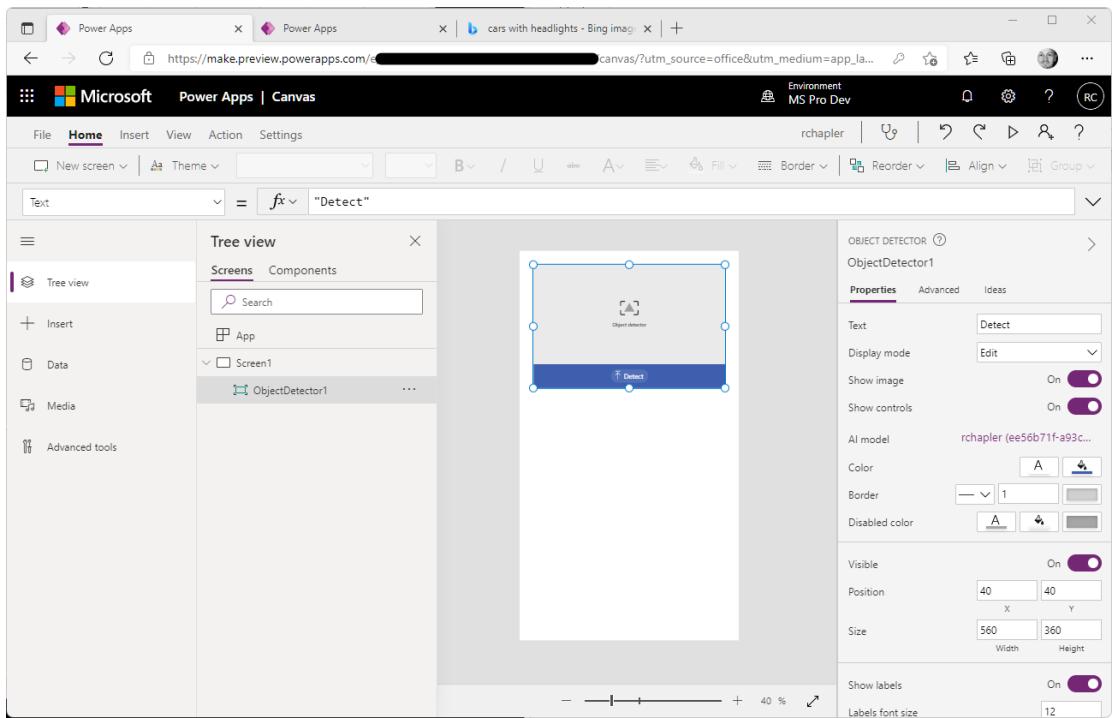


In the snip above, we see that **Performance** {i.e., model performance expectation %} of 46 for the example images included in training.

In the future, you can improve model quality by including more images and re-training the model.

Click the **Publish** button.

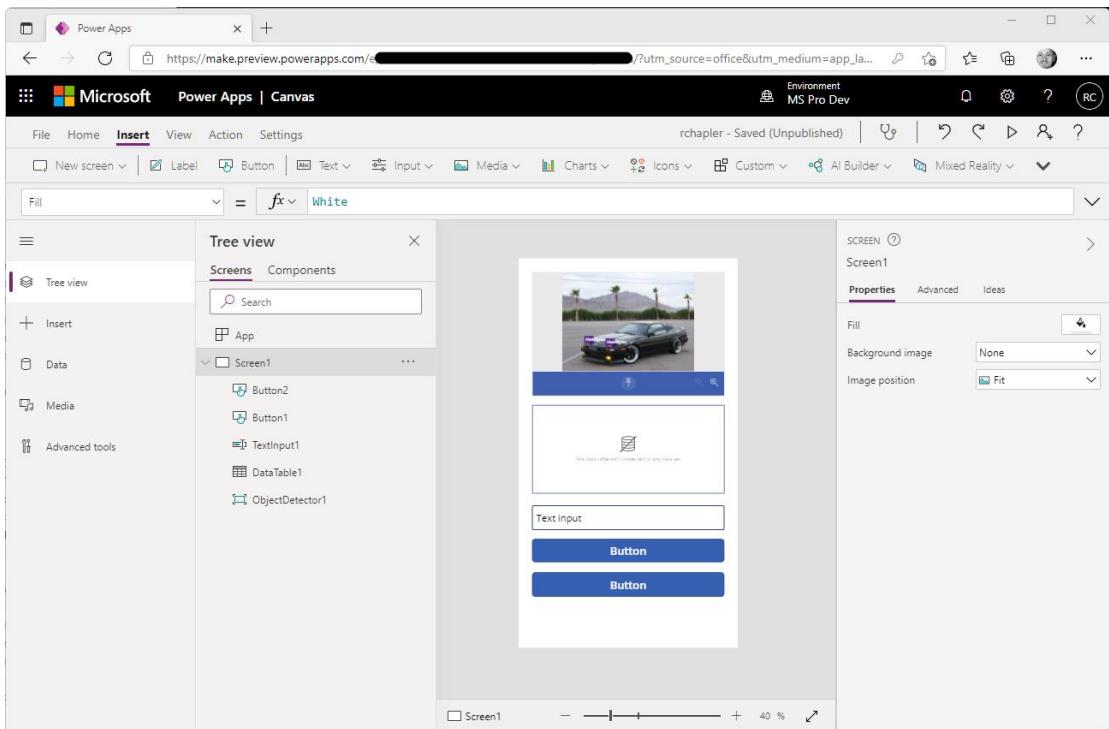
Return to Power Apps and complete the **ObjectDetector1** component by selecting your newly created model.



Note: Consider resizing the component to improve visibility.

Step 8: Insert / Configure Controls

Return by closing Preview Mode.

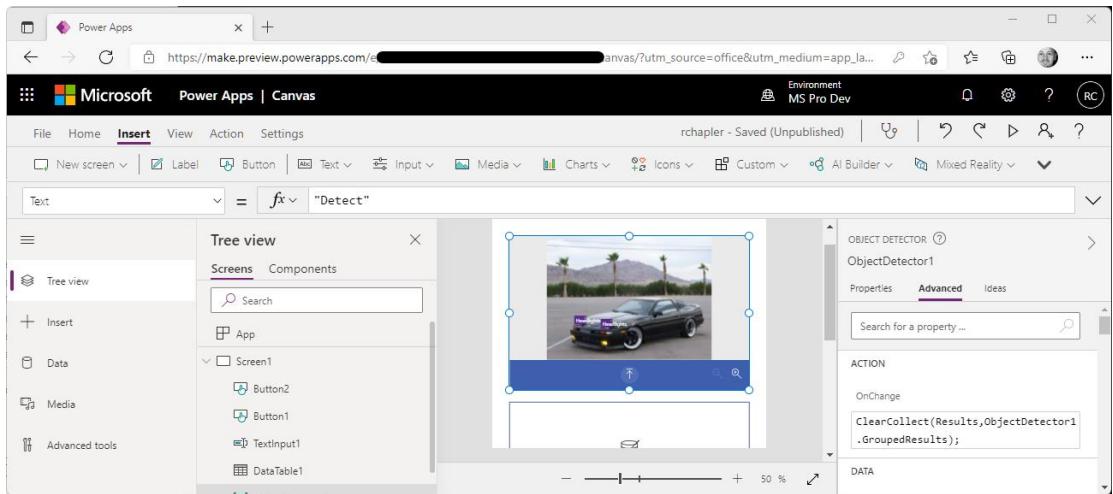


Insert the following controls:

- **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

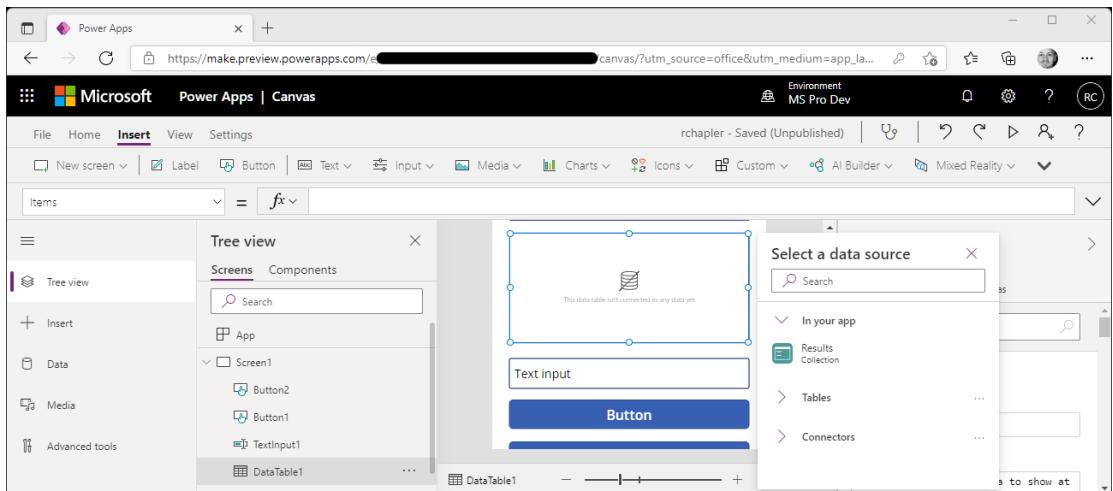


Click on **ObjectDetector1** in the left-hand **Tree View**. On the resulting right-hand popout, click the **Advanced** tab.

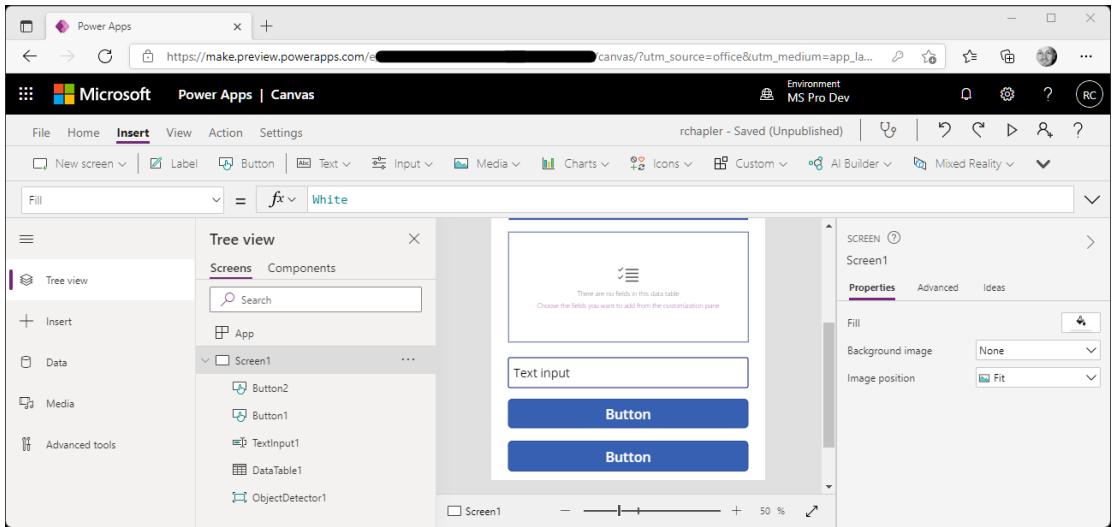
Paste the following logic into **Action | OnChange**:

```
ClearCollect(Results, ObjectDetector1.GroupedResults);
```

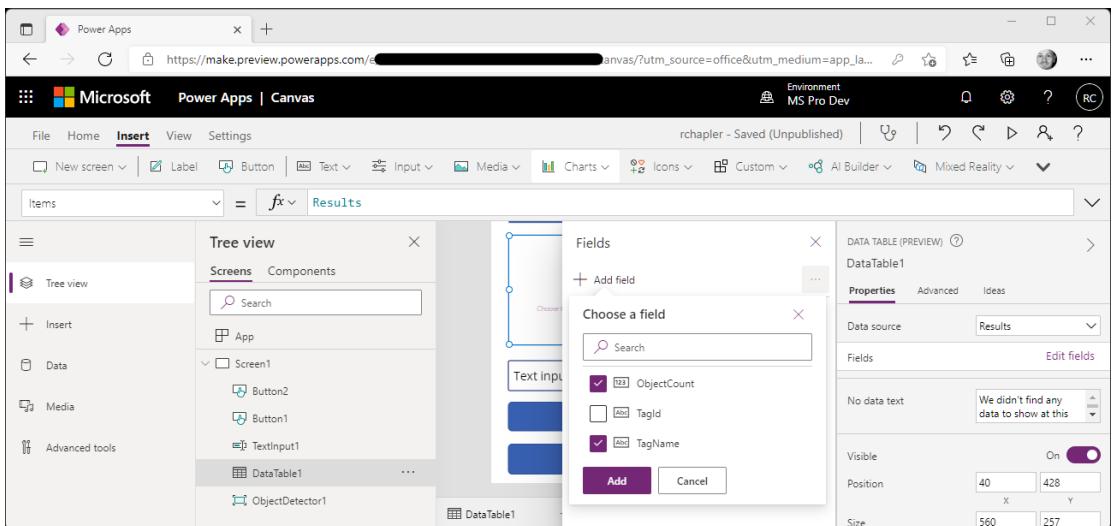
DataTable1



Click on **DataTable1** in the left-hand **Tree View**. On the resulting popup, select “**In your app**” | **Results**.



