Get started with Azure Stream Analytics

In this exercise you'll provision an Azure Stream Analytics job in your Azure subscription, and use it to query and summarize a stream of real-time event data and store the results in Azure Storage.

This exercise should take approximately **15** minutes to complete.

Before you start

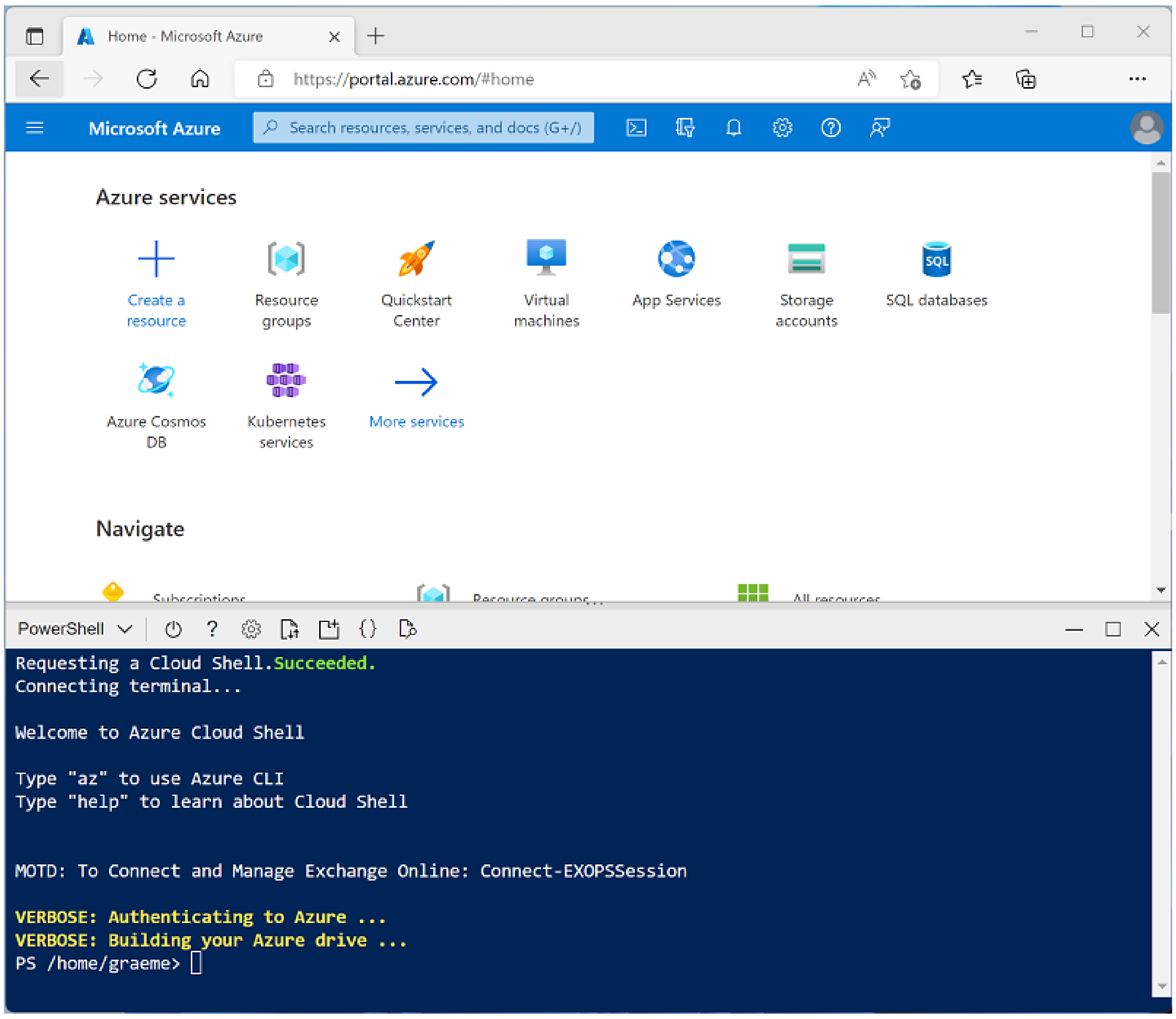
You'll need an Azure subscription in which you have administrative-level access.

