FY19 Intro to Databricks Hack – Attendee Guide

Introduction to using Spark on Azure Databricks

# Goals

Most challenges observed by customers in these realms are in stitching multiple services together. As such, where possible, we have tried to place key concepts in the context of a broader example.

Once all hackathon challenges are completed, you should be able to:

* Create an Azure Databricks cluster
* Create Azure Databricks workspace and notebook for data exploration
* Understand the difference between Spark 1 and Spark 2 (RDD API vs. Dataframe API)
* How to use Spark SQL API to wrangle datasets using a familiar syntax

# Background Knowledge

This hackathon is meant for data engineers & data scientists on Azure. Previous experience with the following may be helpful but is not required:

* + Spark
  + Azure HD Insight
  + Python
  + Jupyter notebooks

# Introduction

In this challenge, you will provision a Databricks workspace and a Spark cluster. You will then interact with the Spark

cluster & begin to explore data interactively.

Getting Started:

To complete the challenges, 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

• Microsoft Teams (challenge datasets / scripts are loaded into files section on Teams)

**Note**: To set up the required environment for the challenge, follow the instructions in the [**Setup**](https://microsoft-my.sharepoint.com/personal/laedell_microsoft_com/Documents/Hackathons/ML/Databricks/databricks_intro/databricks-introFinal/Setup%20Guide.docx)document for this workshop.

Pre-Requisites (completed during workshop)

**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 Azure portal, create a **new Azure Databricks workspace**

\* **helpful tips:**

* + - * **Resource Group:** *Create a new resource group (it will help when cleaning up resources at the end of hackathon)*

Provision a Storage Account

1. Create a new storage account

• **helpful tips:**

* + - * **general storage type V1 is sufficient**
      * **for location**: *Choose the same location as your Databricks workspace*
      * **replication:** Locally-redundant storage (LRD) <it is more cost effective for this hackathon>
      * **choose the resource group created for your databricks workspace above**

2. Create new **Container**

**\* helpful tips:**

• set **public access level**: Private (no anonymous access)

3. Make note of your Account name , container name + Access Key1 below (settings section for your blob)

– you will need these in the next procedure.

Create a Spark Cluster

1. In the Azure portal, browse to the Databricks workspace to **Launch your instance**
2. **Create a cluster – use the latest version available of Spark and Python 3**

*Use your own experience and judgement for cluster sizing*

* + - **helpful tips:**
      * **Min workers = 1 / Max workers = 2**
      * **Spark Config**: Add two key-value pairs for your storage account and key like this:
        + fs.azure.account.key.***your\_storage\_account***.blob.core.windows.net ***your\_key1\_value***
        + spark.hadoop.fs.azure.account.key.***your\_storage\_account***.blob.core.windows.net ***your\_key1\_value***

*\*\*Why? The 1st one enables you to access storage account using the newer DataFrame API / the 2nd is for RDD API access to storage*

## During the challenges below, you will find it helpful to keep track of your various keys in a text file.

Storage

* Azure Blob Storage Connection String:
* Azure Blob Storage name:
* Container name:
* Azure Blob Access Key1:

Databricks

* + Azure Databricks Workspace name:
* Resource group:

## Challenge 1: Intro to Spark

This challenge involves loading textual data in Azure blob storage (for this exercise, we will be using President Kennedy’s inaugural speech as our content source) and will use Spark - either python or scala (your choice) - to filter and transform the dataset according to the challenge summary.

**Challenge Considerations**

You will use Spark Resilient Distributed Datasets (RDDs) to load and explore data. The RDD-based API is an original module of Spark, and has largely been superseded by a newer Dataframe-based API;

however, there are many production systems (and code examples on the Web) that use RDDs, so it’s worth starting your exploration of Spark there.

**Challenge Summary**

Your team is preparing a proof of concept, and your group has been tasked to suggest answers to the following questions:

1. Can you determine the intent of President Kennedy’s speech <hint: look for words like *freedom* & create filtered views for those keywords to answer the question>

2. Are there any additional Spark concepts that you would use to help bring value to your analysis?

3. What are some potential ways you could save time during your analysis <hint: less lines of code following one API approach vs. another means faster time to value>

4. Are there any notable differences when analyzing Unstructured vs. Structured datasets?

Business Case 1a – Unstructured Analysis using President Kennedy’s Speech

* Setup assistance & Introduction to Azure Databricks concept
* Challenge 1.1: Setting up Azure Databricks: Create a cluster
* Challenge 1.2: Setting up Azure Databricks: Create a workspace & your first notebook
* Challenge 1.3: Use RDD API (python or scala) to transform unstructured data file from Azure Blob Storage

Business Case 1b – Structured Analysis of Traffic Accidents

* Challenge 1.4: Setting up Azure Databricks: Create new notebook & label it **Dataframes.ipynb**
* Challenge 1.5: Use Dataframes API (python or scale) (Spark 2.0) to analyze new STRUCTURED datasets (related to traffic accidents) : Read data in from Azure Blob storage
* Challenge 1.6: Create dataframes to analyze traffic patterns & vehicle densities by location;
* Challenge 1.7: Create top and bottom views of accident data to better under emerging patterns <if they exist>
* Challenge 1.8: Transform existing dataframes into Temporary Tables or Permanent Tables if your query results are deemed meaningful by your table using Spark SQL API

**Follow up Activities:**

* Discuss approaches and analysis with Table