On the updated control, click the “Choose the fields you want to add...” link.

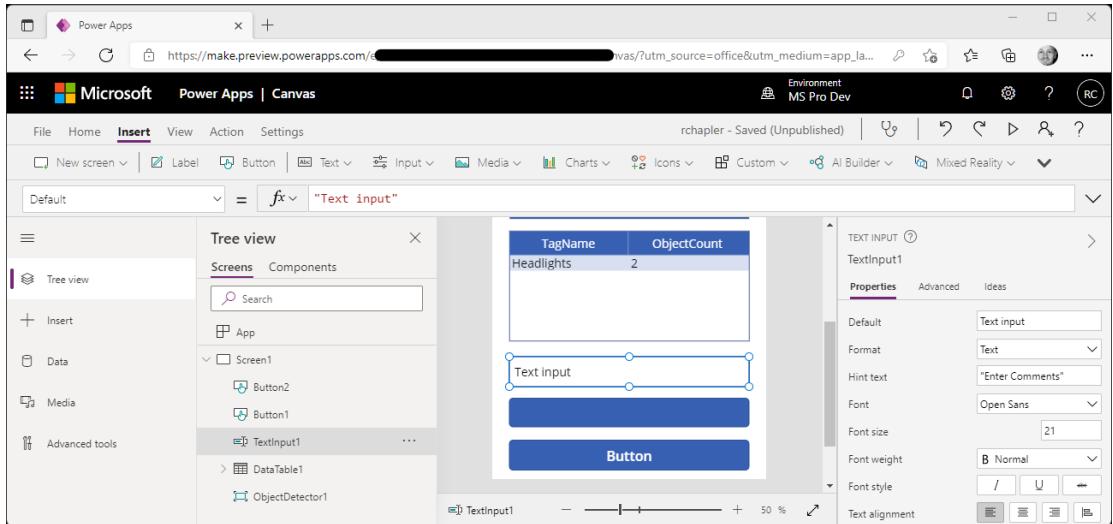


On the resulting **Fields** popout, click “+ Add field”.

In the resulting “Choose a field” popup, check **TagName** and then **ObjectCount**.

Click the **Add** button.

TextInput1

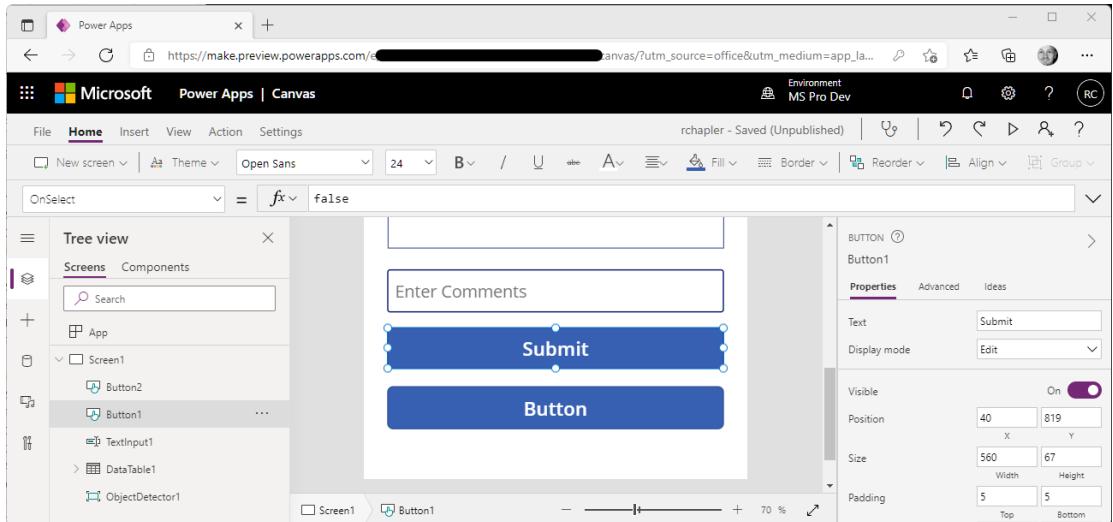


Click on **TextInput1** in the left-hand **Tree View**.

On the resulting “**TEXT INPUT**” popout, **Advanced** tab, enter the following items:

Default	""
Hint Text	“Enter Comments”

Button1



Click on **Button1** in the left-hand **Tree View**.

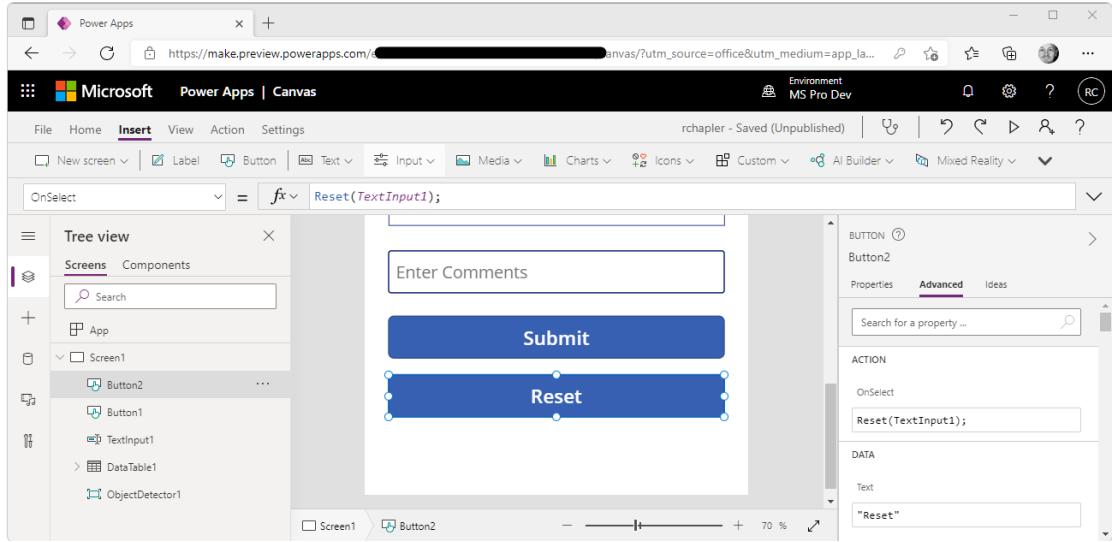
On the resulting right-hand popout, click the **Properties** tab, and enter the word “Submit” in the **Text** input.

On the resulting right-hand popout, click the **Advanced** tab,

Paste the following logic into **Action | OnSelect**:

```
Patch(myTable,Defaults(myTable),{Name:TextInput1.Text,Picture:ObjectDetector1.OriginalImage})
```

Button2



Click on **Button2** in the left-hand **Tree View**.

On the resulting right-hand popout, click the **Properties** tab, and enter the word “Reset” in the **Text** input.

On the resulting right-hand popout, click the **Advanced** tab,

Paste the following logic into **Action | OnSelect**:

```
Reset(TextInput1);
```

Step 9: 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 page, click the “**Preview the app**” button {i.e., Play icon} in the upper-right.

Power Apps

https://make.preview.powerapps.com/.../canvas?utm_source=office&utm_medium=app_la...

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

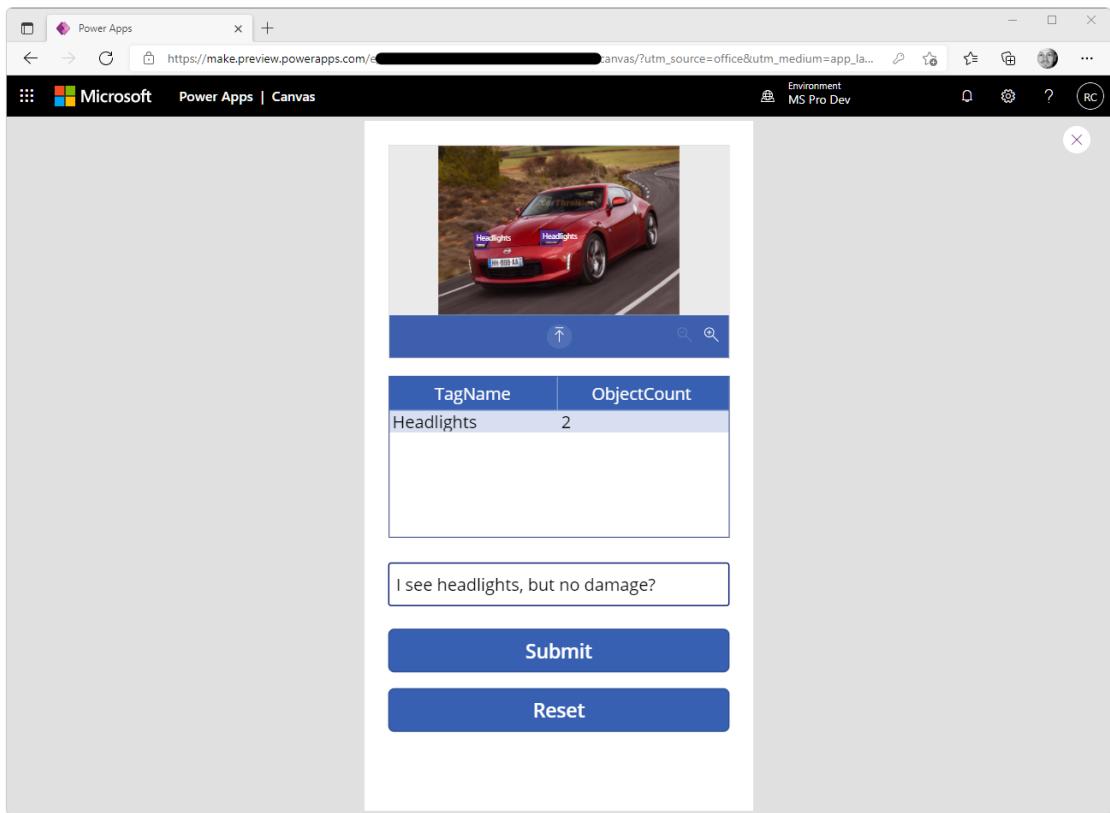
white

ImagePosition

ImagePosition.Fit

The screenshot shows the Microsoft Power Apps Canvas interface. On the left, the 'Tree view' pane displays the app structure with 'Screen1' selected, containing components like 'Button2', 'Button1', 'TextInput1', 'DataTable1', and 'ObjectDetector1'. The main canvas area shows a red sports car on a road, with a data table below it. The table has two columns: 'TagName' and 'ObjectCount'. One row shows 'Headlights' with a value of '2'. Below the table is a text input field labeled 'Enter Comments' and two blue buttons labeled 'Submit' and 'Reset'. The right side of the interface shows the 'Advanced' tab of the 'Screen1' properties panel, with sections for ACTION (OnVisible, OnHidden), DATA (BackgroundImage, ContentLanguage, etc.), and DESIGN (Fill, ImagePosition). A preview window on the right shows the current state of the app.

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.

The screenshot shows the Microsoft Azure (Preview) portal interface. On the left, there's a sidebar with various service links: Overview, Activity log, Tags, Diagnose and solve problems, Quick start, Query editor (preview), Power Platform (Power BI (preview), Power Apps (preview), Power Automate (preview)), Settings (Compute + storage, Connection strings, Maintenance, Properties, Locks), Data management (Replicas, Sync to other databases), and Integrations (Stream analytics (preview)). The main content area is titled "practicumsd (practicumsd/practicumsd) | Query editor (preview)". It shows a message: "Showing limited object explorer here. For full capability please open SSDT." Below this is a tree view of database objects: Tables (dbo.myTable, ...), Views, and Stored Procedures. In the Query 2 tab, a query is run: "SELECT TOP (1000) * FROM [dbo].[myTable]". The results show two rows:

ID	Name	Picture
1	Nice car, but no apparent quality d...	/9j/4AAQSkJlRgABAQEAAAAAAA...
2	I see headlights, but no damage?	/9j/2wBDAAYEBQYFBAYGBQYHbwY...

