Cloudera Runtime 7.3.2

Cloudera Iceberg REST Catalog for Data Sharing Reference

Date published: 2020-07-28
Date modified: 2026-03-31

https://docs.cloudera.com/

Legal Notice

Contents

Sample Spark workload to access data

See an example of an end to end sample workload flow that describes the process for enabling a Spark session, connecting to the HMS REST Catalog server, and running a Spark engine query to access data.

Note: Based on the type of client that is accessing the REST Catalog, refer to the appropriate client-side documentation to configure your client. For example, see the Tabular documentation.

Before you run the workload

Obtain the OAuth client ID and client secret for your Data Catalog consumer from the **Manage Users** export (CSV columns clientId and secret). Pass them to the sample script as a single argument, ClientId:Secret (a colon between the two values, no spaces).

From a shell, run your PySpark script and supply that value for --credential. Replace your_script.py with your script filename and substitute your real client credentials for the placeholder.

```
python your_script.py --credential '[***CLIENT-ID:CLIENT-SECRET***]'
```

Important: knox.token.ttl controls the lifetime of your Knox access token received in exchange for the Client ID and Secret of your external users. Once it expires, external users need to request a new access token. For more information, see Editing Knox topologies.

PySpark workload example

The PySpark workload connects to Cloudera HMS REST Catalog with CLIENT_ID and CLIENT_SECRET as credentials and runs workloads from an external system like Databricks, Snowflake, Standalone Spark in Docker, and so on. The following example expects the same ClientId:Secret string to be passed in at run time via args.credential (see the previous section).

```
# © 2024 by Cloudera, Inc. All rights reserved.
# Scripts and sample code are licensed under the Apache License,
# Version 2.0
import pyspark
from pyspark.sql import SparkSession
import argparse

parser = argparse.ArgumentParser(description="Spark WorkLoad Script")
parser.add_argument("--credential", help="ClientId:Secret")
args = parser.parse_args()

conf = (
    pyspark.SparkConf()
        .setAppName('Fetch Employees')
        .setMaster('[***SPARK-MASTER-URL***]')
        .set('spark.jars', '[***PATH-TO-ICEBERG-SPARK-RUNTIME-JAR***]')
        .set('spark.files', '[***PATH-TO-LOG4J2-PROPERTIES***]')
        #packages
        .set('spark.jars.packages', '[***ICEBERG-SPARK-RUNTIME-
PACKAGES***]')
        #SQL Extensions
        .set('spark.sql.extensions', 'org.apache.iceberg.spark.extensions
.IcebergSparkSessionExtensions')
        #Configuring Catalog
        .set('spark.sql.defaultCatalog', 'demo')
        .set('spark.sql.catalog.demo', 'org.apache.iceberg.spark.SparkCat
alog')
```

```
        .set('spark.sql.catalog.demo.type', 'rest')
        .set('spark.sql.catalog.demo.uri', 'https://[***DATALAKE-
HOSTNAME***]/[***DATALAKE-NAME***]/cdp-datashare-access/hms-api')
        .set('spark.sql.catalog.demo.io-impl', '[***ICEBERG-FILE-IO-
IMPL***]')
        .set('spark.sql.catalog.demo.s3.client-factory-impl', '[***ICEBERG-
S3-FILE-IO-CLIENT-FACTORY-IMPL***]')
        .set('spark.sql.catalog.demo.credential', args.credential)
        .set('spark.sql.catalog.demo.default-namespace', '[***DEFAULT-
NAMESPACE***]')
)

## Start Spark Session
spark = SparkSession.builder.config(conf=conf).getOrCreate()
spark.sparkContext.setLogLevel("DEBUG")
print("Spark Job Running...")
print("####### Credential: #######")
print(args.credential)
## list databases
dblist=spark.catalog.listDatabases()
print("###### List Databases ########")
print(dblist)
spark.sparkContext.parallelize([dblist]).coalesce(1).saveAsTextFile("file:
///[***OUTPUT-DATABASES-DIR***]")

## list tables
tableList=spark.catalog.listTables("demo.[***DEFAULT-NAMESPACE***]")
print("###### List Tables #######")
print(tableList)
spark.sparkContext.parallelize([tableList]).coalesce(1).saveAsTextFile("file
:///[***OUTPUT-TABLES-DIR***]")

## Run a Query
print("##### Query: fetch all the employees of 'department -> d006' ######")
results=spark.sql("select [***DEFAULT-NAMESPACE***].employees.first
_name, [***DEFAULT-NAMESPACE***].employees.last_name, [***DEFAULT-
NAMESPACE***].departments.dept_name "
                  "from [***DEFAULT-NAMESPACE***].employees, [***DEFAULT-
NAMESPACE***].departments, [***DEFAULT-NAMESPACE***].dept_emp "
                  "where [***DEFAULT-NAMESPACE***].employees.emp_n
o=[***DEFAULT-NAMESPACE***].dept_emp.emp_no "
                  "and [***DEFAULT-NAMESPACE***].dept_emp.dept_n
o=[***DEFAULT-NAMESPACE***].departments.dept_no "
                  "and [***DEFAULT-NAMESPACE***].departments.dept_no
='[***SAMPLE-DEPT-NO***]'")
print(results.show())
results.coalesce(1).write.option("header", "true").csv("file:///[***OUTPUT-
QUERY-RESULTS-DIR***]")
```

