Analyzing Data with Spark in Azure Databricks

Lab 2 – Running a Spark Job

Overview

In this lab, you will run a Spark job to process data.

What You'll Need

To complete the labs, you will need the following:

- A web browser
- A Microsoft account
- A Microsoft Azure subscription
- A Windows, Linux, or Mac OS X computer
- Azure Storage Explorer
- The lab files for this course

Note: To set up the required environment for the lab, follow the instructions in the **Setup** document for this course. Specifically, you must have signed up for an Azure subscription.

Provisioning Azure Resources

Note: If you already have an Azure Databricks Spark cluster and an Azure blob storage account, you can skip this section.

Provision a Databricks Workspace

1. In a web browser, navigate to http://portal.azure.com, and if prompted, sign in using the Microsoft account that is associated with your Azure subscription.
2. In the Microsoft Azure portal, click **+ Create a resource**. Then in the **Analytics** section select **Azure Databricks** and create a new Azure Databricks workspace with the following settings:
    - **Workspace name**: *Enter a unique name (and make a note of it!)*
    - **Subscription:** *Select your Azure subscription*
    - **Resource Group:** *Create a new resource group with a unique name (and make a note of it!)*
    - **Location:** *Choose any available data center location.*
    - **Pricing Tier:** Standard
3. In the Azure portal, view **Notifications** to verify that deployment has started. Then wait for the workspace to be deployed (this can take few minutes).

Provision a Storage Account

1. In the Azure portal tab in your browser, and click **+ Create a resource**.
2. In the **Storage** category, click **Storage account**.
3. Create a new storage account with the following settings:
    - **Name**: *Specify a unique name (and make a note of it)*
    - **Deployment model**: Resource manager
    - **Account kind**: Storage (general purpose v1)
    - **Location**: *Choose the same location as your Databricks workspace*
    - **Replication:** Locally-redundant storage (LRD)
    - **Performance:** Standard
    - **Secure transfer required:** Disabled
    - **Subscription:** *Choose your Azure subscription*
    - **Resource group:** *Choose the existing resource group for your Databricks workspace*
    - **Virtual networks:** Disabled
4. Wait for the resource to be deployed. Then view the newly deployed storage account.
5. In the blade for your storage account, click **Blobs**.
6. In the **Browse blobs** blade, click **+ Container**, and create a new container with the following settings:
    - **Name**: spark
    - **Public access level**: Private (no anonymous access)
7. In the **Settings** section of the blade for your blob store, click **Access keys** and note the **Storage account name** and **key1** values on this blade – you will need these in the next exercise.

Creating a Spark Job

Spark jobs enable you to run data processing code on-demand or at scheduled intervals. This enables you to build data processing solutions for unattended execution.

Upload Source Data to Azure Storage

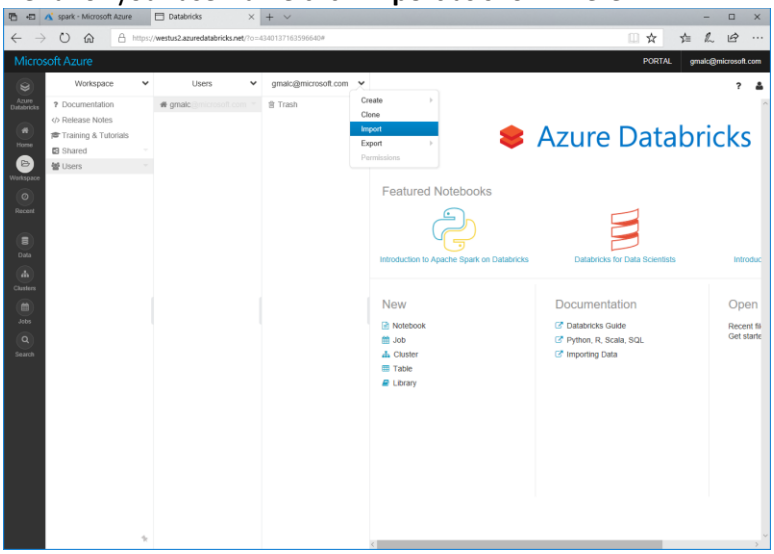
In this exercise, you will create a Spark job to process a web server log file. Before you can do this, you must store the log file you want to process in a blob storage container where it can be accessed by your cluster. The instructions here assume you will use Azure Storage Explorer to do this, but you can use any Azure Storage tool you prefer.

