Table of Contents

Introduction

"hms-mirror" is a utility used to bridge the gap between two clusters and migrate hive *metadata*. HMS-Mirror is distributed under the APLv2 ([License APLv2](#)) license.

The application will migrate hive metastore data (metadata) between two clusters. With SQL ([SQL](#)) and EXPORT_IMPORT ([EXPORT_IMPORT](#)) data strategies, we can move data between the two clusters using Hive SQL.

As an alternative to using Hive SQL to move data between two clusters, hms-mirror can build distcp plans and scripts that can be run in concert with the metadata scripts to complete the migration.

You start by picking a 'Data Strategy' that fits your needs. A **Data Strategy** defines how you're choosing to migrate 'metadata' and 'data' between two clusters. Some of the strategies are 'metadata' only, and some are 'metadata' and 'data'. Others are used to migrate data 'within' a cluster (STORAGE_MIGRATION) to facilitate changes in the storage layer. For example: Move data into an encrypted zone or to a different storage layer, like Ozone.

There are two interface (working on a third) for hms-mirror: CLI ([CLI Interface](#)) and WebUI ([Web Interface](#))

From both interfaces, reports are generated that detail the actions taken by the application. You can direct it to run the conversion scripts automatically or just generate the scripts for you to run later.

Release Notes

Known Issues

The latest set of known issues can be found here (https://github.com/cloudera-labs/hms-mirror/issues?q=is%3Aissue+is%3Aopen+label%3Abug)

Enhancement Requests

The latest set of enhancement requests can be found here (https://github.com/cloudera-labs/hms-mirror/issues?q=is%3Aissue+is%3Aopen+label%3Aenhancement).

If there is something you'd like to see, add a new issue here (https://github.com/cloudera-labs/hms-mirror/issues)

2.3.0.13

What's New

Enhance logging to show which instance is handling a connection/job in case of using multiple HS2 instances (https://github.com/cloudera-labs/hms-mirror/issues/183)

> ⚠️ When using multiple HS2 instances, it can be difficult to determine which instance is handling a connection or job. This enhancement will allow you to add a `set` statement that identifies which instance is handling the connection or job. Use the `-po[r|l]` property override values if you'd like to include this information in the log output. `-po hive.server2.thrift.bind.host` will add the statement to both LEFT and RIGHT output. The option is also available in the Web UI through "Property Overrides".

2.3.0.12

Bug Fixes

The Hikari Connection Pool settings are causing intermittent connection failures, cause table transfer failures. (https://github.com/cloudera-labs/hms-mirror/issues/182)

2.3.0.10

Bug Fixes

Second run in WebUI Fails (https://github.com/cloudera-labs/hms-mirror/issues/181)

dbRegEx not being processed. Throws MISC_ERROR because it can't find any databases. (https://github.com/cloudera-labs/hms-mirror/issues/180)

Legacy DBPROPERTIES are causing ERROR when attempting to set on CDP (https://github.com/cloudera-labs/hms-mirror/issues/179)

Issue with jobs not completing when some schema's were already present.

Address lingering connections after run completes.

Fixed counters for CLI screen output.

Fixed an issue with tables being processed multiple times under some conditions.

Partition discovery for SHADOW table when source is a Managed table shouldn't try to build partitions with ALTER (https://github.com/cloudera-labs/hms-mirror/issues/178)

LEFT side SQL when running 'execute' mode for SQL data strategy isn't being run. (https://github.com/cloudera-labs/hms-mirror/issues/177)

CLI App version fails when attempting to set 'concurrency' option (https://github.com/cloudera-labs/hms-mirror/issues/176)

MSCK for Shadow table not generated when 'metastore_direct' on the LEFT isn't defined (https://github.com/cloudera-labs/hms-mirror/issues/175)

2.3.0.4

What's New

BETA Iceberg conversion support for the STORAGE_MIGRATION data strategy. See Iceberg Conversion (ICEBERG_MIGRATION) for more details. To activate this beta feature for the WebUI, add `--hms-mirror. config.beta=true` to the startup command. EG: `hms-mirror --service --hms-mirror.config.beta=true`

Bugs (Fixed)

Add to connection init the ability to set the queue AND trigger an engine resource (https://github.com/cloudera-labs/hms-mirror/issues/173)

Validate SQL elements before making changes to the Cluster (https://github.com/cloudera-labs/hms-mirror/issues/168)

Extend the HS2 connection validation with Tez task validation (https://github.com/cloudera-labs/hms-mirror/issues/167)

The reset-to-default-location doesn't seem to be working in v2 (https://github.com/cloudera-labs/hms-mirror/issues/165)

2.2.0.19.6 (pre-release)

What's New

BEHAVIOR CHANGE - Drop any shadow table definitions created during migration automatically (https://github.com/cloudera-labs/hms-mirror/issues/163)

> ⚠️ This adjustment will remove shadow tables that were created during the migration process. This will help to keep the source and target clusters clean of any artifacts created during the migration process. An option `saveWorkingTables` has been added to the configuration to allow you to keep these tables if you need them for any audits or other reasons. The default is `false`, which means that the tables will be dropped automatically. An audit 'cleanup' file with the drop commands will be created in the report directory, regardless of the setting.

2.2.0.19.2 (pre-release)

What's New

Error event is not logged for table skipped because RIGHT already has the table (https://github.com/cloudera-labs/hms-mirror/issues/162) Would be great to be able to omit warnings from the end of logs (https://github.com/cloudera-labs/hms-mirror/issues/161) Be able to reduce spring framework related log entries (https://github.com/cloudera-labs/hms-mirror/issues/160)

2.2.0.19.1

Bug Fixes

- Doubling of mapping locations for partitions in `distcp` for STORAGE_MIGRATION.

- STORAGE_MIGRATION is setting ACID to ON, regardless. (https://github.com/cloudera-labs/hms-mirror/issues/158)

What's New

- Validate JDBC Jar Files in config. (https://github.com/cloudera-labs/hms-mirror/issues/159)

- Ability to turn-on `strict` mode for Storage Migration. This will cause `distcp` to fail when non-standard locations are used. To turn off, use the `-sms|--storage-migration-strict` flag via the CLI.

Behavior Changes

The default behavior for Storage Migration 'strict' has changed from `true` to `false`. The intent behind the `strict` mode was to ensure `distcp` would fail when non-standard locations are used. The combination of `metastore_direct` and knowing the partition location details gives us a better chance on making these mappings work for `distcp`. When the scenario arises, we do **HIGHLY** recommend that you validate the plans created. The new default behavior will allow `distcp` to continue when non-standard locations are encountered, while throwing a warning. This will allow the migration to continue, but you should validate the results.

> ⚠ You will need to reset the `strict` mode flag in you configuration yaml. It will be set to `true` if the configuration was created before this release. You will need to set it to `false` to maintain this new 'preferred' behavior.
>
> ```
> transfer:
>   storageMigration:
>     strict: false
> ```

2.2.0.18.1

What's New

Beta Flag to be used for future beta features. To activate:

CLI

```
hms-mirror --beta
```

Web UI

```
hms-mirror --service --hms-mirror.config.beta=true
```

Bugs (Fixed)

- Property overrides - po|pol|por - are not honored in v2, were working in 1.6.x (https://github.com/cloudera-labs/hms-mirror/issues/156)

- Partition auto-discovery config value is misconfigured during setup (https://github.com/cloudera-labs/hms-mirror/issues/155)

- Enhancement for hms-mirror setup (https://github.com/cloudera-labs/hms-mirror/issues/154)

- Remove automatic DB location as db_name.db in db location (https://github.com/cloudera-labs/hms-mirror/issues/153)

- Add support for Hive table backup in DISTCP mode (https://github.com/cloudera-labs/hms-mirror/issues/152)

2.2.0.17.1

What's New

- Extend / Split 'transfer-ownership' to apply to database and/or table (https://github.com/cloudera-labs/hms-mirror/issues/151)

- Improved Logging output

- Support alternate db name for STORAGE_MIGRATION (https://github.com/cloudera-labs/hms-mirror/issues/149)

- CLI Option to control `data-movement-strategy`
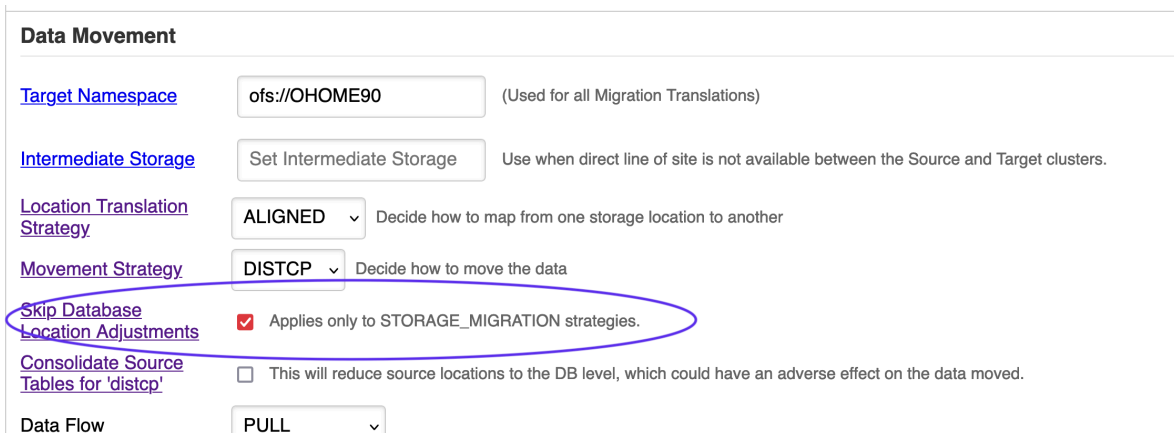
Bugs (Fixed)

- DB Prefix and DB Rename not working (https://github.com/cloudera-labs/hms-mirror/issues/150)

- When Metastore isn't configured for ALIGNED/DISTCP, it's not writing MSCK (https://github.com/cloudera-labs/hms-mirror/issues/131)

- Fixed missing CR/LF is clean up scripts.

2.2.0.15.1

What's New

Feature that allows you to 'skip' modifying the database location during a storage migration. This is useful if you're trying to archive tables in a database to another storage system, but want to leave the database location as is for new tables in the database. For STORAGE_MIGRATION, add option that would skip any Database Location Adjustments (https://github.com/cloudera-labs/hms-mirror/issues/147)

Data Movement

Target Namespace	ofs://OHOME90	(Used for all Migration Translations)
Intermediate Storage	Set Intermediate Storage	Use when direct line of site is not available between the Source and Target clusters.
Location Translation Strategy	ALIGNED ⌄	Decide how to map from one storage location to another
Movement Strategy	DISTCP ⌄	Decide how to move the data
Skip Database Location Adjustments	☑	Applies only to STORAGE_MIGRATION strategies.
Consolidate Source Tables for 'distcp'	☐	This will reduce source locations to the DB level, which could have an adverse effect on the data moved.
Data Flow	PULL ⌄	

sm_skip_dblocs.png

2.2.0.15

What's New

- Add support for Oracle Metastore Direct Connection

- SQL Strategy only uses MSCK for partition discovery for Shadow Table
  (https://github.com/cloudera-labs/hms-mirror/issues/145)

Bugs (Fixed)

- Output and Report Directory Consistency between CLI and Web UI. See docs for more
  details.

- Postgres Metastore Direct Connection Fixes

- SQL Data Strategy Validation Blockers for Acid tables-

2.2.0.12

What's New

- For STORAGE_MIGRATION to Ozone, ensure valid Volume/Bucket names
  (https://github.com/cloudera-labs/hms-mirror/issues/142)

Bugs (Fixed)

- Expose Stats in Web UI Reports (https://github.com/cloudera-labs/hms-mirror/issues/141)

- Fixed the ability to Encrypt/Decrypt password in the UI.

- Execute run for Strategies without RIGHT cluster definition fail to write reports
  (https://github.com/cloudera-labs/hms-mirror/issues/144)

- STORAGE_MIGRATION distcp efficiency option (https://github.com/cloudera-labs/hms-mirror/issues/143)

2.2.0.10

What's New

- [Hive 4 DB OWNER DDL syntax for ALTERing DB ONWER requires 'USER']
  (https://github. com/cloudera-labs/hms-mirror/issues/139)

This changed resulted in a simplification of how we determine what the cluster platform is. Previously we used two attributes (`legacyHive` and `hdpHive3`) to determine the platform. This information would direct logic around translations and other features.

Unfortunately, this isn't enough for us to determine all the scenarios we're encountering. These attributes have been replaced with a new attribute call `platformType`.

We will make automatic translations of legacy configurations to the new `platformType` attribute. The translation will be pretty basic and result in either the platform type being defined as `HDP2` or `CDP_7.1`. If you have a more complex configuration, you'll need to adjust the `platformType` attribute manually. Future persisted configurations will use the new `platformType` attribute and drop the `legacyHive` and `hdpHive3` attributes.

- Add "Property Overrides" to Web Interface (https://github.com/cloudera-labs/hms-mirror/issues/111)

A feature that was late in making it into the Web UI is now here.

- For Web UI Service, default to prefer IPV4 (https://github.com/cloudera-labs/hms-mirror/issues/134)

To ensure the right IP stack is used when the Web UI starts up, we're forcing this JDK configuration with the Web UI.

- Forcibly set Java Home via -Duser.home (https://github.com/cloudera-labs/hms-mirror/issues/136)

We had a few requests and issues with implementations were the target environment isn't always setup with normal user 'home' standards that we can rely on. This change allows us to set the 'home' directory for the user running the application and ensure its translated correctly in hms-mirror for storing and reading configurations, reports, and logs.

If you are in an environment that doesn't follow user `$HOME` standards, you can set the `HOME` environment variable to a custom directory **BEFORE** starting `hms-mirror` to alter

the default behavior.

Cleanup SQL has been added to Web Reporting UI

We've added a 'Cleanup SQL' tab to the Web Reporting UI. This will show you the SQL that was generated to clean up the source cluster after the migration. This is useful to see what will be done before you execute the migration.

Bugs (Fixed)

- DATABASE set OWNER ALTER statement is incorrect (https://github.com/cloudera-labs/hms-mirror/issues/135)

- SQL ACID Migrations from HDPHive3 cluster Ordering (https://github.com/cloudera-labs/hms-mirror/issues/138)

- DB Location for HDP3 migrations is flipped (https://github.com/cloudera-labs/hms-mirror/issues/140)

2.2.0.9

Bugs (Fixed)

- Allow process path that doesn't require Metastore Direct Connection (https://github.com/cloudera-labs/hms-mirror/issues/128)

- Fix Kerberos Connection Issues to HS2 (https://github.com/cloudera-labs/hms-mirror/issues/129)

- Job Progress getting stuck at the end while writing reports (https://github.com/cloudera-labs/hms-mirror/issues/130)

Enhancements

Increase build dependencies to CDP 7.1.9 SP1. Rework Pass Key Management. Additional details in Connection Validation.

2.2.0.8

Bugs (Fixed)

- Not parsing abfs protocol locations correctly (https://github.com/cloudera-labs/hms-mirror/issues/124)

- Database input duplication (https://github.com/cloudera-labs/hms-mirror/issues/125)

2.2.0.7

Bugs (Fixed)

- Non Configs blocking WEB UI Create (https://github.com/cloudera-labs/hms-mirror/issues/122)

- SCHEMA_ONLY with ALIGNED, DISTCP and Partitions is too granular on partition distcp (https://github.com/cloudera-labs/hms-mirror/issues/123)

- Expose rid in web ui (https://github.com/cloudera-labs/hms-mirror/issues/121)

2.2.0.5

Bugs (Fixed)

- CLI Setup Issues (https://github.com/cloudera-labs/hms-mirror/issues/120)

- Stale Config after several runs (https://github.com/cloudera-labs/hms-mirror/issues/118)

- SQL Strategy Transfer Fixes (https://github.com/cloudera-labs/hms-mirror/issues/117)

- Partition Reductions for Distcp (https://github.com/cloudera-labs/hms-mirror/issues/116)

2.2.0.4

Bugs (Fixed)

- Report Output Directory Issue (https://github.com/cloudera-labs/hms-mirror/issues/115)

- Filters Optimizations and Fixes (https://github.com/cloudera-labs/hms-mirror/issues/116)

2.2.0.2

This is a big release for hms-mirror. We've added a Web interface to hms-mirror that makes it easier to configure and run varies scenarios.

Along with the Web interface, we've made some significant adjustments to the hms-mirror engine which is much more complete than the previous release. The engine now supports a wider range of strategies and has a more robust configuration system.

We do our best to guide you through configurations that make sense, help you build plans and manage complex scenarios.

Automatic Configuration Adjustments

To ensure that configuration settings are properly set, the application will automatically adjust the configuration settings to match a valid scenario. These changes will be recorded in the 'run status config messages' section and can be seen on reports or the web interface.

Changes are mostly related to the acceptable strategy configurations. See Location Alignment (Location Alignment) for more details.

Property Overrides

Not yet available in Web UI. Coming soon issue 111 (https://github.com/cloudera-labs/hms-mirror/issues/111).

This feature, introduced in the CLI, allows you to add/override Hive properties on the LEFT, RIGHT, or BOTH clusters for custom control of running Hive jobs. Most commonly used with SQL migration strategies.

Evaluate Partition Locations and Reset to Default Location

These properties are no longer valid. An added property called 'translationType' is used to determine this functionality.

Before the epl|evaluate-partition-locations would gather partition location information from the Metastore Direct connection to ensure they were aligned. We've

adjusted/simplified the concept with `translationType`, which defined either `RELATIVE` or `ALIGNED` strategy types.

See Location Alignment ([Location Alignment](#)) for more details.

Concurrency

In previous releases using the CLI, concurrency could be set through the configuration files `transfer:concurrency` setting. The default was 4 if no setting was provided. This setting in the control file is NO LONGER supported and will be ignored. The new default concurrency setting is `10` and can be overridden only during the application startup.

See Concurrency ([Concurrency](#)) for more details.

Global Location Maps

Previous releases had a fairly basic implementation of 'Global Location Maps'. These could be supplied through the cli option `-glm`, which is still supported, but limited in functionality. The improved implementation work from the concept of building 'Warehouse Plans' which are then used to build the 'Global Location Maps'.

See Warehouse Plans ([Database Warehouse Plans](#)) for more details.

The `-glm` option can take an addition element to identify the mapping for a particular table type. As a result, any configuration files save with this setting will not be loaded and will need to be updated.

While the `-glm` option will still honor the old format of `source_dir=target_dir`, the new format is `source_dir:<table_type>:target_dir`. The `table_type` is a new addition to the configuration and is required for the new implementation. When omitted, the mapping will be created for both EXTERNAL and MANAGED tables.

`<table_type>` can be one of: `EXTERNAL_TABLE` or `MANAGED_TABLE`.

`-glm /tpsds_base_dir=EXTERNAL_TABLE:/alt/ext/location`

Old Format

```
globalLocationMap:
  /tpcds_base_dir: "/alt/ext/location"
  /tpcds_base_dir2/web: "/alt/ext/location"
```

New Format

```
userGlobalLocationMap:
  /tpcds_base_dir:
    EXTERNAL_TABLE: "/alt/ext/location"
  /tpcds_base_dir2/web:
    EXTERNAL_TABLE: "/alt/ext/location"
```

JDK 11 Support

The application now supports JDK 11, as well as JDK 8.

Kerberos Support and Platform Libraries

We are still working to replicate the options available in previous release with regard to Kerberos connections. Currently, hms-mirror can only support a single Kerberos connection. This is the same as it was previously. hms-mirror packaging includes the core Hadoop classes required for Kerberos connections pulled from the latest CDP release.

In the past, we 'could' support kerberos connections to lower versions of Hadoop clusters (HDP and CDH) by running hms-mirror on a cluster with those hadoop libraries installed and specifying --hadoop-classpath on the commandline. This is no longer supported, as the packaging required to support the Web and REST interfaces is now different.

We are investigating the possibility of supporting kerberos connections to lower clusters in the future.

Metastore Direct Access

In later 1.6 releases we introduced a 'Metastore Direct' connection type when defining a LEFT(source) cluster. To help build a more complete picture of locations in the metadata, we found it necessary to gather detailed location information for each partition of the datasets being inspected. Because Hive was so configurable regarding location preferences and the ability to set locations at the partition level, we needed to ensure that the locations were aligned. The only sure way to get this complete picture was to connect directly to the Metastore backend database. We currently support 'MYSQL' and 'POSTGRES' metastore backends. 'Oracle' coming soon.

Getting Started

Check out the sub-sections to get up and running quickly.

Video Series

Installation

Introduction

Logs

Application Logs are a detailed record of the events that have occured in the application. Logs are useful for debugging and troubleshooting issues.

Logs are stored in the $HOME/.hms-mirror/logs directory. Unless the user has specified a different **output directory** when running the CLI interface. When the -o <directory> option is used, the logs will be stored in the output directory along with the job files and reports.

For the --service or WebUI version of hms-mirror, the logs are stored in the $HOME/.hms-mirror/logs directory as hms-mirror-service.log. This log file will rotate every 100Mb and moved to the $HOME/. hms-mirror/logs/archived directory. The log file will be named with the timestamp of when the log was rotated.

Limits

Let's cover some practical limits of `hms-mirror` in the area of scale.

`hms-mirror` is designed to migrate databases. While you can migrate 'every' databases in a schema using the `dbRegEx` we don't recommend this. You should construct a plan to migrate databases individually.

The architecture of `hms-mirror` does everything in memory. Each table and partition consumes memory and contributes to the load placed on systems we're using.

We've done practical tests to do a SCHEMA_ONLY migration of 10k tables between two clusters. On our relatively modest hardware, we accomplished this in about 8 minutes with the default memory settings.

Writing the reports, which happens at the end takes time. It this case 1-2 minutes.

If you need to process more than what we've tested, please do a 'Dry-Run' first. Watch the memory of the application on the host where its running. You may need to increase the memory profile of the application and try again. See Memory Settings ([Memory Settings](#)).

Impact on Source and Target Systems

`hms-mirror` uses HiveServer2 connections to 'extract' and 'replay' schema's between systems. It also utilizes Metastore Direct connections to pull partition level details that can't be efficiently collected through the HiveServer2 SQL interface.

The concurrency (default of 10), will act like 10 users making a whole lot of DDL requests. The impact to existing workloads is a possibility. If you have a large amount of data(metadata) to migrate/extract and you're concerned about the impact to the user base, you should 'isolate' the HiveServer2 AND Hive Metastore for this process. That could mean setting up an additional HiveServer2/Metastore pair that is used specifically by 'hms-mirror'.

Databases with a Large Number of Tables

As we mentioned, migrations are a database at a time. If the table in the database exceed some of these limits, consider using various `table filters` to process filtered lists of tables in the database at a time.

Reports

Reports are created at the database level as well. When the database has a large number of tables, these reports can be extremely long.

Runtime Interfaces

The classic 'hms-mirror' interface is the command line interface (CLI). This is the interface that most users have used in the past. Support for this is still provided but is quickly being replaced by the Web interface.

The CLI interface requires users to define a configuration file that the is used to control connection endpoints, and cluster attributes. Runtime operations are controlled by command line options that are passed to the application.

The Web interface provides a much more complete experience for users and allows configurations to be built and validated in a more interactive way. The Web interface is the preferred interface for users who are new to the application and we encourage existing users to adopt this interface as well since it's capabilities and usability a vast improvement over the CLI.

> ⚠️ We understand that a big part of the attraction to the CLI was its simple commandline interface and feedback UI. And under many circumstances, the CLI was a natural choice due to security restrictions and port exposure in CDP environments. However, the Web interface offers a much more robust and user-friendly experience. See below for some suggestions on how to gain access to the Web Interface in environments with security restrictions.

Accessing the Web Interface in Secure Environments

Ports for the Web Interface may not be available in secure enviroments. By default, the Web Interface is available on port '8090'.

Option #1: Alternate Port

If '8090' isn't available, you can change the port by adding the following to the service start up command:

```
hms-mirror --service --server.port=<new_port>
```

Option #2: SSH Tunnel

If you can SSH into the host where the service is running, you can most likely create a tunnel to the service port that will allow you to access the Web Interface. Here's an example:

```
ssh -L 8090:<remote_host>:8090 <user>@<remote_host>
```

This will create a tunnel from your local machine to the remote host. Any traffic you send to localhost:8090 will be forwarded to the remote host's port 8090. Once the tunnel is established, you can open a browser and navigate to http://localhost:8090/hms-mirror to access the Web Interface.

Here is a good article on SSH Tunnels: SSH Tunneling (https://www.ssh.com/ssh/tunneling/example)

Option #3: Dynamic Port Forwarding

Again, this method relies on SSH access to the remote host. It also requires advanced SOCKS configuration in your browser. You first create a dynamic port forward with SSH:

```
ssh -D <choose_a_port> <user>@<remote_host>
```

Once that tunnel is established, you can configure your browser to use a SOCKS proxy on localhost:<choose_a_port>. This will allow you to access the Web Interface through the tunnel.

To access the Web Interface, you would navigate to http://<remote_host>:8090/hms-mirror (SOCKS proxy will handle the routing).

Concurrency

This controls how many parallel operations can be performed at once. The default is 10 and can be overridden during application startup.

The 'concurrency' setting was previously in the configuration file, but is now only set at startup. The setting in the configuration file will be ignored.

WebUI

To adjust the concurrency setting for the Web Service, add hms-mirror.concurrency.max-threads=n to the startup command.
hms-mirror --service --hms-mirror.concurrency.max-threads=n

CLI

To adjust the concurrency setting, use the -c|--concurrency option when starting the application.

This setting dictates the number of connections made to the various endpoints like both Hive Server 2's (LEFT and RIGHT) and the Metastore Direct connections.

> ⚠ These services need to be able to support these connections.

Web Interface

Starting

The hms-mirror web application is started by running:

```
hms-mirror --service
```

This will start the web application on the default port of 8090. The port can be changed by using the --server.port option during startup.

```
hms-mirror --service --server.port=8080
```

Point your browser to http://server-host:8090/hms-mirror to access the web application.

The web application will use the user home directory to store configuration files, logs, and reports. This isn't as much a concern as it is for the cli version since the web application will manage all this from the user's browser.

The web application works on the premise of a 'session'. Although the application is stateless, the session is used to store the user's configuration and state while they are using the application. This allows the user to navigate the application and make changes without losing their work. Currently, the session is global throughout the application and is not tied to a specific user browser session. An 'hms-mirror' session is NOT synonymous with a browser session.

An instance of the web application can only run ONE session at a time.

Stopping

hms-mirror --stop

Security

Coming Soon...

Where to Start?

There are three methods to building/loading a configuration. From the main page, select 'Initialize'.

web_init_menu

This will bring up the 'Initialize' page where one of the following options can be selected.

web_init.png

> ⚠ Configurations are specific to a Data Strategy ([Strategies](#)). Once created, you can NOT change the Data Strategy but, you can clone the configuration and change the Data Strategy (last option).

Pick an Existing Configuration

Load a previously saved configuration. Changes can be made and saved back to the same file or a new file.

Create a New Configuration

Create a new configuration from scratch.

Clone an Existing Configuration

Using an existing configuration, clone it and change the Data Strategy. This will allow you to maintain any previously configured endpoints to Hive Server 2 and Metastore Direct connections.

Managing the Session Configuration

There are 2 states for a session. The in-memory state and the persisted state.

When you choose to 'Edit' a session configuration, you'll need to 'Save' the configuration for those changes to be applied to the session. 'Saving' the configuration will ONLY update the in-memory state of the session.

session_mngt.png
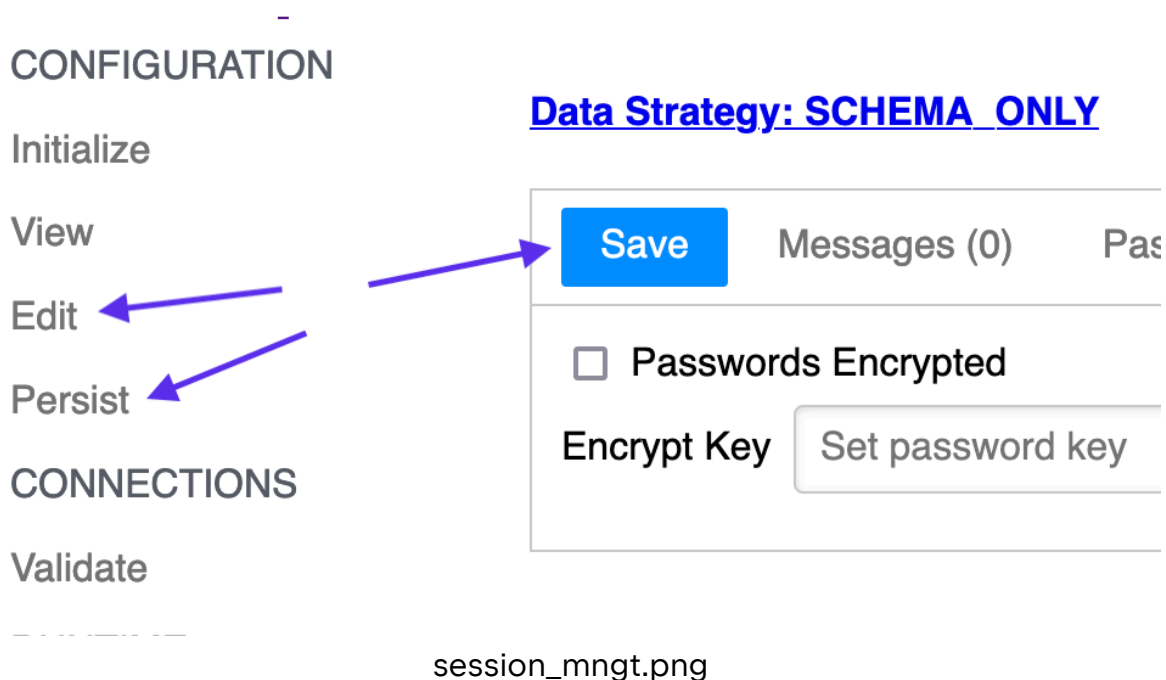
You will need to 'Persist' the session to save it for future use.

CLI Interface

This is the legacy interface and is a Command Line interface that you run from a terminal.

First you need to build a configuration file `default.yaml`, that's placed in `${HOME}/.hms-mirror/cfg`. When you follow the above location and name, this configuration is loaded by default upon application start.

To control the run behavior of the CLI, you can add many commandline parameters. The application output is describe here (Output).

The CLI and Web versions use the same engine and share the same output/results. Although, the web interface has an online viewer for the results. With the CLI version, you need to collect the output, possibly bring it to you localhost and view it through a 'markdown' viewer.