Related Information

Running Apache Spark Applications

Verifying external access to a Data Share

Cloudera Data Catalog Data Sharing Concurrent User Tests

Performance testing of Data Sharing with Cloudera Data Catalog in Cloudera on cloud 7.3.2.0 revealed stable operation with up to 100 concurrent users, while higher loads caused significant errors and timeouts. Recommendations include limiting concurrent users to 100 for optimal performance.

Test Environment

Tests were run for 30 minutes per user level against a cluster running Cloudera on cloud 7.3.2.0. The data set consisted of 50 databases with 100 tables and 1000 columns each. 1000 data shares were created, each sharing 5 tables with 200 external users.

Table 1: Test Results

Users	Requests	Avg. Response Time (miliseconds)	Error
50	9877	9087	0
100	9945	17484	0
150	10113	25222	728 (7.2%)

Conclusion

Data Catalog stress tests showed reliable performance with up to 100 concurrent users. At 150 concurrent users, a significantly higher error rate was observed when listing external users and listing data shares. Response times exceeded 60 seconds, resulting in request timeouts. The errors seen at 150 users were:

```
504 Gateway Time-out
Status Code: 504; Error Code: UNKNOWN_ERROR; Service: datacatalog; Operatio
n: listExternalUsers; Request ID: Unknown;
```

Note: The concurrent user count is the determining factor for this degradation, not the number of external users or data shares. Testing confirmed that 150 concurrent users caused timeouts even when the number of external users per Data Share was reduced.

Recommendation

Cloudera recommends keeping the number of concurrent users accessing the Cloudera Data Catalog APIs at or below 100 for stable operation. Exceeding this threshold is an extreme scenario that may lead to degraded performance and timeout errors.

Data Sharing longevity test results

The following are the results of a 50-user longevity test performed on the Cloudera Data Catalog and Cloudera Iceberg REST Catalog.

Test environment and dataset

The longevity test was conducted on an AWS Enterprise Data Lake cluster with the following dataset configuration:

Table 2: Cloudera Data Catalog longevity test dataset

Configuration item	Value
Databases	50
Tables	100
Columns	1000
Snapshots	0
Data Shares	1000
Tables per Data Share	5

Configuration item	Value
Users	200
Users per Data Share	200

Table 3: Cloudera Iceberg REST Catalog longevity test dataset

Configuration item	Value
Databases	250
Tables	20
Columns	1000
Snapshots	100
Data Shares	100
Tables per Data Share	15
Users per Data Share	50[1]

The following table displays the longevity test results for 50 concurrent users on the AWS EDL cluster environment:

Component	Users/Threads	Duration	Throughput	Total requests	Failed requests	Error %
Cloudera Data Catalog [2]	50	72 hours	2.03/s	527373	62	0.01%
Cloudera Iceberg REST Catalog	50	72 hours	33.90/s	8787271	0	0.00%

Note: During the Cloudera Data Catalog test, some requests failed with errors such as 504 Gateway Time-out and 500 UNKNOWN.

[1] Each user accessed one table from one database up to 50 Data Shares.
[2] The default Ranger heap size was 1 GB.