

Tutorial Spark en Databricks

1. Introducción

En este tutorial vamos a aprender a ejecutar procedimientos de Apache Spark en la nube de Databricks. Databricks es una compañía fundada por los creadores de Spark (y varias otras tecnologías) que ofrece soluciones en tema de manejo de datos. La ventaja de usar Databricks es aliviar el proceso de configuración del *cluster* para centrarse en el problema de manejo de datos.

En general, Databricks provee *notebooks* interactivos que se conectan a un *backend* en AWS, Google Cloud Platform o Microsoft Azure. Sin embargo, para este tutorial vamos a utilizar una versión gratuita más “liviana”, que es gratis para fines educacionales.

2. Creando la cuenta

Para comenzar, necesitamos una cuenta en Databricks. Para esto, accedemos a <https://databricks.com/try-databricks>, en donde nos vamos a encontrar con el formulario de la Figura 1.

The image shows the 'Try Databricks for free' registration page. At the top, there's a navigation bar with links: Platform, Solutions, Learn, Customers, Partners, and Company. The main heading is 'Try Databricks for free'. Below it, a subheading reads: 'An open and unified data analytics platform for data engineering, data science, machine learning, and analytics.' Further down, it says: 'From the original creators of Apache Spark™, Delta lake, MLflow, and Koalas.' There are logos for AWS, Microsoft Azure, and Google Cloud. A section titled 'Databricks trial:' lists three bullet points: 'Collaborative environment for data teams to build solutions together.', 'Interactive notebooks to use Apache Spark™, SQL, Python, Scala, Delta Lake, MLflow, TensorFlow, Keras, Scikit-learn and more.', and 'Available as a 14-day full trial in your own cloud, or as a lightweight trial hosted by Databricks.' Below this, 'Used by:' lists logos for Virgin hyperloop, edmunds, nielsen, RIOT GAMES, REGENERON, and Grab. On the right side, there's a form titled 'Please tell us about yourself' with fields for 'First Name: *', 'Last Name: *', 'Company *', 'Company Email *', 'Title *', and 'Phone Number'. There's a checkbox for 'Keep me informed with occasional updates about Databricks and related open source products'. At the bottom, it says 'By Clicking "Get Started For Free", you agree to the Privacy Policy.' and there's a red button labeled 'GET STARTED FOR FREE'.

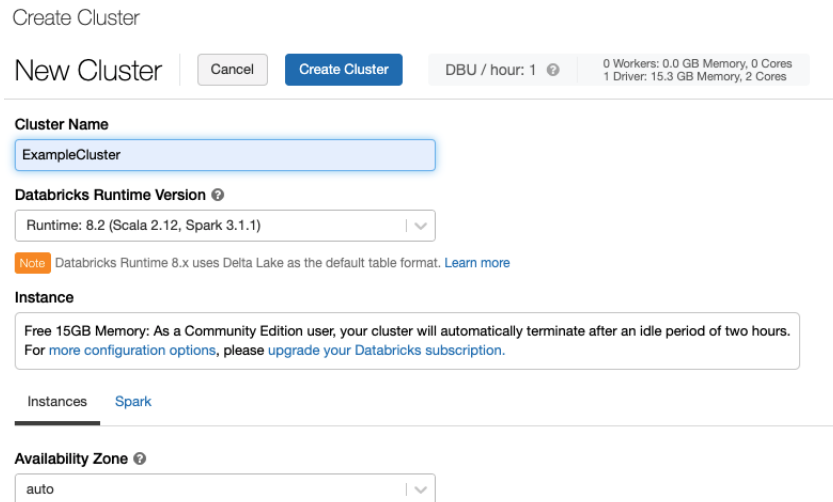
Figura 1: Formulario para crear una cuenta en Databricks.

Aquí al llenar nuestros datos, se nos va a enviar un enlace a nuestro correo para validar la cuenta. Una vez realizado todo este proceso, se nos dará escoger que tipo de cuenta queremos: un *trial* de 14 días

de una versión completa sobre los *backend* mencionados anteriormente, o bien, **acceso a una cuenta para la versión *Community* de Databricks**. Esta segunda es la opción que queremos. Una vez listo con este paso, iniciamos sesión en la versión *Community* de Databricks en <https://community.cloud.databricks.com/login.html>.

3. Subiendo el archivo

Lo primero que vamos a hacer es crear un *cluster*. Este *cluster* permanece activo por un tiempo razonable (1 - 2 horas después de que no exista uso). La pestaña para crear el *cluster* se ve en la Figura 2.



