Storage Handler Integration – HIVE-4331

Design Document

Problem Statement:

This document presents the design of the integrated **StorageHandler** in **Hive 0.12** and beyond. Currently there are two code paths available to access an HBase table one in HCat codebase (residing as a Hive subproject) using the "**HCatHBaseStorageHandler [2]**" and the other in Hive using the "**HBaseStorageHandler [1]**". Unfortunately a table created using one storage handler cannot be read using the other. For example an HBase table created using the "**HBaseStorageHandler**" from the Hive command line cannot be written using the "**HCatHBaseStorageHandler**" using a Pig script. This creates a problem for users who want to create a table using Hive and then populate the same using Pig **[refer Appendix]**.

Proposal:

We intend to accomplish the following for the Storage Handler Integration work which will remove this inconsistency in the behavior between the HCat and Hive code base.

1) Remove all "**HCatStorageHandler**" instances in the HCat code and replace this with the "**HiveStorageHandler**" or **"DefaultStorageHandler"**. This will require modifying all the underlying code in the "**HCatUtil**", "**HCatInputFormat**" and removing the "**FosterStorageHandler**".
2) N**ew unit tests** in HCat will now be able to use the "**HiveStorageHandler**". These tests will be usable by Pig through the **HCatLoader** as well as using the Map Reduce program by supplying the "**HCatOutputFormat**" and "**HCatInputFormat**"
3) Design a "**HivePassThroughOutputFormat**" which will derive from the "**HiveOutputFormat**" so that StorageHandler's OutputFormat may not be dependent on the "**HiveOutputFormat**" and can extend the normal "**OutputFormat**" of **its choice**. "**HivePassThroughOutputFormat**" makres sure the "**HiveOutputFormat**" does not leak into HCatalog code. This would make it easier to deprecate this class and move to storage handlers in the future. This gives the StorageHandler implementer much more flexibility. Here is a skeleton of the "**HivePassThroughOutputFormat**"

```
// HivePassThroughOutputFormat

public class HivePassThroughOutputFormat<K, V> implements Configurable,
HiveOutputFormat<K, V>{

  public HivePassThroughOutputFormat() { }

   private void createActualOF() throws IOException { }

   public void checkOutputSpecs(FileSystem ignored, JobConf job) {}
```

```
  public org.apache.hadoop.mapred.RecordWriter<K, V> getRecordWriter(…)  { }

  public org.apache.hadoop.hive.ql.exec.FileSinkOperator.RecordWriter getHiveRecordWriter(..){ }

  public Configuration getConf() {}

  public void setConf(Configuration config) { }

}
```

4) Design a corresponding "**HivePassThroughRecordReader**" which will allow the StorageHandler to use any "**OutputFormat**" so that it is not tied to the "**HiveRecordReader**". This will open the StorageHandler for use with the RecordReader of its choice.

```
// HivePassThroughRecordWriter

public class HivePassThroughRecordWriter <K extends WritableComparable<?>, V extends
Writable> implements RecordWriter {

  public HivePassThroughRecordWriter() { }

  public void write(Writable r) throws IOException { }

  public void close(boolean abort) throws IOException { }

   }
```

5) With the use of the "**HivePassThroughOutputFormat**" in the Hive code the "**HBaseStorageHandler**" "**HiveHBaseTableOutputFormat**" can intuitively use the "**TableOutputFormat**"

   public class HiveHBaseTableOutput"Format extends TableOutputFormat<ImmutableBytesWritable>

   Instead of:

   public class HiveHBaseTableOutputFormat extends HiveOutputFormat<ImmutableBytesWritable

6) Ensure that all the HBaseHandler unit tests of Hive pass with the new "**HivePassThroughOutputFormat**"
7) Ensure that **backward compatibility** is maintained so that old tables create using the "**HCatStorageHander**" function as expected.

