package com.twitter.search.core.earlybird.facets;

import java.util.HashMap;

import java.util.HashSet;

import java.util.Iterator;

import java.util.Map;

import java.util.Set;

import com.google.common.collect.Sets;

import com.twitter.search.common.schema.base.Schema;

/\*\*

\* Maintains internal state during one facet count request.

\*/

public final class FacetCountState<R> {

private final Set<Schema.FieldInfo> fieldsToCount = new HashSet<>();

private final Map<String, FacetFieldResults<R>> facetfieldResults =

new HashMap<>();

private final int minNumFacetResults;

private final Schema schema;

public FacetCountState(Schema schema, int minNumFacetResults) {

this.schema = schema;

this.minNumFacetResults = minNumFacetResults;

}

/\*\*

\* Adds a facet to be counted in this request.

\*/

public void addFacet(String facetName, int numResultsRequested) {

facetfieldResults.put(facetName, new FacetFieldResults(facetName,

Math.max(numResultsRequested, minNumFacetResults)));

Schema.FieldInfo field = schema.getFacetFieldByFacetName(facetName);

fieldsToCount.add(field);

}

public Schema getSchema() {

return schema;

}

public int getNumFieldsToCount() {

return fieldsToCount.size();

}

/\*\*

\* Returns whether or not there is a field to be counted for which no skip list is stored

\*/

public boolean hasFieldToCountWithoutSkipList() {

for (Schema.FieldInfo facetField: fieldsToCount) {

if (!facetField.getFieldType().isStoreFacetSkiplist()) {

return true;

}

}

return false;

}

public Set<Schema.FieldInfo> getFacetFieldsToCountWithSkipLists() {

return Sets.filter(

fieldsToCount,

facetField -> facetField.getFieldType().isStoreFacetSkiplist());

}

public boolean isCountField(Schema.FieldInfo field) {

return fieldsToCount.contains(field);

}

public Iterator<FacetFieldResults<R>> getFacetFieldResultsIterator() {

return facetfieldResults.values().iterator();

}

public static final class FacetFieldResults<R> {

public final String facetName;

public final int numResultsRequested;

public R results;

public int numResultsFound;

public boolean finished = false;

private FacetFieldResults(String facetName, int numResultsRequested) {

this.facetName = facetName;

this.numResultsRequested = numResultsRequested;

}

public boolean isFinished() {

return finished || results != null && numResultsFound >= numResultsRequested;

}

}

}