Using the Hive Metastore with Apache Kudu

Date published:
Date modified:

https://docs.cloudera.com/

Legal Notice

Contents

Integrating the Hive Metastore with Apache Kudu

You can integrate Kudu's own catalog with the Hive Metastore (HMS) allowing external HMS-aware tool to discover and use Kudu tables.

The HMS is the de-facto standard catalog and metadata provider in the Hadoop ecosystem. When the HMS integration is enabled, Kudu tables can be discovered and used by external HMS-aware tools, even if they are not otherwise aware of, or integrated with Kudu. There is no need to create Impala external tables to map any Kudu tables created directly using API or from Spark. Additionally, these components can use the HMS to discover necessary information to connect to the Kudu cluster which owns the table, such as the Kudu master addresses.

Databases and Table Names

When Hive Matastore (HMS) integration is enabled in Kudu, both the namespace and table names properties change to match the HMS model.

With the Hive Metastore integration disabled, Kudu presents tables as a single flat namespace, with no hierarchy or concept of a database. Additionally, Kudu's only restriction on table names is that they be a valid UTF-8 encoded string. When the HMS integration is enabled in Kudu, both of these properties change in order to match the HMS model: the table name must indicate the table's membership of a Hive database, and table name identifiers (i.e. the table name and database name) are subject to the Hive table name identifier constraints.

Databases

Hive has the concept of a database, which is a collection of individual tables. Each database forms its own independent namespace of table names.

In order to fit into this model, Kudu tables must be assigned a database when the HMS integration is enabled. No new APIs have been added to create or delete databases, nor are there APIs to assign an existing Kudu table to a database. Instead, a new convention has been introduced that Kudu table names must be in the format <hive-database-name>.<hive-table-name>. Thus, databases are an implicit part of the Kudu table name. By including databases as an implicit part of the Kudu table name, existing applications that use Kudu tables can operate on non-HMS-integrated and HMS-integrated table names with minimal or no changes.

Kudu provides no additional tooling to create or drop Hive databases. Administrators or users should use existing Hive tools such as the Beeline Shell or Impala to do so.

Naming Constraints

When the Hive Metastore integration is enabled, the database and table names of Kudu tables must follow the Hive Metastore naming constraints. Namely, the database and table name must contain only alphanumeric ASCII characters and underscores.

When the hive.support.special.characters.tablename Hive configuration is true, the forward-slash (/) character in table name identifiers (i.e. the table name and database name) is also supported.

Additionally, the Hive Metastore does not enforce case sensitivity for table name identifiers. As such, when enabled, Kudu will follow suit and disallow tables from being created when one already exists whose table name identifier differs only by case. Operations that open, alter, or drop tables will also be case-insensitive for the table name identifiers.

Warning: Given the case insensitivity upon enabling the integration, if multiple Kudu tables exist whose names only differ by case, the Kudu master(s) will fail to start up. Be sure to rename such conflicting tables before enabling the Hive Metastore integration.

Metadata Synchronization

When the Hive Metastore integration is enabled, Kudu will automatically synchronize metadata changes to Kudu tables between Kudu and the HMS. As such, it is important to always ensure that the Kudu and HMS have a consistent view of existing tables, using the administrative tools. Failure to do so may result in issues like Kudu tables not being discoverable or usable by external, HMS-aware components (e.g. Apache Sentry, Apache Impala).

The Hive Metastore automatically creates directories for Kudu tables. These directories are benign and can safely be ignored.

Impala has notions of internal and external Kudu tables. When dropping an internal table from Impala, the table's data is dropped in Kudu; in contrast when dropping an external table, the table's data is not dropped in Kudu. External tables may refer to tables by names that are different from the names of the underlying Kudu tables, while internal tables must use the same names as those stored in Kudu. Additionally, multiple external tables may refer to the same underlying Kudu table. Thus, since external tables may not map one-to-one with Kudu tables, the Hive Metastore integration and tooling will only automatically synchronize metadata for internal tables.

Related Information

Administrative tools for Hive Metastore integration

Administrative tools for Hive Metastore integration

Kudu provides the command line tools kudu hms list, kudu hms precheck, kudu hms check, and kudu hms fix to allow administrators to find and fix metadata inconsistencies between the internal Kudu catalog and the Hive Metastore catalog.

The kudu hms tools should be run from the command line as the Kudu admin user. They require the full list of master addresses to be specified:

```
$ sudo -u kudu kudu hms check master-name-1:7051,master-name-2:7051,master-n
ame-3:7051
```

To see a full list of the options available with the kudu hms tool, use the --help flag.

When fine-grained authorization is enabled, the Kudu admin user, commonly "kudu", needs to have access to all the Kudu tables to be able to run the kudu hms tools. This can be done by configuring the user as a trusted user via the --trusted_user_acl master configuration.