At the bottom, a message says "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.

Objective Complete... congratulations!

Audit Usage

Follow these instructions to support regular audit of and alerting on key data products.

Step 1: Instantiate Prerequisites

This exercise requires the following resource(s):

- Synapse

Step 2: Configure Auditing

Navigate to the **Overview** page in Synapse, Dedicated SQL Pool.

The screenshot shows the Microsoft Azure portal interface for a Dedicated SQL Pool named "practicumdsp". The left sidebar navigation includes "Overview", "Activity log", "Access control (IAM)", "Tags", "Settings", "Workload management", "Maintenance schedule", "Geo-backup policy", "Connection strings", "Properties", "Locks", "Security" (which is expanded to show "Auditing", "Data Discovery & Classification", "Dynamic Data Masking", and "Security Center"), and "Notifications (0)", "Features (4)", and "Tasks (2)". The main content area is titled "Essentials" and lists the following details:

Resource group (change)	Workspace name
practicumsg	practicumsaw
Status	Performance level
Online	DW100c
Location	Connection strings
West US 2	Show database connection strings
Subscription (change)	Maintenance schedule
rchapler	Sun 02:00 UTC (8h) / Thu 00:00 UTC (8h)
Subscription ID	
91e9fddc-ef15-416c-9be1-085f8b1b46ed	
Tags (change)	
Click here to add tags	

Below this, under "Features (4)", are two sections: "Transparent data encryption" (NOT CONFIGURED) and "Auditing" (NOT CONFIGURED). The "Auditing" section 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

practicumdsp

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

Objective Complete... congratulations!

Deploy

The following sections describe automatic instantiation and deployment of the Azure resources necessary for the objectives in this book.

Infrastructure-as-Code

Step 1: Understand Use Case

Work with your customer to understand requirements; example notes:

- “We want to **automate infrastructure deployment** to TEST and PROD environments”
- “We want to **protect our investment** in infrastructure / configuration with source control”

Step 2: Instantiate Prerequisites

This exercise requires either of the following resource(s):

- [DevOps](#) (with project and repository)
- [GitHub Enterprise](#) (with repository)
- [Service Principal](#) (with contributor permissions to the subscription)

Step 3: Create Template

In this section, we will prepare a bare-minimum template, and then manually deploy that template.

Create Custom Template

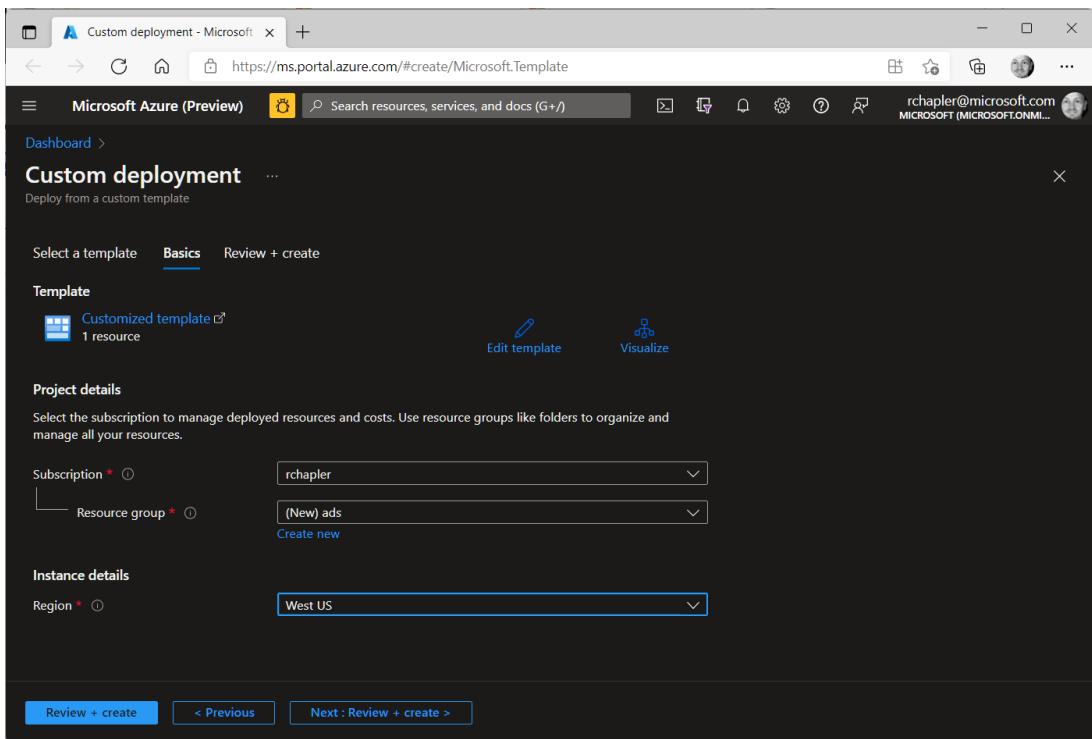
Navigate to <https://ms.portal.azure.com/#create/Microsoft.Template> and then click on the “**Build your own template in the editor**” link. On the resulting “**Edit template**” screen, replace the default JSON with the following prepared, “bare minimum” JSON:

```
{  
  "$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"  
    }  
  ]  
}
```

Special logic callouts...

[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.



On the “Custom deployment” page, enter values for the following items:

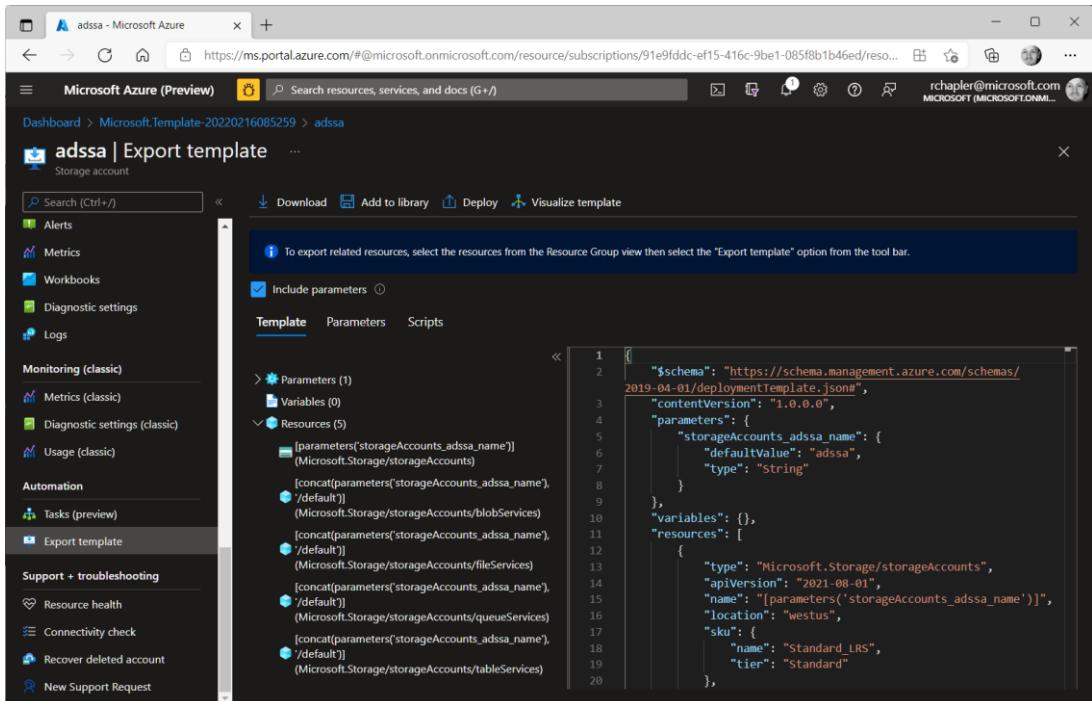
Resource Group	Select existing or “Create new”
Region	Select a region appropriate for your situation

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 from Existing Resource

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.



```
1  {
2      "$schema": "https://schema.management.azure.com/schemas/
2019-04-01/deploymentTemplate.json#",
3      "contentVersion": "1.0.0.0",
4      "parameters": {
5          "storageAccounts_adssa_name": {
6              "defaultValue": "adssa",
7              "type": "String"
8          }
9      },
10     "variables": {},
11     "resources": [
12         {
13             "type": "Microsoft.Storage/storageAccounts",
14             "apiVersion": "2021-08-01",
15             "name": "[parameters('storageAccounts_adssa_name')]",
16             "location": "westus",
17             "sku": {
18                 "name": "Standard_LRS",
19                 "tier": "Standard"
20             }
21         }
22     ]
23 }
```

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.

Step 4: Deploy Template

...via DevOps Pipeline

In this section, we will create, save, and run a DevOps Pipeline that deploys an ARM template.

Step 1: Upload Template

Upload the ARM Template to the DevOps repo by completing the following steps:

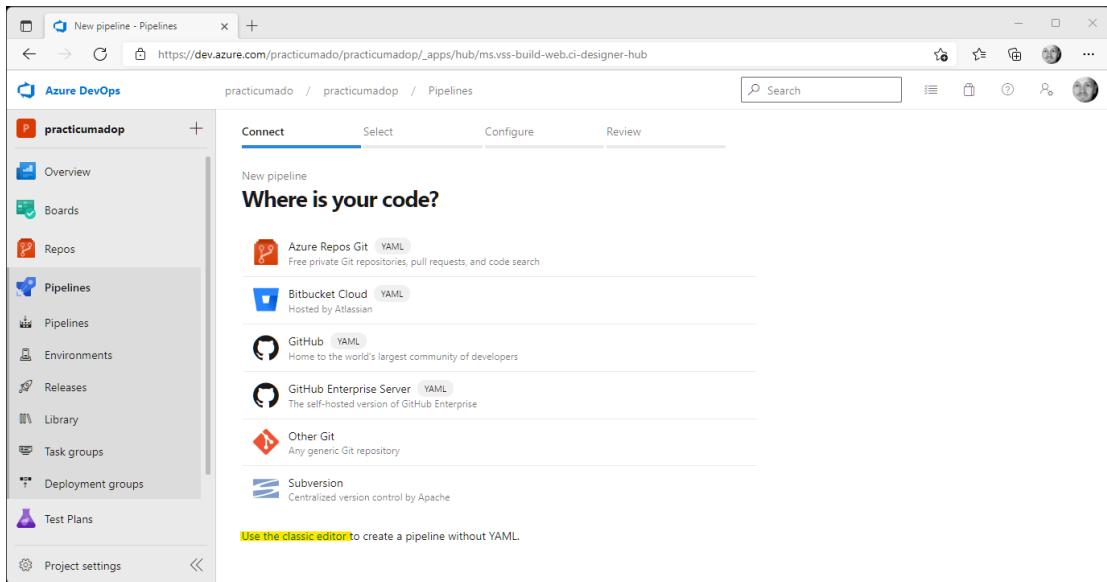
- Navigate to DevOps, select your project, and then click on **Repos** in the navigation pane
- Click the + icon just to the right of your project name and select “**New repository**” from the resulting dropdown
- In the resulting “**Create a repository**” popout, enter a “**Repository name**” {e.g., “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 dropdown
- 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.

Step 2: Create Pipeline

Navigate to DevOps, click on **Pipelines** in the navigation pane, and then click the “**New Pipeline**” button.



The screenshot shows the Azure DevOps interface for creating a new pipeline. The left sidebar shows the project 'practicumadop' and the 'Pipelines' option is selected. The main area has a title 'Where is your code?' and a list of code sources:

- Azure Repos Git (YAML)
- Bitbucket Cloud (YAML)
- GitHub (YAML)
- Github Enterprise Server (YAML)
- Other Git (Any generic Git repository)
- Subversion (Centralized version control by Apache)

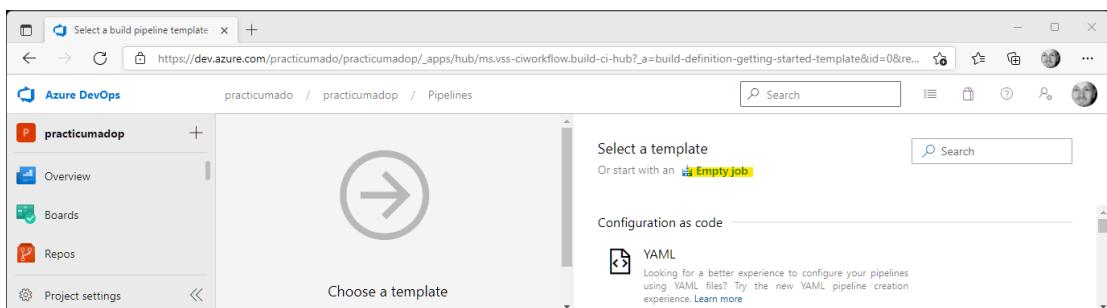
At the bottom, there is a link: [Use the classic editor](#) to create a pipeline without YAML.