Commandline Help (Options)

Flags	Argument	Description
-accept, --accept		Accept ALL confirmations and silence prompts
-ap, --acid-partition-count		Set the limit of partitions that the ACID strategy will work with. '-1' means no-limit.
-asm, --avro-schema-migration		Migrate AVRO Schema Files referenced in TBLPROPERTIES by 'avro.schema.url'. Without migration it is expected that the file will exist on the other cluster and match the 'url' defined in the schema DDL. If it's not present, schema creation will FAIL. Specifying this option REQUIRES the LEFT and RIGHT cluster to be LINKED. See docs: https://github.com/cloudera-labs/hms-mirror#linking-clusters-storage-layers
-at, --auto-tune		Auto-tune Session Settings for SELECT's and DISTRIBUTION for Partition INSERT's.
-c, --concurrency		Set application concurrency, default is 10
-cfg, --config		Config with details for the HMS-Mirror. Default: $HOME/.hms-mirror/cfg/default.yaml

-cine, --create-if-not-exist		CREATE table/partition statements will be adjusted to include 'IF NOT EXISTS'. This will ensure all remaining sql statements will be run. This can be used to sync partition definitions for existing tables.
-cs, --common-storage		Common Storage used with Data Strategy HYBRID, SQL, EXPORT_IMPORT. This will change the way these methods are implemented by using the specified storage location as an 'common' storage point between two clusters. In this case, the cluster do NOT need to be 'linked'. Each cluster DOES need to have access to the location and authorization to interact with the location. This may mean additional configuration requirements for 'hdfs' to ensure this seamless access.
-cto, --compress-text-output		Data movement (SQL/STORAGE_MIGRATION) of TEXT based file formats will be compressed in the new table.
-d, --data-strategy		Specify how the data will follow the schema. [DUMP, SCHEMA_ONLY, LINKED, SQL, EXPORT_IMPORT, HYBRID, CONVERT_LINKED, STORAGE_MIGRATION, COMMON, ICEBERG_CONVERSION]
-da, --downgrade-acid		Downgrade ACID tables to EXTERNAL tables with purge.
-db, --data		Comma separated list of Databases (upto 100).

base		
-dbo, --database-only		Migrate the Database definitions as they exist from LEFT to RIGHT
-dbp, --db-prefix		Optional: A prefix to add to the RIGHT cluster DB Name. Usually used for testing.
-dbr, --db-rename		Optional: Rename target db to ... This option is only valid when '1' database is listed in -db.
-dbRegEx, --database-regex		RegEx of Database to include in process.
-dbsp, --database-skip-properties		Comma separated list of database properties (regex) to skip during the migration process. This will prevent the property from being set on the target cluster.
-dc, --dist		Build the 'distcp' workplans. Optional argument (PULL, PUSH) to define which cluster is running the distcp commands. Default is PULL.

cp		
-dms, --data-moveme nt-str ategy		Specify how the data will follow the schema. [SQL, DISTCP, MANUAL]
-dp, --decrypt-p assw ord		Used this in conjunction with '-pkey' to decrypt the generated pass code from `-p`.
-ds, --dum p-so urce		Specify which 'cluster' is the source for the DUMP strategy (LEFT|RIGHT).
-dtd, --du mp-t est-d ata		Used to dump a data set that can be feed into the process for testing.
-e, --exec ute		Execute actions request, without this flag the process is a dry-run.
-ep, --exp ort-p artitio		Set the limit of partitions that the EXPORT_IMPORT strategy will work with.

n-co unt		
-epl, --eva luate -parti tion-l ocati on		For SCHEMA_ONLY and DUMP data-strategies, review the partition locations and build partition metadata calls to create them is they c an't be located via 'MSCK'.
-ewd, --ext ernal -ware hous e-dir ector y		The external warehouse directory path. Should not include the nam espace OR the database directory. This will be used to set the LOC ATION database option.
-f, --f lip		Flip the definitions for LEFT and RIGHT. Allows the same config to b e used in reverse.
-fel, - -forc e-ext ernal -loca tion		Under some conditions, the LOCATION element for EXTERNAL tabl es is removed (ie: -rdl). In which case we rely on the settings of the database definition to control the EXTERNAL table data location. B ut for some older Hive versions, the LOCATION element in the data base is NOT honored. Even when the database LOCATION is set, th e EXTERNAL table LOCATION defaults to the system wide warehou se settings. This flag will ensure the LOCATION element remains in the CREATE definition of the table to force it's location.
-glm, --glo bal-lo	<key=v alue>	Comma separated key=value pairs of Locations to Map. IE: /myorig/ data/finance=/data/ec/finance. This reviews 'EXTERNAL' table locat ions for the path '/myorig/data/finance' and replaces it with '/data/e

catio n-ma p		c/finance'. Option can be used alone or with -rdl. Only applies to 'EX TERNAL' tables and if the tables location doesn't contain one of the supplied maps, it will be translated according to -rdl rules if -rdl is s pecified. If -rdl is not specified, the conversion for that table is skip ped.
-h, -- help		Help
-ip, -- in-pla ce		Downgrade ACID tables to EXTERNAL tables with purge.
-is, -- inter medi ate-s torag e		Intermediate Storage used with Data Strategy HYBRID, SQL, EXPO RT_IMPORT. This will change the way these methods are implemen ted by using the specified storage location as an intermediate trans fer point between two clusters. In this case, the cluster do NOT nee d to be 'linked'. Each cluster DOES need to have access to the locati on and authorization to interact with the location. This may mean ad ditional configuration requirements for 'hdfs' to ensure this seamles s access.
-itpo, --ice berg- table -prop erty- overri des	<key=v alue>	Comma separated key=value pairs of Iceberg Table Properties to se t/override.
-iv, -- icebe		Specify the Iceberg Version to use. Specify 1 or 2. Default is 2.

rg-ve rsion		
-ltd, -- load -test- data		Use the data saved by the -dtd option to test the process.
-ma, --mig rate- acid	\<buck et-thr eshold (2)>	Migrate ACID tables (if strategy allows). Optional: ArtificialBucketTh reshold count that will remove the bucket definition if it's below thi s. Use this as a way to remove artificial bucket definitions that were added 'artificially' in legacy Hive. (default: 2)
-mao, --mig rate- acid- only	\<buck et-thr eshold (2)>	Migrate ACID tables ONLY (if strategy allows). Optional: ArtificialBu cketThreshold count that will remove the bucket definition if it's bel ow this. Use this as a way to remove artificial bucket definitions that were added 'artificially' in legacy Hive. (default: 2)
-mn n, -- migra te-no n-nat ive		Migrate Non-Native tables (if strategy allows). These include table definitions that rely on external connections to systems like: HBase, Kafka, JDBC
-mnn o, -- migra te-no n-nat ive-o nly		Migrate Non-Native tables (if strategy allows). These include table definitions that rely on external connections to systems like: HBase, Kafka, JDBC

-np, --no-purge		For SCHEMA_ONLY, COMMON, and LINKED data strategies set RIGHT table to NOT purge on DROP
-o, --output-dir		Output Directory (default: $HOME/.hms-mirror/reports/<yyyy-MM-dd_HH-mm-ss>)
-p, --password		Used this in conjunction with '-pkey' to generate the encrypted password that you'll add to the configs for the JDBC connections.
-pkey, --password-key		The key used to encrypt / decrypt the cluster jdbc passwords. If not present, the passwords will be processed as is (clear text) from the config file.
-po, --property-overrides	<key=value>	Comma separated key=value pairs of Hive properties you wish to set/override.
-pol, --property-overrides-left	<key=value>	Comma separated key=value pairs of Hive properties you wish to set/override for LEFT cluster.
-por, --property-over	<key=value>	Comma separated key=value pairs of Hive properties you wish to set/override for RIGHT cluster.

rides -right		
-pt, --pass -thro ugh	<key=v alue>	Key=value property to pass-through to the configuration. This will all ow you to set properties that are not part of the HMS-Mirror configu ration or part of spring's configuration.
-q, -- quiet		Reduce screen reporting output. Good for background processes with output redirects to a file
-rdl, --rese t-to- defau lt-loc ation		Strip 'LOCATION' from all target cluster definitions. This will allow th e system defaults to take over and define the location of the new da tasets.
-repl ay, --r eplay		Use to replay process from the report output.
-rid, --right -is-di scon necte d		Don't attempt to connect to the 'right' cluster and run in this mode
-ro, -- -read -only		For SCHEMA_ONLY, COMMON, and LINKED data strategies set RI GHT table to NOT purge on DROP. Intended for use with replication distcp strategies and has restrictions about existing DB's on RIGHT and PATH elements. To simply NOT set the purge flag for applicable tables, use -np.

-rr, --reset-right		Use this for testing to remove the database on the RIGHT using CASCADE.
-s, --sync		For SCHEMA_ONLY, COMMON, and LINKED data strategies. Drop and Recreate Schema's when different. Best to use with RO to ensure table/partition drops don't delete data. When used WITHOUT `-tf` it will compare all the tables in a database and sync (bi-directional). Meaning it will DROP tables on the RIGHT that aren't in the LEFT and ADD tables to the RIGHT that are missing. When used with `-ro`, table schemas can be updated by dropping and recreating. When used with `-tf`, only the tables that match the filter (on both sides) will be considered. When used with HYBRID, SQL, and EXPORT_IMPORT data strategies and ACID tables are involved, the tables will be dropped and recreated. The data in this case WILL be dropped and replaced.
-scw, --suppress-cli-warnings		Suppress CLI Warnings from the final on-screen report.
-sdpi, --sort-dynamic-partition-in-serts		Used to set `hive.optimize.sort.dynamic.partition` in TEZ for optimal partition inserts. When not specified, will use prescriptive sorting by adding 'DISTRIBUTE BY' to transfer SQL. default: false
-sf, --skip-f		Skip Features evaluation.

eatur es		
-slc, --skip -link- chec k		Skip Link Check. Use when going between or to Cloud Storage to a void having to configure hms-mirror with storage credentials and li braries. This does NOT preclude your Hive Server 2 and compute en vironment from such requirements.
-slt, --skip -lega cy-tr ansla tion		Skip Schema Upgrades and Serde Translations
-smn, --stor age- migra tion- name spac e		Optional: Used with the 'data strategy STORAGE_MIGRATION to sp ecify the target namespace.
-sms, --stor age- migra tion- strict		Use 'strict' location translations for storage migration.
-so, --skip -opti		Skip any optimizations during data movement, like dynamic sorting or distribute by

mizat ions		
-sp, - -sql- partiti on-c ount		Set the limit of partitions that the SQL strategy will work with. '-1' me ans no-limit.
-sql, --sql- outp ut		. This option is no longer required to get SQL out in a report. That is the default behavior.
-ssc, --ski p-sta ts-co llecti on		Skip collecting basic FS stats for a table. This WILL affect the optim izer and our ability to determine the best strategy for moving data.
-su, - -setu p		Setup a default configuration file through a series of questions
-swt, --sav e-wo rking -tabl es		Save working tables (shadow tables) created during the migration p rocess.
-tef, - -tabl e-exc		Filter tables (excludes) with name matching RegEx. Comparison do ne with 'show tables' results. Check case, that's important. Hive tabl

lude-filter		es are generally stored in LOWERCASE. Make sure you double-quote the expression on the commandline.
-tf, --table-filter		Filter tables (inclusive) with name matching RegEx. Comparison done with 'show tables' results. Check case, that's important. Hive tables are generally stored in LOWERCASE. Make sure you double-quote the expression on the commandline.
-tfp, --table-filter-partition-count-limit		Filter partition tables OUT that are have more than specified here. Non Partitioned table aren't filtered.
-tfs, --table-filter-size-limit		Filter tables OUT that are above the indicated size. Expressed in MB
-to, --transfer-ownership		If available (supported) on LEFT cluster, extract and transfer the tables owner to the RIGHT cluster. Note: This will make an 'exta' SQL call on the LEFT cluster to determine the ownership. This won't be supported on CDH 5 and some other legacy Hive platforms. Beware the cost of this extra call for EVERY table, as it may slow down the process for a large volume of tables.

-tod b, --t ransf er-o wner ship- datab ase		If available (supported) on LEFT cluster, extract and transfer the DB owner to the RIGHT cluster. Note: This will make an 'exta' SQL call on the LEFT cluster to determine the ownership. This won't be supported on CDH 5 and some other legacy Hive platforms.
-totb l, --tr ansfe r-ow nersh ip-ta ble		If available (supported) on LEFT cluster, extract and transfer the tables owner to the RIGHT cluster. Note: This will make an 'exta' SQL call on the LEFT cluster to determine the ownership. This won't be supported on CDH 5 and some other legacy Hive platforms. Beware the cost of this extra call for EVERY table, as it may slow down the process for a large volume of tables.
-tt, --transl ation -type		Translation Strategy when migrating data. (ALIGNED|RELATIVE) Default is RELATIVE
-v, --views -only		Process VIEWs ONLY
-wd, --war ehou se-di recto ry		The warehouse directory path. Should not include the namespace OR the database directory. This will be used to set the MANAGEDLOCATION database option.
-wps, --war	<db=e xt-dir:	The warehouse plans by database. Defines a plan for a database wit

ehou se-pl ans	mngd- dir[,db =ext-di r:mng d-di r]...>	h 'external' and 'managed' directories.

Output

The output from `hms-mirror` will, by default, be sent to `$HOME/.hms-mirror/reports`. Each run will be place in a sub-directory with a timestamp. You can choose to redirect the output to a different location with the `-o` option. In this case the directory will be created if it doesn't exist and the output will be written to that location (without the timestamp sub-directory).

If you wish to have the reports written to a different location AND have the timestamp sub-directory, use a symbolic link redirect the `$HOME/.hms-mirror/reports` directory to the desired output directory.

A report for each **database** processed will be created in the output directory. Files for a database will be prefixed with the database name. This applies to each of the following report/script types.

Application Report

The output report is in markdown format. You can use a markdown renderer to view the report. If you don't have a renderer, you can still read the report, it will just be harder to read.

The report include various stats regarding the run and details for each tables migration process. In this report, you'll find details on "why" a particular table was skipped, or what actions were taken to migrate the table. The report will even list issue encountered during the process.

SQL Scripts

`hms-mirror` will produce the SQL scripts used to migrate the data. These scripts are written to the output directory. The scripts are prefixed with the **database** name.

- `<db_name>_LEFT_Clueanup_execute.sql` – When present, this scripts represents SQL statements that should be run on the LEFT cluster to cleanup artifacts from the migration process.

- `<db_name>_LEFT_execute.sql` – When present, this scripts represents SQL statements that should be run on the LEFT cluster to migrate the data. If the `-e` option was used, the contents of this script will be executed on the LEFT cluster by hms-

mirror. If the -e option was NOT specified, these script can be verified and executed manually on the LEFT cluster.

- <db_name>_RIGHT_execute.sql – When present, this scripts represents SQL statements that should be run on the RIGHT cluster to migrate the data. If the -e option was used, the contents of this script will be executed on the RIGHT cluster by hms-mirror. If the -e option was NOT specified, these script can be verified and executed manually on the RIGHT cluster.

- <db_name>_RIGHT_Clueanup_execute.sql – When present, this scripts represents SQL statements that should be run on the RIGHT cluster to cleanup artifacts from the migration process.

YAML Output

The <db_name>_hms-mirror.yaml file is a full listing of the migration process as a document. Use this file to programmatically determine what actions were taken during the migration process.

Runbook

The <db_name>_runbook.md is a markdown file that is a workbook of 'what' to do. It lays out the steps taken and the steps to be taken to complete the migration process.

distcp Scripts and Workbook

When you include the -dc|--distcp option when running hms-mirror, we'll build a template distcp job for each database that has data to be migrated. The result is a set of **bash scripts** and source files listing the contents to be used in the migration.

Depending other influencing options, there may be a distcp script for the LEFT and RIGHT clusters. The scripts will be prefixed with the database name.

The various **distcp** reports include:

- <db_name>_RIGHT_n_distcp_source.txt – A list of the source directories to be copied to the RIGHT cluster. The n will increment for each one of the jobs created for the database being migrated. These files must be copied to the RIGHT clusters HDFS

filesystem. When running the distcp bash shell script, set the bash environment variable $HCFS_BASE_DIR to set the 'directory' these are copied to.

- <db_name>_RIGHT_distcp_script.sh – The bash script created that will run the distcp jobs. This script will be run on the RIGHT cluster. Review the comments in the script for details on how to run it.

- <db_name>_RIGHT_distcp_workbook.md – A markdown report table that breakdown what will be moved by the process.

** Example distcp Workbook **

Database	Target	Sources
tpcds_bin_partitioned_orc_10		
	hdfs://HOME90/apps/hive/warehouse/tpcds_bin_partitioned_or	hdfs://HDP50/apps/hive/warehouse/tpcds_bin_partitioned_orc_10.db/call_center hdfs://HDP50/apps/hive/warehouse/tpcds_bin_partitioned_orc_10.db/catalog_page hdfs://HDP50/apps/hive/warehouse/tpcds_bin_partitioned_orc_10.db/catalog_returns hdfs://HDP50/apps/hive/warehouse/tpcds_bin_partitioned_orc_10.db/catalog_sales hdfs://HDP50/apps/hive/warehouse/tpcds_bin_partitioned_orc_10.db/customer hdfs://HDP50/apps/hive/warehouse/tpcds_bin_partitioned_orc_10.db/customer_address hdfs://HDP50/apps/hive/warehouse/tpcds_bin_partitioned_orc_10.db/customer_demographics hdfs://HDP50/apps/hive/warehouse/tpcds_bin_partitioned_orc_10.db/date_dim hdfs://HDP50/apps/hive/warehouse/tpcds_bin_partitioned_orc_10.db/household_demographics hdfs://HDP50/apps/hive/warehouse/tpcds_bin_partitioned_orc_10.db/income_band hdfs://HDP50/apps/hive/warehouse/tpcds_bin_partitioned_orc_10.db/inventory hdfs://HDP50/apps/hive/warehouse/tpcds_bin_partiti

c_10. db	oned_orc_10.db/item hdfs://HDP50/apps/hive/warehouse/tpcds_bin_partitioned_orc_10.db/promotion hdfs://HDP50/apps/hive/warehouse/tpcds_bin_partitioned_orc_10.db/reason hdfs://HDP50/apps/hive/warehouse/tpcds_bin_partitioned_orc_10.db/ship_mode hdfs://HDP50/apps/hive/warehouse/tpcds_bin_partitioned_orc_10.db/store hdfs://HDP50/apps/hive/warehouse/tpcds_bin_partitioned_orc_10.db/store_returns hdfs://HDP50/apps/hive/warehouse/tpcds_bin_partitioned_orc_10.db/store_sales hdfs://HDP50/apps/hive/warehouse/tpcds_bin_partitioned_orc_10.db/time_dim hdfs://HDP50/apps/hive/warehouse/tpcds_bin_partitioned_orc_10.db/warehouse hdfs://HDP50/apps/hive/warehouse/tpcds_bin_partitioned_orc_10.db/web_page hdfs://HDP50/apps/hive/warehouse/tpcds_bin_partitioned_orc_10.db/web_returns hdfs://HDP50/apps/hive/warehouse/tpcds_bin_partitioned_orc_10.db/web_sales hdfs://HDP50/apps/hive/warehouse/tpcds_bin_partitioned_orc_10.db/web_site

Logs

Logs, as of 1.6.5.6 are now in the same output directory as the reports.

REST Interface (Technical Preview)

Start hms-mirror with the --service option.

```
hms-mirror --service
```

> ⚠️ This feature is in technical preview and is subject to change. It is not complete and may not be fully functional.

The REST Swagger documentation is available at http://server-host:8090/hms-mirror/swagger-ui/index.html.

The REST base endpoint is http://server-host:8090/hms-mirror/api/v1.

The REST service controls the 'current' session in hms-mirror. This session is the same as the session available through the Web Interface.

Reports

Both the Web ([Web Interface](#)) and CLI ([CLI Interface](#)) interfaces generate reports. By default, the reports are generated in the $HOME/.hms-mirror/reports directory with the timestamp as the 'name' of the report. The report is directory of several files that include configurations, conversions, and job 'yaml' files of what was done. Each database has their own reports. The database name prefixes the report file name for the various reports.

From the CLI, you'll need to download the report directory to your local machine and use a 'Markdown' viewer to read it easily.

The Web interface has built in report viewing capabilities and makes it much easier to review the reports.

Output Directory

You can control where the reports are generated by using the `-o <directory>` option. Default behavior is to use the `$HOME/.hms-mirror/reports` directory. If you specify a different directory, the reports will be generated in the specific directory. If you use the `-o <dir>` option with the Web Interface, the reports will be in that 'base directory' with a subdirectory of the timestamp of the report.

For the CLI, the report will be placed in the directory. If the directory exists, the report will be placed in an increment of that directory to avoid overwriting an existing report. EG: /my/report -> /my/report_1

Options

There are several views for the reports that include a detailed view of the process, workbooks for 'distcp', scripts for 'distcp' , job 'yaml' files, the configuration used to run the job, etc.

You have options to view all these reports within the Web Interface. You can further 'download' a zip file of the report contents and view them locally.

To keep things organized, there is an 'archive' option that will move the reports to an 'archive' directory. This is useful to help keep the reports directory clean and organized. If

you need to view those reports again, you can always move them back to the reports directory. Although, that process is manual.

> ⚠️ If you run the process via the CLI and are using defaults, you'll be able to view the reports through the Web Interface since they are both using the same location.

Quick Start Scenarios

- On-prem Sidecar Migrations ([On Prem Legacy Hive to Non-Legacy Hive](#))

- Hybrid Migrations ([Hybrid Data LakeHouse](#))

- Cloud to Cloud DR with Datahub ([Cloud to Cloud DR (AWS)](#))

Pre-Requisites

User Home Directory

A lot of what happens in `hms-mirror` depends on a 'user' $HOME$ directory. Whatever user runs `hms-mirror` we will use the $HOME environment variable to store configurations, logs, and reports.

If your environment doesn't follow a typical home environment setup, you will NEED to set the $HOME environment variable before running the application to ensure that the application can find and store the necessary files.

Hive/TEZ Properties Whitelist Requirements

HiveServer2 has restrictions on what properties can be set by the user in a session. To ensure that `hms-mirror` will be able to set the properties it needs, add `hive.security.authorization.sqlstd.confwhitelist.append` ([https://cwiki.apache.org/confluence/display/Hive/Configuration+Properties#Configurat ionProperties-hive.security.authorization.sqlstd.confwhitelist.append](https://cwiki.apache.org/confluence/display/Hive/Configuration+Properties#ConfigurationProperties-hive.security.authorization.sqlstd.confwhitelist.append)) property in the HiveServer2 Advanced Configuration Snippet (Safety Valve) for `hive-site.xml` with at least the following value(s) so `hms-mirror` can set the properties it needs:

```
tez\.grouping\..*
```

Backups

DO NOT SKIP THIS!!!

The `hms-mirror` process DOES 'DROP' tables when asked. If those tables *manage* data like a *legacy* managed, *ACID*, or *external.table.purge=true* scenario, we do our best NOT to DROP those and ERROR out. But, protect yourself and make backups of the areas you'll be working in.

HDFS Snapshots

Use HDFS Snapshots to make a quick backup of directories you'll be working on. Do this, especially in the *LEFT* cluster. We only drop tables, so a snapshot of the database directory is good. BUT, if you are manually doing any DROP DATABASE <x> CASCADE

operations, that will delete the snapshotted directory (and the snapshot). In this case, create the *snapshot* one level above the database directory.

Metastore Backups

Take a DB backup of your metastore and keep it in a safe place before starting.

Shared Authentication

The clusters must share a common authentication model to support cross-cluster HDFS access when HDFS is the underlying storage layer for the datasets. This means that a **kerberos** ticket used in the RIGHT cluster must be valid for the LEFT cluster.

For cloud storage, the two clusters must have rights to the target storage bucket.

If you can `distcp` between the clusters ([Linking Cluster Storage Layers](#)), you have the basic connectivity required to start working with `hms-mirror`.

Warning

Building METADATA

Rebuilding METADATA can be an expensive scenario. Especially when you are trying to reconstruct the entire metastore in a short time period, consider this in your planning. Know the number of partitions and buckets you will be moving and account for this. Test on smaller datasets (volume and metadata elements). Progress to testing higher volumes/partition counts to find limits and make adjustments to your strategy.

Using the SQL and EXPORT_IMPORT strategies will move metadata AND data, but rebuilding the metastore elements can be pretty expensive. So consider migrating the metadata separately from the data (distcp) and use MSCK on the RIGHT cluster to discover the data. This will be considerably more efficient.

If you will be doing a lot of metadata work on the RIGHT cluster. That cluster also serves a current user base; consider setting up separate HS2 pods for the migration to minimize the impact on the current user community. Isolate Migration Activities ("Isolate Migration Activities" in "Optimizations")

Partition Handling for Data Transfers

There are three settings in the configuration to control how and to what extent we'll attempt to migrate *DATA* for tables with partitions.

For non-ACID/transactional tables the setting in:

```
hybrid:
  exportImportPartitionLimit: 100
  sqlPartitionLimit: 500
```

Control both the HYBRID strategy for selecting either EXPORT_IMPORT or SQL and the SQL *LIMIT* for how many partitions we'll attempt. When the SQL limit is exceeded, you will need to use SCHEMA_ONLY to migrate the schema followed by distcp to move the data.

For ACID/transactional tables, the setting in:

```
migrateACID:
  partitionLimit: 500
```

Effectively draws the same limit as above.

Why do we have these limits? Mass migration of datasets via SQL and EXPORT_IMPORT with many partitions is costly and NOT very efficient. It's best that when these limits are reached that you separate the METADATA and DATA migration to DDL and distcp.

Permissions

We use a cross-cluster technique to back metadata in the RIGHT cluster with datasets in the LEFT cluster for data strategies: LINKED, HYBRID, EXPORT_IMPORT, SQL, and SCHEMA_ONLY (with `-ams` AVRO Migrate Schema).

See Linking Clusters Storage Layers (Linking Cluster Storage Layers) for details on configuring this state.

Permissions

In both the METADATA and STORAGE phases of `hms-mirror` the RIGHT cluster will reach down into the LEFT clusters storage layer to either *use* or *copy* the data.

`hms-mirror` access each cluster via JDBC and use the RIGHT cluster for *storage* layer access.

When the RIGHT cluster is using 'non-impersonation' (hive `doas=false`), the *hive* service account on the **RIGHT** cluster (usually `hive`) needs access to the storage layer on the **LEFT** cluster to use this data to support sidecar testing, where we use the data of the LEFT cluster but *mirror* the metadata.

> ⚠ Having Ranger on both clusters helps because you can create additional ACLs to provide the access required.

OR

> ⚠ Checked permissions of '<submitting_user>': Found that the '<submitting_user>' user was NOT the owner of the files in these directories. The user running the process needs to be in 'dfs.permissions.superusergroup' for the LEFT clusters 'hdfs' service. Ambari 2.6 has issues setting this property: https://jira.cloudera.com/browse/EAR-7805

> ⚠ Follow workaround above or add user to the 'hdfs' group. I had to use '/var/lib/ambari-server/resources/scripts/configs.py' to set it manually for Ambari.

> ⚠ `sudo ./configs.py --host=k01.streever.local --port=8080 -u admin -p admin -n hdp50 -c hdfs-site -a set -k dfs.permissions.superusergroup -v hdfs_admin`

Linking Cluster Storage Layers

For the `hms-mirror` process to work, it relies on the RIGHT clusters' ability to *SEE* and *ACCESS* data in the LEFT clusters HDFS namespace. This is the same access/configuration required to support DISTCP for an HA environment and accounts for failovers.

We suggest that `distcp` operations be run from the RIGHT cluster, which usually has the greater 'hdfs' version in a migration scenario.

The RIGHT cluster HCFS namespace requires access to the LEFT clusters HCFS namespace. RIGHT clusters with a greater HDFS version support **LIMITED** functionality for data access in the LEFT cluster.

NOTE: This isn't designed to be a permanent solution and should only be used for testing and migration purposes.

Goal

What does it take to support HDFS visibility between these two clusters?

Can that integration be used to support the Higher Clusters' use of the Lower Clusters HDFS Layer for distcp AND Hive External Table support?

Scenario #1

HDP 2.6.5 (Hadoop 2.7.x)

Kerberized - sharing same KDC as CDP Base Cluster

Configuration Changes

The *namenode kerberos* principal MUST be changed from `nn` to `hdfs` to match the namenode principal of the CDP cluster.

Note: You may need to add/adjust the `auth_to_local` settings to match this change.

If this isn't done, `spark-shell` and `spark-submit` will fail to initialize. When changing this in Ambari on HDP, you will need to *reset* the HDFS zkfc `ha` zNode in Zookeeper and reinitialize the hdfs `zkfc`.

From a Zookeeper Client: `/usr/hdp/current/zookeeper-client/bin/zkCli.sh -server`

localhost

```
rmr /hadoop-ha
```

Initialize zkfc

```
hdfs zkfc -formatZK
```

core-site.xml

```
hadoop.rpc.protection=true
dfs.encrypt.data.transfer=true
dfs.encrypt.data.transfer.algorithm=3des
dfs.encrypt.data.transfer.cipher.key.bitlength=256
```

CDP 7.1.4 (Hadoop 3.1.x)

Kerberized, TLS Enabled

Configuration Changes

Requirements that allow this (upper) cluster to negotiate and communicate with the lower environment.

Cluster Wide hdfs-site.xml Safety Value

```
ipc.client.fallback-to-simple-auth-allowed=true
```

HDFS Service Advanced Config hdfs-site.xml

```
# For this Clusters Name Service
dfs.internal.nameservices=HOME90

# For the target (lower) environment HA NN Services
dfs.ha.namenodes.HDP50=nn1,nn2
dfs.namenode.rpc-address.HDP50.nn1=k01.streever.local:8020
dfs.namenode.rpc-address.HDP50.nn2=k02.streever.local:8020
dfs.namenode.http-address.HDP50.nn1=k01.streever.local:50070
dfs.namenode.http-address.HDP50.nn2=k02.streever.local:50070
dfs.namenode.https
```

```
  address.HDP50.nn1=k01.streever.local:50471
dfs.namenode.https-address.HDP50.nn2=k02.streever.local:50470
dfs.client.failover.proxy.provider.HDP50=org.apache.hadoop.hdfs.se
rver.namenode.ha.ConfiguredFailoverProxyProvider

# For Available Name Services
dfs.nameservices=HOME90,HDP50
```

Running distcp from the RIGHT Cluster

NOTE: Running distcp from the **LEFT** cluster isn't supported since the hcfs client is not forward compatible.

Copy 'from' Lower Cluster

```
hadoop distcp hdfs://HDP50/user/dstreev/sstats/queues/2020-10.txt
/user/dstreev/temp
```

Copy 'to' Lower Cluster

```
hadoop distcp
/warehouse/tablespace/external/hive/cvs_hathi_workload.db/queue/20
20-10.txt hdfs://HDP50/user/dstreev/temp
```

Sourcing Data from Lower Cluster to Support Upper Cluster External Tables

Proxy Permissions

The lower cluster must allow the upper clusters *Hive Server 2* host as a 'hive' proxy. The setting in the lower clusters *custom* core-site.xml may limit this to that clusters (lower) HS2 hosts. Open it up to include the upper clusters HS2 host.

Custom core-site.xml in Lower Cluster

```
hadoop.proxyuser.hive.hosts=*
```

Credentials from the 'upper' cluster will be projected down to the 'lower' cluster. The hive user in the upper cluster, when running with 'non-impersonation' will require access to the datasets in the lower cluster HDFS.

For table creation in the 'upper' clusters Metastore, a permissions check will be done on the lower environments directory for the submitting user. So, both the service user AND `hive` will require access to the directory location specified in the lower cluster.

When the two clusters *share* accounts, and the same accounts are used between environments for users and service accounts, then access should be simple.

When a different set of accounts are used, the 'principal' from the upper clusters service account for 'hive' and the 'user' principal will be used in the lower cluster. This means additional HDFS policies in the lower cluster may be required to support this cross-environment work.

Configuration

LEFT and RIGHT Clusters

hms-mirror defines clusters as LEFT and RIGHT. The LEFT cluster is the source of the metadata and the RIGHT cluster is the target. The LEFT cluster is usually the older cluster version. Regardless, under specific scenario's, hms-mirror will use an HDFS client to check directories and move small amounts of data (AVRO schema files). hms-mirror will depend on the configuration of the node it's running on to locate the 'hcfs filesystem'. This means that the /etc/hadoop/conf directory should contain all the environments settings to successfully connect to hcfs (Hadoop Compatible File System.

The configuration is done via a 'yaml' file, details below.

There are two ways to get started:

- The first time you run hms-mirror and it can't find a configuration, it will walk you through building one and save it to $HOME/.hms-mirror/cfg/default.yaml. Here's what you'll need to complete the setup:

  - URI's for each clusters HiveServer2

  - STANDALONE jar files for EACH Hive version.

    - We support the Apache Hive based drivers for Hive 1 and Hive2/3.

    - Recently added support for the Cloudera JDBC driver for CDP.

  - Username and Password for non-kerberized connections.

    - Note: hms-mirror will only support one kerberos connection. For the other, use another AUTH method.

  - The hcfs (Hadoop Compatible FileSystem) protocol and prefix used for the hive table locations in EACH cluster.

- Use the template yaml ([Default Configuration Template](#)) for reference and create a default.yaml in the running users $HOME/.hms-mirror/cfg directory.

