package com.twitter.search.earlybird\_root;

import java.util.Map;

import com.google.common.collect.Maps;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import com.twitter.search.common.partitioning.base.PartitionMappingManager;

import com.twitter.search.earlybird\_root.visitors.MultiTermDisjunctionPerPartitionVisitor;

import com.twitter.search.queryparser.query.Query;

import com.twitter.search.queryparser.query.QueryParserException;

public final class EarlybirdRootQueryUtils {

private static final Logger LOG = LoggerFactory.getLogger(EarlybirdRootQueryUtils.class);

private EarlybirdRootQueryUtils() {

}

/\*\*

\* Rewrite 'multi\_term\_disjunction from\_user\_id' or 'multi\_term\_disjunction id' based on partition

\* for USER\_ID/TWEET\_ID partitioned cluster

\* @return a map with partition id as key and rewritten query as value.

\* If there is no 'multi\_term\_disjunction from\_user\_id/id' in query, the map will be empty; if all

\* ids are truncated for a partition, it will add a NO\_MATCH\_CONJUNCTION here.

\*/

public static Map<Integer, Query> rewriteMultiTermDisjunctionPerPartitionFilter(

Query query, PartitionMappingManager partitionMappingManager, int numPartitions) {

Map<Integer, Query> m = Maps.newHashMap();

// If there is no parsed query, just return

if (query == null) {

return m;

}

for (int i = 0; i < numPartitions; ++i) {

MultiTermDisjunctionPerPartitionVisitor visitor =

new MultiTermDisjunctionPerPartitionVisitor(partitionMappingManager, i);

try {

Query q = query.accept(visitor);

if (q != null && q != query) {

m.put(i, q);

}

} catch (QueryParserException e) {

// Should not happen, put and log error here just in case

m.put(i, query);

LOG.error(

"MultiTermDisjuctionPerPartitionVisitor cannot process query: " + query.serialize());

}

}

return m;

}

}