On the Connect tab {i.e., “New Pipeline ... Where is your code?”} page, click the “**Use classic editor**” link.

On the “**Select a source**” popout, select “**Azure Repos Git**” and then enter values for the following items:

Team Project	Select your DevOps project
Repository	Select your DevOps repository {e.g., infrastructure }
Default Branch...	Confirm default selection, main

Click the **Continue** button.



The screenshot shows the 'Select a build pipeline template' page. The left sidebar shows the project 'practicumadop'. The main area features a large circular arrow icon with an arrow pointing right. Below it, there is a section titled 'Choose a template' with a 'YAML' option. A note below says: 'Looking for a better experience to configure your pipelines using YAML files? Try the new YAML pipeline creation experience. [Learn more](#)'.

Click “Empty job” on the “Select a template” popout.

The screenshot shows the Azure DevOps Pipelines interface. On the left, the navigation bar is visible with 'practicumadop' selected. The main area displays a 'Build pipeline' named 'practicumadop-Cl'. The pipeline consists of a single task, 'Get sources', which is configured to pull from the 'practicumadop_infrastructure' repository. Below the tasks, there is a section for 'Agent job 1' with the option 'Run on agent'. To the right, there are configuration fields for 'Name' (set to 'practicumadop-Cl'), 'Agent pool' (set to 'Azure Pipelines'), and 'Agent Specification' (set to 'windows-2019'). A 'Parameters' section is present with a note about pipeline parameters. At the bottom right, there is a 'Save & queue' button.

Confirm default values for **Name**, **Agent Pool**, and **Agent Specification**.

The screenshot shows the same Azure DevOps Pipelines interface as before, but with a modal window open over the pipeline details. The modal is titled 'Add tasks' and contains a search bar with the term 'arm'. Below the search bar, a list of extensions is shown. The first item is 'ARM template deployment' by Microsoft Corporation, which is highlighted with a blue background. An 'Add' button is located at the bottom right of this card. Other items in the list include 'ARM Outputs' and 'Marketplace'.

Click the “+” button to the right of “Agent Job 1”.

Search for and then select “**ARM template deployment**”.

Click the **Add** button.

On the “**ARM template deployment**” popout, enter values for the following items:

Deployment Scope	Confirm default, “ Resource Group ”
Azure Resource Manager...	Select your subscription
Subscription	Select your subscription
Action	Select “Create or update resource group”
Resource Group	Select your resource group
Location	Match resource group selection
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 dropdown.

Confirm values on the resulting “**Run pipeline**” popout, and then click the “**Save and run**” button.

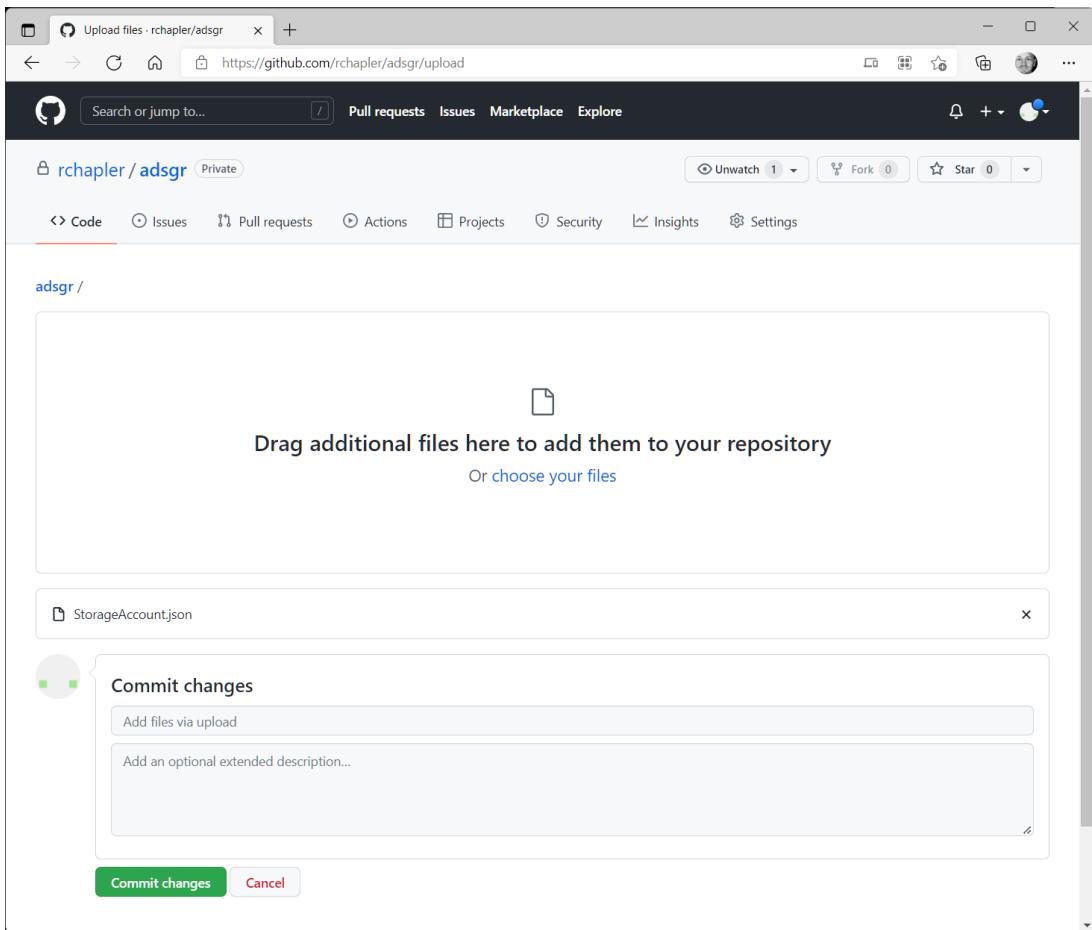
...via GitHub Action

In this section, we will create, save, and run a GitHub Action that deploys an ARM template.

Step 1: Upload Template

Modify (with your values) and navigate to the following URL:

<https://github.com/rchaplert/adsgr/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.

Step 2: 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": "{Subscription GUID}",  
  "tenantId": "{Tenant GUID}"  
}
```

You should use the values produced when you created the Service Principal.

Step 3: Create Secret

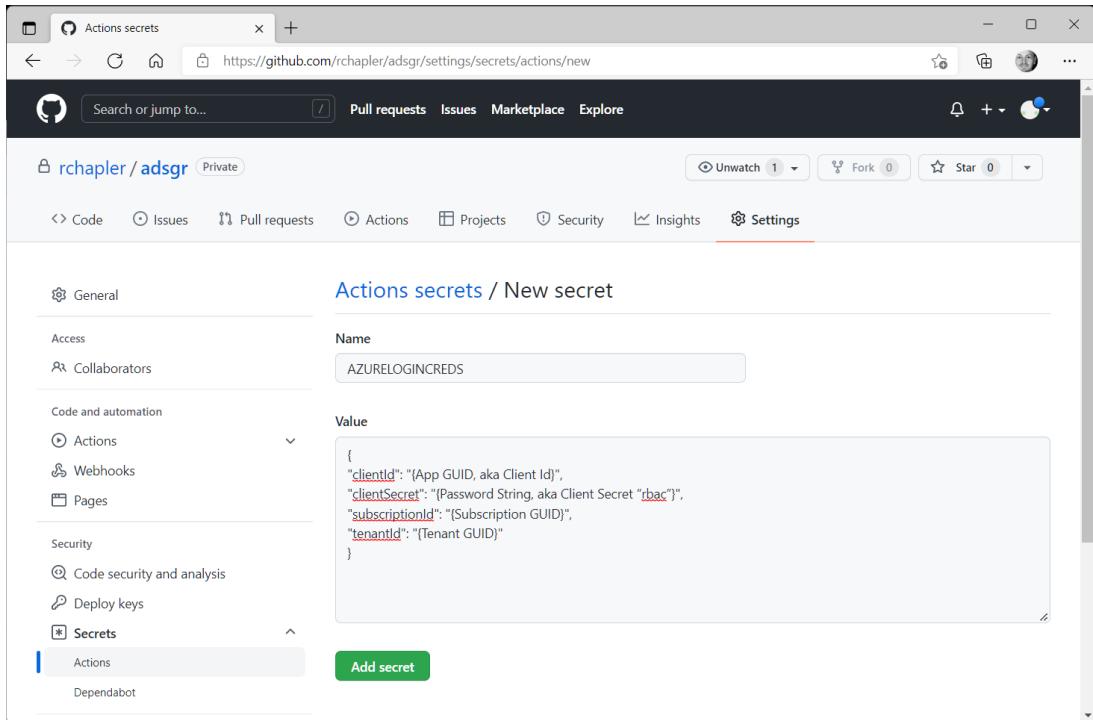
Modify and navigate to the following URL:

[https://github.com/rchapler/shared/settings/secrets/actions.](https://github.com/rchapler/shared/settings/secrets/actions)

Click the “**New repository secret**” button.

On the “**Actions secrets / new secret**” page, enter values for the following items:

Name	Enter a meaningful name aligned with standards
Value	Paste the credential string prepared in the prior step



The screenshot shows the "Actions secrets / New secret" page on GitHub. The "Name" field contains "AZURELOGINCRED\$". The "Value" field contains the following JSON code:

```
{
  "clientId": "[App GUID, aka Client Id]",
  "clientSecret": "[Password String, aka Client Secret \"rbac\"]",
  "subscriptionId": "[Subscription GUID]",
  "tenantId": "[Tenant GUID]"
}
```

The left sidebar shows navigation links for General, Access, Collaborators, Code and automation (Actions, Webhooks, Pages), Security, Code security and analysis, Deploy keys, and Secrets. The "Actions" link is currently selected. At the bottom right, there is a green "Add secret" button.

Click the “**Add secret**” button.

Step 4: Create Workflow

The screenshot shows a browser window for GitHub Actions. The URL is <https://github.com/rchapler/adsgr/actions/new?category=none&query=simple+workflow>. The GitHub header includes 'Actions · rchapler/adsgr' and a search bar. The main navigation bar has tabs for 'Pull requests', 'Issues', 'Marketplace', 'Explore', 'Actions' (which is highlighted), 'Projects', 'Security', 'Insights', and 'Settings'. Below the navigation, there are buttons for 'Unwatch 1', 'Fork 0', and 'Star 0'. A sidebar on the left lists categories: 'Automation', 'Continuous integration', and 'Deployment'. A search bar at the top right contains the query 'simple workflow'. The results section says 'Found 2 workflows' and lists two options: 'Simple workflow' (By GitHub) and 'Manual workflow' (By GitHub Actions). Both cards have a 'Configure' button.

Click **Actions** in the menubar, search for “**Simple workflow**”, and then click the **Configure** button.

```
name: rchapler20220217
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 YML.

```
name: rchapler20220216
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
```

Click the “Start commit” button and then the “Commit new file” button on the resulting popup.

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 file: <https://github.com/rchapler/adsgr/blob/main/.github/workflows/rchapler20220217.yml>.

On the resulting page, click on the link for the committed YML 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 status of the run is 'In progress'.

Click on the run {i.e., “Create rchapler20220217.yml”}.