Create Cluster

New Cluster | Cancel | Create Cluster | DBU / hour: 1 | 0 Workers: 0.0 GB Memory, 0 Cores
1 Driver: 15.3 GB Memory, 2 Cores

Cluster Name
ExampleCluster

Databricks Runtime Version ⓘ
Runtime: 8.2 (Scala 2.12, Spark 3.1.1) | v

Note Databricks Runtime 8.x uses Delta Lake as the default table format. [Learn more](#)

Instance
Free 15GB Memory: As a Community Edition user, your cluster will automatically terminate after an idle period of two hours.
For more configuration options, please [upgrade your Databricks subscription](#).

Instances Spark

Availability Zone ⓘ
auto | v

Figura 2: Pestaña para crear un cluster en Databricks.

Una vez que creemos un *cluster*, necesitamos crear una tabla¹. **Importante:** no vamos a utilizar la tabla, sino que vamos a seguir este procedimiento para subir un archivo. En el menú para crear la tabla, vamos a subir el archivo al que le queremos contar las palabras, que en este caso se llama `example.txt`². Un ejemplo de esto se ve en la Figura 3

Ahora para comprobar que el archivo se haya subido correctamente, vamos a la pestaña para crear una tabla, y luego en la parte superior hacemos click en DBFS. Si seguimos la estructura de directorios de la Figura 4, podremos comprobar si el archivo se subió correctamente. Notemos además, que abajo se nos menciona la ruta del archivo.

4. Corriendo nuestro primer programa

Ahora que el archivo ya está subido, creamos un nuevo *notebook* asociado a nuestro *cluster*. Para esto, en el panel izquierdo nos vamos a la opción create y hacemos click en *notebook* según muestra la Figura 5.

Ahora, en la primera celda, escribimos el siguiente programa³:

¹Esto se hace en la opción Data del menú en la izquierda.

²Estamos subiendo el mismo archivo de ejemplo de clases.

³Que es el mismo programa visto en clases.

Create New Table

Data source ⓘ

Upload File S3 DBFS Other Data Sources Partner Integrations

DBFS Target Directory ⓘ

/FileStore/tables/ (optional) Select

Files uploaded to DBFS are accessible by everyone who has access to this workspace. [Learn more](#)

Files ⓘ

example.txt ✓

62 b
[Remove file](#)

✓ File uploaded to /FileStore/tables/example.txt

Create Table with UI Create Table in Notebook ⓘ

Select a Cluster to Preview the Table

Choose a cluster with which you will read and preview the data.

Cluster ⓘ

ExampleCluster | v

Preview Table

Figura 3: Pestaña para subir un archivo en Databricks.

```

1  val textFile = sc.textFile("/FileStore/tables/example.txt")
2  val counts = textFile.flatMap(line => line.split(" ")).
3    map(word => (word, 1)).reduceByKey((a, b) => a + b)
4  counts.saveAsTextFile("/FileStore/tables/output")

```

Donde el archivo es el que subimos anteriormente (tenemos que comprobar el *path* según lo que vimos anteriormente). Ahora, para poder ver el *output* vamos a tener que usar un pequeño *workaround*. Lo que vamos a hacer primero es ver la URL que estamos utilizando. Esta se ve de la forma:

```
community.cloud.databricks.com/?o=...#
```

En donde entre el *?o=* y el *#* hay un número. Este número lo tenemos que guardar. Ahora vamos a ver, en la pestaña de crear una tabla, la estructura de directorios en el DBFS. Ahora vamos a buscar los archivos de *output*. Esto se ve en la Figura 6
Y en nuestro navegador, vamos a ingresar la URL:

```
community.cloud.databricks.com/files/...
```

Donde ... se rellena con la estructura de directorios y nombre de archivo (sin el FileStore), que en este caso, según la Figura 6 generaría la URL:

```
community.cloud.databricks.com/files/tables/output/part-00000
```

Ahora bien, hay que agregar el token de autenticación, que es el número que guardamos más arriba. Así, la URL final sería:

