package com.twitter.search.earlybird.util;

import java.util.List;

import java.util.Map;

import java.util.Set;

import javax.annotation.Nullable;

import com.google.common.collect.ImmutableMap;

import com.twitter.search.common.constants.thriftjava.ThriftLanguage;

import com.twitter.search.common.database.DatabaseConfig;

import com.twitter.search.common.query.thriftjava.EarlyTerminationInfo;

import com.twitter.search.common.util.earlybird.ResultsUtil;

import com.twitter.search.common.util.earlybird.ThriftSearchResultUtil;

import com.twitter.search.common.util.earlybird.ThriftSearchResultsRelevanceStatsUtil;

import com.twitter.search.core.earlybird.facets.LanguageHistogram;

import com.twitter.search.earlybird.partition.PartitionConfig;

import com.twitter.search.earlybird.search.Hit;

import com.twitter.search.earlybird.search.SearchResultsInfo;

import com.twitter.search.earlybird.search.SimpleSearchResults;

import com.twitter.search.earlybird.search.relevance.RelevanceSearchResults;

import com.twitter.search.earlybird.thrift.ThriftSearchResult;

import com.twitter.search.earlybird.thrift.ThriftSearchResultDebugInfo;

import com.twitter.search.earlybird.thrift.ThriftSearchResultMetadata;

import com.twitter.search.earlybird.thrift.ThriftSearchResults;

import com.twitter.search.earlybird.thrift.ThriftSearchResultsRelevanceStats;

// EarlybirdSearchResultUtil contains some simple static methods for constructing

// ThriftSearchResult objects.

public final class EarlybirdSearchResultUtil {

public static final double MIN\_LANGUAGE\_RATIO\_TO\_KEEP = 0.002;

private EarlybirdSearchResultUtil() { }

/\*\*

\* Update result stats on the ThriftSearchResult.

\*/

public static void setResultStatistics(ThriftSearchResults results, SearchResultsInfo info) {

results.setNumHitsProcessed(info.getNumHitsProcessed());

results.setNumPartitionsEarlyTerminated(info.isEarlyTerminated() ? 1 : 0);

if (info.isSetSearchedStatusIDs()) {

results.setMaxSearchedStatusID(info.getMaxSearchedStatusID());

results.setMinSearchedStatusID(info.getMinSearchedStatusID());

}

if (info.isSetSearchedTimes()) {

results.setMaxSearchedTimeSinceEpoch(info.getMaxSearchedTime());

results.setMinSearchedTimeSinceEpoch(info.getMinSearchedTime());

}

}

/\*\*

\* Create an EarlyTerminationInfo based on information inside a SearchResultsInfo.

\*/

public static EarlyTerminationInfo prepareEarlyTerminationInfo(SearchResultsInfo info) {

EarlyTerminationInfo earlyTerminationInfo = new EarlyTerminationInfo(info.isEarlyTerminated());

if (info.isEarlyTerminated()) {

earlyTerminationInfo.setEarlyTerminationReason(info.getEarlyTerminationReason());

}

return earlyTerminationInfo;

}

/\*\*

\* Populate language histogram inside ThriftSerachResults.

\*/

public static void setLanguageHistogram(ThriftSearchResults results,

LanguageHistogram languageHistogram) {

int sum = 0;

for (int value : languageHistogram.getLanguageHistogram()) {

sum += value;

}

if (sum == 0) {

return;

}

ImmutableMap.Builder<ThriftLanguage, Integer> builder = ImmutableMap.builder();

int threshold = (int) (sum \* MIN\_LANGUAGE\_RATIO\_TO\_KEEP);

for (Map.Entry<ThriftLanguage, Integer> entry : languageHistogram.getLanguageHistogramAsMap()

.entrySet()) {

if (entry.getValue() > threshold) {

builder.put(entry.getKey(), entry.getValue());

}

}

Map<ThriftLanguage, Integer> langCounts = builder.build();

if (langCounts.size() > 0) {

results.setLanguageHistogram(langCounts);