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

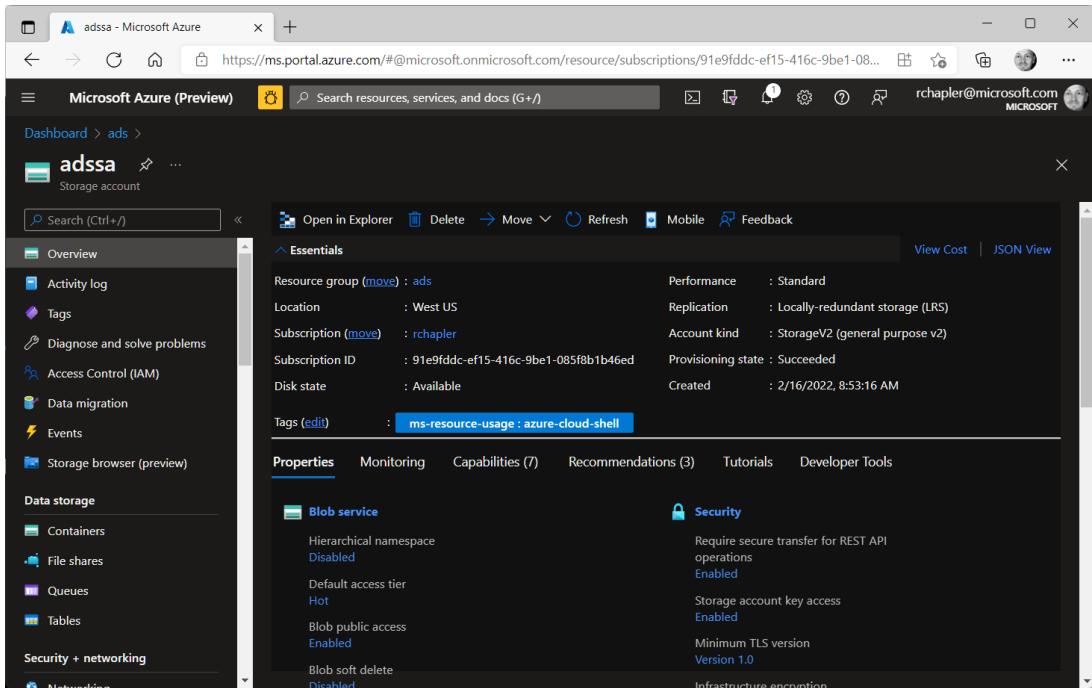
Run ARM deploy

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 (Preview) interface. The left sidebar shows the navigation path: Dashboard > ads > adssa. The main content area displays the 'adssa' storage account details. The 'Properties' tab is selected, showing the following information:

Resource group	: ads	Performance	: Standard
Location	: West US	Replication	: Locally-redundant storage (LRS)
Subscription	: rchabler	Account kind	: StorageV2 (general purpose v2)
Subscription ID	: 91e9ffddc-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		

Below the properties, there are sections for 'Blob service' and 'Security'. The 'Blob service' section shows settings like Hierarchical namespace (Disabled), Default access tier (Hot), Blob public access (Enabled), and Blob soft delete (Disabled). The 'Security' section shows settings like Require secure transfer for REST API operations (Enabled), Storage account key access (Enabled), Minimum TLS version (Version 1.0), and Infrastructure encryption.

Objective Complete... congratulations!

Step 5: Incorporate Additional Templates

The templates in this section...

- ...are “bare bones” {i.e., include only the minimum amount of code necessary for instantiation}
- ...use the latest API version rather than a specific dated version; as such, schema might require update over time

File Sync (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"  
      }  
    }  
  ]  
}
```

Storage Account

```
{  
  "$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

```
{  
  "$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": "[toLowerCase(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]",  
      "tags": {  
        "NetworkSecurityGroup": "true"  
      },  
      "properties": {  
        "ipConfigurations": [  
          {  
            "name": "ipConfig1",  
            "publicIPAddress": "[resourceId('Microsoft.Network/publicIPAddresses', concat(resourceGroup().name, '-pia'))]",  
            "subnet": "[resourceId('Microsoft.Network/virtualNetworkSubnets', concat(resourceGroup().name, '/vnet1'), 'Subnet1')]"  
          }  
        ]  
      }  
    }  
  ]  
}
```

```

"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": "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": "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]"
    }
}
}

```

Source Control

Follow these instructions to connect Synapse to DevOps, create branches, and pull requests.

Step 1: Understand Use Case

Work with your customer to understand requirements; example notes:

- “We need multiple developers to collaborate on parts of a large use case”

Step 2: Instantiate Prerequisites

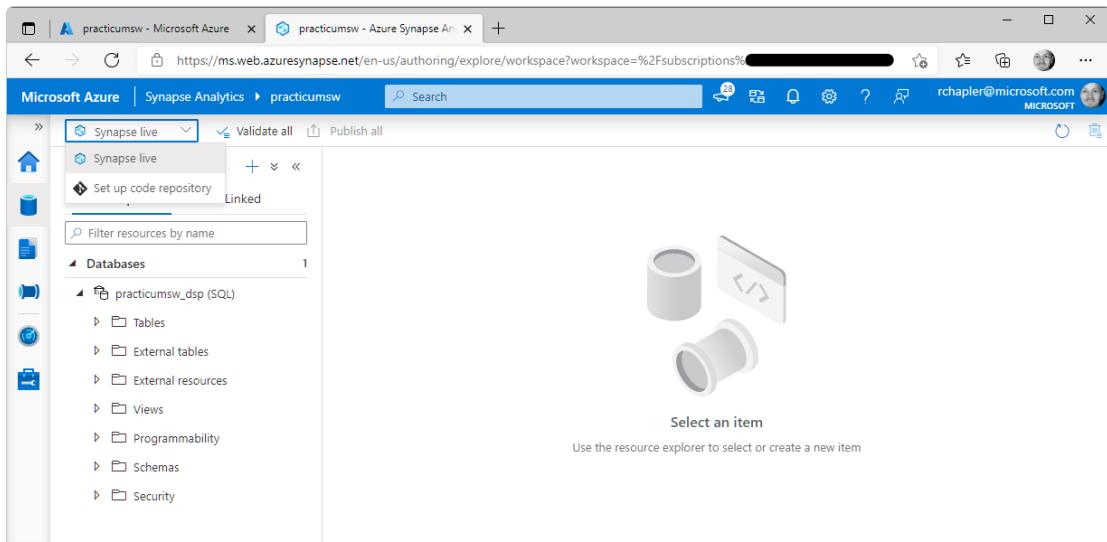
This exercise requires the following resource(s):

- DevOps
- Synapse (with Dedicated SQL Pool)

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

Step 3: Setup Code Repository

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



Click “**Synapse live**” and then select “**Set up code repository**” from the resulting dropdown.

Configure a repository

Specify the settings that you want to use when connecting to your repository.

Repository type * Azure DevOps Git

Azure Active Directory Microsoft

Continue **Cancel**

On the “Configure a repository” popout, enter values for the following items:

Repository Type Select “Azure DevOps Git” from the dropdown

Azure Active Directory Select the value appropriate for your organization

Click the **Continue** button.

Configure a repository

Specify the settings that you want to use when connecting to your repository.

Select repository Use repository link

Azure DevOps organization name * practicumad0

Project name * practicumad0p

Repository name * practicumad0p

Collaboration branch * collaboration_synapse

Publish branch * workspace_publish

Root folder * /

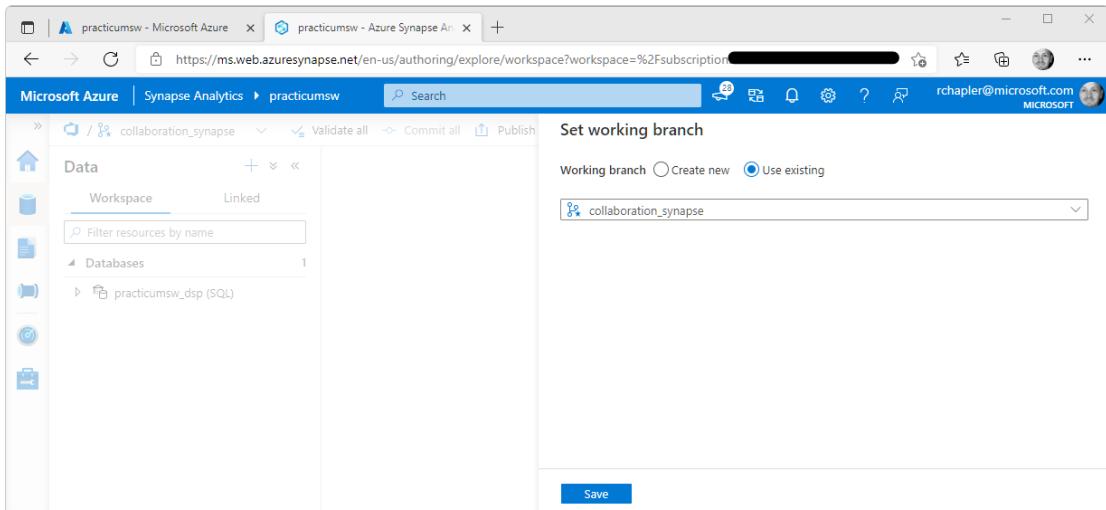
Import existing resource Import existing resources to repository

Apply **Back** **Cancel**

On the second “Configure a repository” popout, enter values for the following items:

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	Dropdown, click “+ Create new”, enter an appropriate name in the resulting popup, and click Create
Publish Branch	Confirm default, workspace_publish
Root Folder	Confirm default, /
Import Existing Resources...	Unchecked

Click the **Apply** button.



Confirm values and then click the **Save** button.

When processing is complete, you will note that we are now working in the new branch.

Step 4: Confirm Success

In this section, we will confirm success by simulating a deployment workflow {i.e., modification, and then pull request}.

Modify one or more of the following:

- SQL Script
- Spark Notebook
- Data Flow
- Pipeline

Pull Request

Next, we will promote changes from the collaboration branch to the master branch via Pull Request. Click on the “**{collaboration} branch**” dropdown control and select “**Create pull request...**” from the resulting dropdown.

A third tab in your browser (pointing to <https://dev.azure.com...>) will open.

The screenshot shows the 'New pull request' interface in Azure DevOps. On the left, there's a sidebar with project navigation links like Overview, Boards, Repos, Files, Commits, Pushes, Branches, Tags, Pipelines, Test Plans, Artifacts, and Compliance. The 'Pull requests' link is highlighted. The main area shows a 'New pull request' form. At the top, it says 'collaboration_synapse' into 'master'. Below that, the 'Overview' tab is selected, showing 'Files 6' and 'Commits 4'. The main form has fields for 'Title' (containing 'My Pull Request'), 'Description' (with placeholder 'Describe the code that is being reviewed'), 'Reviewers' (with placeholder 'Search users and groups to add as reviewers'), 'Work items to link' (with placeholder 'Search work items by ID or title'), and 'Tags' (an empty field). A 'Create' button is at the bottom right. The browser address bar shows the URL for creating a pull request.

On the second “**New pull request**” page, enter values for the following items:

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)

Enter values for items like **Reviewers** and “**Work items...**” as appropriate.

No additional changes are required. Review settings on the remaining tabs.

Click the **Create** button.

The screenshot shows the 'My Pull Request' page in Azure DevOps. On the left, the 'Pull requests' menu item is selected. The main area displays a pull request from 'Rich Chapler' titled 'collaboration_synapse into master'. The 'Reviewers' section indicates 'Required' status with 'No required reviewers' listed. The 'Optional' section shows 'No optional reviewers'. Below this is a 'Description' input field and a 'Tags' section with 'No tags'. The 'Work items' section shows 'No work items'. A comment from 'Rich Chapler' is visible, stating 'Rich Chapler created the pull request' at 'Just now'. An 'Add a comment...' input field is also present.

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 pull request' dialog open. The dialog includes fields for 'Merge type' (set to 'Merge (no fast forward)'), a merge diagram, and 'Post-completion options' with checkboxes for '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**” popout.

Click the “**Complete merge**” button.

The screenshot shows the Azure DevOps interface for a pull request. On the left, there's a sidebar with 'practicumadop' selected under 'Repos'. The main area is titled 'My Pull Request' and shows a completed merge from 'collaboration_synapse' into 'master'. It includes tabs for 'Overview', 'Files', 'Updates', and 'Commits'. Below the title, it says 'Rich Chapler completed this pull request Just now' with 'Cherry-pick' and 'Revert' buttons. A message box shows 'Merged PR 2: My Pull Request' with commit hash 'edae84b1' and 'Rich Chapler Just now'. It also says 'No merge conflicts Last checked Just now'. To the right, there are sections for 'Reviewers' (Required: No required reviewers), 'Optional' (No optional reviewers), 'Tags' (No tags), and 'Work items' (No work items). At the bottom, there's a comment input field 'Add a comment...' and a log entry '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.

Objective Complete... congratulations!

Schema Comparison

Follow these instructions to **compare two database schemas**.