Provision Azure resources

In this exercise, you'll capture a stream of simulated sales transaction data, process it, and store the results in a blob container in Azure Storage. You'll need an Azure Event Hubs namespace to which streaming data can be sent, and an Azure Storage account in which the results of stream processing will be stored.

You'll use a combination of a PowerShell script and an ARM template to provision these resources.

1. Sign into the Azure portal at `https://portal.azure.com`.
2. Use the **[>_]** button to the right of the search bar at the top of the page to create a new Cloud Shell in the Azure portal, selecting a ***PowerShell*** environment and creating storage if prompted. The cloud shell provides a command line interface in a pane at the bottom of the Azure portal, as shown here:



```
Requesting a Cloud Shell.Succeeded.
Connecting terminal...

Welcome to Azure Cloud Shell

Type "az" to use Azure CLI
Type "help" to learn about Cloud Shell


MOTD: To Connect and Manage Exchange Online: Connect-EXOPSSession

VERBOSE: Authenticating to Azure ...
VERBOSE: Building your Azure drive ...
PS /home/graeme> 
```

> **Note**: If you have previously created a cloud shell that uses a *Bash* environment, use the the drop-down menu at the top left of the cloud shell pane to change it to ***PowerShell***.

3. Note that you can resize the cloud shell by dragging the separator bar at the top of the pane, or by using the **—**, **◻**, and **X** icons at the top right of the pane to minimize, maximize, and close the pane. For more information about using the Azure Cloud Shell, see the Azure Cloud Shell documentation.

4. In the PowerShell pane, enter the following commands to clone the repo containing this exercise:

```
rm -r dp-203 -f
git clone https://github.com/MicrosoftLearning/dp-203-azure-data-engineer dp-203
```

5. After the repo has been cloned, enter the following commands to change to the folder for this exercise and run the **setup.ps1** script it contains:

```
cd dp-203/Allfiles/labs/17
./setup.ps1
```

6. If prompted, choose which subscription you want to use (this will only happen if you have access to multiple Azure subscriptions).
7. Wait for the script to complete - this typically takes around 5 minutes, but in some cases may take longer. While you are waiting, review the Welcome to Azure Stream Analytics article in the Azure Stream Analytics documentation.

View the streaming data source

Before creating an Azure Stream Analytics job to process real-time data, let's take a look at the data stream it will need to query.

1. When the setup script has finished running, resize or minimize the cloud shell pane so you can see the Azure portal (you'll return to the cloud shell later). Then in the Azure portal, go to the **dp203-*xxxxxxx*** resource group that it created, and notice that this resource group contains an Azure Storage account and an Event Hubs namespace.

   Note the **Location** where the resources have been provisioned - later, you'll create an Azure Stream Analytics job in the same location.

2. Re-open the cloud shell pane, and enter the following command to run a client app that sends 100 simulated orders to Azure Event Hubs:

```
node ~/dp-203/Allfiles/labs/17/orderclient
```

3. Observe the sales order data as it is sent - each order consists of a product ID and a quantity. The app will end after sending 1000 orders, which takes a minute or so.

Create an Azure Stream Analytics job

Now you're ready to create an Azure Stream Analytics job to process the sales transaction data as it arrives in the event hub.

1. In the Azure portal, on the **dp203-*xxxxxxx*** page, select **+ Create** and search for `Stream Analytics job`. Then create a **Stream Analytics job** with the following properties:

   - **Basics**:

     - **Subscription**: Your Azure subscription

        - **Resource group**: Select the existing **dp203-*xxxxxxx*** resource group.
        - **Name**: `process-orders`
        - **Region**: Select the region where your other Azure resources are provisioned.
        - **Hosting environment**: Cloud
        - **Streaming units**: 1
    - **Storage**:
        - **Secure private data in storage account**: Unselected
    - **Tags**:
        - *None*
2. Wait for deployment to complete and then go to the deployed Stream Analytics job resource.

