Databricks on AWS

Last updated: December 15, 2023

Migration guide for init scripts to Workspace File System (WSFS)

Where should I move my init scripts?

We recommend using Databricks Runtime version 13.3 LTS and above and Unity Catalog.

The following recommendations for init script use are organized by Databricks Runtime version and Unity Catalog enablement.

- Databricks Runtime 13.3 LTS and above with Unity Catalog. Store init scripts in Unity Catalog volumes.
- Databricks Runtime 11.3 LTS and above without Unity Catalog. Store init scripts as workspace files (AWS).
- Databricks Runtime 10.4 LTS and below. Store init scripts using cloud object storage (AWS). In cases where your init scripts are "self-contained," i.e., DO NOT reference other files such as libraries, configuration files, or shell scripts, store init scripts as workspace files (AWS).

Important! Note:

- If you use init scripts from cluster policies, you **must** update the cluster policies AND all clusters already using those policies. Cluster policy changes do not propagate to all clusters using that policy.
- WSFS has a file size limitation of 200mb. If you have larger files, please use cloud storage or volumes instead.

Migration guide for cluster-scoped init scripts to Workspace File System (WSFS)

{AWS version}

For Databricks Runtime 11.3 LTS and above without Unity Catalog, migrate cluster-scoped init scripts from DBFS to Workspace File System (WSFS). This guidance is for customers who use shared or single-user access modes with Unity Catalog enabled.

In general, you must do the following:

1. Create a folder on WSFS and check that the cluster creator can access the folder.
2. Copy all your init scripts and files referenced by the init scripts from DBFS to your WSFS folder.
    a. [Optional] Modify init scripts to reference files on WSFS if necessary.
3. Update cluster configuration and cluster policies to reference the init scripts on WSFS.

Follow the steps below to migrate init scripts from DBFS to WSFS.

Note: If you are using Terraform, follow these instructions instead.

1) Create a folder on WSFS and check that the cluster creator can access the folder.

Create a new shared folder in the Databricks Workspace and set CAN_VIEW permission on the shared folder to access files in the folder, i.e., other shell scripts referenced by the init scripts or libraries to be installed. To create a folder, use the UI or the Databricks CLI.

If you use the Databricks CLI, install it and configure the appropriate authentication method. Perform the following steps:

1. Create a folder for init scripts, for example, /InitScripts:

```
databricks workspace mkdirs '/InitScripts'
```

2. Set CAN_VIEW permission on the folder to a given cluster creators' (in the example group *cluster_admins*) access to the folder.

```
databricks workspace update-permissions directories \
  $(databricks workspace get-status '/InitScripts' -o json|jq .object_id)
--json '{
    "access_control_list": [
      {
        "group_name": "cluster_admins",
        "permission_level": "CAN_VIEW"
      }
  ]
}'
```

2) Copy all init scripts, and files referenced by init scripts, to your WSFS folder.

First, identify all objects (clusters, jobs, DLT pipelines, etc.) that use init scripts on DBFS. Use the init script detection notebook to prepare a list of those individual init scripts.

You can copy files from DBFS to WSFS using the Databricks CLI. Perform the following steps:

1. Copy the file from the DBFS to the local disk:

```
databricks fs cp 'dbfs:/FileStore/init-script.sh' init-script.sh
```

2. Copy the file from the local disk into the folder created in Databricks workspace:

```
databricks workspace import --format AUTO \
  --file init-script.sh '/InitScripts/init-script.sh'
```

3. If you are accessing configuration files, libraries, and other scripts or files from within the init scripts, you must copy them to the WSFS folder and update the paths. In your init script, change all the DBFS paths from DBFS to WSFS: Append the /Workspace string to the workspace path (that you can obtain from the UI), in the form of /Workspace/<path-to-workspace-file>. Example:

```
#!/bin/bash

ls /Workspace/Users/user@domain.com/init-script.sh
```

3) Update cluster configuration and cluster policies to reference the init scripts on WSFS.

Change the init script path in each of the objects identified above to point to WSFS instead of DBFS (if you use init scripts detection notebook, click links in generated HTML tables):

1. **Clusters**: Edit your clusters, remove init scripts that use files on DBFS, and add init scripts with source "Workspace," providing the path to the file in workspace, e.g. `/Users/user@domain.com/init_script.sh`
   Note: If you use init scripts from cluster policies, you **must** update the cluster policies AND all clusters using those policies. Cluster policy changes do not propagate to clusters already using that policy.
2. **Jobs**: Edit the definitions of each task that uses a dedicated job cluster and each shared job cluster.
3. **Cluster policies**: Edit Cluster Policy. In the policy definition, search for blocks like the following, where N is the item number:

```
{
  "init_scripts.N.dbfs.destination": {
    "type": "fixed",
    "value": "dbfs:/FileStore/init-scripts/empty_init_script.sh"
  }
}
```

and replace them with the following, adjusting the file path:

```
{
  "init_scripts.N.workspace.destination": {
      "type": "fixed",
      "value": "/Users/user@domain.com/init_script.sh"
  }
}
```

4. **Delta Live Tables pipelines**: Open pipeline settings, select the "JSON" tab, and in the cluster definition(s), change entries in the `init_scripts` array from

```
{
 "dbfs": {
   "destination": "dbfs:/FileStore/init-scripts/empty_init_script.sh"
 }
}
```

to

```
{
 "workspace": {
   "destination": "/Users/user@domain.com/init_script.sh"
 }
}
```

This works for both UC and non-UC DLT pipelines in the CURRENT channel.

Migration to Workspace Files using Terraform

To migrate existing Terraform code, you must make the following changes in the code:

- Switch from using databricks_dbfs_file to databricks_workspace_file resource to continue using the init scripts in the workspace. These resources have the same parameters, so only the resource type needs to be changed. For example, if you use the following resource block to upload a file with an init script to DBFS:

```
resource "databricks_dbfs_file" "init_script" {
  source = "${path.module}/init-script.sh"
  path   = "/FileStore/init-script.sh"
}
```

then change the resource type and adjust the path to the init script in the Workspace:

```
resource "databricks_workspace_file" "init_script" {
  source = "${path.module}/init-script.sh"
  path   = "/Users/user@domain.com/init-script.sh"
}
```

- Set permissions on the workspace files to be readable by clusters/jobs/… owners.
- Change the init_scripts blocks in the databricks_cluster, databricks_job, and databricks_pipeline resources from

Unset

```
init_scripts {
 dbfs {
  destination = databricks_dbfs_file.init_script.dbfs_path
 }
}
```

to

Unset

```
init_scripts {
 workspace {
  destination = databricks_workspace_file.init_script.path
 }
}
```

- If using init scripts in databricks_cluster_policy, adjust the paths to point to the workspace file reference.