Hive APIs Overview

This page aims to catalogue and describe the various public facing APIs exposed by Hive in order to inform developers wishing to integrate their applications and frameworks with the Hive ecosystem. To date the following APIs have been identified in the Hive project that are either considered public, or widely used in the public domain:

API categories

The APIs can be segmented into two conceptual categories: operation based APIs and query based APIs.

Operation based APIs

Operation based APIs expose many tightly scoped methods that each implement a very specific Hive operation. Such methods usually accept and return strongly typed values appropriate to their respective operation. The implementations of the operations usually target very specific layers or subsystems within Hive and are therefore likely to be efficient in use. However, the outcome of an operation may diverge from that of the equivalent HQL as the different code paths may be invoked in each case. Operation based APIs are used for constructing processes that need to interact in a repetitive, declarative manner and provide a greater degree of compile-time checking.

Query based APIs

Query based APIs permit the submission and execution of some subset of HQL. It is often necessary for the API client to parse and interpret any returned values as the return types are frequently quite broad in scope. The implementations of such APIs usually target the 'query language' subsystem of Hive which parses the query and executes it as needed. Given that most query based APIs share a similar execution pathway, it is likely that any operation submitted via the API will have a similar outcome to equivalent HQL submitted via the Hive CLI. Query based APIs are often used for constructing processes where Hive API operations are created dynamically or where an HQL equivalent outcome is important. Drawbacks of this type of API include: lack of compile time checking, possible inefficiencies in working at a higher level of abstraction, and potential susceptibility to SQL-injection like attacks.

Available APIs

HCatClient (Java)

Operation based Java API that presents a number of DDL type operations, however it is not as comprehensive as the Metastore API. The HCatClient was intended to be the Java based entry point to WebHCat HCatalog API although this was never realised. Currently `HCatClientHMSImpl` is the only concrete implementation of the API; it integrates directly with the Metastore using the Metastore API and does not utilise WebHCat whatsoever despite being packaged inside the WebHCat project. The `HCatClientHMSImpl` was originally provided as a reference implementation but it has over time gained traction as a public client. Anecdotally, it is now the preferred API for issuing DDL type operations from external programs; and feature contributions are encouraged. There is some minimal documentation on the HCatalog wiki in the form of a design document describing the interface but not the implementation.

HCatalog Storage Handlers (Java)

Operation based Java API. This is well documented on the wiki.

TODO

Requires overview.

HiveServer2 API

Query based Java API.

TODO

Describe.

HCatalog CLI (Command Line)

Query based API. This is well documented on the wiki.

TODO

Requires overview. Describe differences with `hive` CLI.

Metastore (Java)

A Thrift operation based API with Java bindings, described by the `IMetaStoreClient` interface. The API decouples the metastore storage layer from other Hive internals. Because Hive itself uses this internally, it is required to implement a comprehensive feature set which makes it attractive to developers who might find the other APIs lacking. It was not originally intended to be a public API although it became public in version 1.0.0 (HIVE-3280) and there is a proposal that it be documented more fully (HIVE-9363). Anecdotally, its use outside of the Hive project is not currently recommended.

TODO: API usage

There are numerous ways of instantiating the metastore API including: `HCatUtil.getHiveMetastoreClient()`, `new HiveMetaStoreClient.HiveMetaStoreClient(...)`. It may be useful to make some recommendations on the preferred approach.

Hive (Java)

Refers to the `org.apache.hadoop.hive.ql.metadata.Hive` class. Appears to be a distinct concrete implementation of a variation of the metastore API. Delegates to the metastore API but does not directly extend/implement it.

TODO

I suspect its use is not encouraged. Seeking clarification on the motivations behind this class and thoughts on its use outside of Hive.

Driver (Java)

Query based API with Java endpoint. Refers to the `org.apache.hadoop.hive.ql.Driver` class.

TODO

Describe the role of `Driver`, when to use it, etc.

WebHCat (REST)

WebHCat is a REST operation based API for HCatalog. This is well documented on the wiki.

TODO

Is this API actively used? Is its use encouraged? When should one use it?

Streaming Data Ingest (Java)

Operation based Java API focused on the writing of continuous streams of data into transactional tables using Hive's ACID feature. New data is inserted into tables using small batches and short-lived transactions. Documented on the wiki and has package level Javadoc. Introduced in Hive version 0.13.0 (HIVE-5687).

Streaming Mutation (Java)

Operation based Java API focused on mutating (insert/update/delete) records into transactional tables using Hive's ACID feature. Large volumes of mutations are applied atomically in a single long-lived transaction. Documented on the wiki. Scheduled for release in Hive version 2.0.0 (HIVE-10165).

hive-jdbc (JDBC)

Query based API.

TODO

Seeking documentation. Is this just for reading data? writing data? does it support DDL operations?