You'll need JDBC driver jar files that are * *specific* to the clusters you'll integrate. If the **LEFT** cluster isn't the same version as the **RIGHT** cluster, don't use the same JDBC jar file, especially when integrating Hive 1 and Hive 3 services. The Hive 3 driver is NOT backwardly compatible with Hive 1.

See the running ([Running](#)) section for examples on running `hms-mirror` for various environment types and connections.

Setup

Binary Package

Download the latest version of `hms-mirror-<version>-dist.tar.gz`
(https://github.com/cloudera-labs/hms-mirror/releases)

HMS-Mirror Setup from Binary Distribution

On the edgenode:

- Remove previous install directory `rm -rf hms-mirror-install-<version>`

  - If you don't remove the previous install directory, the default `tar` behaviour will NOT overwrite the existing directory, hence you won't get the new version.

- Expand the tarball `tar zxvf hms-mirror-<version>-dist.tar.gz`.

  > ⚠️ This produces a child `hms-mirror-install-<version>` directory.

- Make sure you **STOP** any previous release of `hms-mirror` before attempting install.

  - `hms-mirror --stop`

- Two options for installation:

  - As the root user (or `sudo`), run `hms-mirror-install-<version>/setup.sh`. This will install the `hms-mirror` packages in `/usr/local/hms-mirror` and create symlinks for the executables in `/usr/local/bin`. At this point, `hms-mirror` should be available to all user and in the default path.

  - As the local user, run `hms-mirror-install-<version>/setup.sh`. This will install the `hms-mirror` packages in `$HOME/.hms-mirror` and create symlink in `$HOME/bin`. Ensure `$HOME/bin` is in the users path and run `hms-mirror`.

DO NOT RUN `hms-mirror` *from the installation directory.*

If you install both options, your environment PATH will determine which one is run. Make

note of this because an upgrade may not be reachable.

Quick Start

After installation (above), run `hms-mirror` for the particular interface that interests you.

Web UI

> Run:
>
> `hms-mirror --service`
>
> Open a browser to `http://hostname:8090/hms-mirror` to access the Web UI.

CLI

> `hms-mirror` requires a configuration file describing the LEFT (source) and RIGHT (target) cluster connections. There are two ways to create the config:
>
> - Use the default config template ([Default Configuration Template](#))) as a starting point. Edit and place a copy here `$HOME/.hms-mirror/cfg/default.yaml`.
>
> - `hms-mirror --setup` - Prompts a series of questions about the LEFT and RIGHT clusters to build the default configuration file.

If either or both clusters are Kerberized, please review the detailed configuration guidance here (["Running Against a LEGACY (Non-CDP) Kerberized HiveServer2" in "Running"](#)) and here (["Kerberized Connections" in "Running"](#)).

General Guidance

- Run `hms-mirror` from the RIGHT cluster on an Edge Node.

> ⚠️ `hms-mirror` is built (default setting) with CDP libraries and will connect natively using those libraries. The edge node should have the hdfs client installed and configured for that cluster. The application will use this connection for some migration strategies.

- If running from the LEFT cluster, note that the `-ro/--read-only` feature examines HDFS on the RIGHT cluster. The HDFS client on the LEFT cluster may not support this access.

- Connecting to HS2 through KNOX (in both clusters, if possible) reduces the complexities of the connection by removing Kerberos from the picture.

- The libraries will only support a Kerberos connection to a 'single' version of Hadoop at a time. This is relevant for 'Kerberized' connections to Hive Server 2. The default libraries will support a kerberized connection to a CDP clusters HS2 and HDFS. If the LEFT (source) cluster is Kerberized, including HS2, you will need to make some adjustments.

  - The LEFT clusters HS2 needs to support any auth mechanism BUT Kerberos.

  - Use an Ambari Group to setup an independent HS2 instance for this exercise or use KNOX.

Hive JDBC Drivers and Configuration

hms-mirror requires JDBC drivers to connect to the various end-points needed to perform tasks. The LEFT and RIGHT cluster endpoints for HiveServer2 require the standalone JDBC drivers that are specific to that Hive version.

hms-mirror supports the Apache Hive packaged **standalone** drivers that are found with your distribution. You can find a copy of this driver in:

Platform	Driver Location/Pattern
HDP	/usr/hdp/current/hive-client/jdbc/hive-jdbc-<hive-platform-version>-standalone.jar
CDP	/opt/cloudera/parcels/CDH/jars/hive-jdbc-<hive-platform-version>-standalone.jar

For CDP, we also support to Cloudera JDBC driver found and maintained at on the Cloudera Hive JDBC Downloads Page (https://www.cloudera.com/downloads/connectors/hive/jdbc). Note that the URL configurations between the Apache and Cloudera JDBC drivers are different.

Hive JDBC Drivers need to be inline with the version of HS2 you're connecting to. If the cluster is an HDP cluster, get the appropriate **standalone** driver from that cluster. These drivers(jar files) should be stored locally on the machine running hms-mirror and referenced in the configuration file.

> ⚠️ Do NOT put these drivers in ${HOME}/.hms-mirror/aux_libs or any sub-directory of that location. `hms-mirror` connects to different versions of Hive and the drivers need to be specific to the version of Hive you're connecting to. To do this, we need to manage the classpath and the drivers in a more controlled manner. They should NOT be in the applications main classpath

> which includes jar files in `$HOME/.hms-mirror/aux_libs`, this will cause
> connectivity issues.

Web UI

Hive Server 2 Configuration

hs2_cfg.png

CLI

```
hiveServer2:
  uri: "<cloudera_jdbc_url>"
  driverClassName: "com.cloudera.hive.jdbc.HS2Driver"
  connectionProperties:
    user: "xxx"
    password: "xxx"
```

Starting with the Apache Standalone driver shipped with **CDP 7.1.8 cummulative** hot fix parcels, you will need to include additional jars in the configuration jarFile configuration, due to some packaging adjustments.

For example: jarFile: "<cdp_parcel_jars>/hive-jdbc-3.1.3000.7.1.8.28-1-standalone.jar:<cdp_parcel_jars>/log4j-1.2-api-2.18.0.jar:<cdp_parcel_jars>/log4j-api-2.18.0.jar:

`<cdp_parcel_jars>/log4j-core-2.18.0.jar"` NOTE: The jar file with the Hive Driver MUST be the first in the list of jar files.

The Cloudera JDBC driver shouldn't require additional jars.

Kerberized HS2 Connections

We currently have validated **kerberos** HS2 connections to CDP clusters using the Hive JDBC driver you'll find in your target CDP distribution.

> ⚠️ Connections to Kerberized HS2 endpoints on NON-CDP clusters is NOT currently supported. You will need to use KNOX in HDP to connect to a kerberized HS2 endpoint. For CDH, you can setup a non-kerberized HS2 endpoint to support the migration.

> ℹ️ This process has CHANGED compared to v1.x of `hms-mirror`. Please adjust your configurations accordingly.

We NO LONGER need to have the hive JDBC driver in the `aux_libs` directory ($HOME/.hms-mirror/aux_libs). The driver should be stored locally on the machine running `hms-mirror` and referenced in the configuration file via the `jarFile` attribute. Follow the same procedure as above for **Kerberized** connections as is done for non-kerberized connections.

hs2_kerb_config.png

At this point, just like the previous version of hms-mirror, you'll need to have a valid kerberos ticket on the machine running hms-mirror. This is required to authenticate to the kerberized HS2 endpoint.

REMOVE all 'hive' related JDBC jar files from aux_libs. Leaving them there WILL cause conflicts during the service startup.

Validating HS2 Connectivity

Once you have everything configured, you can validate all connections required by hms-mirror through the 'CONNECTIONS --> Validate' left menu option in the UI. This will test the connectivity to the various endpoints required by hms-mirror.

HDP 3 Connections

The JDBC driver for HDP Hive 3 has some embedded classes for log4j that conflict with the log4j classes in the hms-mirror application. To resolve this, you can use the Cloudera Apache JDBC driver for HDP 3 Hive. This driver is compatible with HDP 3 and does not have the log4j conflict.

Hive JDBC Driver Connection Pool Settings

Hive 1/2, which includes HDP 2.x and CDH 5.x / 6.x environments will use the Apache Commons DBCP2 Connection Pool libraries. Hive 3/4, which includes HDP 3.x and CDH 7.x environments will use the HikariCP Connection Pool libraries.

There are a few settings that you can adjust in the hms-mirror configuration file to tune the connection pool settings.

For the DBCP2 connection pool, you can set the maxWaitMillis setting, which has a default of 5000 milliseconds.

For the HikariCP connection pool, the following connection properties can be adjusted: connectionTimeout (default: 60000 milliseconds) validationTimeout (default: 30000 milliseconds) initializationFailTimeout (default: 10000 milliseconds)

The default setting have been pretty successful in our testing, but you can adjust these settings to meet your needs.

CLI

The DBCP2 connection pool settings can be adjustment through the CLI via the `-pt|--pass-through` option using one of more of the following setting:

- `dbcp2.maxWaitMillis`

EG: `-pt dbcp2.maxWaitMillis=10000`

The Hikari connection pool settings can be adjusted through the CLI via the `-pt|--pass-through` option using one of more of the following setting:

- `hikari.connectionTimeout`

- `hikari.validationTimeout`

- `hikari.initializationFailTimeout`

> ℹ️ You can have multiple `-pt` options on the command line

EG: `-pt hikari.connectionTimeout=60000 -pt hikari.validationTimeout=30000 -pt hikari.initializationFailTimeout=10000`

Web UI

Hive Server 2

Driver Class	org.apache.hive.jdbc.HiveDriver
JDBC URI	jdbc:hive2:// c201.streever.local:2181/;transportMode=http;httpPath=cliservic ......
Username	hms-mirror
Password	Set Hive Server2 Password
DBCP2 Max Wait(ms)	5000
Hikari Connection Timeout(ms)	
Hikari Validation Timeout(ms)	
Hikari Initialization Fail Timeout(ms)	
JDBC Jar File(s)	/opt/cloudera/parcels/CDH/jars/hive-jdbc-3.1.3000.7.3.1.0-197-standalone.jar:/opt/cloudera/parcels/CDH/jars/log4j-core-2.18.0.jar:/opt/cloudera/parcels/CDH/jars/log4j-

hs2_cp_settings.png

Filters ▾   Attributes ▾   Cluster(s) ▾   Mappings ▾

Databases

[   ] Add

filter_databases.png

Config File

```
clusters:
  LEFT:
    environment: "LEFT|RIGHT"
    hiveServer2:
```

```yaml
        uri: "..."
    connectionProperties:
      maxWaitMillis: "5000" # DBCP2 connection pool setting
      hikari.validationTimeout: "30000"
      hikari.initializationFailTimeout: "10000"
      hikari.connectionTimeout: "60000"
```

Example URL's

CDP Hive via Knox Gateway

Doesn't require Kerberos. Knox is SSL, so depending on whether you've self-signed your certs you may need to make adjustments.

- Apache Hive and CDP Packaged Apache Hive JDBC Driver

```
jdbc:hive2://s03.streever.local:8443/;ssl=true;transportMode=http;
httpPath=gateway/cdp-proxy-
api/hive;sslTrustStore=/Users/dstreev/bin/certs/gateway-client-
trust.jks;trustStorePassword=changeit
```

- Cloudera JDBC Driver

```
jdbc:hive2://s03.streever.local:8443;transportMode=http;AuthMech=3
;httpPath=gateway/cdp-proxy-api/hive;SSL=1;AllowSelfSignedCerts=1
```

CDP Hive direct with Kerberos

When connecting to via Kerberos, configure the jar files the same way as non-kerberized connections.

- Apache Hive and CDP Packaged Apache Hive JDBC Driver

```
jdbc:hive2://s04.streever.local:10001/;ssl=true;transportMode=http
;httpPath=cliservice;sslTrustStore=/home/dstreev/bin/certs/gateway
-client-
trust.jks;trustStorePassword=changeit;principal=hive/_HOST@STREEVE
R.LOCAL
```

> ⚠️ NOTE: This configuration includes a certificate reference for SSL. If you're using self-signed certs, you'll need to adjust the `sslTrustStore` and `trustStorePassword` values.

- Cloudera JDBC Driver

```
jdbc:hive2://s04.streever.local:10001;transportMode=http;AuthMech=
1;KrbRealm=STREEVER.LOCAL;KrbHostFQDN=s04.streever.local;KrbServic
eName=hive;KrbAuthType=2;httpPath=cliservice;SSL=1;AllowSelfSigned
Certs=1
```

HDP2 HS2 with No Auth

Since CDP is usually kerberized AND `hms-mirror` doesn't support the simultanous connections to 2 different kerberos environments, I've setup an HS2 on HDP2 specifically for this effort. NOTE: You need to specify a `username` when connecting to let Hive know what the user is. No password required.

- Apache Hive Standalone Driver shipped with HDP2.

```
jdbc:hive2://k02.streever.local:10000
```

Direct Metastore DB Access

The `LEFT` and `RIGHT` configurations also suppport 'direct' metastore access to collect detailed partition information. The support this feature, get the JDBC driver that is appropriate for your metastore(s) backend dbs and place it in `$HOME/.hms-mirror/aux_libs` directory.

Metastore JDBC Drivers and Configuration

For some data points, we revert to direct access to the metastore RDBMS. To support this, you need to place the appropriate JDBC drivers for the metastore RDBMS in $HOME/.hms-mirror/aux_libs directory.

You'll have to get the drivers from your RDBMS vendor.

Web UI

metastore_direct_cfg.png

CLI

Storage Systems (Distributed File Systems)

By default, `hms-mirror` is built with native support to communicate with 'hdfs', and 'ozone' distributed file systems. The `hms-mirror` can be extended to support other distributed file systems by implementing an `hcfs` (hadoop compatible filesystem) interface. These include Amazon S3, Google Cloud Storage, Azure Blob Storage, etc.

To support communication with any of these cloud platforms or other distributed files systems, you'll need to ensure the libraries needed to support that communication are available in the classpath for 'hms-mirror'.

Make sure these are available in the classpath for `hms-mirror` **BEFORE** you start the application (Web and CLI).

We'll cover the libraries in the following sections. These libraries need to be copied to the `$HOME/. hms-mirror/aux_libs` directory.

All these libraries are available in the Cloudera distribution. The below file listings have been sourced through our community testing and may not be exhaustive. If you find we've missed any, please let us know by logging an issue at hms-mirror Github Issues (https://github.com/cloudera-labs/hms-mirror/issues)

Amazon S3

This obviously includes Amazon S3, but also includes other S3 compatible storage systems like Minio, etc. that are compatible with the S3 API.

```
hadoop-aws-<platform-version>.jar
aws-java-sdk-bundle-<platform-version>.jar
ranger-raz-hook-s3-<platform-version>.jar
```

Microsoft Azure

abfs

```
hadoop-azure-<platform-version>9.jar
```

```
ranger-raz-hook-abfs-<platform-version>.jar
```

Google Cloud Storage (GFS)

```
google-cloud-storage-<platform-version>.jar
```

Password Security

Secure Passwords in Configuration

There are multiple passwords stored in the configuration files. By default, the passwords are in clear text in the configuration file. This usually isn't an issue since the file can be protected at the UNIX level from peering eyes. But if you need to protect those passwords, `hms-mirror` supports storing an encrypted version of the password in the configuration.

When you're using this feature, you need to have a `password-key`. This is a key used to encrypt and decrypt the password in the configuration. The **same** `password-key` must be used for **ALL** passwords in the configuration file.

WEB UI

Passwords are saved in the configuration and can easily be encrypted and decrypted using the Web UI. If the password (s) are encrypted, the 'Passwords Encrypted' checkbox will be checked.

pwd_mngd.png

If the passwords are encrypted, you'll need to specify the 'Encrypt Key' before running or connection to any endpoints. Set the 'Encrypt Key' and click 'Save' to save the key for the session.

> ⚠️ The 'Encrypt Key' is only used for the current session and will NOT be saved if you 'persist' the session.

Once encrypted, you'll need to specify the 'password key' with that session to decrypt them for use.

CLI

Generate the Encrypted Password

Use the `-pkey` and `-p` options of `hms-mirror` to generate and decrypt the password(s).

`hms-mirror -pkey cloudera -p have-a-nice-day`

Will generate:

```
=== Errors ===
    38:Password en/de crypt

=== Warnings ===
    56:Encrypted password: HD1eNF8NMFahA2smLM9c4g==
```

Ignore the error 38, it's just a warning that the password is being encrypted. The encrypted password is the value after the `Encrypted password:` string.

Copy this encrypted password and place it in your configuration file for the JDBC connection. Repeat for the other passwords, if it's different, and paste it in the configuration as well.

Running hms-mirror with Encrypted Passwords

Using the **same** `-pkey` you used to generate the encrypted password, we'll run `hms-mirror`

`hms-mirror -db <db> -pkey cloudera ...`

When the `-pkey` option is specified **WITHOUT** the `-p` option (used previously), `hms-mirror` will understand to * *decrypt ** the configuration passwords before connecting to jdbc. If you receive jdbc connection exceptions, recheck the `-pkey` and encrypted password from before.

Testing the Encrypted Password

If you're unsure if the password is being decrypted correctly, you can use the `-dp` option to decrypt the hashed password and print it to the console.

`hms-mirror -pkey cloudera -dp HD1eNF8NMFahA2smLM9c4g==`

Will generate:

```
=== Errors ===
    38:Password en/de crypt

=== Warnings ===
    57:Decrypted password: have-a-nice-day
```

Again, ignore the error 38, it's just a warning that the password is being decrypted. The decrypted password is the value after the `Decrypted password:` string.

This should match the password you used to generate the encrypted password.

Memory Settings

The default memory footprint of hms-mirror is set in the startup script here (https://github.com/cloudera-labs/hms-mirror/blob/560166ca90c0d7ce243f8215ba9f29ca1a10cba2/bin/hms-mirror#L70) which declares an 8GB max footprint.

If you need to increase this, you can opt to export APP_JAVA_OPTS before running hms-mirror. You'll need to include:

- Xms

- Xmx

- Garbage Collection info.

For example:

export APP_JAVA_OPTS="-Xms8192m -Xmx16384m -XX:+UseG1GC

This raises the memory limit to 16Gb.

Default Configuration Template

Use this as a template for the `default.yaml` configuration file used by the `cli` interface. You can also build a configuration file in the 'web' interface and reference it in the `cli` interface.

```yaml
# Copyright 2021 Cloudera, Inc. All Rights Reserved.
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
#       http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing,
software
# distributed under the License is distributed on an "AS IS"
BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.
# See the License for the specific language governing permissions
and
# limitations under the License.

transfer:
  # Optional (default: 4)
  concurrency:        10
  # Optional (default: 'transfer_')
  transferPrefix:      "hms_mirror_transfer_"
  # This directory is appended to the 'clusters:...:hcfsNamespace'
value to store the transfer package for hive export/import.
  # Optional (default: '/apps/hive/warehouse/export_')
  exportBaseDirPrefix: "/apps/hive/warehouse/export_"
clusters:
  LEFT:
    # Set for Hive 1/2 environments
```

```yaml
    # Remove, see 'platformType' legacyHive:     true
    # platform type
    platformType: 'HDP2'
    # Is the 'Hadoop COMPATIBLE File System' used to prefix data
locations for this cluster.
    # It is mainly used as the transfer location for metadata
(export)
    # If the primary storage for this cluster is 'hdfs' than use
'hdfs://...'
    # If the primary storage for this action is cloud storage, use
the
    #     cloud storage prefix. IE: s3a://my_bucket
    hcfsNamespace: "<hdfs://namespace>"
    hiveServer2:
      # URI is the Hive JDBC URL in the form of:
      # jdbc:hive2://<server>:<port>
      # See docs for restrictions
      uri:      "<LEFT-cluster-jdbc-url>"
      connectionProperties:
        user:     "*****"
        password: "*****"
    # Standalone jar file used to connect via JDBC to the LEFT
environment Hive Server 2
    # NOTE-1: Hive 3 jars will NOT work against Hive 1.  The
protocol isn't compatible.
    # NOTE-2: You can specify a 'colon' separated list of jar
files in the jarFile property. We've found this
    #         useful when the JDBC driver requires additional jar
files to be present in the classpath to support it.
    #         For example, the CDH 5 Hive JDBC driver requires
that the `hadoop-common` jar file to be present in the
    #         classpath. You can specify this as follows: jarFile:
"/path/to/hive-jdbc.jar:/path/to/hadoop-common.jar"
    #     The order of the jar files is important. The first jar
file in the list MUST have the JDBC driver class file.
      jarFile: "<environment-specific-jdbc-standalone-driver>"
    # Optional.  Required only for (-epl) with DUMP or SCHEMA_ONLY
    # This will require the user to install the jdbc driver for
```

```yaml
the metastoreDirect in $HOME/.hms-mirror/aux_libs
    metastore_direct:
      uri: "<jdbc_url_to_metastore_db_including_db>"
      type: MYSQL|POSTRES|ORACLE
      connectionProperties:
        user: "<db_user>"
        password: "<db_password>"
      connectionPool:
        min: 3
        max: 5
  RIGHT:
    # Removed, see platformType. ...legacyHive:    false
    platformType: 'CDP7_1'
    # Is the 'Hadoop COMPATIBLE File System' used to prefix data
locations for this cluster.
    # It is mainly used to as a baseline for where "DATA" will be
transfered in the
    # STORAGE stage.  The data location in the source location
will be move to this
    # base location + the extended path where it existed in the
source system.
    # The intent is to keep the data in the same relative location
for this new cluster
    # as the old cluster.
    hcfsNamespace: "<hdfs://namespace>"
    hiveServer2:
      # URI is the Hive JDBC URL in the form of:
      # jdbc:hive2://<server>:<port>
      # See docs for restrictions
      uri:     "<RIGHT-cluster-jdbc-url>"
      connectionProperties:
        user:      "*****"
        password: "*****"
      # Standalone jar file used to connect via JDBC to the LEFT
environment Hive Server 2
      # NOTE-1: Hive 3 jars will NOT work against Hive 1.  The
protocol isn't compatible.
      # NOTE-2: You can specify a 'colon' separated list of jar
```

```yaml
files in the jarFile property. We've found this
      #        useful when the JDBC driver requires additional jar
files to be present in the classpath to support it.
      #        For example, the CDH 5 Hive JDBC driver requires
that the `hadoop-common` jar file to be present in the
      #        classpath. You can specify this as follows: jarFile:
"/path/to/hive-jdbc.jar:/path/to/hadoop-common.jar"
      #     The order of the jar files is important. The first jar
file in the list MUST have the JDBC driver class file.
      jarFile: "<environment-specific-jdbc-standalone-driver>"
    partitionDiscovery:
      # Addition HMS configuration needed for this
"discover.partitions"="true"
      auto:     true
      # When a table is created, run MSCK when there are
partitions.
      initMSCK: true
```

Running

After running the `setup.sh` script, `hms-mirror` will be available in the `$PATH` in a default configuration.

Assumptions

1. This process will only 'migrate' EXTERNAL and MANAGED (non-ACID/Transactional) table METADATA (not data, except with SQL ([SQL](#)) and EXPORT_IMPORT ([EXPORT_IMPORT](#))).

2. MANAGED tables replicated to the **RIGHT ** cluster will be converted to "EXTERNAL" tables for the 'metadata' stage. They will be tagged as 'legacy managed' in the **RIGHT ** cluster. They will be assigned the `external.table.purge=true` flag, to continue the behaviors of the legacy managed tables.

3. The **RIGHT** cluster has 'line of sight' to the **LEFT** cluster.

4. The **RIGHT** cluster has been configured to access the **LEFT ** cluster storage. See link clusters ([Linking Cluster Storage Layers](#)). This is the same configuration required to support `distcp` from the **RIGHT** cluster to the **LEFT** cluster.

5. The movement of metadata/data is from the **LEFT** cluster to the **RIGHT** cluster.

6. With Kerberos, each cluster must share the same trust mechanism.

- The **RIGHT** cluster must be Kerberized IF the **LEFT** cluster is.

- The **LEFT** cluster does NOT need to be kerberized if the **RIGHT** cluster is kerberized.

7. The * *LEFT* cluster does NOT have access to the **RIGHT** cluster.

8. The credentials use by 'hive' (doas=false) in the **RIGHT ** cluster must have access to the required storage (hdfs) locations on the lower cluster.

- If the **RIGHT ** cluster is running impersonation (doas=true), that user must have access to the required storage (hdfs) locations on the lower cluster.

Transfer DATA, beyond the METADATA

HMS-Mirror does NOT migrate data between clusters unless you're using the SQL ([SQL](#)) or EXPORT_IMPORT ([EXPORT_IMPORT](#)) data strategies. In some cases where data is co-located, you don't need to move it. IE: Cloud to Cloud. As long as the new cluster environment has access to the original location. This is the intended target for strategies COMMON ([COMMON](#)) and to some extend LINKED ([LINKED](#)).

When you do need to move data, `hms-mirror` creates a workbook of 'source' and 'target' locations in an output file called `distcp_workbook.md`. Use this to help build a transfer job in `distcp` using the `-f` option to specify multiple sources.

Application Return Codes

The `hms-mirror` application returns `0` when everything is ok. If there is a configuration validation issue, the return code will be a negative value who's absolute value represents the bitSets cumulative `OR` value. See: MessageCodes (https://github.com/cloudera-labs/hms-mirror/blob/main/src/main/java/com/cloudera/utils/hadoop/hms/mirror/MessageCode.java) for values and Messages.java for the calculation (https://github.com/cloudera-labs/hms-mirror/blob/df9df251803d8722ef67426a73cbcfb86f981d3e/src/main/java/com/cloudera/utils/hadoop/hms/mirror/Messages.java#L26).

When you receive an error code (negative value), you'll also get the items printed to the screen and the log that make up that error code.

For example, the following would yield a code of `-2305843009214742528` (20 and 61).

```
******* ERRORS *********
20:STORAGE_MIGRATION requires you to specify PATH location for
'managed' and 'external' tables (-wd, -ewd) to migrate storage.
These will be appended to the -smn (storage-migration-namespace)
parameter and used to set the 'database' LOCATION and
MANAGEDLOCATION properties
61:You're using the same namespace in STORAGE_MIGRATION, without
`-rdl` you'll need to ensure you have `-glm` set to map locations.
```

`((2^20)+(2^61))*-1=-2305843009214742528`

Running Against a LEGACY (Non-CDP) Kerberized HiveServer2

`hms-mirror` is pre-built with CDP libraries and WILL NOT be compatible with LEGACY kerberos environments. A Kerberos connection can only be made to ONE cluster when the clusters are NOT running the same 'major' version of Hadoop.

To attach to a LEGACY HS2, run `hms-mirror` with the `--hadoop-classpath` command-line option. This will strip the CDP libraries from `hms-mirror` and use the hosts Hadoop libraries by calling `hadoop classpath` to locate the binaries needed to do this.

On-Prem to Cloud Migrations

On-Prem to Cloud Migrations should run `hms-mirror` from the LEFT cluster since visibility in this scenario is usually restricted to LEFT->RIGHT.

If the cluster is an older version of Hadoop (HDP 2, CDH 5), your connection to the LEFT HS2 should NOT be kerberized. Use LDAP or NO_AUTH.

The clusters LEFT hcfsNamespace (clusters:LEFT:hcfsNamespace) should be the LEFT clusters HDFS service endpoint. The RIGHT hcfsNamespace (clusters:RIGHT:hcfsNamespace) should be the *target* root cloud storage location. The LEFT clusters configuration (/etc/hadoop/conf) should have all the necessary credentials to access this location. Ensure that the cloud storage connectors are available in the LEFT environment.

There are different strategies available for migrations between on-prem and cloud environments.

SCHEMA_ONLY

This is a schema-only transfer, where the `hcfsNamespace` in the metadata definitions is 'replaced' with the `hcfsNamespace` value defined on the RIGHT. NOTE: The 'relative' directory location is maintained in the migration.

No data will be migrated in this case.

There will be a `distcp` Planning Workbook (["distcp Planning Workbook and Scripts" in "Features"](#)) generated with a plan that can be used to build the data migration process with `distcp`.

Connections

hms-mirror connects to 3 endpoints. The hive jdbc endpoints for each cluster (2) and the hdfs environment configured on the running host. This means you'll need:

- JDBC drivers to match the JDBC endpoints

- For **non** CDP 7.x environments and Kerberos connections, an edge node with the current Hadoop libraries.

See the config (Configuration) section to setup the config file for hms-mirror.

Configuring the Libraries

AUX_LIBS - CLASSPATH Additions

S3

The directory $HOME/.hms-mirror/aux_libs will be scanned for 'jar' files. Each 'jar' will be added the java classpath of the application. Add any required libraries here.

The application contains all the necessary hdfs classes already. You will need to add to the aux_libs directory the following:

- JDBC driver for HS2 Connectivity (only when using Kerberos)

- AWS S3 Drivers, if s3 is used to store Hive tables. (appropriate versions)

  - hadoop-aws.jar

  - aws-java-sdk-bundle.jar

JDBC Connection Strings for HS2

See the Apache docs (https://cwiki.apache.org/confluence/pages/viewpage.action?pageId=30758725#HiveServer2Clients-JDBC) regarding these details if you are using the environment 'Standalone' JDBC drivers. Other drivers may have different connect string requirements.

The drivers for the various environments are located:

- HDP - `/usr/hdp/current/hive-server2/jdbc/hive-jdbc-<version>-standalone.jar` (NOTE: Use the hive-1 standalone jar file for HDP 2.6.5, not the hive-2 jar)

- CDH/CDP - `/opt/cloudera/parcels/CDH/jars/hive-jdbc-<version>-standalone.jar`

Non-Kerberos Connections

The most effortless connections are 'non-kerberos' JDBC connections either to HS2 with AUTH models that aren't * *Kerberos** or through a **Knox** proxy. Under these conditions, only the __standalone __ JDBC drivers are required. Each of the cluster configurations contains an element `jarFile` to identify those standalone libraries.

```
hiveServer2:
  uri: "<jdbc-url>"
  connectionProperties:
    user: "*****"
    password: "*****"
  jarFile: "<environment-specific-jdbc-standalone-driver>"
```

When dealing with clusters supporting different Hive (Hive 1 vs. Hive 3) versions, the JDBC drivers aren't forward OR backward compatible between these versions. Hence, each JDBC jar file is loaded in a sandbox that allows us to use the same driver class, but isolates it between the two JDBC jars.

Place the two jdbc jar files in any directory **EXCEPT ** `$HOME/.hms-mirror/aux_libs` and reference the full path in the `jarFile` property for that `hiveServer2` configuration.

SAMPLE Commandline

`hms-mirror -db tpcds_bin_partitioned_orc_10`

Kerberized Connections

`hms-mirror` relies on the Hadoop libraries to connect via 'kerberos'. Suppose the clusters are running different versions of Hadoop/Hive. In that case, we can only support connecting to one of the clusters via Kerberos.

> ⚠️ Packaging improvements in version 2.x include all the Kerberos libraries you need to connection to a kerberized Hive Server2 and HDFS. The `--hadoop-

classpath` option is no longer required for Kerberos connections. The application will use the embedded Hadoop 3.1 libraries to connect to the kerberized endpoints. The `--hadoop-classpath` option is still available for connecting to older Hadoop 2.x environments.

ⓘ The `jarFile` property is NOT used for Kerberos connections. The JDBC jar file for the kerberized cluster should be placed in the `$HOME/.hms-mirror/aux_libs` directory.

There are three scenarios for kerberized connections.