Step 1: Understand Use Case

Work with your customer to understand requirements; example notes:

- “We need to articulate changes to the database schema in the development environment (as compared to the database schema in the production environment)”

Step 2: Instantiate Prerequisites

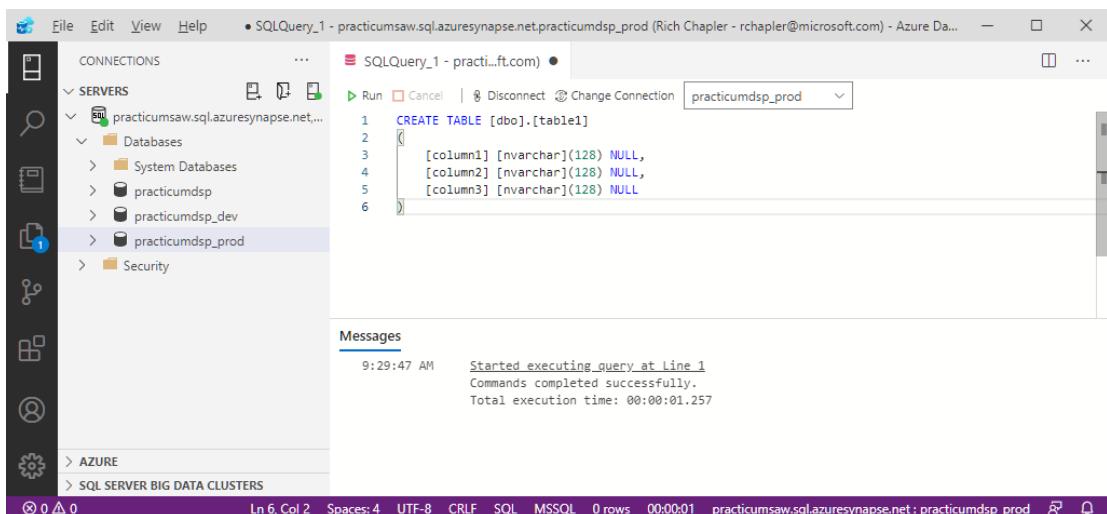
This exercise requires the following resource(s):

- Data Studio
- Synapse (with two SQL pools named **practicumdsp_dev** and **practicumdsp_prod**)

Although this example focuses on Synapse, the same solution applies to SQL.

Step 3: Stage Resources

Open Azure Data Studio.



A screenshot of the Azure Data Studio interface. The left sidebar shows connections and the current connection is 'practicumsaw.sql.azuresynapse.net.practicum dsp_prod'. The main pane displays a T-SQL script for creating a table:

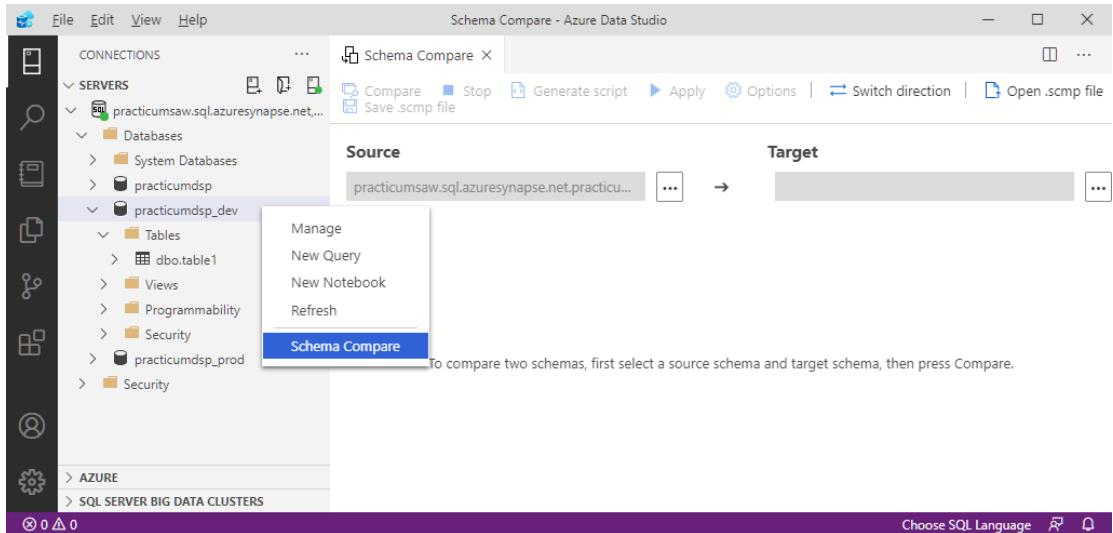
```
CREATE TABLE [dbo].[table1]
(
    [column1] [nvarchar](128) NULL,
    [column2] [nvarchar](128) NULL,
    [column3] [nvarchar](128) NULL
)
```

The status bar at the bottom indicates the command was started at 9:29:47 AM, completed successfully, and took 00:00:01.257. The status bar also shows the connection details: practicumsaw.sql.azuresynapse.net : practicum dsp_prod.

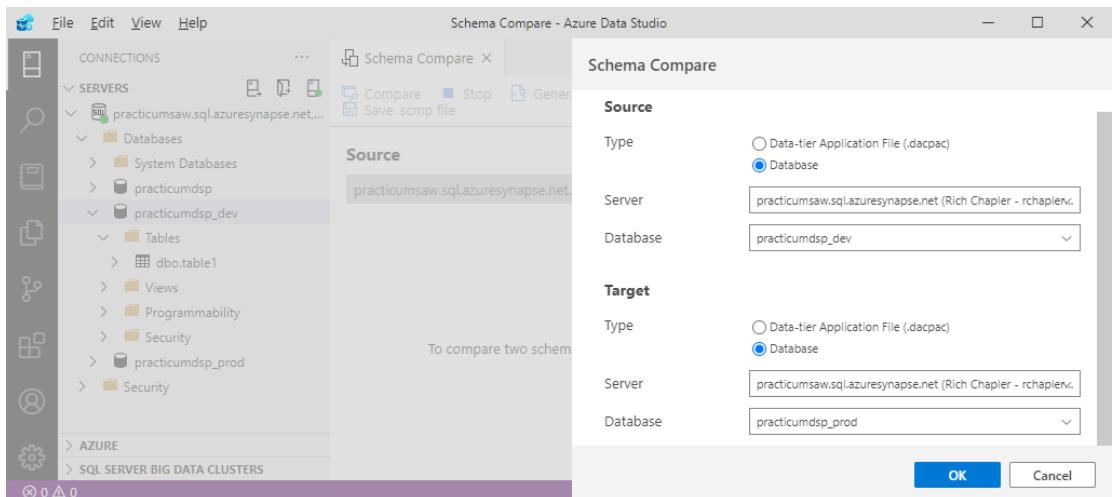
Execute the following T-SQL on **practicum dsp_dev**:

```
CREATE TABLE [dbo].[table1]
(
    [column1] [nvarchar](128) NULL,
    [column2] [nvarchar](128) NULL,
    [column3] [nvarchar](128) NULL
)
```

Step 4: Schema Compare



Right-click on the SQL Pool and click “Schema Compare” in the resulting dropdown.



Click the ellipses button.

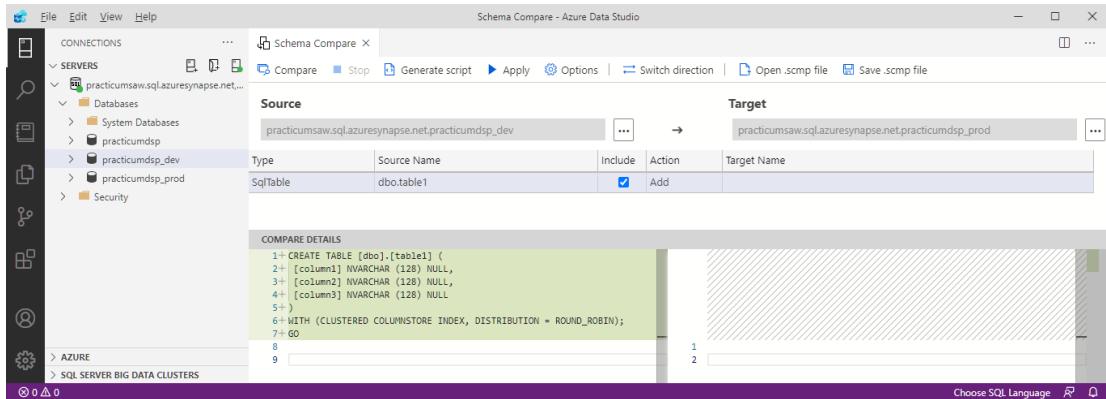
On the “Schema Compare” popout, enter values for the following items:

Source Type	Select the Database radio button
Source Server and Database	Confirm selection of your Synapse server and the practicum dsp_dev database
Target Type	Select the Database radio button

Target | Server and Database

Confirm selection of your Synapse server and the **practicumdsp_prod** database

Click the **OK** button.

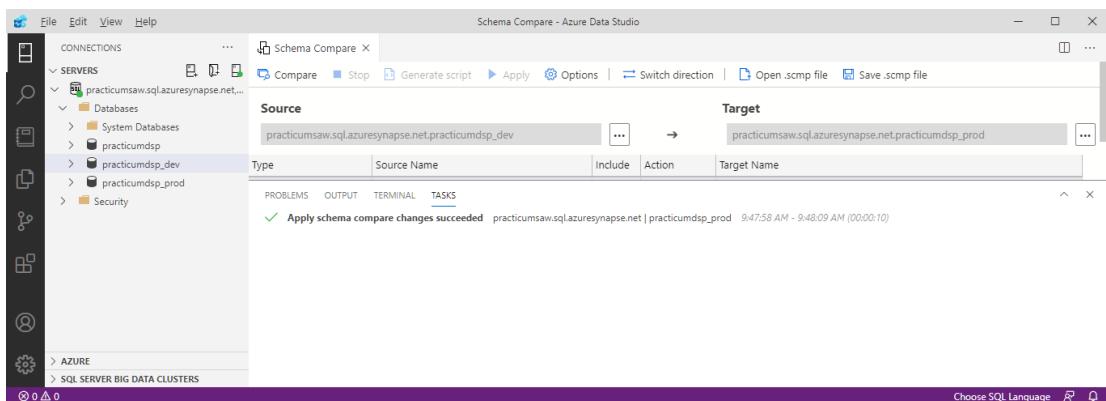


On the “Schema Compare” tab, click the **Compare** button.

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.



Objective Complete... congratulations!

Govern

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.

Step 1: Instantiate Prerequisites

This exercise requires the following resource(s):

- Purview
- SQL

Step 2: 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”** popout.

Click to check the **“Azure SQL Database”** option and then click the **Continue** button.

Microsoft Azure | Purview > practicum

Sources

Register sources (Azure SQL Database)

Name *

Azure subscription

Server name *

Endpoint

Select a collection *

All assets under this source will belong to the collection you select.

Register Back Cancel

On the “**Register sources (Azure SQL Database)**” popout, enter values for the following items:

Name	Enter a meaningful name aligned with standards
Subscription	Select your subscription
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.

Step 3: Scan Data

Navigate to **Sources**, “**Map view**”.

The screenshot shows the Microsoft Azure Purview Studio interface. On the left, there's a navigation sidebar with icons for Sources, Collections, Source management, Scan rule sets, Integration runtimes, Annotation management, Classifications, and Classification rules. The main area is titled 'Sources' and shows a list of registered sources. One item, 'practicump', is expanded to show its contents. Inside 'practicump', there is a single item named 'practicumsd' which is identified as an 'Azure SQL Database'. Below this item are edit, refresh, and delete icons, along with a 'View details' button. To the right of the main list, there's a small preview window showing a hierarchical tree structure.

Click the “New scan” icon on the newly created item.

The screenshot shows the 'Scan "practicumsd"' dialog box. The 'Name' field contains 'Scan-20210910'. The 'Connect via integration runtime' dropdown is set to 'Azure AutoResolveIntegrationRuntime'. The 'Server endpoint' field shows 'practicumsd.database.windows.net'. Under 'Database selection method', the 'From Azure subscription' radio button is selected. In the 'Database name' field, 'practicumsds' is entered. The 'Credential' dropdown is set to 'practicumsds'. The 'Select a collection' dropdown is set to 'practicump'. A note at the bottom states: 'All assets scanned will be included in the collection you select.' At the bottom right are 'Continue', 'Test connection', and 'Cancel' buttons.