1. In the folder where you extracted the lab files for this course on your local computer, in the **data** folder, verify that the **IISlog.txt** file exist. This file contains the data you will process in this exercise.
2. Open **IISlog.txt** in a text editor and view its contents. It contains details of page requests made to a website, including requests to a page named *products.aspx* for which a query string indicates the specific product that the user is viewing. Close the file without saving any changes.
3. Start Azure Storage Explorer, and if you are not already signed in, sign into your Azure subscription.
4. Expand your storage account and the **Blob Containers** folder, and then double-click the **spark** blob container you created previously.
5. In the **Upload** drop-down list, click **Upload Files**. Then upload **IISlog.txt** as a block blob to a folder named **data** in root of the **spark** container.

Upload Code to Process the Data

In this lab, you will use your choice of a Python or Scala script to process the web server log data you have uploaded. Source files containing the necessary code to process the data have been provided.

1. In the Databricks workspace, click **Workspace**. Then click **Users**, click your user name, and in the drop-down menu for your username click **Import** as shown here:



2. Browse to the folder where you extracted the lab files. Then select either **ProcessLog.py** or **ProcessLog.scala**, depending on your preferred choice of language (Python or Scala), and upload it.
3. Open the file you uploaded to view the code it contains, read the comments to understand what it does, and change both instances of *<YOUR_ACCOUNT>* to the name of your Azure Storage account.

Create a Job to Run the Code

Now that you've uploaded the code, you're ready to automate its execution with a job.

1. In the Databricks workspace, click **Jobs**. Then click **+Create Job**.
2. Name the new job **Process Web Log**.
3. For the **Task** option, click **Select Notebook** and select the **ProcessLog** code file you uploaded previously.
4. For the **Parameters** option, Click **Edit** and add the following key-value pair:

logfile	IISlog.txt

Note that these values are case-sensitive.

5. For the **Cluster** option, click **Edit** and configure the cluster for the job as follows:
   - **Cluster Type**: New Cluster
   - **Databricks Runtime Version**: *Choose the latest available version*
   - **Python Version:** 3
   - **Driver Type**: Same as worker
   - **Worker Type**: *Leave the default type*
   - **Workers:** 1
   - **Spark Config**: Add a key-value pair for your storage account and key like this:

   fs.azure.account.key.***your_storage_account***.blob.core.windows.net ***your_access_key1_value***

6. For the **Schedule** option, click **Edit** and note that you can schedule the job to run at regular intervals. Then cancel the edits to the schedule and leave it set to **None**.
7. Under **Active runs**, click **Run Now** to start the job.

View Job Status and Output

Now that you've started the job, you can observe its status and view the output.

1. In the Databricks workspace, click **Jobs**. Note that the **Process Web Log** job is listed with a green ● icon to indicate it is active.
2. Click the **Process Web Log** job. This returns you to the job details page where **Run 1** should be listed under **Active runs**.
3. Note the **Status** value for **Run 1**. Initially this will be **Pending**. Over time it will change to **Running**, and then to **Succeeded**. After it has succeeded, **Run 1** will be moved to the **Completed in the last 60 days** list.
4. Click **Run 1** to view the job output. This includes the code that was executed, and the output returned by the Spark process that ran the code.
5. In the Databricks workspace, click **Clusters**, and in the **Job Clusters** list, note that a new cluster was created for the job and is now in a **Terminated** state.

View the Processed Data

Now that you've run the job, you can view the processed data.

1. Use your preferred Azure storage tool to view the contents of the **spark** container in your Azure storage account. This should contain a new folder named **output** that was created by your job code.
2. In the **output** folder, view the files that have been generated by the job. These include:
    - **_committed_*nnnn…*** (a checkpoint file written when the output was committed)
    - **_started_*nnnn…*** (a checkpoint file written when the job started)
    - **_SUCCESS** (a file indicating the result status of the job)
    - **Part-00000-*nnn….*csv** (the output data produced by the job)
3. Download and view the **Part-00000-*nnnn….*csv** file to view the output generated by the job. This file contains the number of web page views for each product.

Clean Up

Note: If you intend to proceed straight to the next lab, skip this section. Otherwise, follow the steps below to delete your Azure resources and avoid being charged for them when you are not using them.

Delete the Resource Group

1. Close the browser tab containing the databricks workspace if it is open.
2. In the Azure portal, view your **Resource groups** and select the resource group you created for your databricks workspace. This resource group contains your databricks workspace and your storage account.
3. In the blade for your resource group, click **Delete**. When prompted to confirm the deletion, enter the resource group name and click **Delete**.
4. Wait for a notification that your resource group has been deleted.
5. After a few minutes, a second resource group containing the resources for your cluster will automatically be deleted.
6. Close the browser.