databricks

1.4 - Demo - Working with Repos on Databricks

(https://databricks.com)

Demo: Working with Repos

Learning Objectives

1. Describe Repos as a capability centered around continuous integration of assets in Databricks and external Git repositories.
2. Add a repo from an existing Git repository.
3. Describe how to compare, pull, and push changes between Databricks and a Git repository.

Notes for this lesson

- In this lesson, we are going to use a demo repo that is available in Github. If you are going to perform the steps in this demo, you will need to use your own Github account.
- You will generate a Personal Access Token (PAT) in Github during this lesson. PATs are just like a username and password combination and will provide the token's holder the ability to make changes to your Github account. You should treat PATs with the same security as you would a username and password. Also, you must follow your organization's security practices when working with your organization's Github account and any PATs you generate in that account.

Our Hypothetical Situation

For the remainder of this course, let's consider the hypothetical situation that is common for data engineers: You have been given a data file, and you have been asked to:

1. Ingest the file into Databricks
2. Write a notebook that inserts the data into a table and cleans it in that table
3. Share this notebook with other members of the team
4. Create a job that runs on a regular interval that will run the notebook on a regular interval
5. Create a simple visualization from the cleaned data.

We will complete each task on this list by the end of this session.

Creating a Personal Access Token (PAT) in Github

In order to connect Databricks to Github, you need to generate a Personal Access Token (PAT) in Github. Please note that a PAT is just like a username and password and should be treated with the same security that you would use with a username and password. Also, the PAT you will generate in your own Github account will not give other users, or Databricks' employees, access to your Github account. To generate a PAT in Github:

1. Click here (https://github.com/) to navigate to Github. Because you should never trust a link provided by a third-party, please verify that you are now on the official Github site by examining the URL in your browser.
2. If you are not currently signed in to Github, do so now using your own credentials. Again, if you do not have a Github account, or if you do not wish to use your own account, just follow along.
3. In the upper-right corner of Github, click the image associated with your account, and select "Settings."
4. At the bottom of the left navigation bar, click "Developer Settings."
5. At the bottom of the left navigation bar, click "Personal access tokens" and select "Tokens (classic)."
6. In the upper-right corner, click "Generate new token" and select "Generate new token (classic)."
7. Give the token a name in the "Note" field. I recommend "Delete Me."
8. Provide an expiration date for the token. I recommend "7 days." To make the connection between Databricks and Github, you will need to select two scopes: repo and workflow.
9. Tick the boxes next to "repo" and "workflow" to select them.
10. Click "Generate token." The token is now listed and is shown. **This is the only time the token will be shown. You must copy it at this time, or you will have to generate a new one.**
11. Copy the token by clicking the double-square icon. Github will let you know that the token has been copied.

Add the PAT to Databricks

Perform the following steps to add the PAT to Databricks:

1. In the upper-right corner of Databricks, click your username, right-click on "User settings," and open the link in a new tab. This will allow you to refer back to these instructions, as needed.
2. In the top navigation, click "Git integration." Github is the default Git provider, so we can leave this alone.
3. Type your Github username or email address in the field provided.
4. Paste the Token in the field provided.
5. Click "Save." The PAT is added to Databricks. You can close that tab.

Forking the Demo Repo

Now that we have added a PAT to Databricks, we can clone a repo into Databricks. Perform the following steps:

1. Click here (https://github.com/databricks-academy/get-started-with-data-engineering-on-databricks-repo-example) to access the demo repo for this course.
2. Click "Fork" in the upper-right corner of the repo to fork the repo to your own account.
3. If needed, change the "Owner" and select "Create Fork." The demo repo is now forked into your own Github account, and you are viewing your own fork of the repo.

Cloning the Demo Repo into Databricks

We can now clone the fork of the demo repo into Databricks.

1. In the upper-right corner of the repo, click the green "<> Code" button, and click the double-square icon to copy the repo's URL.
2. Back in Databricks, right-click "Workspace" in the left sidebar, and open the Workspace page in a new tab. This will allow you to refer back to these instructions.
3. Click "Repos" to drop open the folders included under that section, and click *your-username*. This will be at the top of the folders just under the word "Repos" and will have a home icon next to it.
4. In the upper-right corner of the page, click "Add" and select "Repo."
5. Paste the URL you copied earlier into the "Git repositiory URL" field.
6. The "Git provider" and "Repository name" fields will be added automatically. Leave these alone for now.

7. Click "Create Repo" to add the repo to Databricks. The repo is added to your account, and you are now viewing the contents of the repo.

Comparing and Committing

Just like with Git, you can compare, commit, and pull from the repo on Github. Perform the following steps:

1. Click "Add" in the upper-right corner, and select "Notebook." We are going to talk about Notebooks in a future lesson. For now, just complete the following:
2. Click in the first cell of the notebook. It is called "Cmd 1" and has a "1" next to it.
3. Type `%md` and press return/enter to move to the next line.
4. Type `### This is my first notebook`.
5. Click in the whitespace below the cell. You should see bold text that says, "This is my first notebook."'
6. Change the name of the notebook by selecting the current notebook name (i.e., "Untitled Notebook ...") and typing "My Notebook" and return/enter. You have now made a change to the repo that is not reflected in the remote version on Github. You can compare your local version to the remote version and commit changes, as needed:
7. Click the word "published" next to the name of the notebook. This opens the repo's comparison window. In this window, you can create branches, change branches, perform a hard reset, and merge. We will not be discussing these features in this course. We can see that there is one changed file, the notebook we just added, and we can see the code in this notebook that changed. Let's commit this to the remote repo on Github.
8. In the lower-right corner, type "added a notebook" in the "Commit message (required)" field.
9. Either press return/enter or click the "Commit & Push" button. The changes are committed to the remote repo on Github.
10. Click the "X" in the upper-right corner to close the window.

Pulling from the Remote Repo

You can use repos to setup CI/CD workflows or otherwise work with members of your team working in other Databricks workspaces. Furthermore, you can clone repos to your local computer and work with them in a local environment, as

needed, just like any other repo. If you need to pull the latest changes from the remote repo, perform the following:

1. Click "published" next to the name of the notebook to open the repo's comparison window. If there are changes that need to be pulled from the remote repo, you will see a number next to the "Pull" button. To pull these changes:
2. Click the "Pull" button.
3. Note what the warning says, and click "Confirm."
4. Click the "X" to close the window.

Conclusion

In the next lesson, we will discuss creating and managing compute resources in Databricks.