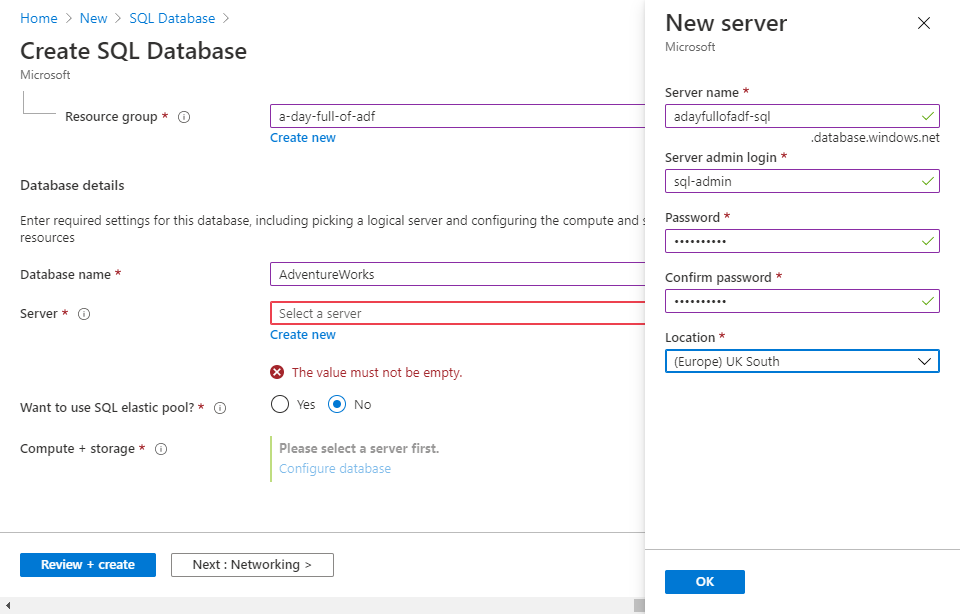
Lab 2 – Build a Copy data pipeline

In this lab you will create a Copy data pipeline to copy data from a source SQL database into your data lake.

# Lab 2.1 – Create a source SQL database

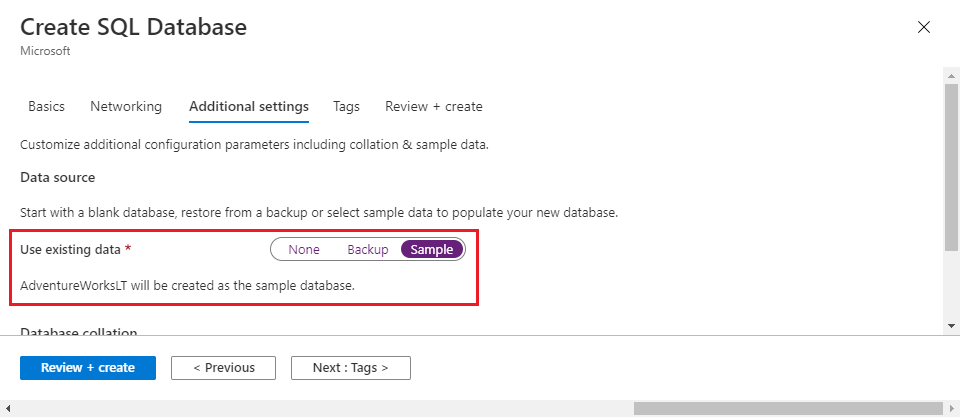
For this lab, you’ll be copying data from AdventureWorks OLTP database. Start by creating a copy in your resource group – this will play the role of a source business system for ADF data processing.