Scenario	LEFT Kerberized/Version	RIGHT Kerberized/Version	Notes	Sample Commandline
1	No HDP2	Yes HDP 3 or CDP 7	1. 'hms-mirror' needs to be run from a node on the HDP3/CDP cluster. 2. place the RIGHT cluster jdbc jar file in `$HOME/.hms-mirror/aux_libs` (yes this contradicts some earlier directions) 3. comment out the `jarFile` property for the RIGHT cluster hiveServer2 setting.	`hms-mirror -db tpcds_bin_partitioned_orc_10 --hadoop-classpath`
2	YES HDP 3 or CDP 7	YES HDP 3 or CDP 7	1. 'hms-mirror' needs to be run from a node on the HDP3/CDP cluster. 2. place the RIGHT cluster jdbc jar file in $HOME/.hms-mirror/aux_libs (yes this contradicts some earlier directions) 3. comment out the `jarFile` property for the LEFT AND RIGHT cluster hiveServer2 settings.	`hms-mirror -db tpcds_bin_partitioned_orc_10 --hadoop-classpath`
3	YES HDP 2 or Hive 1	NO HDP 3 or CDP 7	Limited testing, but you'll need to run `hms-mirror` ON the **LEFT** cluster and include the LEFT clusters hive standalone jdbc driver in `$HOME/.hms-mirror/cfg/aux_libs`.	`hms-mirror -db tpcds_bin_partitioned_orc_10 --hadoop-classpath`

4	YES H DP 2 o r Hive 1	YES H DP 2 or Hive 1	1. The Kerberos credentials must be TRU STED to both clusters 2. Add `--hadoop-classpath` as a comma ndline option to `hms-mirror`. This repl aces the prebuilt Hadoop 3 libraries wi th the current environments Hadoop L ibs. 3. Add the jdbc standalone jar file to `$H OME/.hms-mirror/aux_libs` 4. Comment out/remove the `jarFile` refe rences for BOTH clusters in the config uration file.	`hms-mirror -db tpcds_bin_parti tioned_orc_10 - -hadoop-classp ath`

For Kerberos JDBC connections, ensure you are using an appropriate Kerberized Hive URL.

`jdbc:hive2://s03.streever.local:10000/;principal=hive/_HOST@STREEVER.LOCAL`

ZooKeeper Discovery Connections

You may run into issues connecting to an older cluster using ZK Discovery. This mode brings in a LOT of the Hadoop ecosystem classes and may conflict across environments. We recommend using ZooKeeper discovery on only the RIGHT cluster. Adjust the LEFT cluster to access HS2 directly.

TLS/SSL Connections

If your HS2 connection requires TLS, you will need to include that detail in the jdbc 'uri' you provide. In addition, if the SSL certificate is 'self-signed' you will need to include details about the certificate to the java environment. You have 2 options:

- Set the JAVA_OPTS environment with the details about the certificate.

  - export JAVA_OPTS=-Djavax.net.ssl.trustStore=/home/dstreev/certs/gateway-client-trust.jks -Djavax.net.ssl.trustStorePassword=changeit

- Add `-D` options to the `hms-mirror` commandline to inject those details.

  - `hms-mirror -db test_db -Djavax.net.ssl.trustStore=/home/dstreev/certs/gateway-client-trust.jks -Djavax.net.ssl.trustStorePassword=changeit`

Troubleshooting

If each JDBC endpoint is Kerberized and the connection to the LEFT or RIGHT is successful, both NOT both, and the program seems to hang with no exception... it's most likely that the Kerberos ticket isn't TRUSTED across the two environments. You will only be able to support a Kerberos connection to the cluster where the ticket is trusted. The other cluster connection will need to be anything BUT Kerberos.

Add `--show-cp` to the `hms-mirror` command line to see the classpath used to run.

The argument `--hadoop-classpath` allows us to replace the embedded Hadoop Libs (v3.1) with the libs of the current platform via a call to `hadoop classpath`. This is necessary to connect to kerberized Hadoop v2/Hive v1 environments.

Check the location and references to the JDBC jar files. General rules for Kerberos Connections:

- The JDBC jar file should be in the `$HOME/.hms-mirror/aux_libs`. For Kerberos connections, we've seen issues attempting to load this jar in a sandbox, so this makes it available to the global classpath/loader.

- Get a Kerberos ticket for the running user before launching `hms-mirror`.

"Unrecognized Hadoop major version number: 3.1.1.7.1...0-257"

This happens when you're trying to connect to an HS2 instance.

Optimizations

Moving metadata and data between two clusters is a pretty straightforward process but depends entirely on the proper configurations in each cluster. Listed here are a few tips on some crucial configurations.

HMS-Mirror only moves data with the SQL ([SQL](#)) and EXPORT_IMPORT ([EXPORT_IMPORT](#)) data strategies. All other strategies either use the data as-is (LINKED ([LINKED](#)) or COMMON ([COMMON](#))) or depend on the data being moved by something like `distcp`.

Controlling the YARN Queue that runs the SQL queries from hms-mirror

Use the jdbc url defined in `default.yaml` to set a queue.

`jdbc:hive2://host:10000/.....;...?tez.queue.name=batch`

The commandline properties `-po`, `-pol`, and `-por` can be used to override the queue name as well. For example: `-pol tez.queue.name=batch` will set the queue for the "LEFT" cluster while `-por tez.queue.name=migration` will set the queue for the "RIGHT" cluster.

Make Backups before running hms-mirror

Take snapshots of areas you'll touch:

- The HMS database on the LEFT and RIGHT clusters

- A snapshot of the HDFS directories on BOTH the LEFT and RIGHT clusters will be used/touched.

  > ⚠ NOTE: If you are testing and "DROPPING" dbs, Snapshots of those data directories could protect you from accidental deletions if you don't manage purge options correctly. Don't skip this... A snapshot of the db directory on HDFS will prevent `DROP DATABASE x CASCADE` from removing the DB directory (observed in CDP 7.1.4+ as tested, check your version) and all sub-directories even though tables were NOT configured with `purge` options.

Isolate Migration Activities

The migration of schemas can put a heavy load on HS2 and the HMS server it's using. That impact can manifest itself as 'pauses' for other clients trying to run queries. Extended schema/discovery operations have a 'blocking' tendency in HS2.

To prevent average user operational impact, I suggest establishing an isolated HMS and HS2 environment for the migration process.

Isolate Migration Service Endpoints
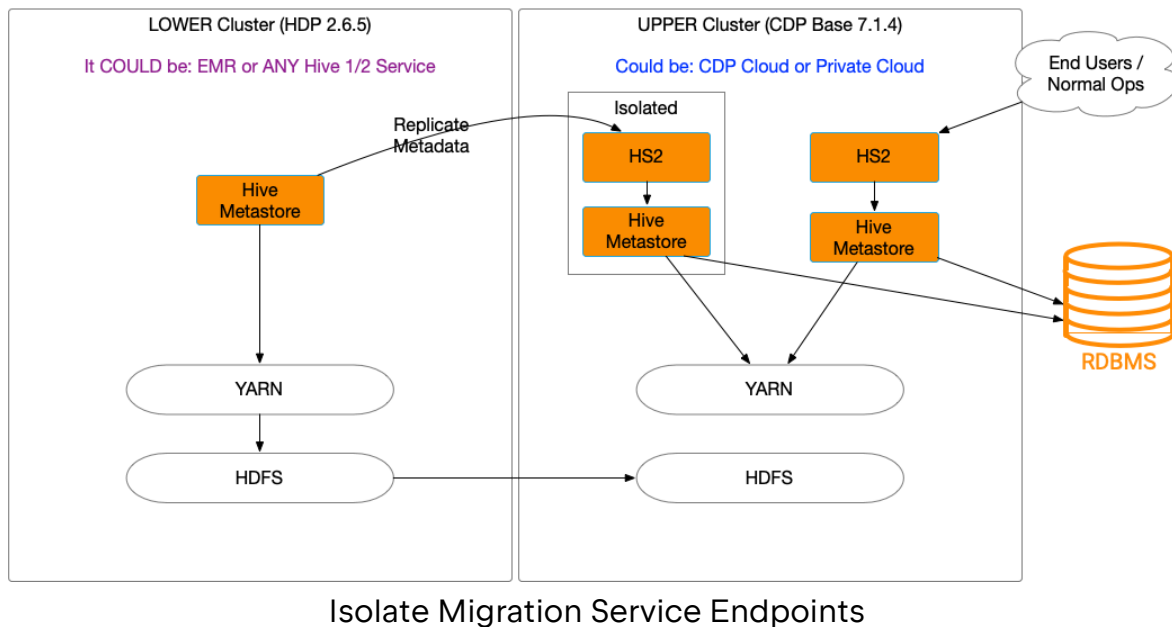
Speed up CREATE/ALTER Table Statements - with existing data

Set `ranger.plugin.hive.urlauth.filesystem.schemes=file` in the Hive Server 2(hive_on_tez) Ranger Plugin Safety Value, via Cloudera Manager.

Safety Value

Add this to the HS2 instance on the RIGHT cluster when Ranger is used for Auth. This skips the check done against every directory at the table location (for CREATE or ALTER LOCATION). It is allowing the process of CREATE/ALTER to run much faster.

The default (true) behavior works well for the interactive use case. Still, bulk operations like this can take a long time if this validation needs to happen for every new partition during creation or discovery.

I recommend turning this back after the migration is complete. This setting exposes permissions issues at the time of CREATE/ALTER. So by skipping this, future access issues may arise if the permissions aren't aligned, which isn't a Ranger/Hive issue, it's a permissions issue.

Turn ON HMS partition discovery

In CDP 7.1.4 and below, the housekeeping threads in HMS used to discover partitions are NOT running. Add metastore.housekeeping.threads.on=true to the HMS Safety Value to activate the partition discovery thread. Once this has been set, the following parameters can be used to modify the default behavior.

```
hive.metastore.partition.management.task.frequency
hive.exec.input.listing.max.threads
hive.load.dynamic.partitions.thread
hive.metastore.fshandler.threads
```

Source Reference

```
METASTORE_HOUSEKEEPING_LEADER_HOSTNAME("metastore.housekeeping.lea
der.hostname",
            "hive.metastore.housekeeping.leader.hostname", "",
"If multiple Thrift metastore services are running, the hostname
of Thrift metastore " +
        "service to run housekeeping tasks at. By default, this
value is empty, which " +
        "means that the current metastore will run the
housekeeping tasks. If configuration" +
        "metastore.thrift.bind.host is set on the intended leader
metastore, this value should " +
        "match that configuration. Otherwise it should be same as
the hostname returned by " +
        "InetAddress#getLocalHost#getHostName(). Given the
uncertainty in the later " +
        "it is desirable to configure metastore.thrift.bind.host
on the intended leader HMS."),

METASTORE_HOUSEKEEPING_THREADS_ON("metastore.housekeeping.threads.
on",
        "hive.metastore.housekeeping.threads.on", false,
        "Whether to run the tasks under
metastore.task.threads.remote on this metastore instance or
not.\n" +
            "Set this to true on one instance of the Thrift
metastore service as part of turning\n" +
            "on Hive transactions. For a complete list of
parameters required for turning on\n" +
            "transactions, see hive.txn.manager."),
```

The default batch size for partition discovery via `msck` is 3000. Adjustments to this can be made via the `hive.msck.repair.batch.size` property in HS2.

Tips

⚠ Run in `screen` or `tmux`

This process can be a long-running process. It depends on how much you've asked it to do. Having the application terminated because the `ssh` session to the edgenode timed out and your computer went to sleep will be very disruptive.

Using either of these session state tools (or another of your choice) while running it on an edgenode will allow you to sign-off without disrupting the process AND reattach to see the interactive progress at a later point.

⚠ Use `dryrun` FIRST

Before you run a process that will make changes, try running `hms-mirror` with the `dry-run` option first. The report generated at the end of the job will provide insight into what issues (if any) you'll run across.

⚠ Start Small

Use `-db` (database) AND `-tf` (table filter) options to limit the scope of what you're processing. Start with a test database that contains various table types you'd like to migrate.

Review the output report for details/problems you encountered in processing.

Since the process expects the user to have adequate permissions in both clusters, you may find you have some prework to do before continuing.

Location Alignment

In the Hive Metastore, Database definitions, Table Schemas, and Partition details include the location of their datasets. These locations contain a full URI to the dataset. Migrating from one cluster to another requires us to make adjustments to these locations.

Data Strategy: SCHEMA_ONLY

datamovement_strategy.png

The most simple translation will change the namespace of the URI so everything is RELATIVE. This helps reduce the impact of other tools that might be using these datasets outside the Hive Metastore definitions.

Using the RELATIVE 'Location Translation Strategy' is suggested for side-car cluster migrations where you want to keep everything the same as much as possible.

When you are reorganizing, consolidating, or changing storage environments then the ALIGNED 'Location Translation Strategy' will aid in that process. We suggest building out

Warehouse Plans ([Warehouse Plans](#)) for each database for maximum control of that movement.

Attributes of location transformations:

- Global Warehouse Directories

- Environment Warehouse Directories

> ⚠ These are pulling automatically from the Hive Environment when available.

- Target Namespace

> ⚠ This is defined in the configuration through the `transfer.targetNamespace` configuration attribute or the `target-namespace` configuration setting. This is used for migrations between two clusters and for STORAGE_MIGRATION's within the cluster.

- Warehouse Plans

> ⚠ Defined for each 'database'. And when defined we should assume that we expect locations of the database, tables, and partitions will be ALIGNED with that location.

Order of Evaluation

Order of Evaluation means that we will evaluate the attribute in the describe order and once a valid mapping is found, we will stop the evaluation. Evaluation order depends on the translation type as well.

ALIGNED

- Target Namespace

- Warehouse Plans (when defined)

- Global Warehouse Directories

- Environment Warehouse Directories (under certain conditions)

RELATIVE

- Target Namespace

Translation Types

Translation types are used to determine how the location should be transformed. The following are the translation types are `ALIGNED` and `RELATIVE`.

Legend

Icon	Description
	Valid
	Invalid
	Ignored
	SQL
	Distcp
	Manual
	Automatic
	Optional
	Metastore Direct
	Global Warehouse
	Warehouse Plan

Translation Scenarios

	Translation Type		Data Movement	Required	Notes
DUMP	✔️	RELATIVE			
	❌	ALIGNED			
SCHEMA_ONLY	✔️	RELATIVE	🗐👤		⚠️ Using distcp 🗐 here when either table or partition locations aren't standard, will result in data loss because we're not inspecting all the locations through the Metastore Direct connection. It's recommended to use `ALIGNED` with `distcp` to build an accurate `distcp` plan.
	✔️	ALIGNED	🗐	🛢️ when Warehouse Plan (s) used 🌐 ⬛ (Optional)	
SQL	✔️	RELA LA			

		TI VE			
	✔	AL IG N E D	[SQL]	🟢 ➡ (O ptional)	
EXP ORT _IMP ORT	✔	RE LA TI VE			
	✔	AL IG N E D		🟢 ➡ (O ptional)	
HYB RID	✘	RE LA TI VE			
	✔	AL IG N E D	[SQL]	🟢 ➡ (O ptional)	

STORAGE_MIGRATION	❌	RELATIVE			
	✅	ALIGNED	🗒️SQL 📑	🛢️ when Warehouse Plan (s) used 📄→ (Optional)	
LINKED	☑	RELATIVE			
	☑	ALIGNED			
COMMON	☑	RELATIVE			
	☑	ALIGN N			

		E D			

Databases

The whole goal of hms-mirror is to move metadata from one cluster to another. Picking which databases to move is a primary function of the application.

There are several ways to select databases for migration. Each with its own benefits.

> ⚠ Once a method is selected to add database(s), the other add options will be limited until the option is cleared.

Add 'database' names to the runtime configuration.

This is the simplest way to select databases. Additional filtered can be applied to tables through the table **RegEx** filters.

Web UI

Ensure you've selected 'Edit' from the Left Navigation Menu. Enter a comma separated list of databases and press 'Add'.

Data Strategy: SCHEMA_ONLY

| Save | Messages (0) | Password Management | Filter | Attributes | Source Cluster(LEFT) |

Manage Databases through Warehouse Plans, Database, or Database RegEx.

Databases   **Warehouse Plans**

[   ] Add   Add

RegEx Filters

Database RegEx   Set RegEx

Table RegEx   Set RegEx

Table Exclude RegEx   Set Exclude RegEx

dbs_by_comma.png

CLI

The `-db|--database` option allows you to list the databases you want to process.

Use the Database RegEx filter to include matching databases found in the source cluster.

Web UI

Ensure you've selected 'Edit' from the Left Navigation Menu. Enter a RegEx pattern to match the databases you want to include and press 'Add'.

CLI

The `-dbRegEx|--database-regex` option allows you to filter matching databases.

Create Warehouse Plans for the databases you want to include.

Web UI

Ensure you've selected 'Edit' from the Left Navigation Menu. Select the 'Warehouse Plans' tab. Create a new Warehouse Plan and add the databases you want to include.

CLI

This feature isn't available through CLI commandline Options. It is possible to use the WebUI to create the configuration and then use the 'persisted' version of that configuration as the configuration for the CLI via `-cfg`.

Warehouse Plans

Warehouse Plans in "hms-mirror" are a database-level configuration mechanism designed to manage and map storage locations within a Hive database during metadata migration. They involve reviewing the database metadata to identify all storage locations (e.g., table and partition locations for both External and Managed tables) and mapping these to predefined Warehouse Plan locations. The intersection of this metadata and the Warehouse Plan is then used to construct **Global Location Maps,** which "hms-mirror" uses internally to provide a consistent mapping between storage systems across clusters or within a single cluster.

There are three types of 'Warehouse Plans'.

- Global

- Per Database

- Environment

When you choose to 'ALIGN' your datasets during the migration (See Location Alignment ([Location Alignment](#)) for settings that are applicable for each strategy), we'll evaluate the warehouse plans to determine where each dataset should land. If you're using 'SQL' to move data, we'll build the schema's and sql that will make the adjustments required to reorganize the data based on the warehouse plan that is found.

It's recommended to define a warehouse plan for each **database** you want to move when you're using the 'ALIGNED' data movement strategy. When this is defined for the database, we'll inspect the all the current locations of tables and partitions in that dataset and make the necessary adjustments to locations.

The 'Warehouse Plans' get converted into 'Global Location Maps' that are inspected during processing to make that conversion.

Standard Locations

When you choose to **ALIGN** the datasets, you are choosing to collect everything in the dataset/database under the same location as you've defined in the **warehouse plan**.

When you choose DISTCP as the movement plan for ALIGNED, we'll build a distcp plan that will make those translations. **BUT**, there are some restrictions to this.

When you choose to use non-standard locations for 'partition specs', we can't build a proper distcp plan. In this case we will throw an 'error' for the offending table and describe to imbalance. You can either fix/adjust the dataset OR choose to use the SQL data movement strategy.

Warehouse Plans Explained

What Are Warehouse Plans For?

- **Purpose**: At the database level, Warehouse Plans define how storage locations in the source database metadata are translated to target locations. This process ensures that the migrated metadata aligns with the desired storage structure on the target system. Unlike the earlier globalLocationMap (["Global Location Maps" in "Release Notes"](#)), which offered a simpler, global key-value mapping, Warehouse Plans operate specifically at the database scope, providing a more targeted and detailed approach.

- **Database Warehouse Plans** ([Database Warehouse Plans](#)): These are the core of the feature, specifying location mappings for a single database. The documentation implies that "Global Warehouse Plans" ([Global Warehouse Plans](#)) may extend this concept across multiple databases, but the primary focus is on individual database-level planning.

The resulting Global Location Maps, derived from Warehouse Plans, serve as an internal framework for "hms-mirror" to handle location translations during migration, ensuring consistency between source and target storage systems.

What to Use Warehouse Plans For?

Warehouse Plans are primarily designed for scenarios where storage locations within a database need to be systematically mapped or reorganized during a migration. Here's an explanation of their use cases, reflecting their database-level scope and original intent for STORAGE_MIGRATION, as well as their broader applicability:

1. **Primary Use Case: Storage Migration Within a Cluster (STORAGE_MIGRATION):**

- **Scenario**: You're migrating the storage layer behind a database's metadata from HDFS to Ozone (or another system like an encrypted zone), as highlighted in the .

- **Use**: Define a Warehouse Plan for the database to map all existing locations (e.g., `/apps/hive/warehouse/my_db`) to a new target (e.g., `ofs://ozone1/vol1/bucket1/my_db`). The metadata is reviewed, and all table and partition locations are updated accordingly. The resulting Global Location Maps ensure the migration reflects the new storage system.

- **Example**: For a database `finance_db`:

```
databaseWarehousePlans:
  finance_db:
    EXTERNAL_TABLE: "/finance_db/ext"
    MANAGED_TABLE: "/finance_db/managed"
```

This maps all External and Managed table locations within `finance_db` to the specified Ozone paths.

- The Ozone Namespace is pulled from the `targetNamespace` which can be set in the config or via the Web UI.

```
transfer:
  targetNamespace: ofs://myOzone
```

2. **Reorganizing Storage During Schema Migration (SCHEMA_ONLY):**

- **Scenario**: You're migrating a database's schema between clusters (e.g., on-premises to cloud) using `SCHEMA_ONLY` (page 150), and you want to reorganize storage locations simultaneously.

- **Use**: A Warehouse Plan can redefine the database's storage locations (e.g., from `/data/my_db` to `s3a://mybucket/my_db`). Since `SCHEMA_ONLY` doesn't move data, pair it with the `-dc|--distcp` option (<u>Commandline Help (Options)</u>) to generate `distcp` plans for separately migrating the data to the new locations.

- **Example**:

```
databaseWarehousePlans:
  my_db:
    EXTERNAL_TABLE: "/my_db/ext"
    MANAGED_TABLE: "/my_db/managed"
```

Run: hms-mirror -d SCHEMA_ONLY -db my_db -dc to get the schema migration and a distcp plan. Again, use the transfer.targetNamespace to set the s3 location. EG s3a://mybucket.

3. **Migrating and Moving Data with SQL Strategy (SQL):**

- **Scenario**: You're using the SQL data strategy ([SQL](#)) to migrate both metadata and data between linked clusters, and you need to adjust storage locations.

- **Use**: Define a Warehouse Plan to map the database's locations to the target cluster's storage (e.g., from hdfs://source/my_db to hdfs://target/my_db). The SQL strategy will move the data to the new locations as part of the migration.

- **Example**:

```
databaseWarehousePlans:
  my_db:
    EXTERNAL_TABLE: "/my_db/ext"
    MANAGED_TABLE: "/my_db/managed"
```

Run: hms-mirror -d SQL -db my_db.

4. **Supporting Data Strategies Without Data Movement:**

- **Scenario**: You're using a strategy like SCHEMA_ONLY ([SCHEMA_ONLY](#)) that doesn't move data, but you need a plan to migrate the data later.

- **Use**: Warehouse Plans map the database's locations, and the -dc|--distcp option generates distcp scripts to handle the data migration separately, aligning with the mapped locations.

- **Example**: For SCHEMA_ONLY:

```
hms-mirror -d SCHMEA_ONLY -db my_db -dc -wps
my_db=/my_db/ext:/my_db/managed
```

The Warehouse Plan ensures the dump reflects the intended target locations, and
`distcp` plans are provided.

5. **Consistency Within a Database Across Table Types**:

- **Scenario**: A database contains both External and Managed tables with various
  locations, and you want a unified storage structure on the target.

- **Use**: A Warehouse Plan reviews all locations in the database metadata and maps
  them to consistent target locations, regardless of table type. This is more precise
  than global mappings, as it's tailored to the database.

- **Example**: Mapping all tables in `sales_db` to a single storage layer:

```
databaseWarehousePlans:
  sales_db:
    EXTERNAL_TABLE: "/new_storage/division_ext"
    MANAGED_TABLE: "/new_storage/division_mngd"
```

How to Use Warehouse Plans

To implement Warehouse Plans, configure them in the `default.yaml` file under the
`databaseWarehousePlans` section at the database level.

Here's how, based on the documentation and your clarification:

- **Syntax**:

```
databaseWarehousePlans:
  <database_name>:
    EXTERNAL_TABLE: "<target_location_for_external>"
    MANAGED_TABLE: "<target_location_for_managed>"
```

- The metadata for `<database_name>` is analyzed, and all locations (for External and
  Managed tables) are mapped to the specified targets. These mappings feed into
  the Global Location Maps used internally.

- NOTE: The database name is appended to the location you specify, so do NOT include that in the location. This allows you to use the same location for multiple databases in the scenario you want to have multiple databases share the same root location.

- **Command-Line Integration**: While Warehouse Plans are configuration-driven, they work with strategies like `STORAGE_MIGRATION`, `SCHEMA_ONLY`, or `SQL`. Use `-dc|--distcp` for strategies without data movement.

  - The Warehouse Plans can be set in the config, via the Web UI, or via the commandline option `-wps <db=ext-dir:mngd-dir[,db=ext-dir:mngd-dir]...>`

  ```
  hms-mirror -d STORAGE_MIGRATION -db my_db -dc -wps
  my_db=/my_db/ext:/my_db/managed
  ```

  or

  ```
  hms-mirror -d SCHEMA_ONLY -db my_db -dc
  ```

- **Execution**: The Warehouse Plan drives the location translation process. For `STORAGE_MIGRATION`, data is moved directly (if using `DISTCP` or `SQL` as the data movement strategy, page 202). For other strategies, `-dc` ensures data migration plans are provided.

Practical Tips

- **Database Scope**: Define Warehouse Plans per database to ensure precise mapping. Unlike the older `globalLocationMap`, they don't apply globally unless explicitly extended via "Global Warehouse Plans."

- **Dry-Run**: Test with a dry-run (`hms-mirror -db <db_name>`) to review the mappings in the output reports (page 111) before executing (`-e`).

- **Storage Access**: Verify permissions for the mapped locations, especially for cross-cluster scenarios (page 69, "Linking Cluster Storage Layers").

- **Original Intent**: While designed for `STORAGE_MIGRATION` (e.g., HDFS to Ozone), their flexibility supports broader reorganization tasks.

In summary, Warehouse Plans are a database-level tool in "hms-mirror" for mapping storage locations within a database's metadata, originally for STORAGE_MIGRATION (e.g., HDFS to Ozone), but also valuable for SCHEMA_ONLY and SQL strategies. They construct Global Location Maps internally, ensuring accurate location translations, and can be paired with -dc|--distcp for separate data movement when needed.

Global Warehouse Plans

global_warehouse.png

Database Warehouse Plans

filter_wp-add.png

wp-add-detail.png

Environment Warehouse

When a job is started, we'll gather details about the environment from Hive via `set;`. Hive defines default global warehouse locations for 'external' and 'managed' tables for databases. If a database doesn't define a `LOCATION` and/or `MANAGEDLOCATION`, these entries will be used to define to locations of a table/partition when it's created.

In a multi-tenant environment where different storage locations and/or namespaces are used, these global default settings are inadequate. In these environments, you should be declaring the locations at the database level to ensure all new datasets end up in the desired locations and you can maintain a true multi-tenant environment.

Hive Conversions

Hive has gone through a lot of changes as it's evolved over the last several years. Especially between Hive 1/2 and Hive 3. The default syntax used to 'create' tables hasn't changed, but the resulting table structure may have.

Understand those changes and what hive flags are available to help influence that structure aren't very clear to even the most seasoned Hive user.

`hms-mirror` uses settings in the 'cluster' configuration to influence **how** tables are translated during the migration.
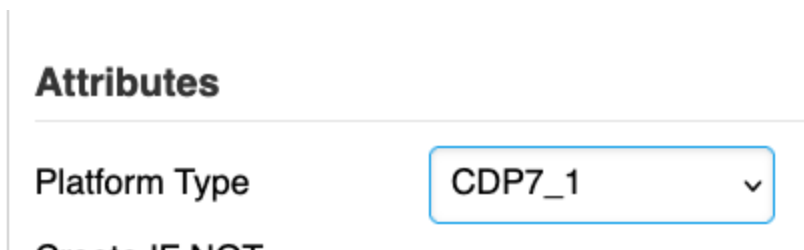
Web Interface

Set the properties in the appropriate cluster configuration(s) for your strategy.

cluster_tabs.png

Set the flags that match you cluster's Hive version.

hive_version_flags.png

CLI

Modify the `hms-mirror` configuration to include the following settings:

```
clusters:
  LEFT|RIGHT:
```

```
platformType: HDP2|HDP3|CHD5|CDH6|CDP7_1|CDP7_2|...
```

If you were upgrading in-place a Hive 1/2 cluster to Hive 3, the Hive upgrade process would convert 'legacy' managed tables to 'EXTERNAL' tables and add a 'PURGE' flag in the table properties so that tables behavior remains consistent with the original Hive 1/2 behavior while also being a compatible Hive 3 table.

Managed non-ACID tables are NOT a valid state in Hive 3.

With the properties set in the configuration for the cluster (above), `hms-mirror` will make these conversions for you, in the same way that the Hive upgrade process would.

Hive 1/2 Table Types History

Tables in Hive 1/2 are typically 'external' tables. This means that the data is managed independently of the Hive Metastore. The table definition in the Hive Metastore points to the location of the data but does NOT manage the data. In this case, if you drop the table (or a partition), the data remains on the filesystem. These are created with the `CREATE EXTERNAL TABLE` command.

Another table type in Hive 1/2 is the 'managed' table. This is where Hive manages the data. If you drop the table (or partition), Hive will clean up the data on the filesystem. These are created with the `CREATE TABLE` command.

There is also a variant of the 'managed' table that is ACID compliant. These are created with the `CREATE TABLE` command with table properties that enable ACID characteristics `transactional=true`. These tables have special characteristics and are not friendly to the 'schema on read' paradigm, since these tables are 'schema on write' and embed special columns in the data files.

Starting with Hive 3, the default behavior of the `CREATE TABLE` command is to create a 'managed' ACID table. The additional table properties for a 'transactional' table are no longer required to create an ACID table. This is a significant change from Hive 1/2.

There is a web application that allows you to experiment with various commands and settings, showing the resulting table structures.

Regardless, use the following web application to help you quickly through the trial and error phase of understanding these new structural and session settings.

Hive Create Path (https://dstreev.github.io/hive/create_path.html)

Features

hms-mirror is designed to migrate schema definitions from one cluster to another or simply provide an extract of the schemas via -d DUMP.

Under certain conditions, hms-mirror will 'move' data too. Using the data strategies -d SQL|EXPORT_IMPORT|HYBRID well use a combination of SQL temporary tables and Linking Clusters Storage Layers (Linking Cluster Storage Layers) to facilitate this.

Iceberg Table Migration via Hive

See Iceberg Migration (ICEBERG_MIGRATION) for details.

File System Stats

SQL based operations, hms-mirror will attempt to gather file system stats for the tables being migrated. This is done by running hdfs dfs -count on the table location. This is done to help determine the best strategy for moving data and allows us to set certain hive session values and distribution strategies in SQL to optimize the data movement.

But some FileSystems may not be very efficient at gathering stats. For example, S3. In these cases, you can disable the stats gathering by adding -ssc|--skip-stats-collection to your command line.

When you have a LOT of tables, collecting stats can have a significant impact on the time it takes to run hms-mirror and the general pressure on the FileSystem to gather this information. In this case, you have to option to disable stats collection through -scc.

CREATE [EXTERNAL] TABLE IF NOT EXISTS Option

Default behavior for hms-mirror is to NOT include the IF NOT EXISTS clause in the CREATE TABLE statements. This is because we want to ensure that the table is created and that the schema is correct. If the table already exists, we want to fail.

But there are some scenarios where the table is present and we don't want the process to fail on the CREATE statement to ensure the remaining SQL statements are executed. In this case, you can modify add the commandline option -cine or add to the configuration:

```
clusters:
  RIGHT|LEFT:
    createIfNotExists: "true"
```

Using this option has the potential to create a table with a different schema than the source. This is not recommended. This option is applied when using the SCHEMA_ONLY data strategy.

Auto Gathering Stats (disabled by default)

CDP Default settings have enabled hive.stats.autogather and hive.stats.column.autogather. This impacts the speed of INSERT statements (used by hms-mirror to migrate data) and for large/numerous tables, the impact can be significant.

The default configuration for hms-mirror is to disable these settings. You can re-enable this in hms-mirror for each side (LEFT|RIGHT) separately. Add the following to your configuration.

```
clusters:
  LEFT|RIGHT:
    enableAutoTableStats: true
    enableAutoColumnStats: true
```

To disable, remove the enableAutoTableStats and enableAutoColumnStats entries or set them to false.

Non-Standard Partition Locations

Partitions created by 'hive' with default locations follow a file system naming convention that allows other partitions of 'hive' to discovery/manage those location and partition associations.

