CLOUDERA

Cloudera Professional Services

BNI
CDP Cluster

Saketh Gadde
Solutions Architect

TABLE OF CONTENTS

CLOUDERA

IMPORTANT NOTICE

Cloudera, Inc.
395 Page Mill Road
Palo Alto, CA
94306
info@cloudera.com
US:
1-888-789-1488
Intl: 1-650-362-0488
www.cloudera.com

Release Information
Version: 1.0 Date: 13/06/2022

Hive Warehouse Connector(HWC)

Spark users cannot just run sparksql code on ACID tables. For governance reasons, users must go through the Hive Warehouse Connector (HWC) and make code changes to access the non-native ACID tables.

Spark is not natively compatible with ACID tables. You need to use the Hive Warehouse Connector (HWC) to read Hive ACID tables from the Hive metastore. There are two modes for HWC: JDBC mode and Direct Reader mode. The HMS translation layer prevents Spark from accessing Hive tables.

The HMS translation layer checks each client connection to determine the capabilities of the client. For example, HMS checks whether or not the client supports ORC and transactional tables. When a Spark client talks to the metastore, it can bypass HiveServer (HS2). Some operations, such as Direct Reader and Hive Streaming, go to Hive directly through HMS and reveal its capabilities to the HMS translation layer. If Spark does not have a connection to HWC associated, when the Spark user tries to get information about the ACID table, the query fails. The HMS translation layer determines that Spark without HWC does not have the required capabilities to access ACID tables, and gives you an error.

HWC JDBC Cluster mode

If the user does not have access to the file systems (restricted access), you can use HWC to submit HiveSQL from Spark with benefits of fine-grained access control (FGAC), row filtering and column masking, to securely access Hive tables from Spark.

However, if the size of your database query returns are less than 1 GB of data, it is recommended that you use HWC JDBC Cluster mode in which Spark executors connect to Hive through JDBC, and execute the query. Larger workloads are not recommended for JDBC reads in production due to slow performance.

HWC Secure access mode mode

If the user does not have access to the file systems (restricted access) and if the size of database query returns are greater than 1 GB of data, it is recommended to use HWC Secure access mode that offers fine-grained access control (FGAC), row filtering and column masking to access Hive table data from Spark.

Secure access mode enables you to set up an HDFS staging location to temporarily store Hive files that users need to read from Spark and secure the data using Ranger FGAC.

HWC Direct Reader mode

If the user has access to the file systems (ETL jobs do not require authorization and run as super user) and if you are accessing Hive managed tables, you can use the HWC Direct reader mode to allow Spark to read directly from the managed table location.

Important: This workload must be run with 'hive' user permissions.

If you are querying Hive external tables, use Spark native readers to read the external tables from Spark.

HWC(JDBC Mode) vs Direct Reader Mode for Handling Transaction Tables

1. Architecture and Integration:

- **JDBC Mode**:
  - Connects to HiveServer2 (HS2) to retrieve transactional information.
  - Secured through Ranger, supporting fine-grained access controls.
  - Involves a single JDBC connection, which can be a bottleneck.
  - Recommended for secure environments requiring strict authorization.
- **Direct Reader Mode**:
  - Bypasses HS2, directly accessing the Hive Metastore (HMS) for transactional snapshots.
  - Reads data directly from the managed table's file location.
  - Offers better performance due to reduced overhead from HS2.

2. Performance:

- **JDBC Mode**:
  - Performance can degrade with large data volumes due to the bottleneck created by HS2 interaction.
  - Optimized for smaller datasets (recommended for workloads of 1GB or less).
  - High latency and slower read speeds compared to Direct Reader Mode.
- **Direct Reader Mode**:
  - High throughput with lower latency as it directly accesses the data.
  - Supports vectorized ORC reads, improving performance on large datasets.
  - Better suited for read-heavy operations in ETL processes where speed is critical.

3. Security:

- **JDBC Mode**:
  - Strong security model with full Ranger integration, enabling column-level security, masking, and row-level filtering.
  - Suitable for environments with stringent data security and compliance requirements.
- **Direct Reader Mode**:
  - Lacks Ranger authorization, making it unsuitable for environments requiring detailed access controls.
  - Assumes that security is handled externally (e.g., at the filesystem level).

4. Compatibility:

- **JDBC Mode**:
  - Compatible with a broader range of use cases, including both transactional and external tables.
  - Supports Spark 2.4.7, PySpark, Zeppelin, and Sparklyr (with limitations).
- **Direct Reader Mode**:
  - Optimized for high-performance reads of transactional tables using ORC format.
  - Requires Spark 2.4.7 or later, but lacks support for Spark 3.

5. Use Cases:

- **JDBC Mode**:
  - Best for secure environments where data governance and compliance are critical.
  - Recommended for smaller datasets or scenarios where fine-grained access control is required.
- **Direct Reader Mode**:
  - Ideal for high-performance ETL jobs where reading speed is crucial.
  - Suitable for environments where security can be managed outside the HWC.

Recommendation for Performance Issues

Given the performance issues with reading large transactional tables using HWC in JDBC mode, the following recommendations are advised:

1. **Switch to Direct Reader Mode**:
    - If the primary concern is performance and security constraints are less stringent, consider switching to Direct Reader Mode. This mode will reduce the latency associated with HS2 and increase the read speed for large datasets.
2. **Optimize JDBC Configuration**:
    - If switching to Direct Reader Mode is not feasible due to security requirements, optimize the existing JDBC setup:
        - Increase the number of parallel JDBC connections.
        - Fine-tune HiveServer2 settings, including memory and connection pooling.
        - Implement caching strategies or pre-fetching data to minimize the overhead during reads.
3. **Hybrid Approach**:
    - Utilize Direct Reader Mode for performance-critical ETL processes and reserve JDBC Mode for operations requiring strict security. This hybrid approach allows leveraging the strengths of both modes depending on the specific use case.

Related Information

Introduction to HWC
Introduction to HWC JDBC read mode
Introduction to HWC Secure access mode
Introduction to HWC Direct reader mode