Create an input for the event stream

Your Azure Stream Analytics job must get input data from the event hub where the sales orders are recorded.

1. On the **process-orders** overview page, select **Add input**. Then on the **Inputs** page, use the **Add stream input** menu to add an **Event Hub** input with the following properties:
    - **Input alias**: `orders`
    - **Select Event Hub from your subscriptions**: Selected
    - **Subscription**: Your Azure subscription
    - **Event Hub namespace**: Select the **events*xxxxxxx*** Event Hubs namespace
    - **Event Hub name**: Select the existing **eventhub*xxxxxxx*** event hub.
    - **Event Hub consumer group**: Select the existing **$Default** consumer group
    - **Authentication mode**: Create system assigned managed identity
    - **Partition key**: *Leave blank*
    - **Event serialization format**: JSON
    - **Encoding**: UTF-8
2. Save the input and wait while it is created. You will see several notifications. Wait for a **Successful connection test** notification.

Create an output for the blob store

You will store the aggregated sales order data in JSON format in an Azure Storage blob container.

1. View the **Outputs** page for the **process-orders** Stream Analytics job. Then use the **Add** menu to add a **Blob storage/ADLS Gen2** output with the following properties:
    - **Output alias**: `blobstore`
    - **Select Select Blob storage/ADLS Gen2 from your subscriptions from your subscriptions**: Selected
    - **Subscription**: Your Azure subscription
    - **Storage account**: Select the **store*xxxxxxx*** storage account
    - **Container**: Select the existing **data** container
    - **Authentication mode**: Managed Identity: System assigned
    - **Event serialization format**: JSON
    - **Format**: Line separated
    - **Encoding**: UTF-8
    - **Write mode**: Append as results arrive
    - **Path pattern**: `{date}`
    - **Date format**: YYYY/MM/DD
    - **Time format**: *Not applicable*
    - **Minimum rows**: 20
    - **Maximum time**: 0 Hours, 1 minutes, 0 seconds
2. Save the output and wait while it is created. You will see several notifications. Wait for a **Successful connection test** notification.

Create a query

Now that you have defined an input and an output for your Azure Stream Analytics job, you can use a query to select, filter, and aggregate data from the input and send the results to the output.

1. View the **Query** page for the **process-orders** Stream Analytics job. Then wait a few moments until the input preview is displayed (based on the sales order events previously captured in the event hub).
2. Observe that the input data includes the **ProductID** and **Quantity** fields in the messages submitted by the client app, as well as additional Event Hubs fields - including the **EventProcessedUtcTime** field that indicates when the event was added to the event hub.
3. Modify the default query as follows:

   Code ⎘ Copy

   ```
   SELECT
       DateAdd(second,-10,System.TimeStamp) AS StartTime,
       System.TimeStamp AS EndTime,
       ProductID,
       SUM(Quantity) AS Orders
   INTO
       [blobstore]
   FROM
       [orders] TIMESTAMP BY EventProcessedUtcTime
   GROUP BY ProductID, TumblingWindow(second, 10)
   HAVING COUNT(*) > 1
   ```

   Observe that this query uses the **System.Timestamp** (based on the **EventProcessedUtcTime** field) to define the start and end of each 10 second *tumbling* (non-overlapping sequential) window in which the total quantity for each product ID is calculated.
4. Use the ▷ **Test query** button to validate the query, and ensure that the **test Results** status indicates **Success** (even though no rows are returned).
5. Save the query.

Run the streaming job

OK, now you're ready to run the job and process some real-time sales order data.

1. View the **Overview** page for the **process-orders** Stream Analytics job, and on the **Properties** tab review the **Inputs**, **Query**, **Outputs**, and **Functions** for the job. If the number of **Inputs** and **Outputs** is 0, use the ↻ **Refresh** button on the **Overview** page to display the **orders** input and **blobstore** output.
2. Select the ▷ **Start** button, and start the streaming job now. Wait until you are notified that the streaming job started successfully.
3. Re-open the cloud shell pane, reconnecting if necessary, and then re-run the following command to submit another 1000 orders.