The standard is for partitions to exist as sub-directories of the table location. For example: Table Location is hdfs://my-cluster/warehouse/tablespace/external/hive/my_test.db/my_table and the partition location is hdfs://my-

cluster/warehouse/tablespace/external/hive/my_test.db/my_table/dt=2020-01-01, assuming the partition column name is dt.

When this convention is not followed, additional steps are required to build the partition metadata. You can't use MSCK REPAIR because it will not find the partitions. You can use ALTER TABLE ADD PARTITION but you'll need to provide the location of the partition. hms-mirror will do this for you when using the data strategies -d DUMP|SCHEMA_ONLY and the commandline flag -epl|--evaluate-partition-location.

In order to make this evaluation efficient, we do NOT use standard HiveSQL to discover the partition details. It is possible to use HiveSQL for this, it's just not meant for operations at scale or tables with a lot of partitions.

Hence, we tap directly into the hive metastore database. In order to use this feature, you will need to add the following configuration definition to your hms-mirror configuration file (default.yaml).

```
clusters:
  LEFT|RIGHT:

    ...
    metastore_direct:
      uri: "<db_url>"
      type: MYSQL|POSTGRES|ORACLE
      connectionProperties:
        user: "<db_user>"
        password: "<db_password>"
      connectionPool:
        min: 3
        max: 5
```

You will also need to place a copy of the RDBMS JDBC driver in $HOME/.hms-mirror/aux_libs. The driver must match the type defined in the configuration file.

Note: Non-Standard Partition Location will affect other strategies like SQL where the LEFT clusters storage is accessible to the RIGHT and is used by the RIGHT to source data. The 'mirror' table used for the transfer will NOT discover the partitions and will NOT transfer data. See: Issue #63 (https://github.com/cloudera-labs/hms-mirror/issues/63) for updates on addressing this scenario. If this is affecting you, I highly recommend you comment on the issue to help us set priorities.

Optimizations

The following configuration settings control the various optimizations taken by `hms-mirror`. These settings are mutually exclusive.

- `-at|--auto-tune`

- `-so|--skip-optimizations`

- `-sdpi|--sort-dynamic-partition-inserts`

Auto-Tune

`-at|--auto-tune`

Auto-tuning will use some basic file level statistics about tables/partitions to provide overrides for the following settings:

- `tez.grouping.max-size`

- `hive.exec.max.dynamic.partitions`

- `hive.exec.reducers.max`

in addition to these session level setting, we'll use those basic file statistics to construct migration scripts that address things like 'small-files' and 'large' partition datasets.

We'll set `hive.optimize.sort.dynamic.partition.threshold=-1` and append `DISTRIBUTE BY` to the SQL migration sql statement, just like we do with `-sdpi`. But we'll go one step further and review the average partition size and add an additional 'grouping' element to the SQL to ensure we get efficient writers to a partition. The means that tables with large partition datasets will have more than the standard single writer per partition, preventing the LONG running hanging task that is trying to write a very large partition.

Sort Dynamic Partition Inserts

`-sdpi|--sort-dynamic-partition-inserts`

This will set the session property `hive.optimize.sort.dynamic.partition.threshold=0`, which will enable plans to distribute multi partition inserts by the partition key, therefore reducing partitions writes to a single 'writer/reducer'.

When this isn't set, we set `hive.optimize.sort.dynamic.partition.threshold=-1`, and append `DISTRIBUTE BY` to the SQL migration sql statement to ensure the same behavior of grouping reducers by partition values.

Skip Optimizations

`-so`

Feature Request #23 (https://github.com/cloudera-labs/hms-mirror/issues/23) was introduced in v1.5.4.2 and give an option to **Skip Optimizations**.

When migrating data via SQL with partitioned tables (OR downgrading an ACID table), there are optimizations that we apply to help hive distribute data more efficiently. One method is to use `hive.optimize.sort.dynamic.partition=true` which will "DISTRIBUTE" data along the partitions via a Reduction task. Another is to declare this in SQL with a `DISTRIBUTE BY` clause.

But there is a corner case where these optimizations can get in the way and cause long-running tasks. If the source table has already been organized into large files (which would be within the partitions already), adding the optimizations above force a single reducer per partition. If the partitions are large and already have good file sizes, we want to skip these optimizations and let hive run the process with only a map task.

HDP3 MANAGEDLOCATION Database Property

HDP3 doesn't support MAANGEDLOCATION (https://github.com/cloudera-labs/hms-mirror/issues/52) so we've added a property to the cluster configuration to allow the system to *SKIP* setting the `MANAGEDLOCATION` database property in HDP 3 / Hive 3 environments.

```
clusters:
  LEFT:
    platformType: 'HDP3'
```

Compress Text Output

`-cto` will control the session level setting for `hive.exec.compress.output'.

VIEWS

`hms-mirror` now supports the migration of VIEWs between two environments. Use the `-v|--views-only` option to execute this path. VIEW creation requires dependent tables to exist.

Run `hms-mirror` to create all the target tables before running it with the `-v` option.

This flag is an `OR` for processing VIEW's `OR` TABLE's. They are NOT processed together.

Requirements

- The dependent tables must exist in the RIGHT cluster

- When using `-dbp|--db-prefix` option, VIEW definitions are NOT modified and will most likely cause VIEW creation to fail.

ACID Tables

`hms-mirror` supports the migration of ACID tables using the `-d HYBRID|SQL|EXPORT_IMPORT` data strategy in combination with the `-ma|--migrate-acid` or `-mao|--migrate-acid-only` flag. You can also simply 'replay' the schema definition (without data) using `-d SCHEMA_ONLY -ma|-mao`. The `-ma|-mao` flag takes an *optional* integer value that sets an 'Artificial Bucket Threshold'. When no parameter is specified, the default is `2`.

Use this value to set a bucket limit where we'll *remove* the bucket definition during the translation. This is helpful for legacy ACID tables which *required* a bucket definition but weren't a part of the intended design. The migration provides an opportunity to correct this artificial design element.

With the default value `2`, we will *remove* CLUSTERING from any ACID table definitions with `2` or fewer buckets defined. If you wish to keep ALL CLUSTERED definitions, regardless of size, set this value to `0`.

There is now an option to 'downgrade' ACID tables to EXTERNAL/PURGE during migration using the `-da` option.

The ACID Migration Process

The ACID migration builds a 'transfer' table on the LEFT cluster, a 'legacy' managed table (when the LEFT is a legacy cluster), or an 'EXTERNAL/PURGE' table. Data is copied to this transfer table from the original ACID table via SQL.

Since the clusters are linked ([Linking Cluster Storage Layers](#)), we build a 'shadow' table that is 'EXTERNAL' on the 'RIGHT' cluster that uses the data in the 'LEFT' cluster. Similar to the LINKED data strategy. If the data is partitioned, we run `MSCK` on this 'shadow' table in the 'RIGHT' cluster to discover all the partitions.

The final ACID table is created in the 'RIGHT' cluster, and SQL is used to copy data from the 'LEFT' cluster via the 'shadow' table.

Requirements

- Data Strategy: `HYBRID`, `SQL`, or `EXPORT_IMPORT`

- Activate Migrate ACID: `-ma|-mao`

- Link Clusters ([Linking Cluster Storage Layers](#)), unless using the `-is|--intermediate-storage` option.

- This is a 'ONE' time transfer. It is not an incremental update process.

- Adequate Storage on LEFT to make an 'EXTERNAL' copy of the ACID table.

- Permissions:

  - From the RIGHT cluster, the submitting user WILL need access to the LEFT cluster's storage layer (HDFS) to create the shadow table (with location) that points across clusters.

  - doas will have a lot to do with the permissions requirements.

  - The 'hive' service account on the RIGHT cluster will need elevated privileges to the LEFT storage LAYER (HDFS). For example: If the hive service accounts on each cluster DO NOT share the same identity, like `hive`, then the RIGHT hive identity MUST also have privileged access to the LEFT clusters HDFS layer.

- Partitioned tables must have data that is 'discoverable' via `MSCK`. NOTE: The METADATA activity and REDUCER restrictions to the number of BUCKETs can dramatically affect this.- The number of partitions in the source ACID tables must be below the `partitionLimit` (default 500). This strategy may not be successful when the partition count is above this, and we won't even attempt the conversion. Check YARN for the progress of jobs with a large number of partitions/buckets. Progress many appear stalled from 'hms-mirror'.

- ACID table migration to Hive 1/2 is NOT supported due to the lack of support for "INSERT OVERWRITE" on transactional tables. Hive 1/2 to Hive 3 IS support and the target of this implementation. Hive 3 to Hive 3 is also supported.

Replace ACID -r or --replace

When downgrading ACID tables during migration, the -r option will give you the option to 'replace' the original ACID table with the a table that is no longer ACID. This option is only available along with the -da and SQL data strategy options.

Intermediate/Common Storage Options

When bridging the gap between two clusters, you may find they can't share/link storage. In this case, using one of these options will help you with the transfer.

The -is or --intermediate-storage option is consider a transient location that both cluster can share, see, and have access to. The strategies for transferring data (EXPORT_IMPORT, SQL, HYBRID) will use this location to facilitate the transfer. This is a common strategy when migrating from on-prem environments to the cloud.

The -cs or --common-storage option is similar to -is but this option ends up being the final resting place for the data, not just the transfer location. And with this option, we can streamline the jumps required to migrate data. Again, this location needs to be accessible to both clusters.

Non-Native Hive Tables (Hbase, KAFKA, JDBC, Druid, etc..)

Any table definition without a LOCATION element is typically a reference to an external system like: HBase, Kafka, Druid, and/or (but not limited to) JDBC.

Requirements

These references require the environment to be:

- Correctly configured to use these resources

- Include the required libraries in the default hive environment.

- The referenced resource must exist already BEFORE the 'hive' DDL will successfully run.

AVRO Tables

AVRO tables can be designed with a 'reference' to a schema file in `TBLPROPERTIES` with `avro.schema.url`. The referenced file needs to be 'copied' to the *RIGHT* cluster BEFORE the `CREATE` statement for the AVRO table will succeed.

Add the `-asm|--avro-schema-move` option at the command line to *copy* the file from the LEFT cluster to the RIGHT cluster.

As long as the clusters are linked ([Linking Cluster Storage Layers](#)) and the cluster `hcfsNamespace` values are accurate, the user's credentials running `hms-mirror` will attempt to copy the schema file to the *RIGHT* cluster BEFORE executing the `CREATE` statement.

Requirements

- Link Clusters ([Linking Cluster Storage Layers](#)) for Data Strategies: `SCHEMA_ONLY`, `SQL`, `EXPORT_IMPORT`, and `HYBRID`

- Running user must have 'namespace' access to the directories identified in the `TBLPROPERTIES` key `avro.schema.url`.

- The user running `hms-mirror` will need enough storage level permissions to copy the file.

- When hive is running with `doas=false`, `hive` will need access to this file.

Warnings

- With the `EXPORT_IMPORT` strategy, the `avro.schema.url` location will NOT be converted. It may lead to an issue reading the table if the location includes a prefix of the cluster's namespace OR the file doesn't exist in the new cluster.

Table Translations

Legacy Managed Tables

`hms-mirror` will convert 'legacy' managed tables in Hive 1 or 2 to EXTERNAL tables in Hive 3. It relies on the `legacyHive` setting in the cluster configurations to accurately make

this conversion. So make sure you've set this correctly.

distcp Planning Workbook and Scripts

hms-mirror will create source files and a shell script that can be used as the basis for the 'distcp' job(s) used to support the databases and tables requested in -db. hms-mirror will NOT run these jobs. It will provide the basic job constructs that match what it did for the schemas. Use these constructs to build your execution plan and run these separately.

The constructs created are intended as a *one-time* transfer. If you are using *SNAPSHOTS* or --update flags in distcp to support incremental updates, you will have to make additional modifications to the scripts/process. Note: For these scenarios, hms-mirror supports options like -ro|--read-only and -sync.

Each time hms-mirror is run, *source* files for each database are created. These source files need to be copied to the distributed filesystem and reference with an -f option in distcp. We also create a *basic* shell script that can be used as a template to run the actual distcp jobs.

Depending on the job size and operational expectations, you may want to use *SNAPSHOTS* to ensure an immutable source or use a diff strategy for more complex migrations. Regardless, you'll need to make modifications to these scripts to suit your purposes.

If your process requires the data to exist BEFORE you migrate the schemas, run hms-mirror in the dry-run mode (default) and use the distcp planning workbook and scripts to transfer the datasets. Then run hms-mirror with the -e|--execute option to migrate the schemas.

These workbooks will NOT include elements for ACID/Transactional tables. Simply copying the dataset for transactional tables will NOT work. Use the HYBRID data strategy migration transactional table schemas and datasets.

ACID Table Downgrades

The default table creation scenario for Hive 3 (CDP and HDP 3) installations is to create ACID transactional tables. If you're moving from a legacy platform like HDP 2 or CDH, this may have caught you off guard and resulted in a lot of ACID tables you did NOT intend on.

The -da|--downgrade-acid option can be used to convert these ACID tables to *EXTERNAL/PURGE* tables.

If you have ACID tables on the current platform and would like to *downgrade* them, but you're not doing a migration, try the -ip|--in-place option. This will archive to existing ACID table and build a new table (with the original table name) that is *EXTERNAL/PURGE*.

Reset to Default Locations

Migrations present an opportunity to clean up a lot of history. While hms-mirror was originally designed to migration data and maintain the **relative** locations of the data, some may want to reorganize the data during the migration.

The option -rdl|--reset-default-location will overwrite the locations originally used and place the datasets in the 'default' locations as defined by -wd|--warehouse-directory and -ewd|--external-warehouse-directory.

The -rdl option requires -wd and -ewd to be specified. These locations will be used to ALTER the databases LOCATION and MANAGEDLOCATION values. After which, all new CREATE \[EXTERNAL\] TABLE definitions don't specify a LOCATION, which means table locations will use the default.

Legacy Row Serde Translations

By default, tables using old row *serde* classes will be converted to the newer serde as the definition is processed by hms-mirror. See here (https://github.com/cloudera-labs/hms-mirror/blob/6f6c309f24fbb8133e5bd52e5b18274094ff5be8/src/main/java/com/cloudera/utils/hadoop/hms/mirror/feature/LegacyTranslations.java#L28) for a list of serdes we look for.

If you do NOT want to apply this translation, add the option -slt|--skip-legacy-translation to the commandline.

Filtering Tables to Process

There are options to filter tables included in hms-mirror process. You can select -tf|--table-filter to "include" only tables that match this 'regular-expression'. Inversely, use -

etf|--exclude-table-filter to omit tables from the list. These options are mutually exclusive.

The filters for -tf and -tef are expressed as a 'regular-expression'. Complex expressions should be enclosed in quotes to ensure the commandline interpreter doesn't split them.

Additional table filters (-tfs|--table-filter-size-limit and -tfp|--table-filter-partition-count-limit) that check a tables data size and partition count limits can also be applied to narrow the range of tables you'll process.

The filter does NOT override the requirement for options like -ma|-mao. It is used as an additional filter.

Migrations between Clusters WITHOUT line of Site

There will be cases where clusters can't be linked (Linking Cluster Storage Layers). And without this line of sight, data movement needs to happen through some other means.

On-Prem to Cloud

This scenario is most common with "on-prem" to "cloud" migrations. Typically, hms-mirror is run from the RIGHT cluster, but in this case the RIGHT cloud cluster doesn't have line of site to the LEFT on-prem cluster. But the on-prem cluster will have limited line of site to the cloud environment. In this case, the only option is to run hms-mirror from the on-prem cluster. The on-prem cluster will need access to either an -is|--intermediate-storage location that both clusters can access or a -cs|--common-storage location that each cluster can see, but can be considered the final resting place for the cloud environment. The -cs option usually means there are fewer data hops required to complete the migration.

You'll run hms-mirror from a LEFT cluster edgenode. This node will require line of site to the Hive Server 2 endpoint in the RIGHT cloud environment. The LEFT cluster will also need to be configured with the appropriate libraries and configurations to write to the location defined in -is|-cs. For example: For S3, the LEFT cluster will need the AWS S3 libraries configured and the appropriate keys for S3 access setup to complete the transfer.

Shared Storage Models (Isilon, Spectrum-Scale, etc.)

There are cases where 'HDFS' isn't the primary data source. So the only thing the cluster share is storage in these 'common' storage units. You want to transfer the schema, but the data doesn't need to move (at least for 'EXTERNAL' (non-transactional) tables). In this case, try the `-d|--data-strategy` COMMON. The schema's will go through all the needed conversions while the data remains in the same location.

Disconnected Mode

Use the `-rid|--right-is-disconnected` mode when you need to build (and/or) transfer schema/datasets from one cluster to another, but you can't connect to both at the same time. See the issues log for details regarding the cases here issue #17 (https://github.com/cloudera-labs/hms-mirror/issues/17).

Use cases:

- Schema Only Transfers

- SQL, EXPORT_IMPORT, and HYBRID only when -is or -cs is used. This might be the case when the clusters are secure (kerberized), but don't share a common kerberos domain/user auth. So an intermediate or common storage location will be used to migrate the data.

- Both clusters (and HS2 endpoints) are Kerberized, but the clusters are NOT the same major hadoop version. In this case, hms-mirror doesn't support connecting to both of these endpoints at the same time. Running in the disconnected mode will help push through with the conversion.

hms-mirror will run as normal, with the exception of examining and running scripts against the right cluster. It will be assumed that the RIGHT cluster elements do NOT exist.

The RIGHT_ 'execution' scripts and distcp commands will need to be run MANUALLY via Beeline on the RIGHT cluster.

Note: This will be know as the "right-is-disconnected" option. Which means the process should be run from a node that has access to the "left" cluster. This is 'counter' to our general recommendation that the process should be run from the 'right' cluster.

No-Purge Option

`-np`

Feature Request #25 (https://github.com/cloudera-labs/hms-mirror/issues/25) was introduced in v1.5.4.2 and gives the user to option to remove the `external.table.purge` option that is added when converting legacy managed tables to external table (Hive 1/2 to 3). This does affect the behavior of the table from the older platforms.

Property Overrides

`-po[l|r] <key=value>[,<key=value>]...`

Feature Request #27 (https://github.com/cloudera-labs/hms-mirror/issues/27) introduced in v1.5.4.2 provides the ability to set a hive properties at the beginning of each migration part. This is a comma separated list of key=value pairs with no space. If spaces are needed, quote the parameter on the commandline.

You can use `-po` to set the properties for BOTH clusters or `-pol|-por` to set them specifically for the 'left' and/or 'right' cluster.

For example: `-po hive.exec.orc.split.strategy=BI,hive.compute.query.using.stats=false`

To provide a consistent list of settings for ALL jobs, add/modify the following section in the configuration file ie: `default.yaml` used for processing. In this case you do NOT need to use the commandline option. Although, you can set basic values in the configuration file and add other via the commandline.

Notice that there are setting for the LEFT and RIGHT clusters.

```
optimization:
  overrides:
    left:
      tez.queue.name: "compaction"
    right:
      tez.queue.name: "migration"
```

Global Location Map

`-glm|--global-location-map <from=to>[,...]`

This is an opportunity to make some specific directory mappings during the migration. You can supply a comma separated list of directory pairs to be use for evaluation.

```
-glm /data/my_original_location=/corp/finance_new_loc,/user/hive=/warehouse/hive
```

These directory mappings ONLY apply to 'EXTERNAL' and 'DOWNGRADED ACID' tables. You can supply 'n' number of mappings to review through the commandline interface as describe above. To provide a consistent set of mappings to ALL jobs, add/modify the following section in the configuration file ie: `default.yaml` used for processing.

```yaml
translator:
  globalLocationMap:
    /data/my_original_location: "/corp/finance_new_loc"
    /user/hive: "/warehouse/hive"
```

The list will be sorted by the length of the string, then alpha-numerically. This will ensure the deepest nested paths are evaluated FIRST. When a path is matched during evaluation, the process will NOT look and any more paths in the list. Therefore, avoiding possible double evaluations that may result when there are nested paths in the list.

Paths are evaluated with 'startsWith' on the original path (minus the original namespace). When a match is found, the path 'part' will be replaced with the value specified. The remaining path will remain intact and regardless of the `-rdl` setting, the LOCATION element will be included in the tables new CREATE statement.

Force External Locations

`-fel|--force-external-location`

Under some conditions, the default warehouse directory hierarchy is not honored. We've seen this in HDP 3. The `-rdl` option collects the external tables in the default warehouse directory by omitting the LOCATION element in the CREATE statement, relying on the default location. The default location is set at the DATABASE level by `hms-mirror`.

In HDP3, the CREATE statement doesn't honor the 'database' LOCATION element and reverts to the system wide warehouse directory configurations. The `-fel` flag will simply include the 'properly' adjusted LOCATION element in the CREATE statement to ensure the tables are created in the desired location. This setting overrides the effects intended by the `-rdl` option which intend to place the tables under the stated warehouse locations by omitting the location from the tables definition and relying on the default location specified in the database.

-fel will use the original location as a starting point. If -wd|-ewd are specified, they aren't not used in the translation, but warnings may be issued if the final location doesn't align with the warehouse directory. The effect change in the location when using -fel, add mappings via -glm.

HDP 3 Hive

HDP 3 (Hive 3) was an incomplete implementation with regards to complex table 'location' management. When you are working in this environment, add to the cluster configuration (LEFT or RIGHT) the setting: hdpHive3: true. There is NOT a commandline switch for this. JUst add it to the configuration file you're using to run the application.

```
clusters:
  LEFT:
    environment: "LEFT"
    legacyHive: false
    hdpHive3: true
```

HDP3 Hive did NOT have a MANAGEDLOCATION attribute for Databases.

The LOCATION element tracked the Manage ACID tables and will control where they go.

This LOCATION will need to be transferred to MANAGEDLOCATION 'after' upgrading to CDP to ensure ACID tables maintain the same behavior. EXTERNAL tables will explicity set there LOCATION element to match the setting in -ewd.

Future external tables, when no location is specified, will be created in the hive.metastore.warehouse.external.dir. This value is global in HDP Hive3 and can NOT be set for individual databases.

Post upgrade to CDP, you should add a specific directory value at the database level for better control.

Schema Fix Features

Schema Fix Features are a way to inject special considerations into the replay of a schema between clusters. Each schema is automatically check is a particular 'feature' applies.

If you find that this features check is causing issues, add the flag `-sf` to the application parameters and the feature checks will be skipped.

BAD_ORC_DEF

`BAD_ORC_DEF` is a feature that corrects poorly executed schema definitions in legacy Hive 1/2 that don't translate into a functioning table in Hive 3. In this case, the legacy definition was defined with:

```
ROW FORMAT DELIMITED
  FIELDS TERMINATED BY '\t'
  LINES TERMINATED BY '\n'
STORED AS INPUTFORMAT
  'org.apache.hadoop.hive.ql.io.orc.OrcInputFormat'
OUTPUTFORMAT
  'org.apache.hadoop.hive.ql.io.orc.OrcOutputFormat'
```

when it should have been created with:

```
STORED AS ORC
```

The result, when not modified and replayed in Hive 3 is a table that isn't functional. The `BAD_ORC_DEF` feature will replace:

```
ROW FORMAT DELIMITED
  FIELDS TERMINATED BY '\t'
  LINES TERMINATED BY '\n'
```

with:

```
ROW FORMAT SERDE
  'org.apache.hadoop.hive.ql.io.orc.OrcSerde'
```

BAD_RC_DEF

`BAD_RC_DEF` is a feature that corrects poorly executed schema definitions in legacy Hive 1/2 that doesn't translate into a functioning table in Hive 3. In this case, the legacy definition was defined with:

```
ROW FORMAT DELIMITED,
    FIELDS TERMINATED BY '|'
 STORED AS INPUTFORMAT
  'org.apache.hadoop.hive.ql.io.RCFileInputFormat'
 OUTPUTFORMAT
  'org.apache.hadoop.hive.ql.io.RCFileOutputFormat'
```

when it should have been created with:

```
STORED AS RCFILE
```

The result, when not modified and replayed in Hive 3 is a table that isn't functional. The BAD_RC_DEF feature will replace:

```
ROW FORMAT DELIMITED,
    FIELDS TERMINATED BY '|'
 STORED AS INPUTFORMAT
```

with:

```
STORED AS RCFILE
```

BAD_TEXTFILE_DEF

Older Textfile schemas somehow are corrupted through subsequent ALTER statements that get the table into a state where you can NOT re-run the contents of SHOW CREATE TABLE. In this case, the issue is that there is a declaration for WITH SERDEPROPERTIES along with a ROW FORMAT DELIMITED clause. These two can NOT exist together. Here is an example of this:

```
ROW FORMAT DELIMITED
    FIELDS TERMINATED BY '|'
    LINES TERMINATED BY '\n'
WITH SERDEPROPERTIES (
    'escape.delim'='\\')
STORED AS INPUTFORMAT
    'org.apache.hadoop.mapred.TextInputFormat'
```

```
OUTPUTFORMAT
    'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
```

In this case, we need to convert the ROW FORMAT DELIMITED * TERMINATED BY * values into the SERDEPROPERTIES and replace it with

```
ROW FORMAT SERDE
  'org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe'
```
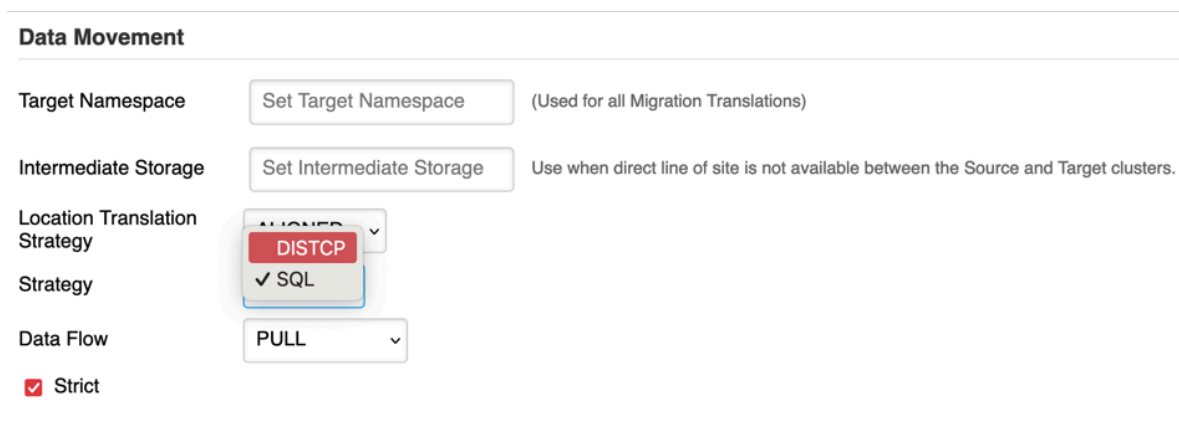
Strategies

A 'strategy' in `hms-mirror` is the method used to migrate metadata and possibly data from one metastore to another.

Some strategies are simply responsible for migrating metadata, while others are responsible for migrating metadata and data.

Data Movement

When the strategy is responsible for moving data as well, you have two options: `SQL` and `distcp`. You can set this option here:

dm_options.png

SQL will rely on the engine and the clusters ability to see across data storage environments to move the data.

For distcp (Distributed Copy) (https://hadoop.apache.org/docs/current/hadoop-distcp/DistCp.html), we'll build a plan that matches what you've asked to migrate.

For a matrix of when this might be available, see Location Alignment (Location Alignment).

SCHEMA_ONLY

This strategy is used to migrate "only" the schema (mostly). When migrating from a `legacy` (Hive 1/2) hive environment to a `non-legacy` (Hive 3+) `hms-mirror` will convert tables that are "managed non-transactional" to "external/purge". This translation is a part of the HSMM (Hive Strict Managed Migration) process that run for 'in-place' upgrades.

`hms-mirror` is mostly designed for "side-car" migrations which involves two separate clusters.

Options

-f|--flip

The configuration yaml has two clusters defined: LEFT and RIGHT. By default, all work moves from LEFT to RIGHT. If you need to perform actions from RIGHT to LEFT, use the `-f` option to switch these clusters. The final report will show that these clusters have been reversed.

-ma|--migrate-acid or -mao|--migrate-acid-only \[optional artificial bucket limit\] \(default is 2\)

The default behaviour is NOT to handle ACID tables. ACID tables require some additional handling. When you need to address ACID tables, use these flags to either include them OR migrate ONLY ACID tables.

Migrations from legacy hive environments may have had to create tables with an unnecessary CLUSTERED BY clause to support bucketing for these ACID tables. The artificial bucket limit is an opportunity to remove those definitions for the migrated tables. For example: If you've defined a table with 2 buckets, the default artificial bucket limit is 2, so the table CLUSTERED BY ... INTO 2 BUCKETS will be removed from the definition.

-mnno|--migrate-non-native-only

Use this to migrate *non-native* hive tables like: Kafka, HBase, JDBC Federated, etc.. This option excludes *native* tables from processing, so you should run it separately.

> ⚠ Many of these tables **REQUIRE** that the supporting references **EXIST** first, or their creation will **FAIL**. For instance, migrating a table with the

> HBase Storage handler requires that the HBase table is *present* AND *visible* from Hive.

-v|--views-only

Like `-mnno` the `-v` option excludes other tables from processing, so it's designed to run along. And like the `-mnno` option, it too requires the supporting tables to exist, otherwise the view create statements will *fail*.

-asm|--avro-schema-migration

AVRO tables have a `TBLPROPERTIES` key `avro.schema.url` that identifies the supporting avro schemas location on the current cluster. When migrating AVRO tables, use this option to check for this setting. As a result, `hms-mirror` will parse out the location and replace it with the same relative location.