}

}

private static void addDebugInfoToResults(List<ThriftSearchResult> resultArray,

@Nullable PartitionConfig partitionConfig) {

if (partitionConfig == null) {

return;

}

ThriftSearchResultDebugInfo debugInfo = new ThriftSearchResultDebugInfo();

debugInfo.setHostname(DatabaseConfig.getLocalHostname());

// These info can also come from EarlybirdServer.get().getPartitionConfig() if we add such a

// getter for partitionConfig().

debugInfo.setPartitionId(partitionConfig.getIndexingHashPartitionID());

debugInfo.setTiername(partitionConfig.getTierName());

debugInfo.setClusterName(partitionConfig.getClusterName());

for (ThriftSearchResult result : resultArray) {

result.setDebugInfo(debugInfo);

}

}

/\*\*

\* Write results into the result array.

\* @param resultArray the result array to write into.

\* @param hits the hits from the search.

\* @param partitionConfig partition config used to fill in debug info. Pass in null if no debug

\* info should be written into results.

\*/

public static void prepareResultsArray(List<ThriftSearchResult> resultArray,

SimpleSearchResults hits,

@Nullable PartitionConfig partitionConfig) {

for (int i = 0; i < hits.numHits(); i++) {

final Hit hit = hits.getHit(i);

final long id = hit.getStatusID();

final ThriftSearchResult result = new ThriftSearchResult(id);

final ThriftSearchResultMetadata resultMetadata = hit.getMetadata();

result.setMetadata(resultMetadata);

resultArray.add(result);

}

addDebugInfoToResults(resultArray, partitionConfig);

}

/\*\*

\* Write results into the result array.

\* @param resultArray the result array to write into.

\* @param hits the hits from the search.

\* @param userIDWhitelist Used to set flag ThriftSearchResultMetadata.dontFilterUser.

\* @param partitionConfig partition config used to fill in debug info. Pass in null if no debug

\* info should be written into results.

\*/

public static void prepareRelevanceResultsArray(List<ThriftSearchResult> resultArray,

RelevanceSearchResults hits,

Set<Long> userIDWhitelist,

@Nullable PartitionConfig partitionConfig) {

for (int i = 0; i < hits.numHits(); i++) {

final long id = hits.getHit(i).getStatusID();

final ThriftSearchResult result = new ThriftSearchResult(id);

final ThriftSearchResultMetadata resultMetadata = hits.resultMetadata[i];

result.setMetadata(resultMetadata);

if (userIDWhitelist != null) {

resultMetadata.setDontFilterUser(userIDWhitelist.contains(resultMetadata.getFromUserId()));

}

resultArray.add(result);

}

addDebugInfoToResults(resultArray, partitionConfig);

}

/\*\*

\* Merge a List of ThriftSearchResults into a single ThriftSearchResults object.

\*/

public static ThriftSearchResults mergeSearchResults(List<ThriftSearchResults> allSearchResults) {

ThriftSearchResults mergedResults = new ThriftSearchResults();

mergedResults.setRelevanceStats(new ThriftSearchResultsRelevanceStats());

mergedResults.setHitCounts(ResultsUtil.aggregateCountMap(allSearchResults,

ThriftSearchResultUtil.HIT\_COUNTS\_MAP\_GETTER));

mergedResults.setLanguageHistogram(ResultsUtil.aggregateCountMap(allSearchResults,

ThriftSearchResultUtil.LANG\_MAP\_GETTER));

for (ThriftSearchResults searchResults : allSearchResults) {

// Add results

mergedResults.getResults().addAll(searchResults.getResults());

// Update counts

ThriftSearchResultUtil.incrementCounts(mergedResults, searchResults);

// Update relevance stats

if (searchResults.getRelevanceStats() != null) {

ThriftSearchResultsRelevanceStatsUtil.addRelevanceStats(mergedResults.getRelevanceStats(),

searchResults.getRelevanceStats());

}

}

return mergedResults;

}

}