   Code ⎘ Copy

   ```
   node ~/dp-203/Allfiles/labs/17/orderclient
   ```
4. While the app is running, in the Azure portal, return to the page for the **dp203-*xxxxxxx*** resource group, and select the **store*xxxxxxxxxxxx*** storage account.
5. In the pane on the left of the storage account blade, select the **Containers** tab.
6. Open the **data** container, and use the ↻ **Refresh** button to refresh the view until you see a folder with the name of the current year.
7. In the **data** container, navigate through the folder hierarchy, which includes the folder for the current year, with subfolders for the month and day.

8. In the folder for the hour, note the file that has been created, which should have a name similar to **0_xxxxxxxxxxxxxxxx.json**.
9. On the **...** menu for the file (to the right of the file details), select **View/edit**, and review the contents of the file; which should consist of a JSON record for each 10 second period, showing the number of orders processed for each product ID, like this:

```
{"StartTime":"2022-11-23T18:16:25.0000000Z","EndTime":"2022-11-23T18:16:35.0000000Z","ProductID":6,"Orders":13.0}
{"StartTime":"2022-11-23T18:16:25.0000000Z","EndTime":"2022-11-23T18:16:35.0000000Z","ProductID":8,"Orders":15.0}
{"StartTime":"2022-11-23T18:16:25.0000000Z","EndTime":"2022-11-23T18:16:35.0000000Z","ProductID":5,"Orders":15.0}
{"StartTime":"2022-11-23T18:16:25.0000000Z","EndTime":"2022-11-23T18:16:35.0000000Z","ProductID":1,"Orders":16.0}
{"StartTime":"2022-11-23T18:16:25.0000000Z","EndTime":"2022-11-23T18:16:35.0000000Z","ProductID":3,"Orders":10.0}
{"StartTime":"2022-11-23T18:16:25.0000000Z","EndTime":"2022-11-23T18:16:35.0000000Z","ProductID":2,"Orders":25.0}
{"StartTime":"2022-11-23T18:16:25.0000000Z","EndTime":"2022-11-23T18:16:35.0000000Z","ProductID":7,"Orders":13.0}
{"StartTime":"2022-11-23T18:16:25.0000000Z","EndTime":"2022-11-23T18:16:35.0000000Z","ProductID":4,"Orders":12.0}
{"StartTime":"2022-11-23T18:16:25.0000000Z","EndTime":"2022-11-23T18:16:35.0000000Z","ProductID":10,"Orders":19.0}
{"StartTime":"2022-11-23T18:16:25.0000000Z","EndTime":"2022-11-23T18:16:35.0000000Z","ProductID":9,"Orders":8.0}
{"StartTime":"2022-11-23T18:16:35.0000000Z","EndTime":"2022-11-23T18:16:45.0000000Z","ProductID":6,"Orders":41.0}
{"StartTime":"2022-11-23T18:16:35.0000000Z","EndTime":"2022-11-23T18:16:45.0000000Z","ProductID":8,"Orders":29.0}
...
```

10. In the Azure Cloud Shell pane, wait for the order client app to finish.
11. In the Azure portal, refresh the file to see the full set of results that were produced.
12. Return to the **dp203-*xxxxxxx*** resource group, and re-open the **process-orders** Stream Analytics job.
13. At the top of the Stream Analytics job page, use the **Stop** button to stop the job, confirming when prompted.

Delete Azure resources

If you've finished exploring Azure Stream Analytics, you should delete the resources you've created to avoid unnecessary Azure costs.

1. In the Azure portal, on the **Home** page, select **Resource groups**.
2. Select the **dp203-*xxxxxxx*** resource group containing your Azure Storage, Event Hubs, and Stream Analytics resources.
3. At the top of the **Overview** page for your resource group, select **Delete resource group**.
4. Enter the **dp203-*xxxxxxx*** resource group name to confirm you want to delete it, and select **Delete**.

   After a few minutes, the resources created in this exercise will be deleted.