`hms-mirror` will use the configured environments `hdfs|core-site.xml` files to launch an `hdfs` client that will copy the supporting avro schema between clusters. For this to work, the clusters MUST be visible to each other ([Linking Cluster Storage Layers](#)).
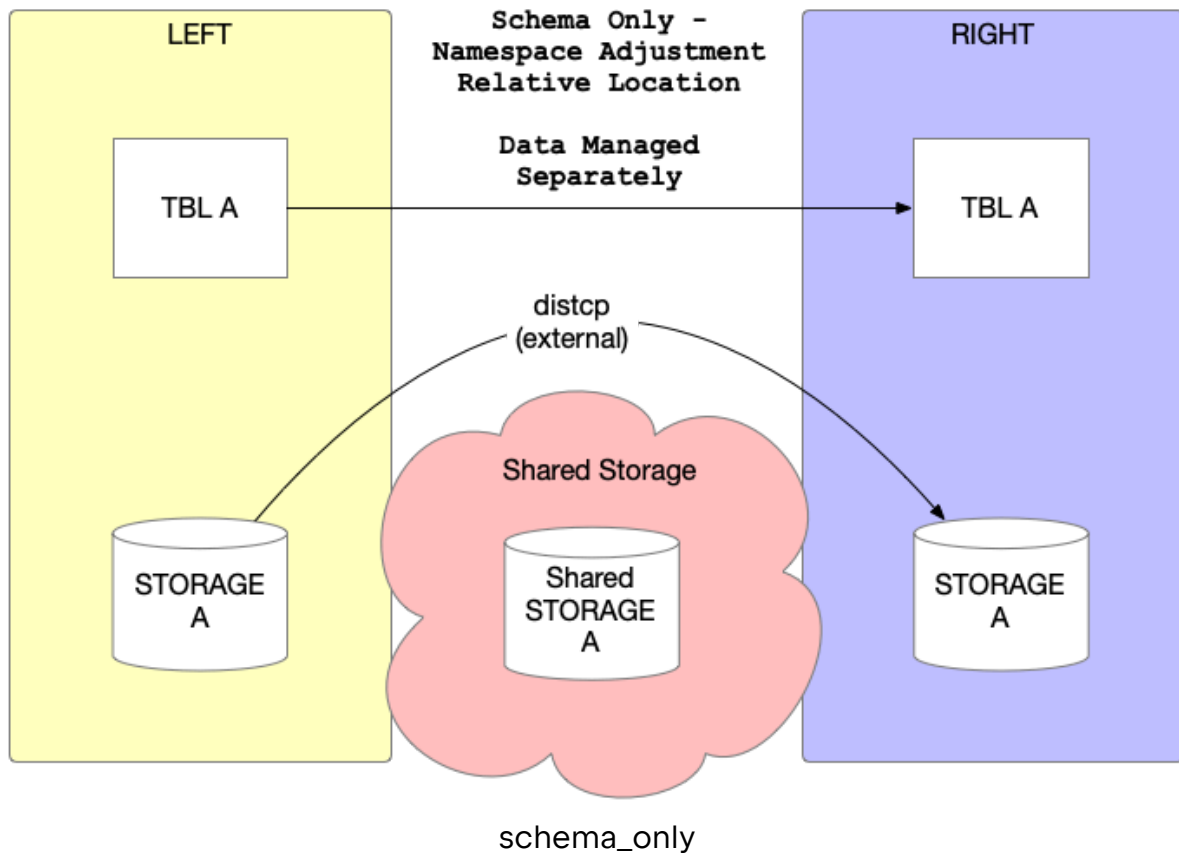
The DUMP strategy is like SCHEMA_ONLY; it just doesn't require the RIGHT cluster to be connected. Although, it does need the following settings for the RIGHT cluster to make the proper adjustments:
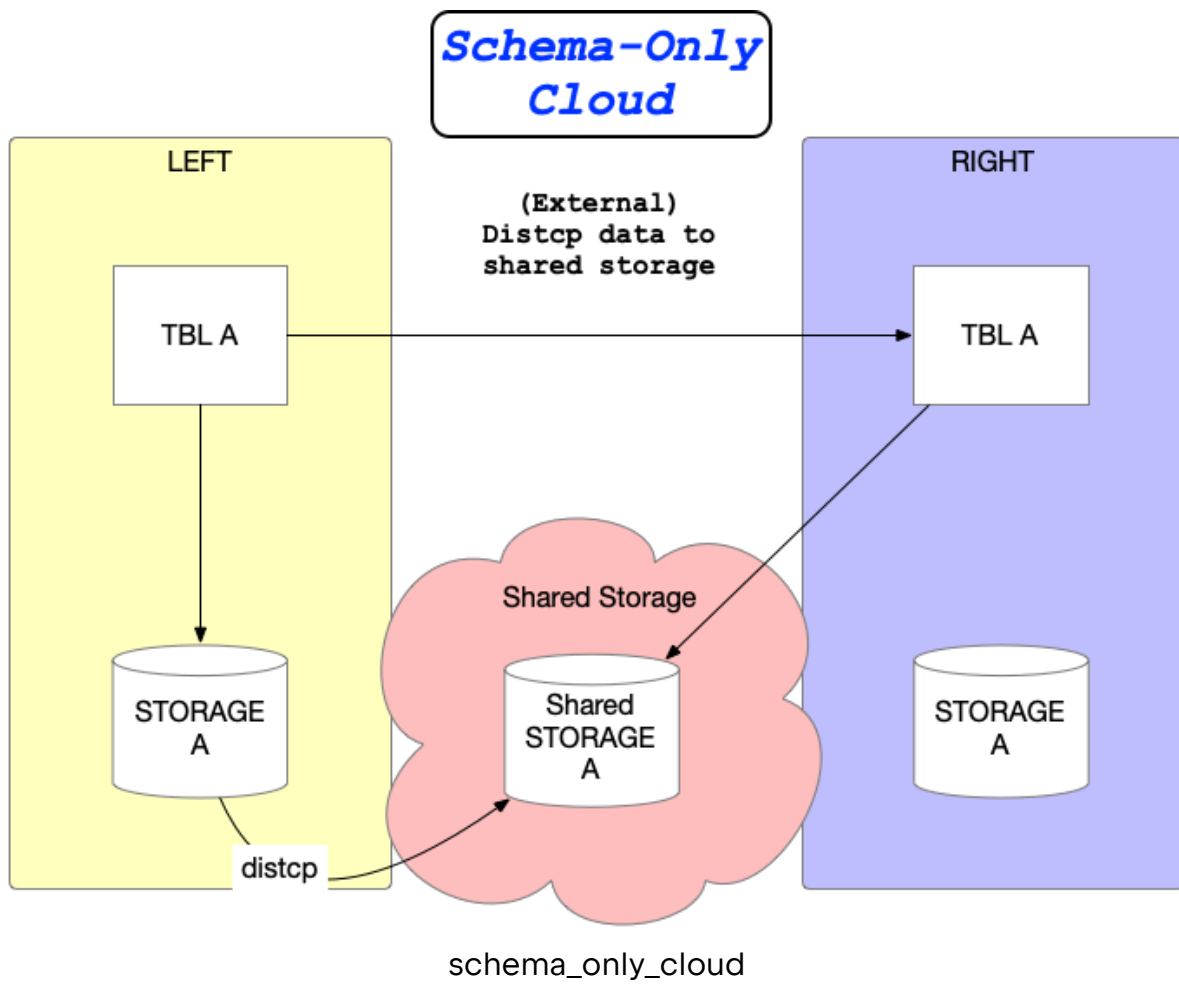
```
legacyHive
hcfsNamespace
hiveServer2 -> partitionDiscovery
```

When the option `-ma` (migrate acid) is specified, the ACID schema's will be migrated/dumped. It is essential to know that the data for ACID tables can NOT simply be copied from one clusters hive instance to another. The data needs to be extracted to a none ACID table, then use an external table definition to read and INSERT the data into the new ACID table on the RIGHT cluster. For those that insist on trying to simply copy the data.... you've been warned ;).

With the DUMP strategy, you'll have a 'translated' (for legacy hive) table DDL that can be run on the new cluster independently.

Schema-Only

schema_only

schema_only_cloud

DUMP

This strategy is used to migrate "only" the schema (mostly). When migrating from a legacy (Hive 1/2) hive environment to a non-legacy (Hive 3+) hms-mirror will convert tables that are "managed non-transactional" to "external/purge". This translation is a part of the HSMM (Hive Strict Managed Migration) process that run for 'in-place' upgrades.

hms-mirror is mostly designed for "side-car" migrations which involves two separate clusters.

Options

-f|--flip

The configuration yaml has two clusters defined: LEFT and RIGHT. By default, all work moves from LEFT to RIGHT. If you need to perform actions from RIGHT to LEFT, use the `-f` option to switch these clusters. The final report will show that these clusters have been reversed.

-ma|--migrate-acid or -mao|--migrate-acid-only

`[optional artificial bucket limit]` (default is 2)

The default behaviour is NOT to handle ACID tables. ACID tables require some additional handling. When you need to address ACID tables, use these flags to either include them OR migrate ONLY ACID tables.

Migrations from legacy hive environments may have had to create tables with an unnecessary `CLUSTERED BY` clause to support `bucketing` for these ACID tables. The `artificial bucket limit` is an opportunity to remove those definitions for the migrated tables. For example: If you've defined a table with 2 buckets, the default `artificial bucket limit` is 2, so the table `CLUSTERED BY … INTO 2 BUCKETS` will be removed from the definition.

[-mnno|--migrate-non-native-only]

See Non-Native Tables (["Non-Native Hive Tables (Hbase, KAFKA, JDBC, Druid, etc..)" in "Features"](#)) for more information.

Use this to migrate *non-native* hive tables like: Kafka, HBase, JDBC Federated, etc.. This option excludes *native* tables from processing, so you should run it separately.

Note that many of these tables **REQUIRE** that the supporting references **EXIST** first, or their creation will **FAIL**. For instance, migrating a table with the HBase Storage handler requires that the HBase table is *present* AND *visible* from Hive.

[-v|--views-only]

See Views (["VIEWS" in "Features"](#)) for more information.

Like `-mnno` the `-v` option excludes other tables from processing, so it's designed to run along. And like the `-mnno` option, it too requires the supporting tables to exist, otherwise the view create statements will *fail*.

-asm|--avro-schema-migration

AVRO tables have a `TBLPROPERTIES` key `avro.schema.url` that identifies the supporting avro schemas location on the current cluster. When migrating AVRO tables, use this option to check for this setting. As a result, `hms-mirror` will parse out the location and replace it with the same relative location.

`hms-mirror` will use the configured environments `hdfs|core-site.xml` files to launch an `hdfs` client that will copy the supporting avro schema between clusters. For this to work, the clusters MUST be visible to each other ([Linking Cluster Storage Layers](#)).
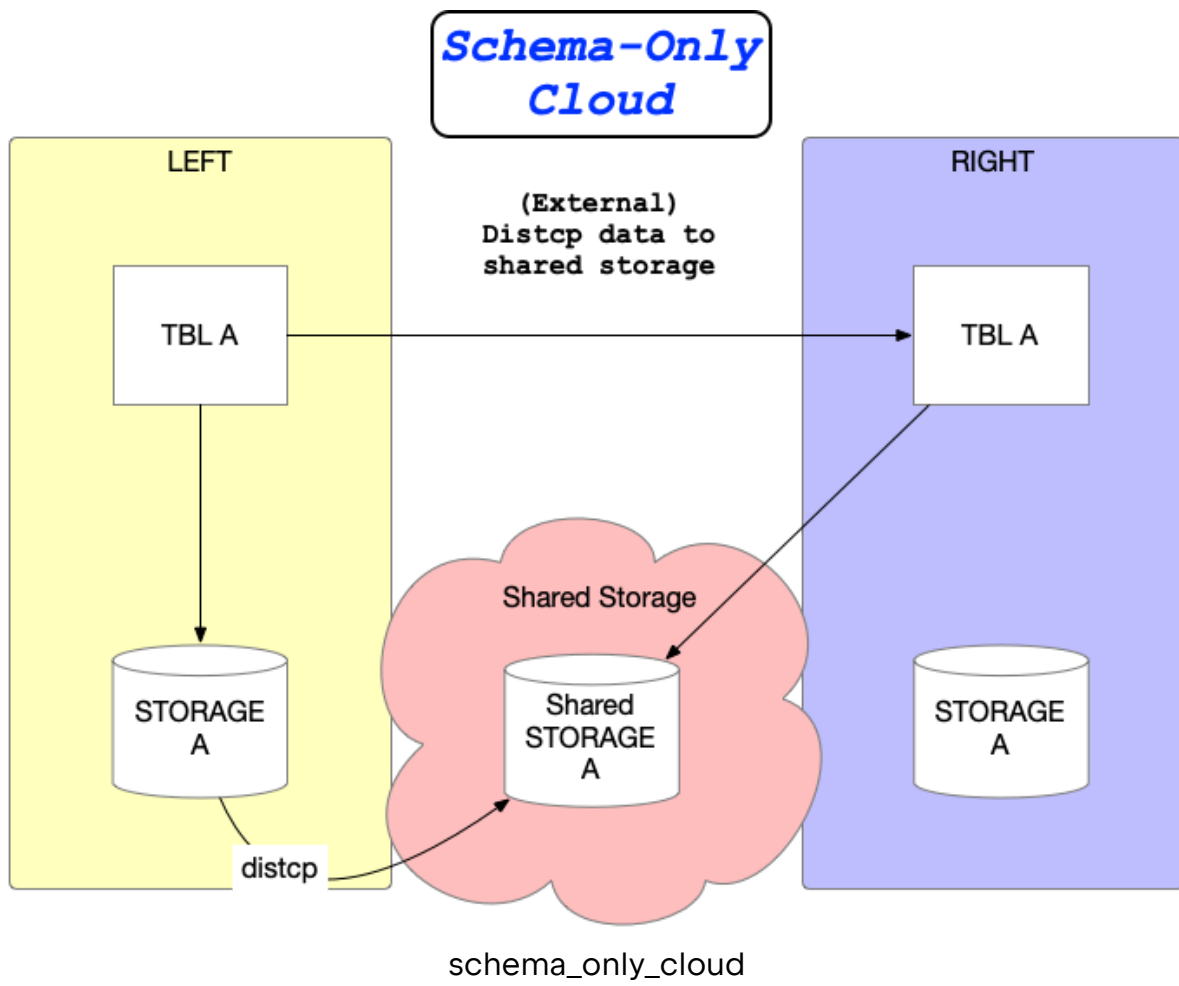
The DUMP strategy is like SCHEMA_ONLY; it just doesn't require the RIGHT cluster to be connected. Although, it does need the following settings for the RIGHT cluster to make the proper adjustments:

```
legacyHive
hcfsNamespace
hiveServer2 -> partitionDiscovery
```

When the option `-ma` (migrate acid) is specified, the ACID schema's will be migrated/dumped. It is essential to know that the data for ACID tables can NOT simply be copied from one clusters hive instance to another. The data needs to be extracted to a none ACID table, then use an external table definition to read and INSERT the data into the new ACID table on the RIGHT cluster. For those that insist on trying to simply copy the data.... you've been warned ;).

With the DUMP strategy, you'll have a 'translated' (for legacy hive) table DDL that can be run on the new cluster independently.

Schema-Only

LEFT   Schema Only – Namespace Adjustment Relative Location   RIGHT

Data Managed Separately

TBL A → TBL A

distcp (external)

Shared Storage

STORAGE A   Shared STORAGE A   STORAGE A

schema_only

schema_only_cloud

LINKED

Assumes the clusters are linked ([Linking Cluster Storage Layers](#)). We'll transfer the schema and leave the location as is on the new cluster.

This provides a means to test hive on the RIGHT cluster using the LEFT cluster's storage.

The `-ma` (migrate acid) tables option is NOT valid in this scenario and will result in an error if specified.

WARNING: If the LOCATION element is specified in the database definition AND you use `DROP DATABASE ... CASCADE` from the RIGHT cluster, YOU WILL DROP THE DATA ON THE LEFT CLUSTER even though the tables are NOT purgeable. This is the DEFAULT behavior of hive 'DROP DATABASE'. So BE CAREFUL!!!!

CONVERT_LINKED

If you migrated schemas with the Linked ([LINKED](#)) strategy and don't want to drop the database and run SCHEMA_ONLY to adjust the locations from the LEFT to the RIGHT cluster, use this strategy to make the adjustment.

This will work only on tables that were migrated already. It will reset the location to the same relative location on the RIGHT clusters `hcfsNamespace`. This will also check to see if the table was a 'legacy' managed table and set the `external.table.purge` flag appropriately. Tables that are 'partitioned' will be handled differently, since each partition has a reference to the 'linked' location. Those tables will first be validated that they NOT `external.table.purge`. If they are, that property will 'UNSET'. Then the table will be dropped, which will remove all the partition information. Then they'll be created again and `MSCK` will be used to discover the partitions again on the RIGHT clusters namespace.

This process does NOT move data. It expects that the data was migrated in the background, usually by `distcp` and has been placed in the same relative locations on the RIGHT clusters `hcfsNameSpace`.

This process does NOT work for ACID tables.

AVRO table locations and SCHEMA location and definitions will be changed and copied.

SQL

The **SQL** data strategy will use Hive SQL to move data between clusters. When the cluster don't have direct line of sight to each other and can NOT be linked ([Linking Cluster Storage Layers](#)), you can use options like `-cs` or `-is` to bridge the gap.

sql
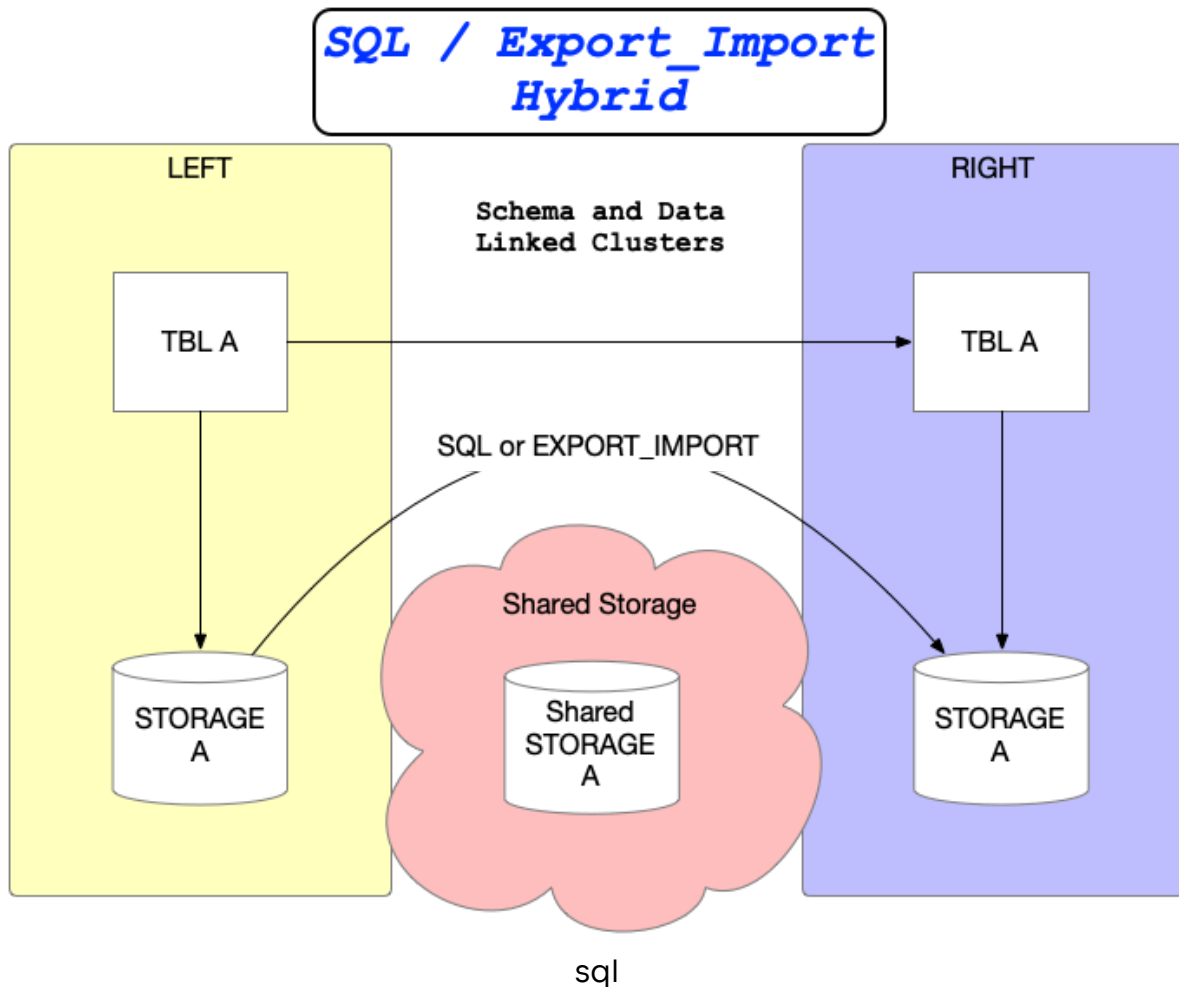
Options

Use `-ep|--export-partition-count <limit>` to set the limit for the number of partitions to use EXPORT_IMPORT. The default is 100. When a table has a partition count that exceeds this value, the SQL ([SQL](#)) strategy will be used.

-sp,--sql-partition-count <limit>

Sets the limit for the number of partitions that the SQL ([SQL](#)) strategy will process. If the value is exceeded, the process will NOT migrate the table. The default is 500 .

To persist a higher value without specifying the `-sp` option, add the following to the config ([Default Configuration Template](#)).

```
hybrid:
  sqlPartitionLimit: <limit>
```

-ma|--migrate-acid or -mao|--migrate-acid-only

Include ACID tables in processing.

-da|--downgrade-acid

Applicable only when `-ma|o` is used. This will take the ACID tables and downgrade them to *EXTERNAL/PURGE* tables. Buckets adjustments (["ACID Tables" in "Features"](#)) are applicable.

-dc|--distcp

For EXTERNAL tables, this will build a `distcp` plan that can be used to transfer the database tables.

When the option `-ma|o` is used, ACID tables will be migrated. *ACID* tables will be converted to EXTERNAL transfer tables for the `distcp` operations. On the other cluster, the tables will be created initially as *EXTERNAL,* then transferred back over to *ACID* tables, unless `-da` is used.

If the `-da` is used the tables will remain *EXTERNAL* on the new cluster.

When `-is` is used, there will be `distcp` operations required on both clusters to handle data movement to/from the `intermediate-storage` area.

-is|--intermediate-storage or -cs|--common-storage

When the clusters aren't linked ([Linking Cluster Storage Layers](#)) these two options provide a way to transfer data through an intermediate/common storage location. Each cluster must have access to these locations. These values are mutually exclusive.

`--intermediate-storage` requires an addition transfer of data. The LEFT transfers data to the `-is` location and the RIGHT cluster uses that data and initiates a transfer from the location to the final hcfsNamespace value set for the cluster. This is a two copy migration.

intermediate
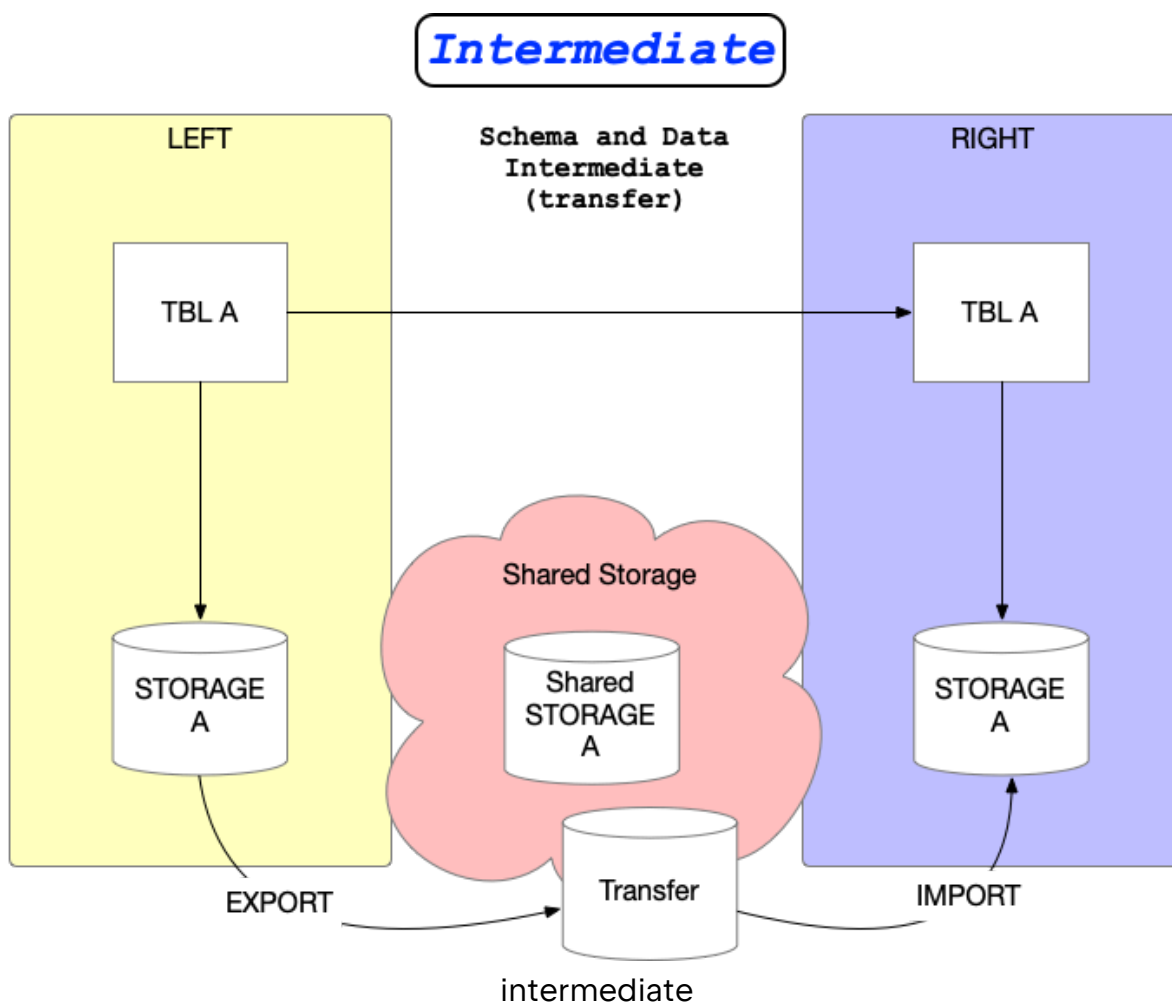
`--common-storage` assumes the location is also the final resting place for the data. Some optimizations are possible (ACID downgrades) that reduce that times data need to move.

This is a single copy migration.

-wd|--warehouse-directory and -ewd|--external-warehouse-directory

Are used to set the *databases* default locations for managed and external tables. This overrides the systems *hive metastore* properties.

-rdl|--reset-to-default-location

Regardless of where the source data *relative* location was on the filesystem, this will reset it to the default location on the new cluster.

If `-dc|--distcp` is used, then the `warehouse` options are required in order for `hms-mirror` to build the `distcp` workplan.

EXPORT_IMPORT

We'll use EXPORT_IMPORT to get the data to the new cluster. The default behavior requires the clusters to be linked ([Linking Cluster Storage Layers](#)).

EXPORT to a location on the LEFT cluster where the RIGHT cluster can pick it up with IMPORT.

When `-ma` (migrate acid) tables are specified, and the LEFT and RIGHT cluster DON'T share the same 'legacy' setting, we will NOT be able to use the EXPORT_IMPORT process due to incompatibilities between the Hive versions. We will still attempt to migrate the table definition and data by copying the data to an EXTERNAL table on the lower cluster and expose this as the source for an INSERT INTO the ACID table on the RIGHT cluster.

export_import
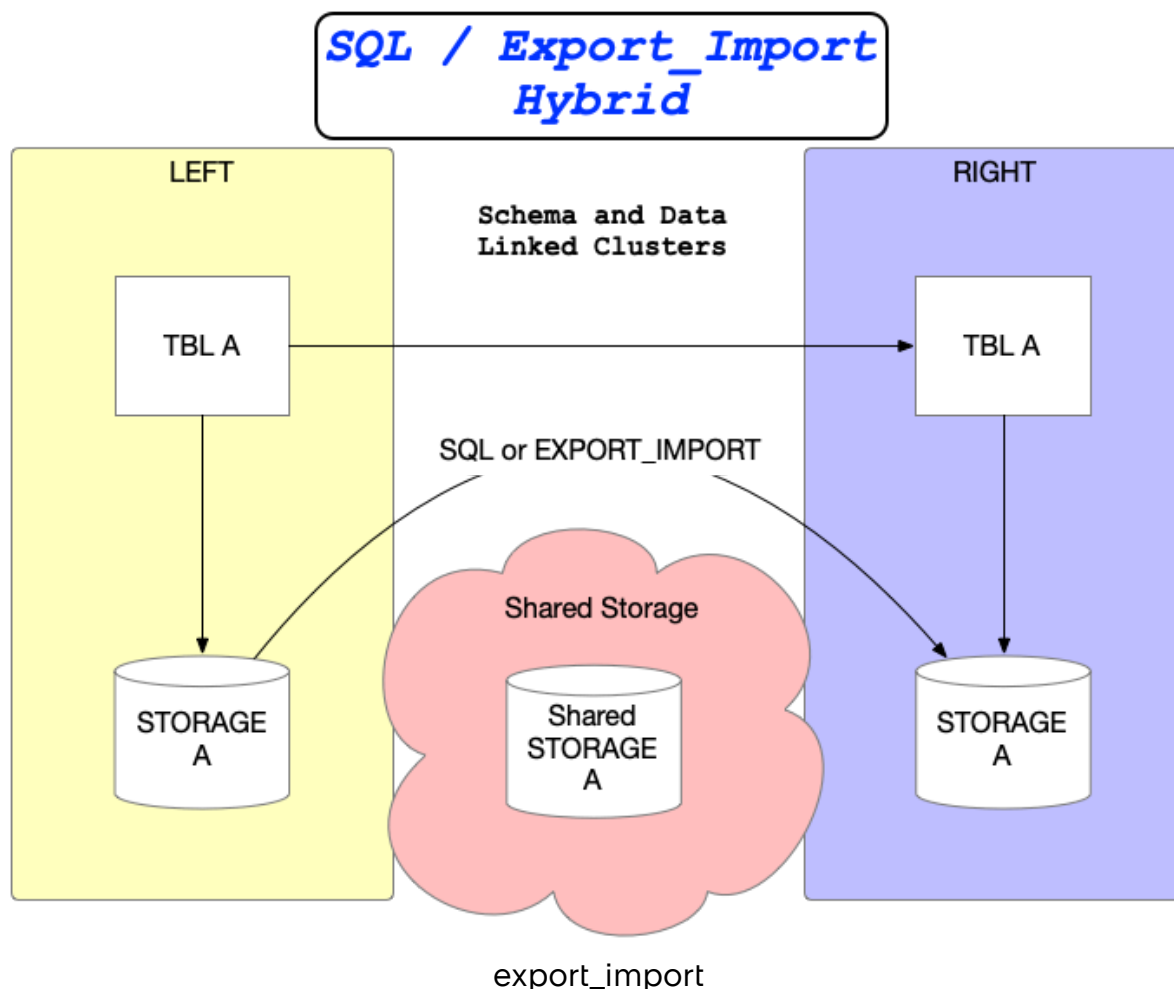
If the `-is <intermediate-storage-path>` is used with this option, we will migrate data to this location and use it as a transfer point between the two clusters. Each cluster will

require access (some configuration adjustment may be required) to the location. In this scenario, the clusters do NOT need to be linked.
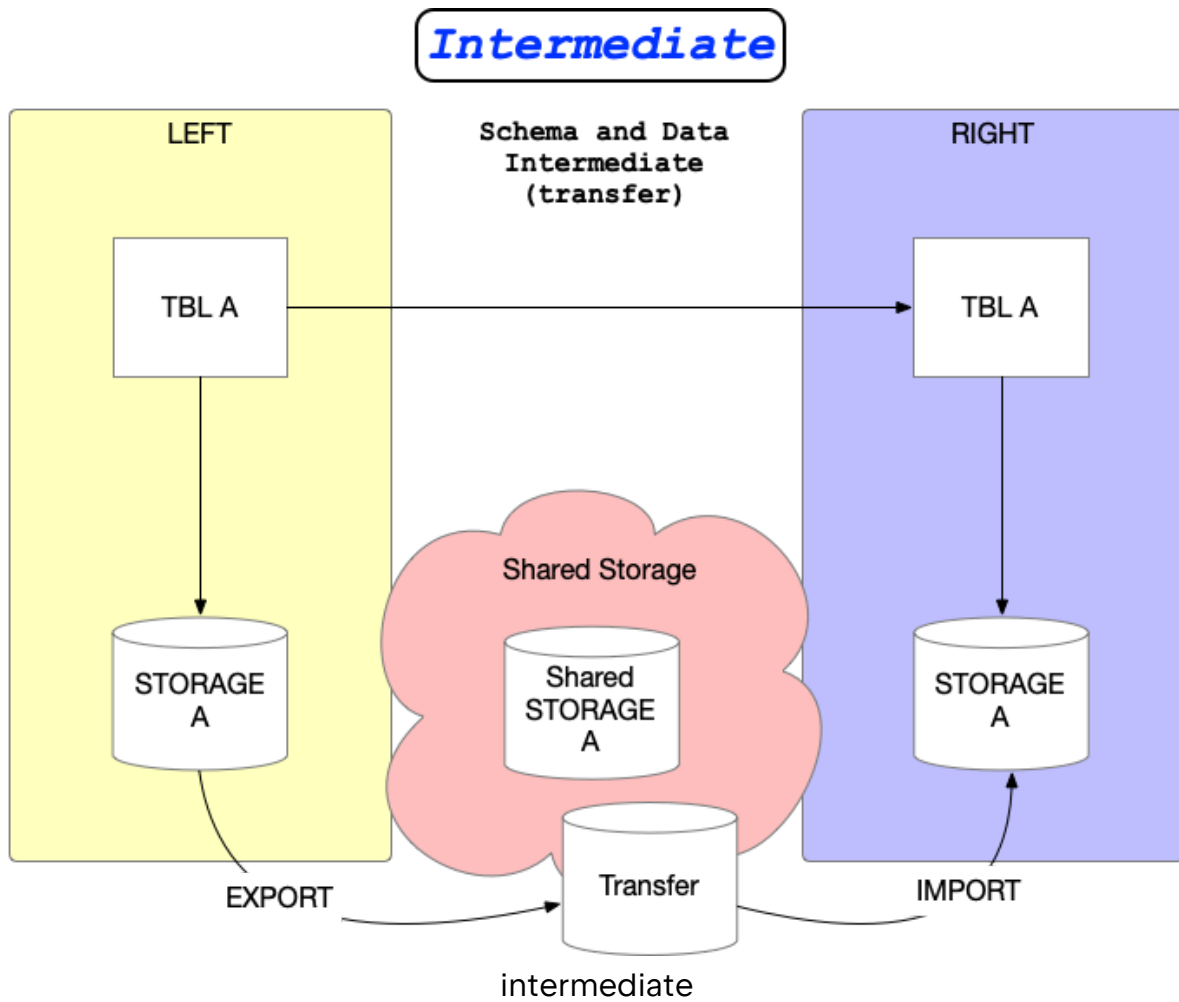
Options

-ep|--export-partition-count <limit>

Sets the limit for the number of partitions to use EXPORT_IMPORT. The default is 100. When a table has a partition count that exceeds this value, the SQL (SQL) strategy will be used.