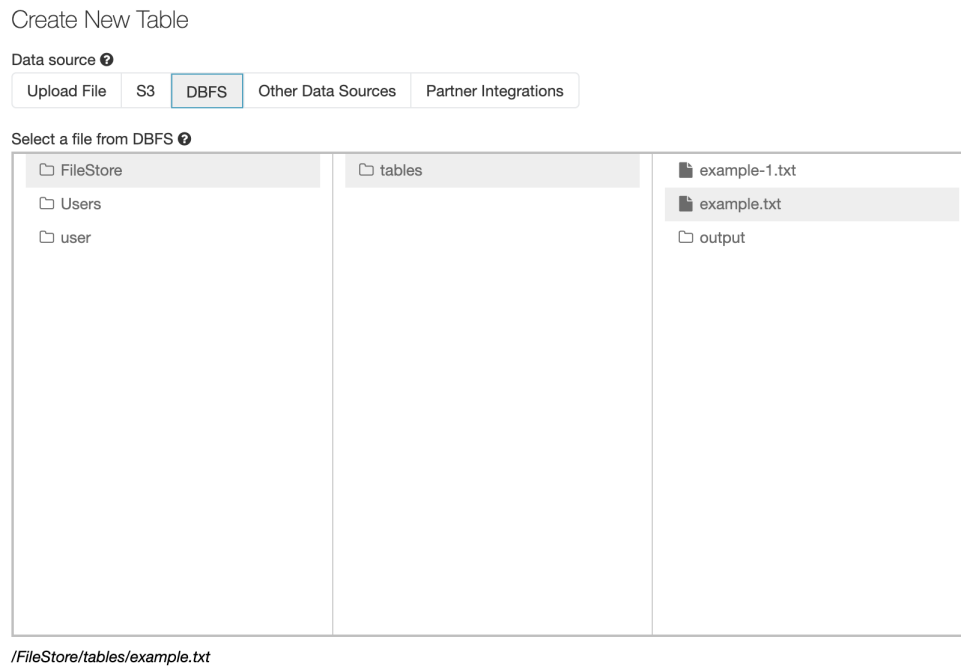


Figura 4: Pestaña para comprobar los archivos del DBFS en Databricks.

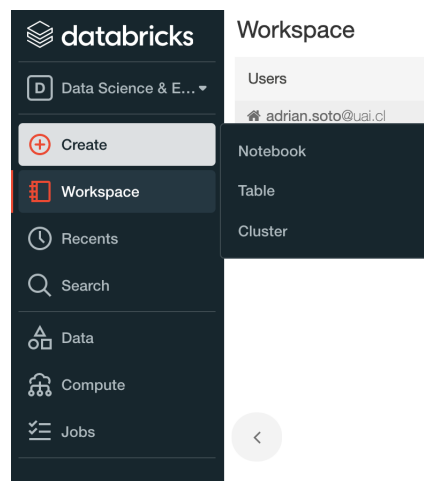


Figura 5: Creando un *notebook* en Databricks.

`community.cloud.databricks.com/files/tables/output/part-00000/?o=...`

Donde ... lo reemplazamos por el número que guardamos más arriba. Al ingresar esta dirección en nuestro navegador, vamos a poder descargar el archivo de *output*. Así, logramos correr nuestro primer programa de Spark en Databricks.

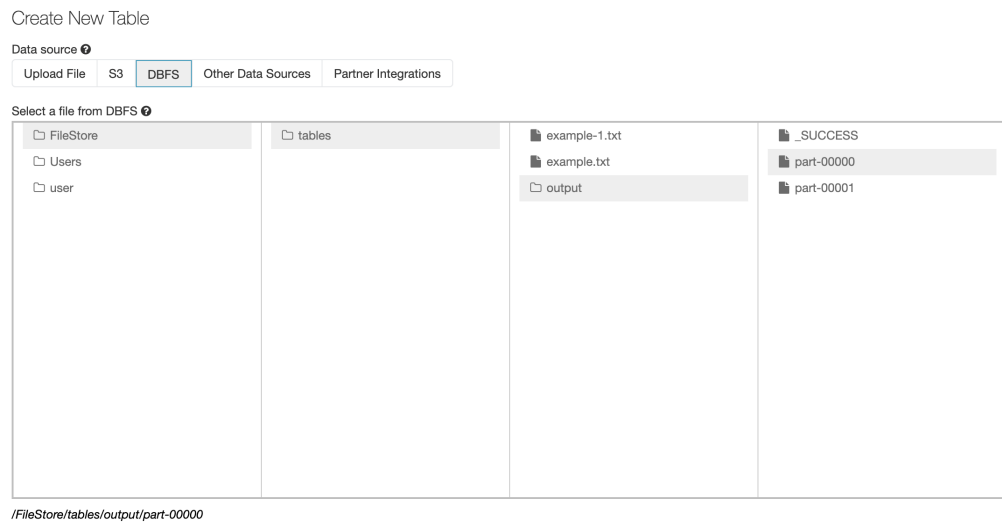


Figura 6: Buscando los archivos de *output*.