1. In the Azure portal, click “Create a resource” and search for “SQL Database”. Click “Create” on the overview screen.
2. Complete the **Basics** tab like this:
   * Choose your subscription and the resource group you created in Lab 1.1.
   * Enter a name for your database
   * Under the “Server” dropdown, click “Create new”
   * On the “New server” blade, provide a globally-unique server name, an admin login name and an admin password. **Make a note of the login and password**. Choose the same location you specified for your resource group, then click “OK”

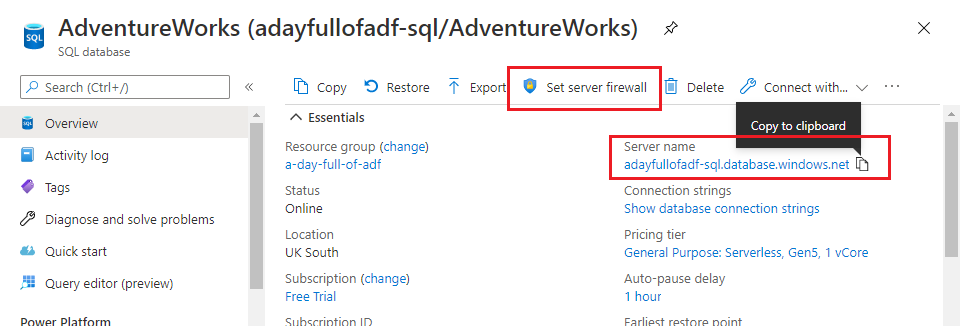


* + Under “Compute + storage”, click “Configure database”, then below “Compute tier” click the “Serverless” tile. Serverless SQL services are automatically paused after a period of inactivity (by default 1 hour) – this is rarely appropriate for production environments but is fine for lab work, and a lot cheaper! Click “Apply”.

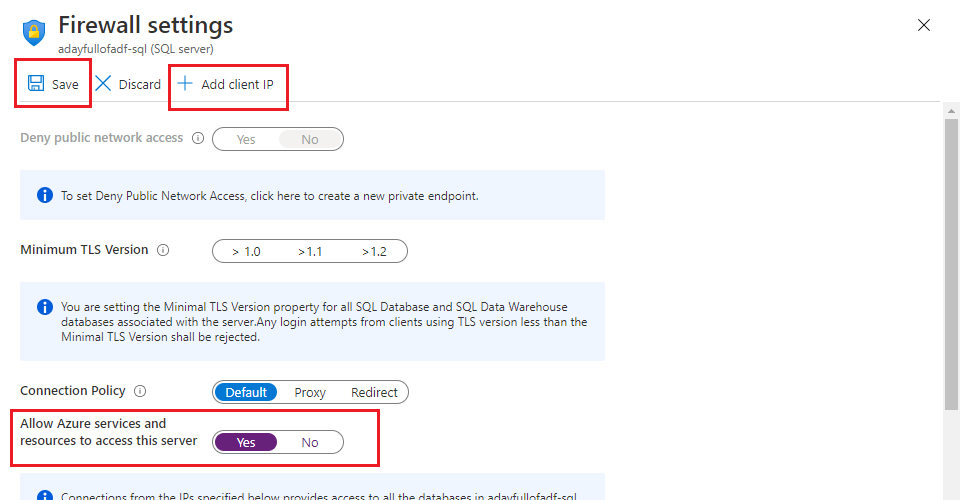
1. Skip forward past the Networking tab to the **Additional settings** tab. Under “Data source”, change the “Use existing data” option to “Sample” – the message “AdventureWorksLT will be created as the sample database” is displayed.



1. Click “Review + create”, then “Create”. When the server and database have finished deploying (this may take a few minutes), click on “Go to resource”.
2. On the database blade you can find the SQL server name which you will need later. Click the menu bar’s “Set Server firewall” button.



1. On the “Firewall settings” blade:
   * Click “+ Add client IP” in the menu bar to add your computer’s IP address to the server firewall.
   * Set “Allow Azure services and resources to access this server” to **Yes**, to allow Azure Data Factory (and other Azure services) through the firewall.
   * Click “Save”.