On the “Scan...” popout, enter values for the following items:

Name	Enter a meaningful name aligned with standards
Database Selection Method	Select the “From Azure subscription” radio button
Database Name	Select your Azure SQL database
Credential	Select your “SQL Authentication” credential

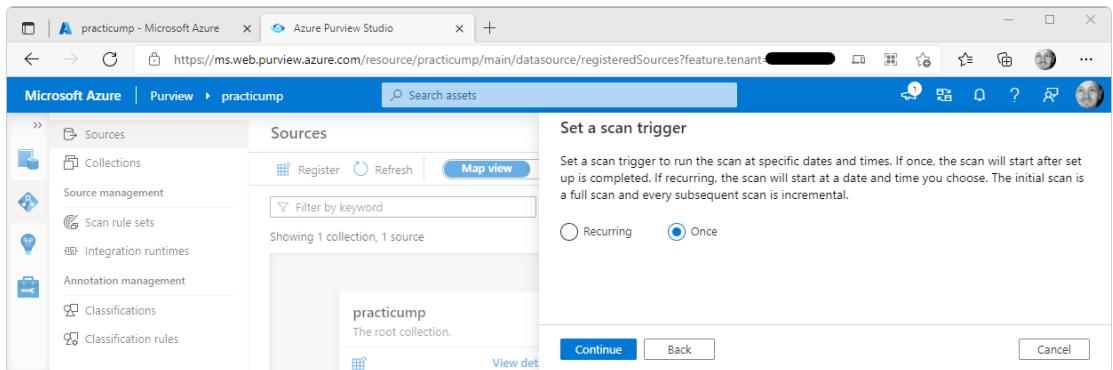
Click “**Test connection**” to confirm successful configuration and then click the **Continue** button.

The screenshot shows the Azure Purview Studio interface. On the left, a sidebar menu includes Sources, Collections, Source management, Scan rule sets, Integration runtimes, Annotation management, Classifications, and Classification rules. The main area displays a 'Sources' list with one collection named 'practicump'. Below it is an 'Azure SQL Database' named 'practicumsd'. A 'Scope your scan' dialog is open on the right, containing a search bar and a list of selected items under a tree view. The selected items include various tables and views from the 'SalesLT' schema, such as 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**” popout and then click the **Continue** button.

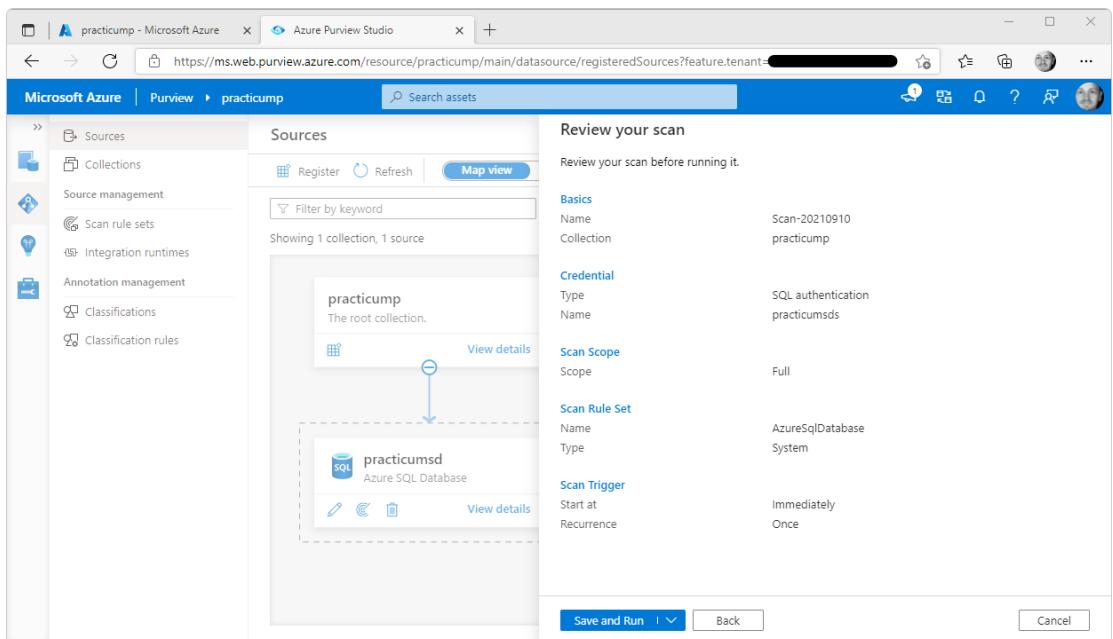
The screenshot shows the Azure Purview Studio interface. The left sidebar and sources list are identical to the previous screenshot. A 'Select a scan rule set' dialog is open on the right. It contains a 'New scan rule set' button, a 'Refresh' button, and a list titled 'Select one scan rule set to be used by your scan.' A single item is listed: 'AzureSqlDatabase [SYSTEM DEFAULT]'. This item is described as a Microsoft default scan rule set that includes all supported system classification rules. Below the description is a 'View detail' link. At the bottom of the dialog are 'Continue', 'Back', and 'Cancel' buttons.

Confirm selection of the default Scan Rule Set on the “Select a scan rule set” popout and then click the **Continue** button.



The screenshot shows the 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 is titled 'Sources' and shows a single collection named 'practicump'. A 'Map view' button is visible. To the right, a 'Set a scan trigger' dialog is open. It contains a description of what a scan trigger does, two radio buttons ('Recurring' and 'Once'), and a 'Continue' button at the bottom. The 'Once' radio button is selected.

Click the **Once** radio button on the “Set a scan trigger” popout and then click the **Continue** button.



The screenshot shows the 'Review your scan' dialog. On the left, the same sidebar and 'Sources' list are visible. The main area shows the 'practicump' collection and a detailed view of a database named 'practicumsd'. On the right, the 'Review your scan' section displays various settings:

- Basics:** Name: Scan-20210910, Collection: practicump
- Credential:** Type: SQL authentication, Name: practicumsds
- Scan Scope:** Scope: Full
- Scan Rule Set:** Name: AzureSqlDatabase, Type: System
- Scan Trigger:** Start at: Immediately, Recurrence: Once

At the bottom, there are 'Save and Run' and 'Cancel' buttons.

Review scan settings and then click the “Save and Run” button.

Confirm Scan

Navigate to Sources, “Table view”.

Azure Purview Studio

Sources

Name	Source type	Collection	Scans	Registered on
AzureSqlDatabase-P3k	Azure SQL Database	practicump	1	09/10/21 01:59 PM

Note that the **Scans** column now shows 1.

Click on the **Name** link to drill through to details. Allow time for processing.

Data map > Sources >

practicumsd

New scan Edit source Delete source Refresh

Overview Scans

Source ID: practicumsd.database.windows.net Registered on 09/11/2021 03:18:28 AM

Scans 1 17

Scanned assets 17

Classified assets 15

Recent scans

Scan name	Last run status	Scan rule set	Last scan time
Scan-20210910	In progress	AzureSqlDatabase	09/11/21 03:25 AM

→ See all applied scans

Recent failed scans

Scan name	Status	Suggestions
No recent failed scan runs		

→ See all failed scan runs

Source hierarchy

```
rchapter
  Subscription
  |
  practicumrg
    Resource group
```

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.

practicump

1 source No assets No glossary terms

Search catalog

Browse assets Explore assets by source type and collection.

Manage glossary Create and edit terms with custom term templates.

Knowledge center Discover learning materials and tutorials.

Click on the “**Browse assets**” button.

Name	Description	Assets	Collection admin
practicump	The root collection.	4	Rich Chapler

Click on the Root Collection link {i.e., name}.

The screenshot shows the Microsoft Azure Purview Studio interface. The left sidebar displays the 'Data catalog' navigation path: Microsoft Azure | Purview | practicum. Below this, there's a 'Browse assets' section with a 'Refresh' button and filters for 'By collection' and 'By source type'. A 'View collection tree' link is also present. The main content area shows a list of assets under the 'practicump' collection, with 23 results found. The results are sorted by Name. The first item listed is 'Address', which is an Azure SQL Table located at mssql://practicumsd.database.windows.net/practicumsds/SalesLT/Address. Other items listed include 'BuildVersion', 'Customer', 'CustomerAddress', and 'dbo'. At the bottom, there are navigation controls for 'Previous', 'Page 1 of 1', and 'Next'.

Explore results.

Click on the first item, **Address**.

The screenshot shows the Azure Purview Studio interface for the 'Address' asset. The top navigation bar includes 'practicump - Microsoft Azure' and 'Azure Purview Studio'. The main content area displays the 'Address' asset, which is an 'Azure SQL Table'. Below the asset name are buttons for 'Edit', 'Refresh', 'Delete', and 'Open in Power BI Desktop'. The asset has tabs for 'Overview' (selected), 'Properties', 'Schema', 'Lineage', 'Contacts', and 'Related'. The 'Overview' tab shows the asset was last updated on September 11, 2021, at 10:29 AM UTC by an automated scan. The 'Collection path' is listed as 'practicump'. The 'Hierarchy' section shows the database structure: 'practicumsd.database.windows.net' (Azure SQL Server) contains 'practicumsd' (Azure SQL Database), which contains 'SalesLT' (Azure SQL Schema), which contains 'Address' (Azure SQL Table). The 'Glossary terms' section indicates there are no glossary terms for this asset.

Notice the automatic “**World Cities**” classification.

Drill-through the tabs {e.g., Properties, Schema, Lineage, etc.}.

Step 4: 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”}.

A screenshot of the Azure Purview Studio interface. The top navigation bar shows 'practicump - Microsoft Azure' and 'Azure Purview Studio'. The main search bar contains the query 'address'. The left sidebar has a 'Data catalog' section with a 'Collection' filter set to 'practicump' (2 results) and other filters for 'Classification', 'Contact', 'Label', and 'Glossary term'. The right pane displays search results for 'address', showing two entries: 'Address' (Azure SQL Table, mssql://practicumsd.database.windows.net/practicumsds/SalesLT/Address) and 'CustomerAddress' (Azure SQL Table, mssql://practicumsd.database.windows.net/practicumsds/SalesLT/CustomerAddress). The results are sorted by Relevance, page 1 of 1.

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.

Address
Azure SQL Table

1 of 2

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/practicumsds/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.

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 Azure Purview Studio interface. The left sidebar has icons for Data Catalog, Purview, and practicum. The main area shows a search result for "Address". The "Related" tab is selected, displaying a list of 13 items under the heading "Showing 1 to 13 of 13 items". The columns are Name, Type, Owner, and Action. The items listed are: 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), and SalesOrderHeader (Azure SQL Table). A blue button at the top right says "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 the Azure Purview Studio interface. The left sidebar has icons for Data Catalog, Purview, and practicum. The main area shows a search result for "Address". The "Contacts" tab is selected, displaying sections for "Experts" and "Owners". Under "Experts", it says "No experts for this asset.". Under "Owners", it says "No owners for this asset.". A blue button at the top right says "Open in Power BI Desktop".

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 snip above, there are no Contacts and you can add information not already captured, click the **Edit** button.

The screenshot shows the 'Edit "Address"' page in Azure Purview Studio. The 'Contacts' tab is selected. A table lists a contact named Rich Chapler with the role 'Curator'. At the bottom are 'Save' and 'Cancel' buttons.

In either the **Experts** or the **Owners** dropdown, search for the user or group in your directory. Click **Save**.

Objective Complete... congratulations!

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.

Step 1: Instantiate Prerequisites

This exercise requires the following resource(s):

- Purview
- SQL

Step 2: 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 popout, confirm selection of “**System default**” and then click the **Continue** button.

The screenshot shows the Microsoft Azure Purview Studio interface for creating a new glossary term. The URL is https://ms.web.purview.azure.com/resource/practicump/main/catalog/term/new?feature.tenant=[REDACTED]. The page has a blue header with 'Microsoft Azure' and 'Purview' tabs, and a search bar. On the left, there's a sidebar with icons for Data catalog, Glossary terms, and other Purview features. The main content area is titled 'New term' and contains the following fields:

- Name**: 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 are 'Create' and 'Cancel' buttons.

On the “New term” page, enter values for the following items:

Name	Enter a meaningful name aligned with standards
Definition	Select your subscription
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.