To persist a higher value without specifying the `-ep` option, add the following to the config (Default Configuration Template).

```
hybrid:
  exportImportPartitionLimit: <limit>
```

HYBRID

This data strategy will use a combination of EXPORT_IMPORT and SQL to move data between clusters.

This strategy will select either SQL (SQL) or EXPORT_IMPORT (EXPORT_IMPORT) based on the table type, and table partition counts.

The initial check will attempt to use EXPORT_IMPORT (EXPORT_IMPORT). If the table is ACID, or the partition count exceeds the limit (-ep), the SQL (SQL) strategy will be used.
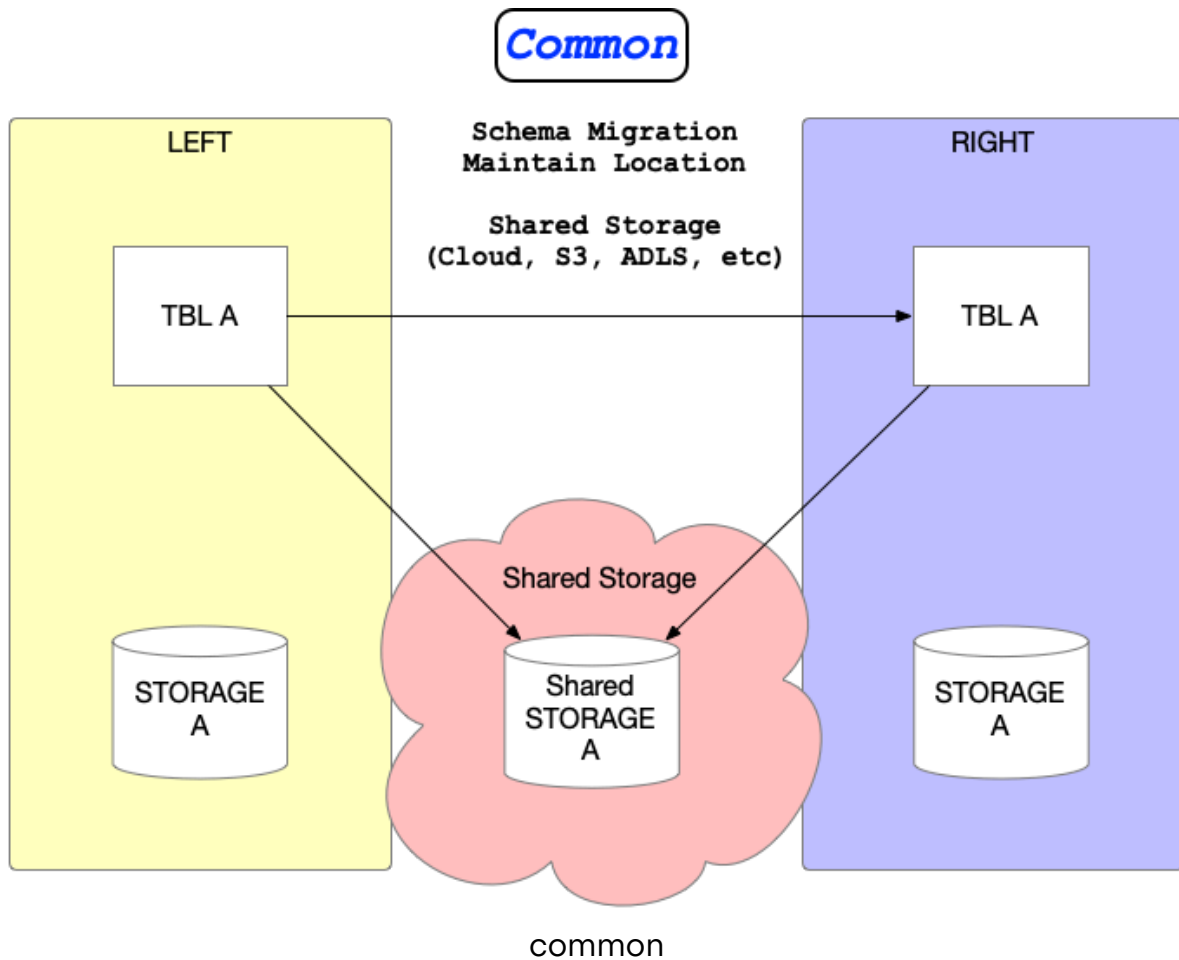
Interesting Options

When the cluster don't have direct line of sight to each other and can NOT be linked (Linking Cluster Storage Layers), you can use options like -cs or -is to bridge the gap.

COMMON

The data storage is shared between the two clusters, and no data migration is required.

Schemas are transferred using the same location.

common

STORAGE_MIGRATION

The Storage Migration feature in `hms-mirror` is used to migrate 'data' from one storage location to another in the same metastore.

For example: You have all of you databases data on 'HDFS' and you want to more it to 'Ozone' while keeping all the same details about the tables, columns, etc. in the metastore so there's minimal impact to the applications using the database/tables.

One of the tricky parts of this type of migration is how you'll want to handle going from a 'single' namespace environment to a 'multi' namespace environment. This is the case when migrating from HDFS to Ozone. Ozone features, and multi-tenant capabilities are different than HDFS. On HDFS, it's ok to have all you databases in a single parent folder (Warehouse Directory). But with Ozone, you do NOT want to do that. This isn't necessarily an 'Ozone' only adjustment. This applies to cloud storage (S3, ADLS, GFS) too.

In this case, you'll want to organize your databases into separate 'namespaces' on the storage system. We're using the term 'namespace' loosely here. It could be a 'bucket' in S3, a 'container' in ADLS, or a 'volume/bucket' in Ozone. Regardless, the point is that we need to be able to handle each database as a separate entity in the storage migration process and allow for each database to be migrated to a different 'namespace' in the target storage system.

Let's take a look at the following example in table format.

Group	Database Name	HDFS External Location	HDFS Managed Location	Ozone External Location	Ozone Managed Location
finance	db1	/warehouse/tablespace/hive/external/db1.db	/warehouse/tablespace/hive/managed/db1.db	/finance/external/db1.db	/finance/managed/db1.db
finance	db2	/warehouse/tablespace/hive/external/db2.db	/warehouse/tablespace/hive/managed/db2.db	/finance/external/db2.db	/finance/managed/db2.db
hr	db3	/warehouse/tablespace/hive/external/db3.db	/warehouse/tablespace/hive/managed/db3.db	/hr/external/db3.db	/hr/managed/db3.db
hr	db4	/warehouse/tablespace/hive/external/db4.db	/warehouse/tablespace/hive/managed/db4.db	/hr/external/db4.db	/hr/managed/db4.db

In the above example, we have 2 owners (finance, hr) and 4 databases (db1, db2, db3, db4). In the legacy environment (HDFS), all databases are in the same parent folder. In the target environment (Ozone), we want to take advantage of the multi-tenant capabilities and separate the databases into separate 'namespaces'. This is done by creating different 'volume/bucket' areas for each 'Owner' and then placing the databases in those areas. To further illustrate separation, look how we established the 'external' and 'managed' locations for each database.

hms-mirror helps you build Warehouse Plans ([Database Warehouse Plans](#)) to handle these types of migrations. Each database to migrate will have a 'Warehouse Plan' that will define how to handle the migration.

Important Properties

The following screen shot shows the properties that are important for the Storage Migration feature.

sm_dm_screen.png

Target Namespace

Identifies the 'namespace' where all the translations will be made to.

Location Translation Strategy

RELATIVE or ALIGNED. In this case, we DO want to ALIGN the locations with the new 'namespace' and locations defined in the Warehouse Plan for the database.

Movement Strategy

DISTCP or SQL. This will direct `hms-mirror` to build a `distcp` plan/scripts or construct `SQL` to move the data.

Skip Database Location Adjustments

Defaults to 'false'. If you want to skip the location adjustments, set this to 'true'. This is useful if you're trying to archive tables in a database to another storage system, but want to leave the database location as is for new tables in the database.

Demo

Here's how the Warehouse Plans would look for the above example:

warehouseplans_01.png

Here's one of the report details for the 'finance.db1' database. See how the 'external' and 'managed' locations are have been altered to reflect the new 'namespace' structure in Ozone we requested in the Warehouse Plan.

Database Details

Issues to review for the LEFT cluster

- This process, when 'executed' will leave the original tables intact in their renamed version. They are NOT automatically cleaned up. Run the produced 'db1_LEFT_CleanUp_execute.sql' file to permanently remove them. Managed and External/Purge table data will be removed when dropping these tables. Ex

SQL for the LEFT cluster

Description	Action
Alter Database Location	ALTER DATABASE db1 SET LOCATION "ofs://OHOME90/finance/external/db1.db"
Alter Database Managed Location	ALTER DATABASE db1 SET MANAGEDLOCATION "ofs://OHOME90/finance/managed/db1.db"

Table Work Status

Table Status Summary: SUCCESS(4)

Phase State 'SUCCESS'

Table	Details			Issues/Steps				SQL		

tbl1

Details			Mark	Description	Action
Data Strategy	STORAGE_MIGRATION		.00	init	
			.69	LEFT	Fetched Schema
Type	External		.06	TRANSFER	STORAGE_MIGRATION
			.22	LEFT	Sql Run SKIPPED (DRY-RUN) for: Selecting DB
Duration (secs)	0		.00	LEFT	Sql Run SKIPPED (DRY-RUN) for: Alter Table Location

LEFT

Description	Action
Selecting DB	USE db1
Alter Table Location	ALTER TABLE tbl1 SET LOCATION "ofs://OHOME90/finance/external/db1.db/tbl1"

tbl2

Details			Mark	Description	Action
Data Strategy	STORAGE_MIGRATION		.00	init	
			.62	LEFT	Fetched Schema
Type	External		.12	TRANSFER	STORAGE_MIGRATION
			.22	LEFT	Sql Run SKIPPED (DRY-RUN) for: Selecting DB
Duration (secs)	0		.00	LEFT	Sql Run SKIPPED (DRY-RUN) for: Alter Table Location

LEFT

Description	Action
Selecting DB	USE db1
Alter Table Location	ALTER TABLE tbl2 SET LOCATION "ofs://OHOME90/finance/external/db1.db/tbl2"

tbl3

Details			Mark	Description	Action
Data Strategy	STORAGE_MIGRATION		.00	init	
			.73	LEFT	Fetched Schema
Type	Managed ACID (Transactio		.01	TRANSACTIONAL	true
			.01	TRANSFER	STORAGE_MIGRATION
Duration (secs)	0		.22	LEFT	Sql Run SKIPPED (DRY-RUN) for: Selecting DB
			.00	LEFT	Sql Run SKIPPED (DRY-RUN) for: Alter Table Location

LEFT

Description	Action
Selecting DB	USE db1
Alter Table Location	ALTER TABLE tbl3 SET LOCATION "ofs://OHOME90/finance/managed/db1.db/tbl3"

tbl4

Details			Mark	Description	Action
Data Strategy	STORAGE_MIGRATION		.00	init	
			.65	LEFT	Fetched Schema
Type	Managed ACID (Transactio		.01	TRANSACTIONAL	true
			.08	TRANSFER	STORAGE_MIGRATION
Duration (secs)	0		.22	LEFT	Sql Run SKIPPED (DRY-RUN) for: Selecting DB
			.00	LEFT	Sql Run SKIPPED (DRY-RUN) for: Alter Table Location

LEFT

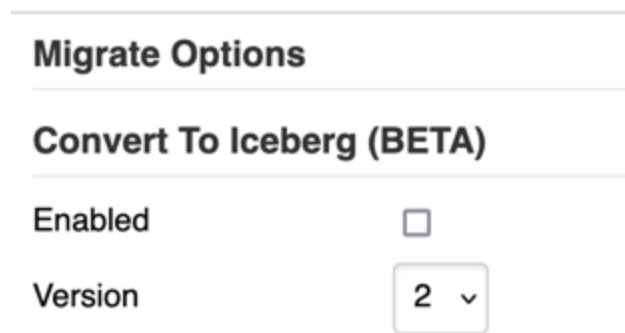| Description	Action
Selecting DB	USE db1
Alter Table Location	ALTER TABLE tbl4 SET LOCATION "ofs://OHOME90/finance/managed/db1.db/tbl4"

warehouseplan_01_rpt.png

ICEBERG_MIGRATION

This process will look at Hive tables, evaluate if the table is a candidate for migration to Iceberg, and then migrate the table to Iceberg.

Currently the conversion of Hive Table Format to Iceberg is supported through the data strategy. Efforts are in place to extend this conversion to SCHEMA_ONLY and SQL data strategies.

iceberg_conversion.png

Simply 'enable' the conversion and all tables covered by the data strategy will be converted to Iceberg. You can also determine which iceberg table format to apply to the table.

Requirements

- Requires Hive with Iceberg Support.

  - CDP Private Cloud Base 7.1.9 Hive does NOT support Iceberg.

  - CDP Private Cloud Base 7.3.1 Hive does support Iceberg.

  - CDP Private Cloud CDW 1.5.1 Hive does support Iceberg. You need CDW Data Services 1.5.1 or higher.

  - CDP Public Cloud Hive does support Iceberg as of August 2023 (Datahub and CDW)

Caution

Make sure you know the component limitations with Iceberg Tables here (https://docs.cloudera.com/cdp-public-cloud/cloud/cdp-iceberg/topics/iceberg-in-cdp.html) so you aren't caught by surprise.

Index of Settings

Copy Avro Schema Urls

CLI

`-asm,--avro-schema-migration` Migrate AVRO Schema Files referenced in TBLPROPERTIES by 'avro.schema.url'. Without migration it is expected that the file will exist on the other cluster and match the 'url' defined in the schema DDL. If it's not present, schema creation will FAIL. Specifying this option REQUIRES the LEFT and RIGHT cluster to be LINKED. See docs: https://github.com/cloudera-labs/hms-mirror#linking-clusters-storage-layers

Web UI

Filters ▾   Attributes ▾   Cluster(s) ▾   Mappings

Miscellaneous

Skip Link Check	☐
Copy Avro Schema URLs	☐
Database Only	☐
Read Only	☐
No Purge	☐
Replace	☐
Skip Features	☐
Skip Legacy Translation	☐
Sync	☐
Transfer 'Database' Ownership	☐
Transfer 'Table' Ownership	☐

attrib_misc.png

Config File

copyAvroSchemaUrls is a boolean value that determines if the Avro schema URLs should be copied from the source to the target. This is useful if you're using Avro schemas in your source and target environments and want to maintain the same schema URLs in both environments. The default value is false.

```
copyAvroSchemaUrls: true|false
```

Data Strategy

Data Strategy controls how schemas and data are migrated. The following values are supported:

- SCHEMA_ONLY

- SQL

- EXPORT_IMPORT

- HYBRID

- DUMP

- STORAGE_MIGRATION

- COMMON

- LINKED

CLI

`-d|--data-strategy <strategy>` 'strategy' is one of the strategies listed above.

Web UI

The Web UI approaches this setting a little differently. When you 'initialized' a migration session, you'll be asked to select a 'Data Strategy' from a drop-down list. Saved sessions will have the 'data-strategy' embedded in them and is NOT editable. You 'can' create a new session based on an existing session and maintain most of the properties of the original session.

init_data-strategy.png

Config File

```
dataStrategy:
"SCHEMA_ONLY|SQL|EXPORT_IMPORT|HYBRID|DUMP|STORAGE_MIGRATION|C
OMMON|LINKED"
```

Database Only

When set we'll only migrate the database objects and not the tables.

CLI

`-dbo|--database-only` will only migrate the database objects and not the tables.

Web UI

attrib_misc.png

Config File

```
databaseOnly: true|false
```

Dump Test Data

Dumping test data uses the running configuration to create a processable artifact that can be used for testing offline.

CLI

dtd|--dump-test-data will dump an artifact that can be used for testing offline.

Web UI

Not an option in the Web UI.

Config File

```
dumpTestData: true|false
```

Load Test Data File

Run the process with a test data file that was created with the --dump-test-data option.

CLI

-ltd|--load-test-data will load the test data file that was created with the --dump-test-data option.

Web UI

Not an option in the Web UI.

Config File

```
loadTestDataFile: "<path_to_test_data_file>"
```

Skip Link Check

When there are issues connecting to the source or target cluster storage systems, you can skip the link check. Doing so may mean that certain features will not work as

expected. EG: Limits used to determine processing that need to review the storage layer or the ability to copy Avro schema files or optimizations made based on file

CLI

`-slc|--skip-link-check` will skip the link check.

Web UI

attrib_misc.png

Config File

Default is `false`.

```
skipLinkCheck: true|false
```

Database Prefix

`dbPrefix` is a value to pre-pend to the database name when creating the database in the target cluster. This is way to avoid conflicts with existing databases in the target cluster. The default value is an empty string.

```
dbPrefix: "<prefix>"
```

Database Rename

`dbRename` is a string value that identifies the new name of the database in the target cluster. This is useful for testing a single database migration to an alternate database in the target cluster. This is only valid for a single database migration.

```
dbRename: "<new_db_name>"
```

Execute

`execute` is a boolean value that determines if the migration should be executed. The default value is `false`, which is the dry-run mode. In the 'dry-run' mode, all the reports are generated and none of the actual migration is done.

> ⚠ You should ALWAYS run a 'dry-run' before executing a migration. This will give you a good idea of what will be done and provide you with the reports to review.

```
execute: true|false
```

Output Directory

Output directory is a string value that identifies the directory where the reports will be written. The default is $HOME/.hms-mirror/reports. When this value is defined, the reports will be written to the specified output directory with the timestamp as the 'name' of the report.

CLI

-o|--output-dir <path_to_output_directory> will set the output directory.

Web UI

Not available in the Web UI.

Config File

```
outputDirectory: "<path_to_output_directory>"
```

Encrypted Passwords

encryptedPasswords is a boolean value that determines if the passwords in the configuration file are encrypted.

```
encryptedPasswords: true|false
```

Read-Only

Read-Only is a boolean value that determines if the migration should be read-only. The default value is false. When this value is set to true, table schema's created will not have a 'purge' flag set to ensure they can't drop data. This is useful for testing migrations and for DR scenarios where you want to limit the exposure of potential changes on the target cluster.

> ⚠ This does NOT prevent data from being manipulated on the target cluster after the migration.

CLI

-ro|--read-only will set the migration to read-only.

Web UI

Config File

```
readOnly: true|false
```

Save Working Tables

saveWorkingTables is a boolean value that determines if the working tables should be saved. The default value is false.

CLI

-swt|--save-working-tables

Web UI

save-working-tables.png

Config File

```
saveWorkingTables: true|false
```

Skip Features

Skip Features is a boolean value that is false by default so feature check will be made. Features are a framework of checks that examine a table definition and make corrections

to it to ensure it's compatible with the target cluster. We've found several circumstances where definitions extracted from the source cluster can NOT be replayed on the target cluster for some reason. These features attempt to correct those issues during the migration.

CLI

`-sf|--skip-features` will skip the feature checks.

Web UI

Config File

```
skipFeatures: true|false
```

Transfer Ownership

You can choose to transfer ownership of objects from the source to target clusters. Ownership is defined as the `owner` of the database or table in the metastore. This is useful if you are using Ranger policies that rely on the owner of the object.

> ⚠ The user running the migration commands will need permissions to make these settings to the table. This would include the ability to `ALTER` the target table. If you're using Ranger, please run some test with the user in beeline to ensure there isn't an issue when running the process.

CLI

`-to|--transfer-ownership` will transfer ownership of databases and tables from the source to the target cluster.
`-todb|--transfer-ownership-database` will transfer ownership of databases from the source to the target cluster.

`-totbl|--transfer-ownership-table` will transfer ownership of tables from the source to the target cluster.

Web UI

Filters ▾   Attributes ▾   Cluster(s) ▾   Mappings

Miscellaneous

Skip Link Check	☐
Copy Avro Schema URLs	☐
Database Only	☐
Read Only	☐
No Purge	☐
Replace	☐
Skip Features	☐
Skip Legacy Translation	☐
Sync	☐
Transfer 'Database' Ownership	☐
Transfer 'Table' Ownership	☐

attrib_misc.png

Config File

```
ownershipTransfer:
  database: true|false
```

```
table: true|false
```

Migrate Options

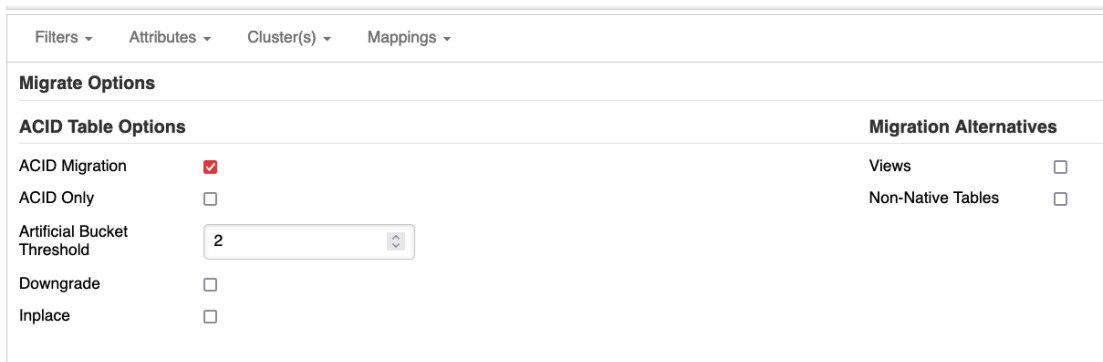There are several options to control the migration of tables and views.

ACID Tables

ACID tables require special handling. By default, ACID tables are NOT migrated unless the following options are set.

CLI

-ma|--migrate-acid will set the migration to include ACID tables. -mao|--migrate-acid-only will set the migration to include ONLY ACID tables.

Web UI

migrate-options.png

Config File

```
migrateACID:
  "on": true
  only: false
```

Non-Native Tables

By default, non-native tables are NOT migrated. Usually because there are requirements that the underlying technology be in place before the migration can happen.
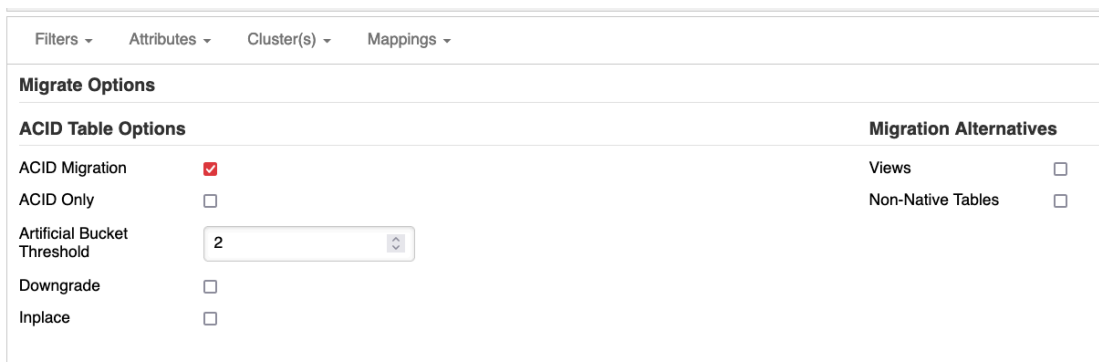
If those prerequisites are met, you can set the `migrateNonNative` option to `true` in the configuration file.

This is typical of tables in Hive that rely on other technologies to store the data. EG: HBase, Kafka, JDBC Federation, etc.

CLI

-mnn|--migrate-non-native will set the migration to include non-native tables. -mnno|--migrate-non-native-only will set the migration to include ONLY non-native tables.

Web UI

migrate-options.png

Config File
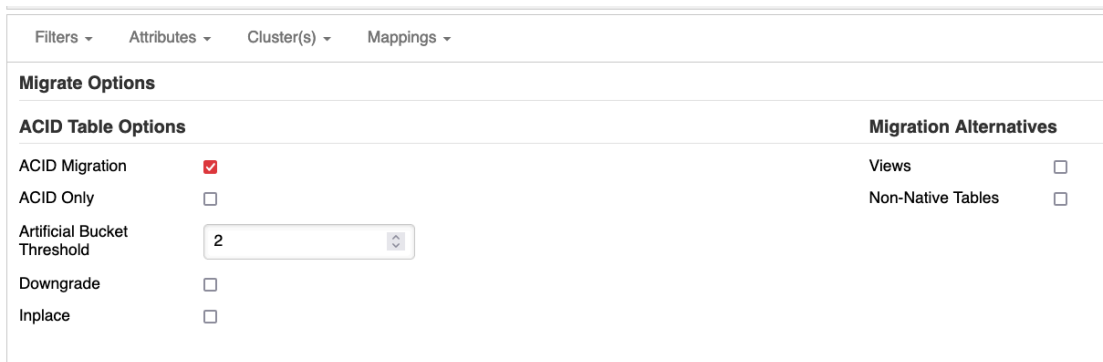
```
migrateNonNative: true
```

Views

Views are NOT migrated by default. Views require the underlying tables to be in place before they can be created. If you have views, migrate the tables first and then rerun the migration for the views.

> ⚠️ Do not rename databases or tables that view depend on. The migration will fail.

CLI

-v|--views-only will set the migration to include ONLY views.

Web UI

migrate-options.png

Config File

```
migrateVIEW:
  "on": false
```

Filter

Databases
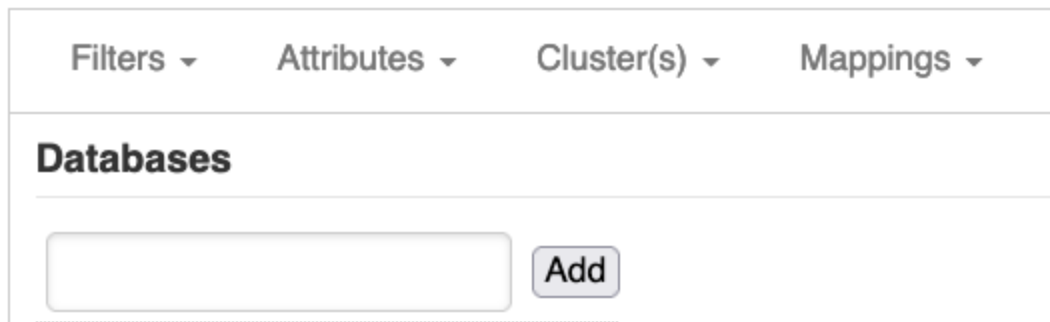
There are several ways to specify which databases to process. The most simple method is to provide a list of database(s) to process.

The first is to set the database(s) to process explicitly.

CLI

`-db <databases>|--database <databases>` 'databases' is a comma separated list of target databases to process.

Web UI

filter_databases.png

Config File
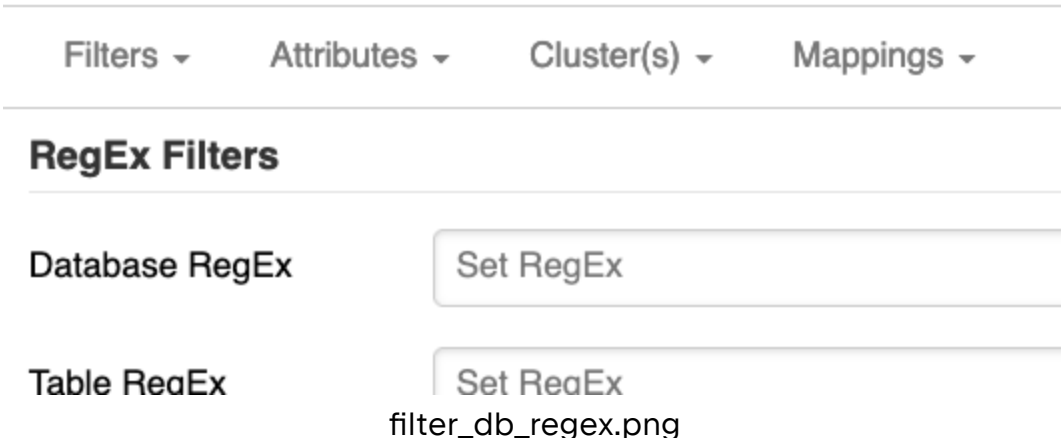
```
databases:
  - db_name
  - db_name2
```

The second method is to use a Regular Expression to match the database(s) to process.

CLI

-dbRegEx <regex>|--database-regex <regex> 'regex' is a valid "Regular Expression" used to filter which databases to process from the target cluster.

Web UI

filter_db_regex.png

Config File

Using the -dbRegEx on the CLI will populate the databases section of the config file with the regular expression matches.

```
databases:
  - db_name
  - db_name2
```

The third method is the most flexible and complete option. Use Warehouse Plans (Warehouse Plans) to define the which databases to process. This method also allows you to define individual database location values.

CLI

-wps,--warehouse-plans <db=ext-dir:mngd-dir[,db=ext-dir:mngd-dir]...>

Web UI
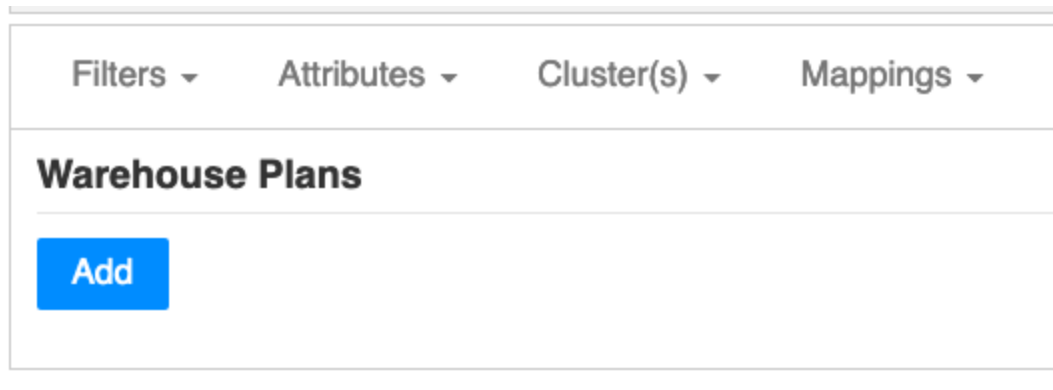
filter_wp.png

Config File

```yaml
translator:
  warehousePlans:
    db1:
      source: "PLAN"
      externalDirectory: "/finance/external"
      managedDirectory: "/finance/managed"
    db2:
        source: "PLAN"
        externalDirectory: "/marketing/external"
        managedDirectory: "/marketing/managed"
```

Database Skip Properties (via RegEx)
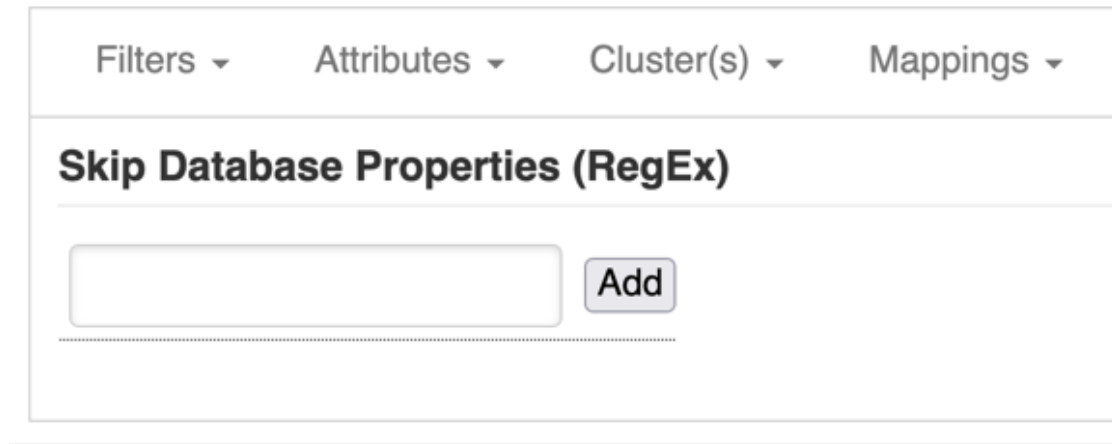
A user managed list of properties that will be filter OUT from the migration. For example: If you don't want to migrate a DBPROPERTY like `repl.incl.test=hello_world`, then add `repl\.incl.*` to this list.