You will now be able to connect to the new database using SSMS or your preferred SQL client. Connect using the server name given on the database blade; choose “SQL Server Authentication” and use the admin username and password you created for the server.

# Lab 2.2 – Create linked service and datasets

To recap, you have created:

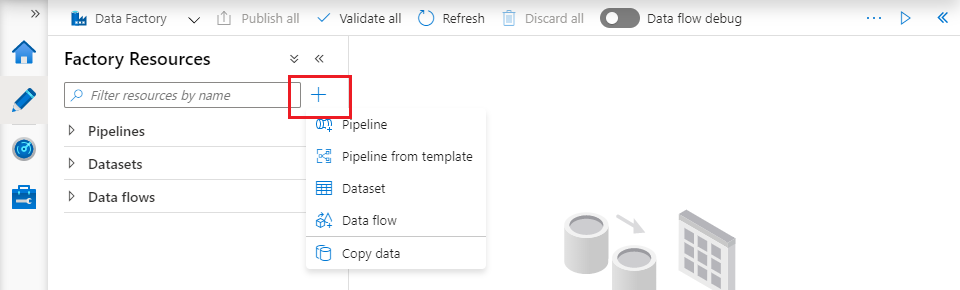
* A SQL database to provide a data source
* A data lake to act as your data sink (Azure Data Factory refers to copy targets/destinations as “sinks”)
* An instance of Azure Data Factory to copy data from source to sink

You’ve already created a linked service connection to your data lake in ADF – now you need to create one for the SQL database.

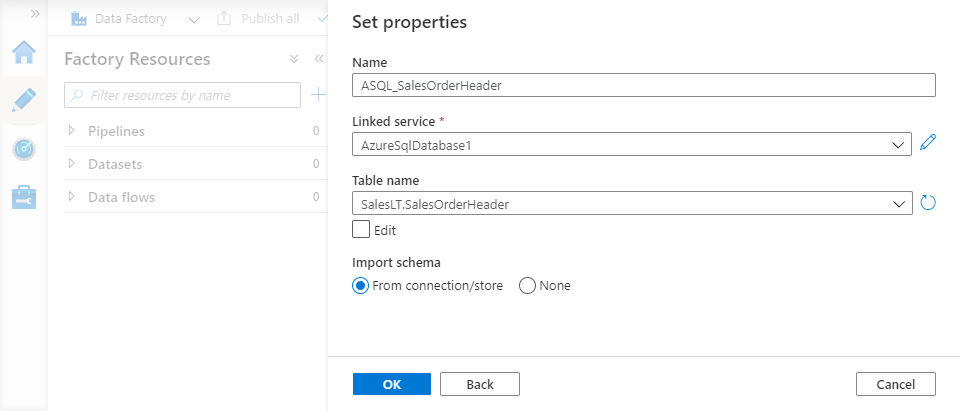
1. In the ADF UX, open the Management Hub and select “Linked services” from the “Connections” section of its sidebar.
2. In the main Linked services pane click “+ New”.
3. Search for “SQL”, then choose “Azure SQL Database” and click “Continue”.
4. Configure linked service details on the “New linked service (Azure SQL Database)” blade:
   * Give it a name.
   * Choose the Azure subscription containing your lab resources.
   * Choose your SQL Server from the “Server name” dropdown, then your new database from the “Database name” dropdown.
   * For “Authentication type”, choose “SQL authentication”, then enter the admin login name and password you noted down in lab 2.1
5. Click “Test connection” at the bottom of the blade. If you have configured your SQL Server and linked service correctly, you will receive a “Connection successful” message.
6. Click “Create” to create the linked service. The linked service is published automatically this time, so that the database username and password aren’t stored in your ADF UX session (or in your Git repo if you’re using one).

Linked services represent connections to external systems – in this example a SQL database. Data stored by those services (in this case as database tables) is represented by an ADF dataset.