Step 3: 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 URL in the address bar is <https://ms.web.purview.azure.com/resource/practicump/main/catalog/entity?guid=288bec77-3c71-41f7-bf47-09f6f6f60006§ion=entity>. The page shows a single asset named "ProductNumber" which is an "Azure SQL Column". Below the asset name are buttons for "Edit", "Refresh", and "Delete". The "Overview" tab is selected. To the right, it says "Updated on September 11, 2021 10:29 AM UTC". On the left sidebar, there are icons for Data catalog, Purview, and practicump. The main content area includes sections for Asset description, Description, Classifications, Fully qualified name, Collection path, Hierarchy, Glossary terms, and Related.

Click the **Edit** button.

A screenshot of the Azure Purview Studio interface showing the "Edit 'ProductNumber'" screen. The URL in the address bar is <https://ms.web.purview.azure.com/resource/practicump/main/catalog/entity/edit?guid=288bec77-3c71-41f7-bf47-09f6f6f60006§ion=edit>. The page title is "Edit 'ProductNumber'". The "Overview" tab is selected. The "Name" field contains "ProductNumber". The "Asset description" field is empty. The "Classifications" dropdown shows "Select...". The "Glossary terms" dropdown shows "Product Number" with a checked checkbox. At the bottom are "Save" and "Cancel" buttons.

Select the new term, "**Product Number**" from the "**Glossary terms**" dropdown.

Click the **Save** button.

Step 4: 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. The URL is https://ms.web.purview.azure.com/resource/practicump/main/catalog/term?section=overview&termGuid=52836041-66f9-43d5-8... . The search bar contains 'product'. The main content area shows a term named 'Product Number' with the following details:

- Term Type:** System default
- Status:** Draft
- Overview** tab is selected.
- Formal name:** Product Number
- Parent:** No parent term for this term
- Last updated:** 09/13/2021 07:37 by Rich Chapler
- Definition:** The stock-keeping unit (SKU) that uniquely identifies a product.
- Acronym:** SKU
- Catalog assets:** 1 asset associated with "Product Number". View asset.
- Resources:** 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. The URL is https://ms.web.purview.azure.com/resource/practicump/main/catalog/entity/edit?guid=81afae99-9619-4559-aa95-b5f6f6f60000&... . The search bar contains 'address'. The main content area shows the schema for an entity named 'Address':

Edit "Address"

Schema tab is selected.

A note says: Making a manual update to the schema will prevent future scans on this asset from updating it.

Column name	Glossary terms	Data type	Asset description
AddressID	Select...	int	
AddressLine1	Select...	nvarchar	
AddressLine2	Select...	nvarchar	
City	Select...	nvarchar	
CountryRegion	Select...	nvarchar	

Use “**Column level classification**” options to characterize all country- / region-type schema elements as a standard classification.

Objective Complete... congratulations!

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.

Step 1: Instantiate Prerequisites

This exercise requires the following resource(s):

- Data Factory
- Data Share
- Purview
- SQL

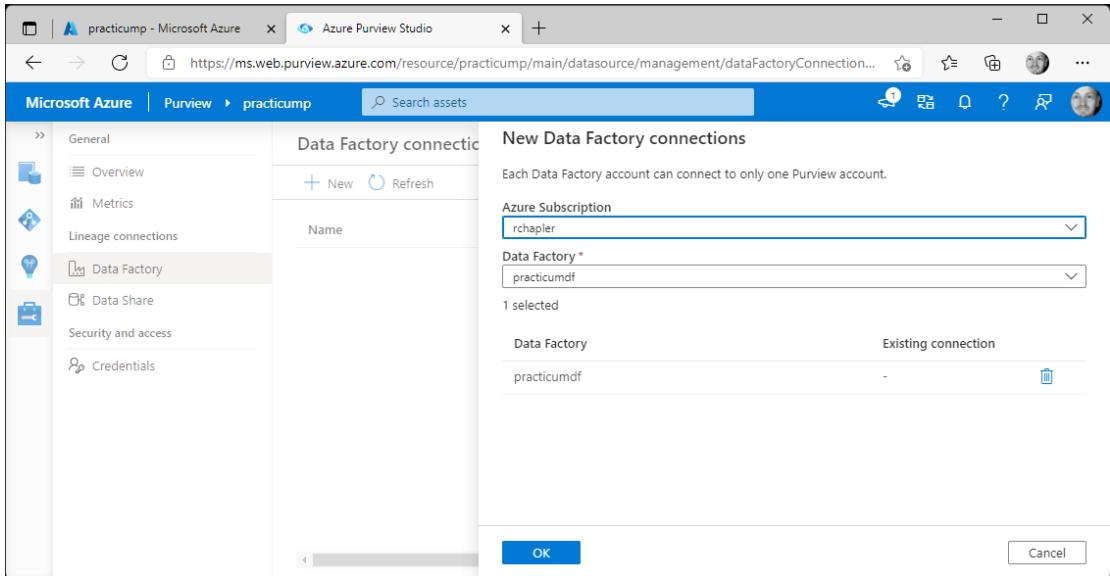
Step 2: Configure Data Factory

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.



On the “**New Data Factory connections**” popout, enter values for the following items:

Azure Subscription

Select your subscription

Data Factory

Select data factory

Click the **OK** button. Confirm connection.

Confirm Success

Purview will automatically detect data movement and capture data about a pipeline when it is executed in a connected 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 lists 'Pipeline' (1 item), 'Dataset' (2 items), 'Data flows' (0), 'Power Query (Preview)' (0), and 'Templates' (0). The main workspace is titled 'pipeline1' and shows a single activity: 'Copy data' with a sub-item 'Copy data1'. Below the activities, there are tabs for 'Parameters', 'Variables', 'Settings', and 'Output'. The 'Output' tab is selected, showing a table with one row: 'Copy data1' (Name), 'Copy data' (Type), '2021-09-15T15:00:19Z' (Run start), '00:01:19' (Duration), and 'Succeeded' (Status). The pipeline run ID is ee386add-a37b-4156-83d0-0e549901c97d.

After successful execution, navigate to Purview.

The screenshot shows the Microsoft Azure Purview Studio interface. The top navigation bar includes 'Microsoft Azure', 'Purview', and the resource name 'practicump'. The main area features a 3D data flow visualization with a central cube labeled 'practicump'. Below the visualization, a search bar contains the text 'pipeline1'. A summary at the bottom left indicates '1 source', '29 assets', and '1 glossary term'. Three cards are displayed at the bottom: '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).

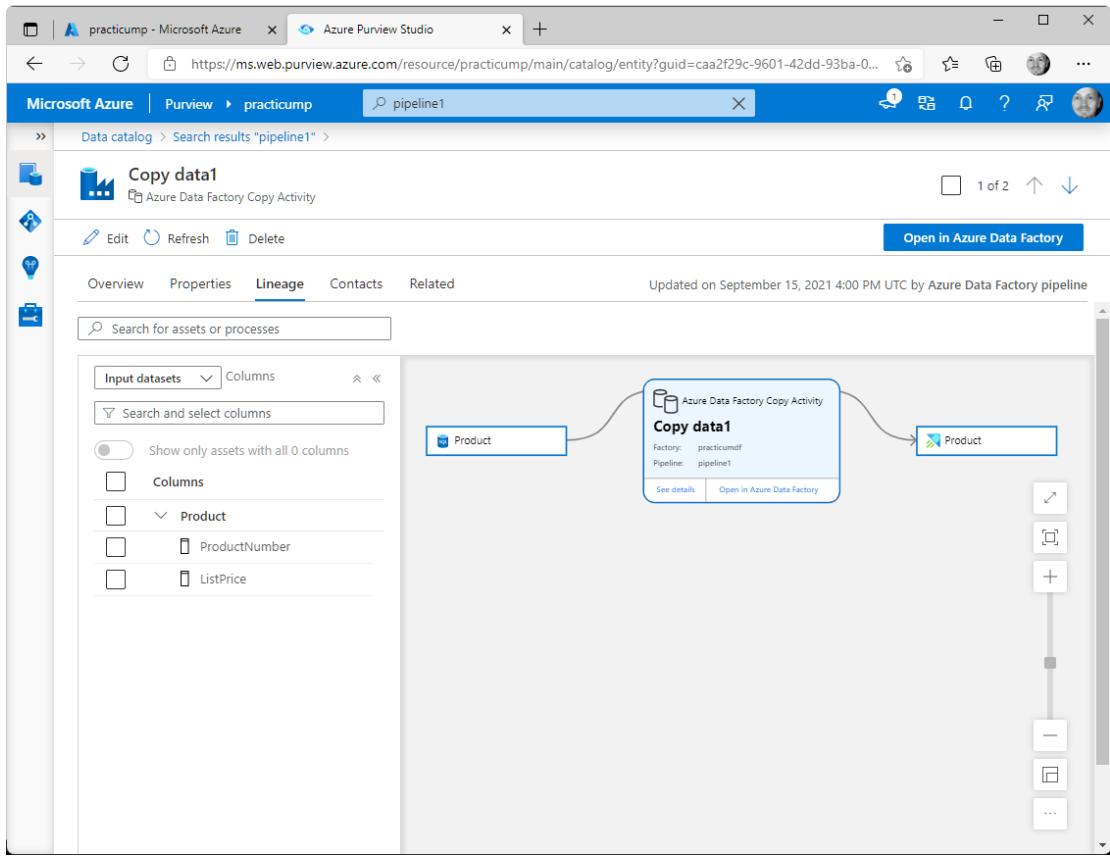
Search for “pipeline1”.

The screenshot shows the Azure Purview Studio interface with a search query "pipeline1". The results page displays two items: "Copy data1" and "pipeline1".

Result Type	Name	Description
Azure Data Factory Copy Activity	Copy data1	Azure Data Factory Copy Activity
Azure Data Factory Pipeline	pipeline1	Azure Data Factory Pipeline

Click on the “Copy data1” result.

Click on the Lineage tab.



Continue to drill-through elements to better understand the capture of Lineage data.

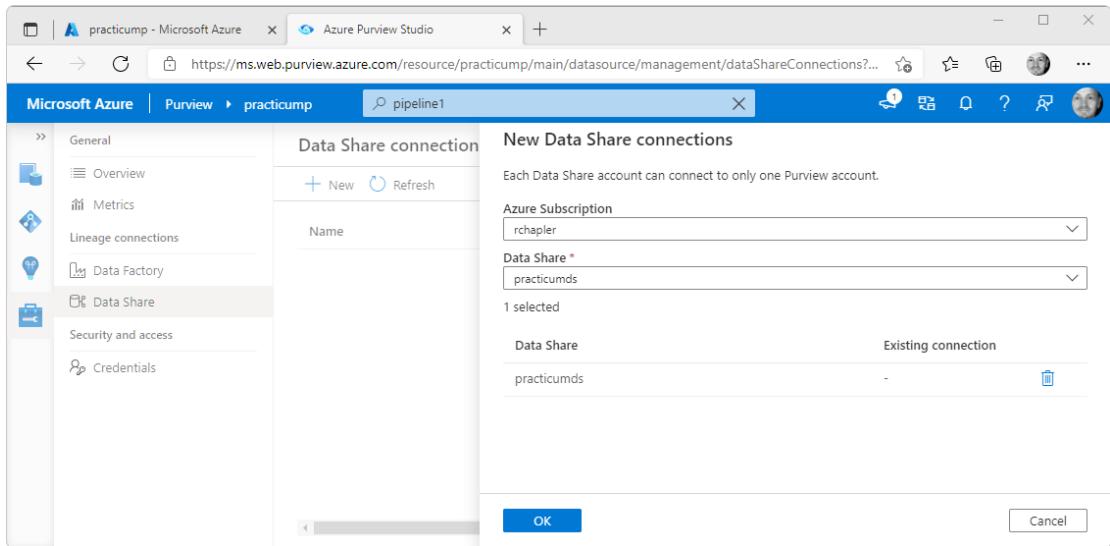
Step 3: Configure Data Share

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.



On the “**New Data Factory connections**” popout, enter values for the following items:

Azure Subscription Select your subscription

Data Share Select data share

Click the **OK** button.

Article: [Connect to Azure Data Share - Azure Purview | Microsoft Docs](#)

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”

Objective Complete... congratulations!

Quick Answers

This section provides quick answers to questions that simply do not merit an extended write-up.

Subscription Identifier

Verbose: “What is my Subscription Identifier?”

Enter the following command in the Cloud Shell:

```
Get-AzADServicePrincipal -DisplayName {Subscription Name}
```

Tenant Identifier

Verbose: “What is my Tenant Identifier?”

Enter the following command in the Cloud Shell:

```
Get-AzADServicePrincipal -DisplayName {Subscription Name}
```