CASCADING - DE-FACTO FOR DATA APPS

- Standard for enterprise data app development
- Your programming language of choice
- Cascading applications that run on MapReduce will also run on Apache Spark, Storm, and …
CASCADING-HIVE

• Cascading-Hive provides integration between Cascading and Apache Hive.

• This project bridges the gap by making it possible to read, write and create Hive tables from within Cascading flows and allow Hive queries to participate in a Cascade.
Key Functionality:

- Run Hive queries within a Cascade
- Read from Hive tables within a Cascading Flow
- Write/create Hive tables from a Cascading Flow
- Write/create partitioned Hive tables from a Cascading Flow
- Deconstruct a Hive view into Taps
- Hive Metastore support
CASCADING-HIVE

Key Components:

https://github.com/Cascading/cascading-hive

- HiveTableDescriptor
  https://github.com/Cascading/cascading-hive/blob/wip-1.1/src/main/java/cascading/tap/hive/HiveTableDescriptor.java

- HiveTap

- HivePartitionTap

- HiveFlow
// create Hive query for returns_catalog tables
String returnsQuery = "SELECT cr_item_sk, COUNT(cr_item_sk) AS quantity_returned " +
"FROM returns_catalog " +
"GROUP BY cr_item_sk ORDER BY quantity_returned DESC LIMIT 20";

// add returnsQuery to array for use in HiveFlow
String queriesReturns[] = {returnsQuery};

// create HiveTableDescriptor for returnsQuery results
HiveTableDescriptor topReturnsTableDescriptor = new HiveTableDescriptor( "Top_20_Returns",
new String[]{"cr_item_sk", "quantity_returned"}, new String[]{"string", "int"} );

// create HiveTap as sink for returnsQuery results
HiveTap topReturnsTap = new HiveTap( topReturnsTableDescriptor, topReturnsTableDescriptor.toScheme(),
REPLACE, true );

// create HiveFlow using returnsQuery, returnsTap (HiveTap) as sources, topReturnsTap (HiveTap) as sink
HiveFlow topReturnsByCategoryHiveFlow = new HiveFlow( "Hive Flow - TopReturnsByCategory",
queriesReturns, Arrays.asList( returnsTap ), topReturnsTap );

// create, connect and complete cascade including HiveFlows
CascadeConnector connector = new CascadeConnector();
Cascade cascade = connector.connect( flow1, flow2, topSalesByCategoryHiveFlow,
topReturnsByCategoryHiveFlow );
cascade.complete();
• Understand the anatomy of your Hive app
• Track execution of queries as single business process
• Identify outlier behavior by comparison with historical runs
• Analyze rich operational meta-data
• Correlate Hive app behavior with other events on cluster
Thank You

Ryan Desmond

ryand@concurrentinc.com
https://www.linkedin.com/in/ryanadesmond