1. Open the ADF authoring canvas by clicking on the pencil button (on the far left, two icons above the Management Hub button).
2. In the “Factory resources” sidebar, click the “+” button to the right of “Filter resources by name”, then choose “Dataset”.



1. Search for “SQL”, then choose “Azure SQL Database” and click “Continue”.
2. Give the dataset a name, then choose your Azure SQL Database from the “Linked service” dropdown. Finally, choose a table from the “Table name” dropdown and click “OK”.

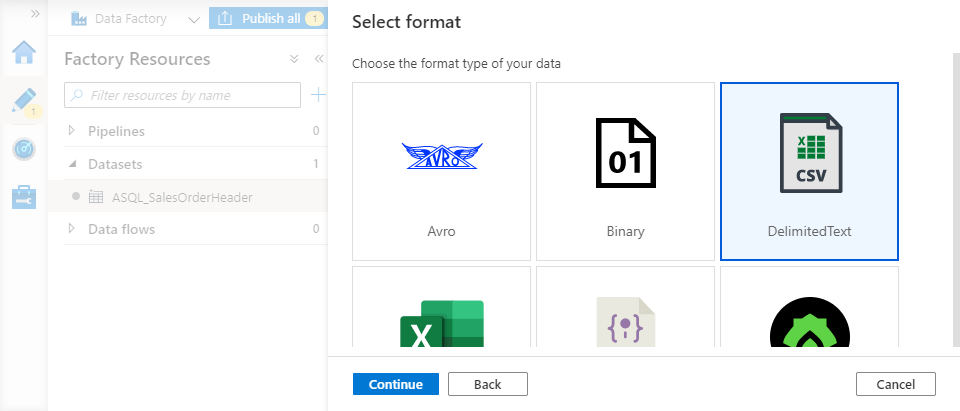


1. Save your changes:

* if you’re using Git, click “Save all”
* otherwise click “Publish all”.

You need a similar dataset to represent the sink file for the Copy data operation.

1. Use the “+” button to create a second dataset. This time search for “data lake”, then choose “Azure Data Lake Storage Gen2” and click “Continue”.
2. For file formats, you need also to specify a file type. Choose “DelimitedText” and click “Continue”.

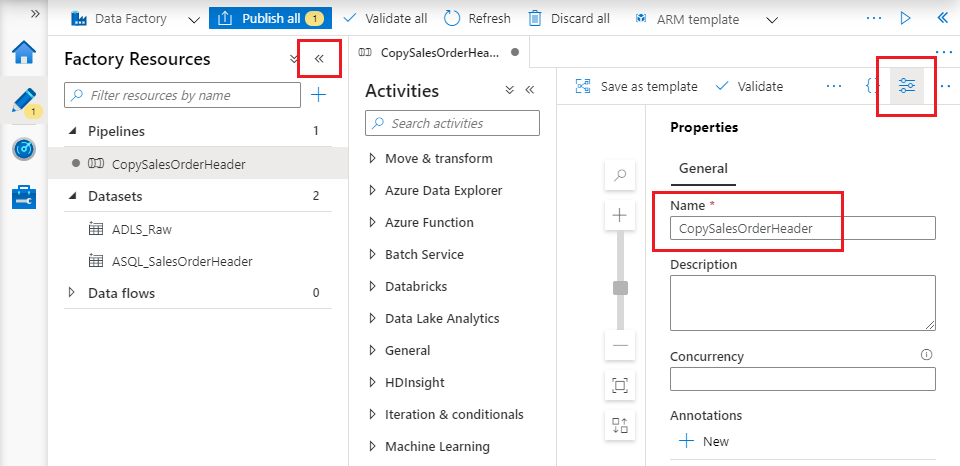


1. On the “Set properties” blade:
   * Give the dataset a name
   * Choose your Azure Data Lake Storage linked service
   * To the right of the three “File path” fields, click the folder icon and browse to the “lakeroot” container’s “Raw” directory. Click “OK”.
   * Ensure that the “First row as header” checkbox is **ticked**, then click “OK”.
2. Save your changes.