Salient Features:

1) Cleaner design which will integrate Hive and HCat storage handlers
2) Use of "**HivePassThroughOutputFormat**" allow the designer of new StorageHandlers to rely on their own OutputFormat rather on the "HiveOutputFormat"
3) Users will only use the HiveStorageHandler, rather than having to deal with numerous other handlers and wrappers which are currently present such as "**HCatStorageHandler**", "**DefaultStorageHandler**"

4) "HivePassThroughOutputFormat" makes sure that the "HiveOutputFormat" does not leak into HCat code. Which would make it easier to deprecate "HiveOutputFormat" and move it to storage handlers in the future.

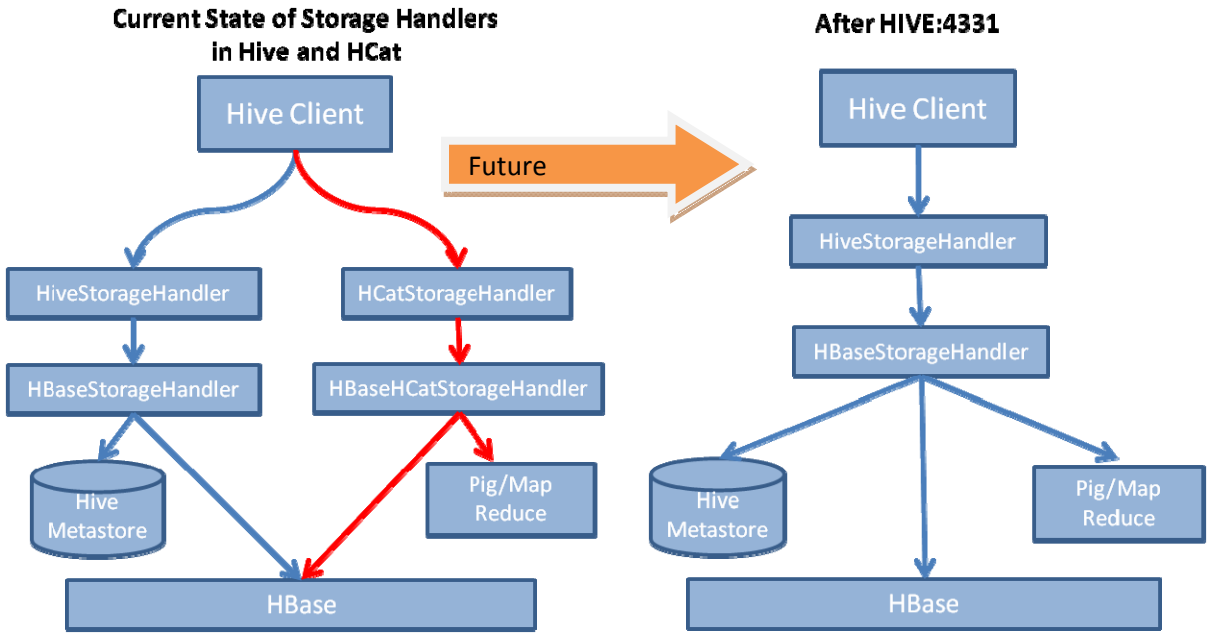
Open Issues:

1) Golden Files in the "**HBaseStorageHandler**" contain the "**HivePassThroughOutputFormat**" when an **explain** is done on the query instead of the "**HiveHBaseTableOutputFormat**"
2) "**HCatStorageHandler**" and "**Revision Manager**" classes have been deprecated and will be removed in the future release of Hive.

Testing

Real tests were run on the Yahoo! clusters to make sure that tables created using the HBaseStorage handler were accessible through Pig using the HCatLoader as well as Map Reduce. We do not have performance numbers as yet.

Appendix:



References:

1. Hive Storage Handler https://cwiki.apache.org/Hive/storagehandlers.html

2. HCat Storage Handler https://cwiki.apache.org/HCATALOG/hcatalog-hbase-integration-design.html