package com.twitter.search.earlybird.search.queries;

import java.io.IOException;

import java.util.Set;

import com.google.common.base.Preconditions;

import org.apache.lucene.index.LeafReader;

import org.apache.lucene.index.LeafReaderContext;

import org.apache.lucene.index.NumericDocValues;

import org.apache.lucene.search.BooleanClause;

import org.apache.lucene.search.BooleanQuery;

import org.apache.lucene.search.DocIdSetIterator;

import org.apache.lucene.search.IndexSearcher;

import org.apache.lucene.search.Query;

import org.apache.lucene.search.ScoreMode;

import org.apache.lucene.search.Weight;

import com.twitter.search.common.query.DefaultFilterWeight;

import com.twitter.search.core.earlybird.index.util.RangeFilterDISI;

public final class FeatureValueInAcceptListOrUnsetFilter extends Query {

private final String featureName;

private final Set<Long> idsAcceptList;

/\*\*

\* Creates a query that filters for hits that have the given feature unset, or that have the

\* given feature set to a value in the given list of IDs.

\*

\* @param featureName The feature.

\* @param ids A list of id values this filter will accept for the given feature.

\* @return A query that filters out all hits that have the given feature set.

\*/

public static Query getFeatureValueInAcceptListOrUnsetFilter(String featureName, Set<Long> ids) {

return new BooleanQuery.Builder()

.add(new FeatureValueInAcceptListOrUnsetFilter(featureName, ids),

BooleanClause.Occur.FILTER)

.build();

}

@Override

public String toString(String s) {

return String.format("FeatureValueInAcceptListOrUnsetFilter(%s, AcceptList = (%s))",

featureName,

idsAcceptList);

}

@Override

public boolean equals(Object obj) {

if (!(obj instanceof FeatureValueInAcceptListOrUnsetFilter)) {

return false;

}

FeatureValueInAcceptListOrUnsetFilter filter =

FeatureValueInAcceptListOrUnsetFilter.class.cast(obj);

return featureName.equals(filter.featureName) && idsAcceptList.equals(filter.idsAcceptList);

}

@Override

public int hashCode() {

return featureName.hashCode() \* 7 + idsAcceptList.hashCode();

}

private FeatureValueInAcceptListOrUnsetFilter(String featureName, Set<Long> ids) {

this.featureName = Preconditions.checkNotNull(featureName);

this.idsAcceptList = Preconditions.checkNotNull(ids);

}

@Override

public Weight createWeight(IndexSearcher searcher, ScoreMode scoreMode, float boost) {

return new DefaultFilterWeight(this) {

@Override

protected DocIdSetIterator getDocIdSetIterator(LeafReaderContext context) throws IOException {

return new FeatureValueInAcceptListOrUnsetDocIdSetIterator(

context.reader(), featureName, idsAcceptList);

}

};

}

private static final class FeatureValueInAcceptListOrUnsetDocIdSetIterator

extends RangeFilterDISI {

private final NumericDocValues featureDocValues;

private final Set<Long> idsAcceptList;

FeatureValueInAcceptListOrUnsetDocIdSetIterator(

LeafReader indexReader, String featureName, Set<Long> ids) throws IOException {

super(indexReader);

this.featureDocValues = indexReader.getNumericDocValues(featureName);

this.idsAcceptList = ids;

}

@Override

public boolean shouldReturnDoc() throws IOException {

// If featureDocValues is null, that means there were no documents indexed with the given

// field in the current segment.

//

// The advanceExact() method returns false if it cannot find the given docId in the

// NumericDocValues instance. So if advanceExact() returns false then we know the feature is

// unset.

// However, for realtime Earlybirds we have a custom implementation of NumericDocValues,

// ColumnStrideFieldDocValues, which will contain an entry for every indexed docId and use a

// value of 0 to indicate that a feature is unset.

//

// So to check if a feature is unset for a given docId, we first need to check if we can find

// the docId, and then we additionally need to check if the feature value is 0.

return featureDocValues == null

|| !featureDocValues.advanceExact(docID())

|| featureDocValues.longValue() == 0

|| idsAcceptList.contains(featureDocValues.longValue());

}

}

}