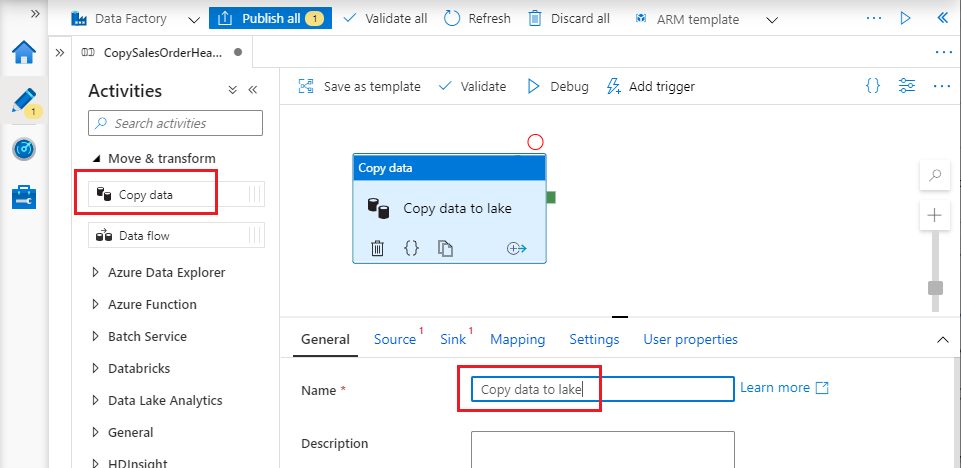
# Lab 2.3 – Create a data factory pipeline

Now that you have datasets able to represent source tables in Azure SQL DB and sink files in the data lake, you can create a pipeline to copy data from one to the other.

1. In the “Factory resources” sidebar, click the “+” button to the right of “Filter resources by name”, then choose “Pipeline”.
2. The pipeline properties blade is displayed automatically with a default name for the pipeline. Change it to something sensible, then dismiss the blade by clicking the slider button immediately above it. If you need more space, collapse the “Factory Resources” sidebar using the left chevron button.



1. Expand the “Move & transform” group in the activity toolbox (headed “Activities”), then drag a “Copy data” activity onto the pipeline canvas. On the “General” tab below the canvas, give the activity a sensible name.

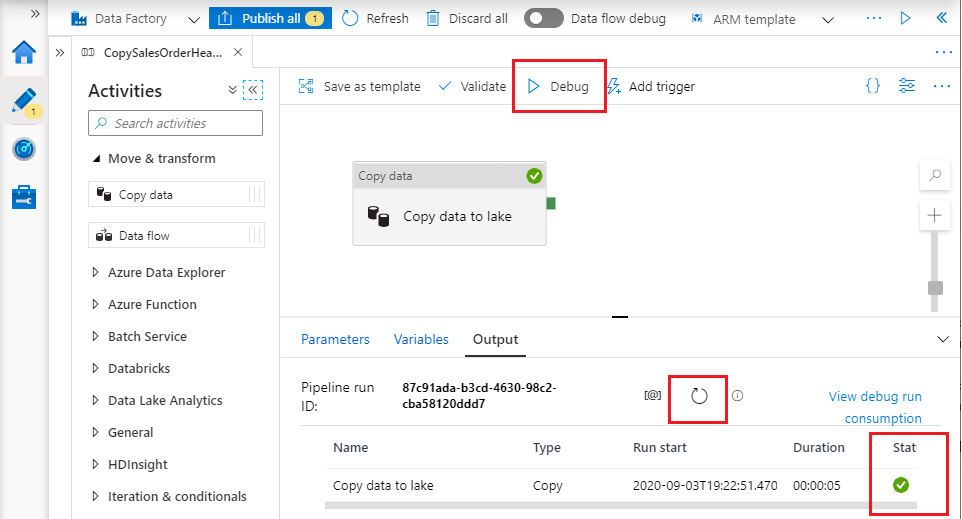


1. Select the “Source” tab below the canvas and select your Azure SQL DB dataset from the “Source dataset” dropdown.
2. Select the “Sink” tab and select your data lake storage dataset from the “Sink dataset” dropdown.
3. Finally, check your pipeline configuration by clicking the “Validate” button above the pipeline canvas.