If the Hive Metastore is configured with fine-grained authorization using Apache Sentry, the Kudu admin user needs to have read and write privileges on HMS table entries. Configured this in the Hive Metastore using the sentry.metastore.service.users property.

kudu hms list

The kudu hms list tool scans the Hive Metastore catalog, and lists the HMS entries (including table name and type) for Kudu tables, as indicated by their HMS storage handler.

kudu hms check

The kudu hms check tool scans the Kudu and Hive Metastore catalogs, and validates that the two catalogs agree on what Kudu tables exist. The tool will make suggestions on how to fix any inconsistencies that are found. Typically, the suggestion will be to run the kudu hms fix tool, however some certain inconsistencies require using Impala Shell for fixing.

kudu hms precheck

The kudu hms precheck tool scans the Kudu catalog and validates that if there are multiple Kudu tables whose names only differ by case and logs the conflicted table names.

kudu hms fix

The kudu hms fix tool analyzes the Kudu and HMS catalogs and attempts to fix any automatically-fixable issues, for instance, by creating a table entry in the HMS for each Kudu table that doesn't

already have one. The --dryrun option shows the proposed fix instead of actually executing it. When no automatic fix is available, it will make suggestions on how a manual fix can help.

kudu hms downgrade

The kudu hms downgrade downgrades the metadata to legacy format for Kudu and the Hive Metastores. It is discouraged to use unless necessary, since the legacy format can be deprecated in future releases.

Upgrading existing Kudu tables for Hive Metastore integration

Before enabling the Kudu-HMS integration upgrade existing Kudu tables to ensure that Kudu and Hive Metstore (HMS) start with a consistent view of existing tables

Before you begin

- Establish a maintenance window. During this time the Kudu cluster will still be available, but tables in Kudu and the Hive Metastore may be altered or renamed as a part of the upgrade process.
- Make note of all external tables using the following command and drop them. This reduces the chance of having naming conflicts with Kudu tables which can lead to errors during upgrading process. It also helps in cases where a catalog upgrade breaks external tables, due to the underlying Kudu tables being renamed. The external tables can be recreated after upgrade is complete.

```
$ sudo -u kudu kudu hms list master-name-1:7051,master-name-2:7051,master-
name-3:7051
```

Procedure

1. Run the kudu hms precheck tool to ensure no Kudu tables only differ by case. If the tool does not report any warnings, you can skip the next step.

```
$ sudo -u kudu kudu table rename_table master-name-1:7051,master-name-2:
7051,master-name-3:7051 <conflicting_table_name> <new_table_name>
```

2. Run the kudu hms check tool using the following command. If the tool does not report any catalog inconsistencies, you can skip the following steps and can continue with *Enabling the Hive Metastore integration.*

```
$ sudo -u kudu kudu hms check master-name-1:7051,master-name-2:7051,mast
er-name-3:7051 --hive_metastore_uris=<hive_metastore_uris> [--ignore_oth
er_clusters=<ignores_other_clusters>]
```

By default, the kudu hms tools will ignore metadata in the HMS that refer to a different Kudu cluster than that being operated on, as indicated by having different masters specified. The tools compare the value of the kudu .master_addresses table property (either supplied at table creation or as --kudu_master_hosts on impalad daemons) in each HMS metadata entry against the RPC endpoints (including the ports) of the Kudu masters. To have the tooling account for and fix metadata entries with different master RPC endpoints specified (e.g. if ports are not specified in the HMS), supply --ignore_other_clusters=false as an argument to the kudu hms check and fix tools.

For example:

```
$ sudo -u kudu kudu hms check master-name-1:7051,master-name-2:7051,mast
er-name-3:7051 --hive_metastore_uris=thrift://hive-metastore:9083 --igno
re_other_clusters=false
```

3. If the kudu hms check tool reports an inconsistent catalog, perform a dry-run of the kudu hms fix tool to understand how the tool will attempt to address the automatically-fixable issues.

```
$ sudo -u kudu kudu hms fix master-name-1:7051,master-name-2:7051,master-
name-3:7051 --hive_metastore_uris=<hive_metastore_uris> --dryrun=true [--
ignore_other_clusters=<ignore_other_clusters>]
```

   For example:

```
$ sudo -u kudu kudu hms check master-name-1:7051,master-name-2:7051,mast
er-name-3:7051 --hive_metastore_uris=thrift://hive-metastore:9083 --dryr
un=true --ignore_other_clusters=false
```

4. Manually fix any issues that are reported by the check tool that cannot be automatically fixed. For example, rename any tables with names that are not Hive-conformant.
5. Run the kudu hms fix tool to automatically fix all the remaining issues.

```
$ sudo -u kudu kudu hms fix master-name-1:7051,master-name-2:7051,master
-name-3:7051 --hive_metastore_uris=<hive_metastore_uris> [--drop_orphan_
hms_tables=<drops_orphan_hms_tables>] [--ignore_other_clusters=<ignore_o
ther_clusters>]
```

   For example:

```
$ sudo -u kudu kudu hms fix master-name-1:7051,master-name-2:7051,master
-name-3:7051 --hive_metastore_uris=thrift://hive-metastore:9083 --ignore
_other_clusters=false
```

   Note: The --drop_orphan_hms_tables argument indicates whether to drop orphan Hive Metastore tables that refer to non-existent Kudu tables. Due to KUDU-2883, this option may fail to drop HMS entries that have no table ID. A workaround to this is to drop the table via the Impala Shell.

6. Recreate any external tables that were dropped when preparing for the upgrade by using Impala Shell.

What to do next

Enabling the Hive Metastore integration.

Related Information

Enabling the Hive Metastore integration

Enabling the Hive Metastore integration

You can enable the Hive Metastore (HMS) integration with Kudu using Cloudera Manager.

Before you begin

Upgrade any tables that may exist in Kudu's or in the HMS's catalog. For more information, see *Upgrading Existing Tables*.

Procedure

1. In Cloudera Manager navigate to Kudu Configuration .
2. Find the HMS Service property and select the Hive Metastore with which Kudu will synchronize its system catalog.

Related Information

Upgrading existing Kudu tables for Hive Metastore integration