> ℹ This is a RegEx, so ensure you follow that syntax.

CLI

> `-dbsp,--database-skip-properties <properties>` Comma separated list of database properties (regex) to skip during the migration process. This will prevent the property from being set on the target cluster.

Web UI

dbSkipProps.png

Config File

```yaml
filter:
  dbPropertySkipList:
    - "repl\\.inc.*"
```

Tables

When nothing is specified, all tables in the processed databases are included. To limit the tables processed, you can use the following options.
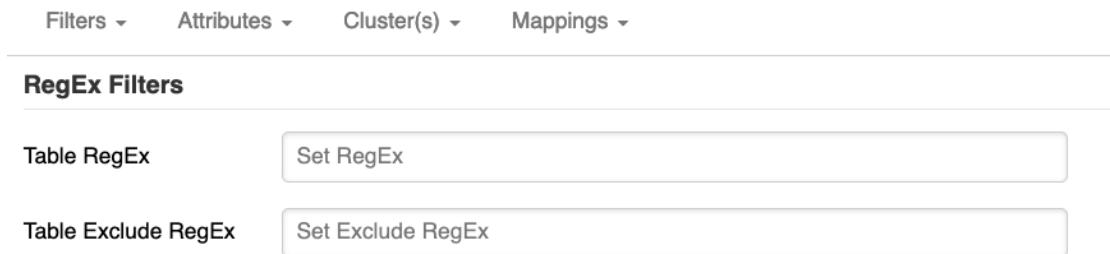
By Regular Expression

CLI

> `-tf <regex>` 'regex' used to match tables in the database to process. process.

> **-tef <regex>** 'regex' used to match tables that would be 'EXCLUDED' in the database to process. process.

Web UI

filter_tbl_regex.png

Config File

```yaml
filter:
  tblRegEx: "test.*"

filter:
  tblExcludeRegEx: "tmp.*"
```

By Limits

CLI

> **-tfp <partition-count>** 'partition-count' would be a limit of the number of partitions in a table that would be processed. Tables with more partitions than the limit would be excluded. A value of **-1** would include all tables.
> **-tfs <size MB>** 'size MB' would be a limit in size for tables that would be processed. Tables greater in size would be excluded. A value of **-1** would include all tables.

Web UI

> A value of **-1** would include all tables.

filter_limits.png

Config File

A value of -1 would include all tables.

```
filter:
  tblSizeLimit: -1
  tblPartitionLimit: -1
```

Cluster

Start typing here...

Transfer

Target Namespace

```
transfer:
  storageMigration:
```

Intermediate Storage

```
transfer:
  storageMigration:
    intermediateStorage: ...
```

Storage Migration

Location Translation Strategy

```
transfer:
  storageMigration:
    translationType: "ALIGNED|RELATIVE"
```

Data Movement Strategy

```
transfer:
  storageMigration:
    dataMovementStrategy: "SQL|DISTCP"
```

Skip Database Location Adjustments

When true, the database location will NOT be adjusted to match the table locations that are migrated. " + "This is useful in the case of an 'archive' strategy where you only want to migrate the table(s) but are not yet " + "ready to migration or stage future tables at the new location.

```
transfer:
  storageMigration:
    skipDatabaseLocationAdjustments: true|false
```

Create Archive

When true (default is false), the tables being migrated will be archived vs. simply changing the location details " + "of the tables metadata. This is relevant only for STORAGE_MIGRATION when using the 'DISTCP' data movement " + "strategy. When the data movement strategy is 'SQL' for STORAGE_MIGRATION, this flag is ignored because the default " + "behavior is to create an archive table anyhow.

```
transfer:
  storageMigration:
    createArchive: true|false
```

Consolidate Source Tables

Used to help define how a `distcp` plan will be built when asked for. The default is `false` which means that will **NOT** be consolidating table locations.

If this is set to `true`, the `distcp` plan will **remove the table location** from the source (which is generally the database location) and use it for all transfers. This looks more simple, but could lead to copying more data than you expect since there's no guarantee that there isn't a lot of other 'extra' data in the source location.

```
transfer:
  storageMigration:
    consolidateTablesForDistcp: true|false
```

Strict Mode

When true and using `distcp`, the migration will fail if the table partition locations aren't standard. The default is `false`. The default mode will allow you to move forward with warnings about the non-standard partition locations. It's highly recommended that you validate the location mappings in the `distcp` plan before running the migration.

```
transfer:
  storageMigration:
    strict: true|false
```

Troubleshooting

Application doesn't seem to be making progress

All the counters for table processing aren't moving (review the hms-mirror.log) or (1.6.1.0+) the on screen logging of what tables are being added and metadata collected for has stopped.

Solution

The application creates a pool of connection to the HiveServer2 instances on the LEFT and RIGHT to be used for processing. When the HiveServer2 doesn't support or doesn't have available the number of connections being requested from hms-mirror, the application will 'wait' forever on getting the connections requested.

Stop the application and lower the concurrency value to a value that can be supported.

```
transfer:
  concurrency: 4
```

Or, you could modify the HiveServer2 instance to handle the number of connections being requested.

Application won't start NoClassDefFoundError

Error

```
Exception in thread "main" java.lang.NoClassDefFoundError:
java/sql/Driver at java.base/java.lang.ClassLoader.defineClass1
```

hms-mirror uses a classloader to separate the various jdbc classes (and versions) used to manage migrations between two different clusters. The application also has a requirement to run on older platforms, so the most common denominator is Java 8. Our method of loading and separating these libraries doesn't work in Java 9+.

Solution

Please use Java 8 to run hms-mirror.

CDP Hive Standalone Driver for CDP 7.1.8 CHF x (Cummulative Hot Fix) won't connect

If you are attempting to connect to a CDP 7.1.8 clusters Hive Server 2 with the CDP Hive Standalone Driver identified in the clusters `jarFile` property, you may not be able to connect. A security item addressed in these drivers changed the required classes.

If you see:

```
java.lang.RuntimeException: java.lang.RuntimeException:
java.lang.NoClassDefFoundError: org/apache/log4j/Level
```

You will need to include additional jars in the `jarFile` property. The following jars are required:

```
log4j-1.2-api-2.18.0.ja
log4j-api-2.18.0.jar
log4j-core-2.18.0.jar
```

The feature enhancement that allows multiple jars to be specified in the `jarFile` property is available in `hms-mirror` 1.6.0.0 and later. See Issue #47 (https://github.com/cloudera-labs/hms-mirror/issues/67)

Solution

Using `hms-mirror` v1.6.0.0 or later, specify the additional jars in the `jarFile` property. For example: jarFile: "<absolute_path_to>/hive-jdbc-3.1.3000.7.1.8.28-1-standalone.jar:<absolute_path_to>/log4j-1.2-api-2.18.0.jar:<absolute_path_to>/log4j-api-2.18.0.jar:<absolute_path_to>/log4j-core-2.18.0.jar"

These jar files can be found on the CDP edge node in `/opt/cloudera/parcels/CDH/jars/`.

Ensure that the standalone driver is list 'FIRST' in the `jarFile` property.

Failed AVRO Table Creation

```
Error while compiling statement: FAILED: Execution Error, return
code 40000 from org.apache.hadoop.hive.ql.ddl.DDLTask.
java.lang.RuntimeException:
```

```
MetaException(message:org.apache.hadoop.hive.serde2.SerDeException
Encountered AvroSerdeException determining schema. Returning
signal schema to indicate problem: Unable to read schema from
given path: /user/dstreev/test.avsc)
```

Solution

Validate that the 'schema' file has been copied over to the new cluster. If that has been done, check the permissions. In a non-impersonation environment (doas=false), the `hive` user must have access to the file.

Table processing completed with ERROR.

We make various checks as we perform the migrations, and when those checks don't pass, the result is an error.

Solution

In tips ([Tips](#)) we suggest running with `dry-run` first (default). This will catch the potential issues first, without taking a whole lot of time. Use this to remediate issues before executing.

If the scenario that causes the `ERROR` is known, a remediation summary will be in the output report under **Issues** for that table. Follow those instructions, then rerun the process with `--retry`. NOTE: `--retry` is currently tech preview and not thoroughly tested.

Connecting to HS2 via Kerberos

Connecting to an HDP cluster running 2.6.5 with Binary protocol and Kerberos triggers an incompatibility issue: `Unrecognized Hadoop major version number: 3.1.1.7.1....`

Solution

The application is built with CDP libraries (excluding the Hive Standalone Driver). When `Kerberos is the` auth` protocol to connect to **Hive 1**, it will get the application libs which will NOT be compatible with the older cluster.

Kerberos connections are only supported to the CDP cluster.

When connecting via `Kerberos, you will need to include the` `--hadoop-classpath when launching` hms-mirror`.
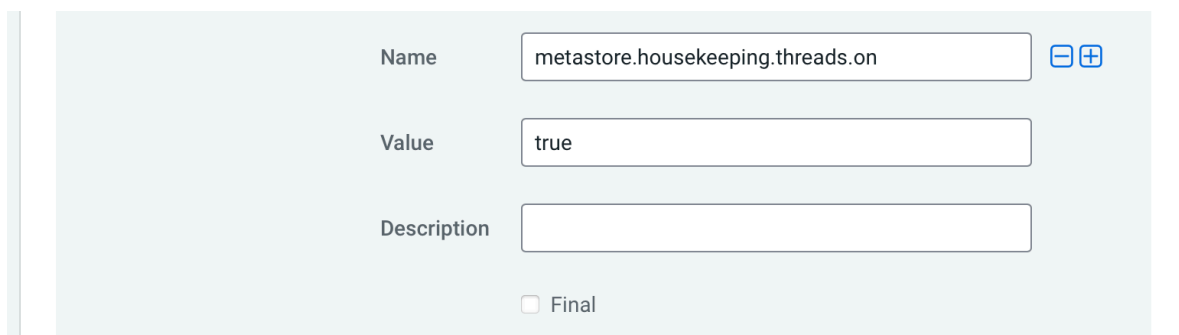
Auto Partition Discovery not working

I've set the `partitionDiscovery:auto` to `true,` but the partitions aren't getting discovered.

Solution

In CDP Base/PVC versions < 7.1.6 have not set the housekeeping thread that runs to activate this feature.

In the Hive metastore configuration in Cloudera Manager, set `metastore.housekeeping.threads.on=true` in the *Hive Service Advanced Configuration Snippet (Safety Valve) for hive-site.xml*

pic

Hive SQL Exception / HDFS Permissions Issues

```
Caused by: java.lang.RuntimeException:
org.apache.hadoop.hive.ql.security.authorization.plugin.HiveAccess
ControlException:Permission denied: user [dstreev] does not have
[ALL] privilege on
[hdfs://HDP50/apps/hive/warehouse/tpcds_bin_partitioned_orc_10.db/
web_site]
```

This error is a permission error to HDFS. For HYBRID, EXPORT_IMPORT, SQL, and SCHEMA_ONLY (with `-ams` enabled), this could be an issue with cross-cluster HDFS access.

Review the output report for details of where this error occurred (LEFT or RIGHT cluster).

When dealing with CREATE DDL statements submitted through HS2 with a `LOCATION` element in them, the submitting *user* **AND** the HS2 *service account* must have

permissions to the directory. Remember, with cross-cluster access, the user identity will originate on the RIGHT cluster and will be **EVALUATED** on the LEFT clusters storage layer.

For migrations, the `hms-mirror` running user (JDBC) and keytab user (HDFS) should be privileged users.

Example and Ambari Hints

After checking permissions of 'dstreev': Found that the 'dstreev' user was NOT the owner of the files in these directories on the LEFT cluster. The user running the process needs to be in 'dfs.permissions.superusergroup' for the lower clusters 'hdfs' service. Ambari 2.6 has issues setting this property: https://jira.cloudera.com/browse/EAR-7805

Follow the workaround above or add the user to the 'hdfs' group. Or use Ranger to allow all access. On my cluster, with no Ranger, I had to use '/var/lib/ambari-server/resources/scripts/configs.py' to set it manually for Ambari.

`sudo ./configs.py --host=k01.streever.local --port=8080 -u admin -p admin -n hdp50 -c hdfs-site -a set -k dfs.permissions.superusergroup -v hdfs_admin`

YARN Submission stuck in ACCEPTED phase

The process uses a connection pool to hive. If the concurrency value for the cluster is too high, you may have reached the maximum ratio of AM (Application Masters) for the YARN queue.

Review the ACCEPTED jobs and review the jobs *Diagnostics* status for details on *why* the jobs is stuck.

Solution

Either of:

1. Reduce the concurrency in the configuration file for `hms-mirror`

2. Increase the AM ratio or Queue size to allow the jobs to be submitted. This can be done while the process is running.

Spark DFS Access

If you have problems accessing HDFS from `spark-shell` or `spark-submit` try adding the following configuration to spark:

```
--conf
spark.yarn.access.hadoopFileSystems=hdfs://<NEW_NAMESPACE>,hdfs://
<OLD_NAMESPACE>
```

Permission Issues

`HiveAccessControlException Permission denied user: [xxxx] does not have [ALL] privileges on ['location'] [state=42000,code=40000]`

and possibly

In HS2 Logs: `Unauthorized connection for super-user`

Solution

Caused by the following:

- The 'user' doesn't have access to the location as indicated in the message. Verify through 'hdfs' that this is true or not. If the user does NOT have access, grant them access and try again.

- The 'hive' service account running HS2 does NOT have access to the location. The message will mask this and present it as a 'user' issue, when it is in fact an issue with the 'hive' service account. Grant the account the appropriate access.

- The 'hive' service does NOT have proxy permissions to the storage layer.

  - Check the `hadoop.proxyuser.hive.hosts|groups` setting in `core-site.xml`. If you are running into this `super-user` error on the RIGHT cluster, while trying to access a storage location on the *LEFT* cluster, ensure the proxy settings include the rights values in the RIGHT clusters `core-site.xml`, since that is where HS2 will pick it up from.

Must use HiveInputFormat to read ACID tables

We've seen this while attempting to migrate ACID tables from older clusters (HDP 2.6). The error occurs when we try to extract the ACID table data to a 'transfer' external table

on the LEFT cluster, which is 'legacy'.

Solution

HDP 2.6.5, the lowest supported cluster version intended for this process, should be using the 'tez' execution engine `set hive.execution.engine=tez`. If the cluster has been upgraded from an older HDP version OR they've simply decided NOT to use the `tez` execution engine', you may get this error.

In `hms-mirror` releases 1.3.0.5 and above, we will explicitly run `set hive.execution.engine=tez` on the LEFT cluster when identified as a 'legacy' cluster. For version 1.3.0.4 (the first version to support ACID transfers), you'll need to set the hive environment for the HS2 instance you're connecting to use `tez` as the execution engine.

ACL issues across cross while using LOWER clusters storage

Are you seeing something like this?

```
org.apache.hadoop.hive.ql.ddl.DDLTask. MetaException(message:Got
exception: org.apache.hadoop.security.AccessControlException
Permission denied: user=hive, access=WRITE,
inode="/apps/hive/warehouse/merge_files.db/merge_files_part_a_smal
l_replacement":dstreev:hadoop:drwxr-xr-x at
```

This is caused when trying to `CREATE` a table on the **RIGHT** cluster that references data on the **LEFT** cluster. When the LEFT cluster is setup differently with regard to impersonation (doas) than the RIGHT, transfer tables are created with POSIX permissions that may not allow the RIGHT cluster/user to access that location.

Solution

Using Ranger on the LEFT cluster, open up the permissions to allow the requesting user access as identified.

License APLv2

```
Apache License
   Version 2.0, January 2004
http://www.apache.org/licenses/
```

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

    "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

    "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

    "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

    "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

    "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

    "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

    "Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

    "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of

authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

   (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and

   (b) You must cause any modified files to carry prominent notices stating that You changed the files; and

   (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and

   (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

   You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

```
To apply the Apache License to your work, attach the following
boilerplate notice, with the fields enclosed by brackets "[]"
replaced with your own identifying information. (Don't include
the brackets!)  The text should be enclosed in the appropriate
comment syntax for the file format. We also recommend that a
file or class name and description of purpose be included on the
same "printed page" as the copyright notice for easier
identification within third-party archives.
```

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at

```
http://www.apache.org/licenses/LICENSE-2.0
```

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Use Cases

On Prem Legacy Hive to Non-Legacy Hive

Environments

HDP 2.6 -> CDP CDH 5.x -> CDP CDH 6.x -> CDP

LEFT clusters are the LEGACY clusters. RIGHT clusters are the NON-LEGACY clusters (hive3).

Assumptions

- LEFT HS2 is NOT Kerberized. Since this is a Legacy cluster (which is kerberized) we need to establish a 'non' kerberized HS2 endpoint. Use KNOX or setup an additional HS2 in the management console that isn't Kerberized. hms-mirror is built (default) with Hadoop 3, of which the libraries are NOT backwardly compatible.

- Standalone JDBC jar files for the LEGACY and NON-LEGACY clusters are available to the running host as specified in the 'configuration'.

- `hms-mirror` is run from an EdgeNode on CDP

  - The edgenode has network access to the Legacy HS2 endpoint

- No ACID tables (HDP)

- No VIEWs

- No Non-Native tables (Hive tables backed by HBase, JDBC, Kafka)

- The HiveServer2's on each cluster have enough concurrency to support the configured connections `transfer->concurrency`. If not specified, the default is 4.

NOTES

`hms-mirror` runs in DRY-RUN mode by default. Add `-e|--execute` to your command to actually run the process on the clusters. Use `--accept` to avoid the verification questions

(but don't deny their meaning).

All actions performed by `hms-mirror` are recorded in the *_execute.sql files. Review them to understand the orchestration and process.

Review the report markdown files (html version also available) for details about the job. Explanations regarding steps, issues, and failure reasons can be found there.

Scenarios

One-time migration of SCHEMA's from LEFT to RIGHT.

This is done with the basic `SCHEMA_ONLY` data strategy (default) and will extract the schema's from the LEFT and replay them on the RIGHT cluster. In this mode, NO DATA is moved.

`hms-mirror -db tpcds_bin_partitioned_orc_10 -o temp`

Examine the reports place in the relative `temp` directory specified with the `-o` option. A report set is generated to each database.

Since no data is transferred in this scenario, the expectation is that the data is managed by another process, like `distcp`. The output of `hms-mirror` will create some template distcp calls with the appropriate location details. You can use this as a starting point to managed this transfer.

DUMP of clusters SCHEMA

We want a simple extraction of a database schema. An optional `-ds LEFT|RIGHT` option can be specified to target which configured cluster to use. The default is `LEFT`. Note: When you specify `RIGHT`, the DUMP output with still show up in the `LEFT` output report.

No translations are done in this scenario. So if the DUMP is taken from a 'legacy' cluster, beware of the implications of 'replaying' it on a non-legacy cluster.

This will use the LEFT cluster configuration for the schema DUMP of `tpcds_bin_partititoned_orc_10`. `hms-mirror -db tpcds_bin_partitioned_orc_10 -o temp -d DUMP`

This will use the RIGHT cluster configuration for the schema DUMP of `tpcds_bin_partititoned_orc_10`. `hms-mirror -db tpcds_bin_partitioned_orc_10 -o temp -d DUMP -ds RIGHT`

Data Migration for Non-ACID tables using SQL

This approach assumes the clusters are linked ([Linking Cluster Storage Layers](#)). The SQL data strategy uses the RIGHT's clusters view into the HDFS filesystem of the LEFT cluster

to facilitate data movement.

```
hms-mirror -d SQL -db tpcds_bin_partitioned_orc_10 -o temp
```

Data Migration for ACID tables using SQL or HYBRID

This approach assumes the clusters are linked (Linking Cluster Storage Layers). The SQL data strategy uses the RIGHT's clusters view into the HDFS filesystem of the LEFT cluster to facilitate data movement.

```
hms-mirror -d SQL -db tpcds_bin_partitioned_orc_10 -o temp -ma
```

Use `-mao` to migrate ONLY ACID tables. `-ma` will migrate ACID and Non-ACID tables.

Schema Migration of specific tables using RegEx

Using a RegEx pattern, filter the tables in the db to migrate.

```
hms-mirror -db tpcds_bin_partitioned_orc_10 -tf call_*. -o temp
```

`-tf|--table-filter` followed by a RegEx to filter tables.

Migrate VIEWS for a specific database

View migration requires the underlying referenced tables exist BEFORE the 'view' can be created. This isn't a requirement of `hms-mirror` rather a Hive requirement. Therefore, you should migrate the tables first as shown above and in a followup process run the following.

```
hms-mirror -db tpcds_bin_partitioned_orc_10 -v
```

Create schema in RIGHT cluster using the LEFT clusters data

This is a helpful scenario for 'testing' workflows on the RIGHT cluster. The tables on the right cluster will NOT be configured with 'PURGE' to avoid the deletion of data on the LEFT cluster. These tables should be considered READ-ONLY. Test this against a sample dataset to THOROUGHLY understand the relationships here. This is NOT intended for 'production' use and should be used only as a validation mechanism for the RIGHT cluster.

The clusters must be linked ([Linking Cluster Storage Layers](#)). Only legacy managed tables, external tables, and views can be linked. ACID tables can NOT be linked.

`hms-mirror -d LINKED -db tpcds_bin_partitioned_orc_10 -o temp`

The tables created on the RIGHT cluster will use the data located on the LEFT cluster. The 'database' will be created to match the database in the LEFT cluster.

WARNING: If the LOCATION element is specified in the database definition AND you use `DROP DATABASE ... CASCADE` from the RIGHT cluster, YOU WILL DROP THE DATA ON THE LEFT CLUSTER even though the tables are NOT purgeable. This is the DEFAULT behavior of hive 'DROP DATABASE'. So BE CAREFUL!!!!

Migrate SCHEMA's and Data using SQL

The clusters must be linked ([Linking Cluster Storage Layers](#)). In this scenario, we'll use the connected clusters and SQL to migrate data from the LEFT cluster to the RIGHT.

There are limits regarding partitioned tables. For SQL migrations, the default is 500. Meaning that tables with more than 500 partitions will NOT attempt this transfer. This can be changed in the 'config' file by adding/changing `hybrid->sqlPartitionLimit`. This was put in place as a general safeguard against attempts at tables with a partition count that may fail. It doesn't mean they'll always fail, it's just a place holder.

`hms-mirror -d SQL -db tpcds_bin_partitioned_orc_10 -o temp`

To transfer ACID tables, add `-ma|-mao`.

This is a one time transfer. Incremental updates aren't supported with this configuration. For incremental schema updates, see:

Migrate SCHEMA's and Data using EXPORT_IMPORT

EXPORT/IMPORT is a basic Hive process used to package table schemas and data into a transferable unit that can be replayed on the new cluster. For `hms-mirror` there is a defined prefix for a transfer directory in the configuration `transfer->exportBaseDirPrefix`. If this isn't defined, the default is `/apps/hive/warehouse/export_`.

There are performance implications to using EXPORT_IMPORT with partitioned tables. The IMPORT process is quite slow at loading partitions. We've defined limits in the config (which can be changed) `hybrid->exportImportPartitionLimit`. The default is 100. If the

number of partitions exceeds this value, we will NOT attempt the transfer and will note this in the output report.

The clusters must be linked ([Linking Cluster Storage Layers](#)). The before mentioned prefix directory on the LEFT cluster is accessed by the IMPORT process that runs on the RIGHT cluster. If the namespace (and permissions) aren't correct, the IMPORT process will fail.

`hms-mirror -d EXPORT_IMPORT -db tpcds_bin_partitioned_orc_10 -o temp`

EXPORT_IMPORT will NOT work for ACIDv1 -> ACIDv2 (Hive 1/2 to 3) conversions. Use `SQL` or `HYBRID` instead.

Migrate SCHEMA's and Data using HYBRID

The `HYBRID` data strategy is a combination of the `SQL` and `EXPORT_IMPORT` data strategies. It uses basic rules to choose the more appropriate method for the table in question.

The clusters must be linked ([Linking Cluster Storage Layers](#)).

The process will first consider using `EXPORT_IMPORT` unless:

- The table is ACIDv1 and you're migrating to ACIDv2 (Legacy to Non-Legacy Clusters)

- The number of partitions exceed the value set by: `hybrid->exportImportPartitionLimit`. The default is 100. When exceeded, the `SQL` method will be used. The `SQL` method will fail is the partition count exceeds the value of `hybrid->sqlPartitionLimit`. The default is 500.

`hms-mirror -d HYBRID -db tpcds_bin_partitioned_orc_10 -o temp`

Disaster Recovery (RIGHT Cluster is DR and READ-ONLY)

The DR scenario will transfer schemas and subsequent runs will update the 'schema' if changes are made. This process does NOT move data. The process will generate a `distcp` work plan for you to get started. You should modify that to use 'snapshot diffs' and managed the data migration through `distcp`.

You should first run the process in `DRY-RUN` mode to get the `distcp` plan. The data must be transferred first! This ensures that the database and table/partition directories are

created BEFORE the schemas are replayed. If the schemas are applied before the distcp with SNAPSHOT diffs, then hive will own the directories and the distcp with snapshots will fail.

WARNING: Do not attempt to DROP DATABASE … CASCADE on the RIGHT cluster, this will modify the filesystem and cause the incremental distcp with snapshots to fail.

`hms-mirror -d SCHEMA_ONLY -db tpcds_bin_partitioned_orc_10 -ro -sync`

This process will review the tables on the LEFT cluster with the RIGHT and either update the schema when it's changed (by dropping and recreating), add missing tables, or drop tables that don't exist anymore.

Tables that are migrated this way will NOT have the PURGE flag set on the RIGHT cluster. This allows us to DROP a table without affecting the data for the -sync process.

Downgrade and Replace ACID tables

In this scenario, you're choosing to downgrade ACID tables that are migrated, as well as the current tables on the source cluster.

`hms-mirror -db tpcds_bin_partitioned_orc_10 -mao -da -r`

This will migrate ACID tables only (-mao vs. -ma), downgrade them to EXTERNAL/PURGE tables, and 'replace' the current ACID table with a MANAGED Non-Transactional table in legacy environments OR a EXTERNAL/PURGE table in Hive3+ environments.

Using the DRY-RUN mode, experiment with options -is and -cs for various implementations of this conversion.

Cloud to Cloud DR (AWS)

We'll cover how to manage a DR scenario where the source cluster is in the cloud and the target cluster is also in the cloud. The main elements to consider are:

- Hive Metadata (Tables, Databases, Views)

- Data on S3

Requirements

- The source and target clusters on AWS are running CDP Cloud with an available HS2 endpoint.

- Provide a mechanism to migrate Hive metadata from the source cluster to the target cluster, to include making adjustments to the metadata to account for differences in the clusters storage locations.

- Establish the RPO and RTO for the DR scenario and ensure the migration process can meet those requirements.

- We'll only target **external** tables for this scenario. Managed tables will require additional considerations, since the data and metadata are intermingled and can't be supported through a simple copy operation.

Assumptions

- The data on S3 is already replicated to the target cluster through some other mechanism (e.g. S3 replication, etc.).

- The S3 replication will meet the RPO and RTO requirements.

- The data replication is in place before DR is invoked and the scripts are run to build the schemas on the target cluster.

- There are no managed tables being migrated. I would recommend setting the database property 'EXTERNAL_TABLES_ONLY'='TRUE' with: ALTER DATABASE

`<db_name> SET TBLPROPERTIES ('EXTERNAL_TABLES_ONLY'='TRUE');` to ensure only external tables can be created.

- Partitions follow standard naming conventions regarding directory names/structures. Tables with non-standard partitioning will require additional considerations. `hms-mirror` doesn't translate partition details and relies on `MSCK REPAIR <table> SYNC PARTITIONS` to discover / rebuild a tables partitions. If the partitions are not in a standard format, the `MSCK REPAIR` will not work and the partitions will need to be manually created.

- We don't support schema evolution. All tables will be created in there current state.

- The "LEFT" clusters `hcfsNamespace` can only address a single namespace at a time. If you have multiple namespaces, you'll need to run `hms-mirror` multiple times, once for each namespace. The "RIGHT" cluster can address multiple namespaces through the `hcfsNamespace` element. This element is used to match and adjust the storage location of the tables on the target cluster.

The Process

The process is fairly straight forward. We'll use `hms-mirror` to migrate the Hive metadata from the source cluster to the target cluster. We'll use the `--common-storage` or set the `hcfsNamespace` element for the RIGHT cluster to ensure the schemas are built with the DR bucket adjustments.

You have a few options regarding the transfer:

- If the target is truly a DR cluster, you can run `hms-mirror` on the source cluster and generate the metadata files locally. Then copy the metadata files to the target cluster and build out the schemas there. This doesn't need to be done until the DR is invoked.

- If you want/need to keep the metadata in-sync between the clusters, you can run `hms-mirror` with the `-ro` and `-sync` flags (and eventually with `-e`) to keep the metadata in-sync between the clusters. Tables created on the source cluster will require the data to be replicated to the target cluster before the table can be created in DR. While we're only migrating *external* tables, they may have set `external.table.purge` to `true` on the source cluster. In this case, these tables will be set to **NON** purge on the target

cluster. This is to prevent the table data (being managed through S3 replication) from being dropped by subsequent sync runs where the tables might have changed.

Running hms-mirror

Configuration

This file should be named `$HOME/.hms-mirror/cfg/default.yaml`

```yaml
clusters:
  LEFT:
    environment: "LEFT"
    legacyHive: false
    hcfsNamespace: "s3a://<my_source_s3_bucket>"
    hiveServer2:
      # Recommend using a KNOX endpoint to remove need for
Kerberos Authentication
      uri: "jdbc:hive2://<my_source_hs2_endpoint>"
      connectionProperties:
        user: "<user>"
        maxWaitMillis: "5000"
        password: "*****"
        maxTotal: "-1"
      jarFile: "<local_location_of_hive-jdbc_driver>"
  RIGHT:
    environment: "RIGHT"
    legacyHive: false
    hcfsNamespace: "s3a://<my_target_s3_bucket>"
    hiveServer2:
      uri: "jdbc:hive2://<my_target_hs2_endpoint>"
      connectionProperties:
        user: "<user>"
        maxWaitMillis: "5000"
        password: "*****"
        maxTotal: "-1"
      jarFile: "<local_location_of_hive-jdbc_driver>"
    partitionDiscovery:
      # Optional, but recommended it the cluster isn't
```

```
overburdened.
    auto: true
    # Required if auto is false and/or you want to ensure the
partitions are in sync after the
    # transfer is made.
    initMSCK: true
```

Command Lines

```
hms-mirror --hadoop-classpath -d SCHEMA_ONLY -db <db_comma_separated_list> -ro
-sync
```

Hybrid Data LakeHouse

On-Prem to Cloud (Direct)

On-Prem to Cloud (In-Direct)

Cloud to On-Prem (In-Direct)