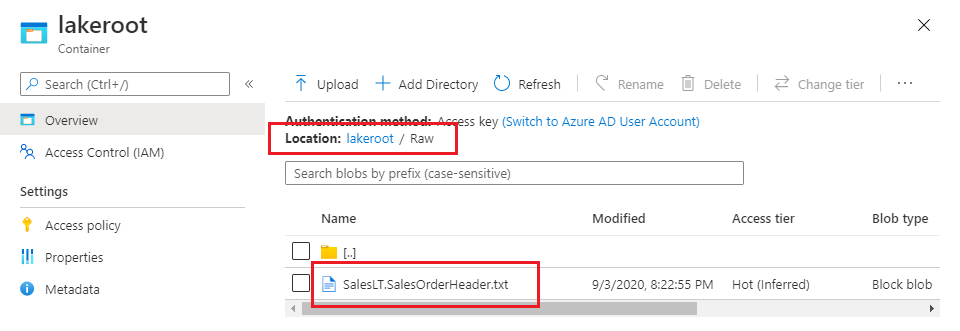
# Lab 2.4 – Test, publish and run the pipeline

You can test your pipeline by running it in “Debug” mode in your ADF UX session.

1. Click “Debug” above the pipeline canvas. The pipeline’s “Output” pane appears below the canvas.
2. The “Output” pane contains a row for each activity execution – in this case, just one for the Copy data activity. The activity execution shows its current status which you can update using the refresh button while the pipeline is running.



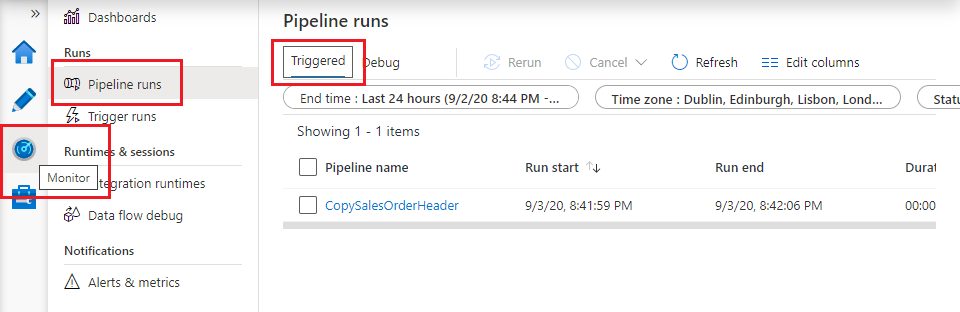
1. “Debug” runs your pipeline without publishing it to the data factory instance, but its effect is just the same – it has the same external dependencies, so has real effects on external resources. Open the “Raw” folder in the Azure portal and you will see the newly-copied file.



1. To be able to use a pipeline outside the ADF UX, it must be published:

* if you’re using Git, click “Save all”, then “Publish”
* otherwise click “Publish all”.

1. Your can run the published pipeline directly from the ADF UX by clicking “Add trigger” above the pipeline canvas and selecting “Trigger now”. A confirmation blade is displayed – click OK to trigger the published pipeline.
2. Published pipelines can be monitored in the ADF UX monitoring experience, accessed by clicking the Gauge icon in the leftmost sidebar. Choose the “Triggered” tab of the “Pipeline runs” page to see run details for published pipelines.



1. Refresh your view of the “Raw” folder in the Azure portal. The file appears as previously, but its “Modified” time has changed – running the published pipeline has overwritten it.