package com.twitter.search.earlybird.index;

import java.io.IOException;

import java.util.Arrays;

import java.util.Set;

import com.google.common.collect.Sets;

import org.apache.lucene.index.LeafReaderContext;

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.common.search.IntArrayDocIdSetIterator;

import com.twitter.search.core.earlybird.index.DocIDToTweetIDMapper;

import com.twitter.search.core.earlybird.index.EarlybirdIndexSegmentAtomicReader;

import com.twitter.search.core.earlybird.index.EarlybirdIndexSegmentData;

public class TweetIDQuery extends Query {

private final Set<Long> tweetIDs = Sets.newHashSet();

public TweetIDQuery(long... tweetIDs) {

for (long tweetID : tweetIDs) {

this.tweetIDs.add(tweetID);

}

}

@Override

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

return new DefaultFilterWeight(this) {

@Override

protected DocIdSetIterator getDocIdSetIterator(LeafReaderContext context) throws IOException {

EarlybirdIndexSegmentData segmentData =

((EarlybirdIndexSegmentAtomicReader) context.reader()).getSegmentData();

DocIDToTweetIDMapper docIdToTweetIdMapper = segmentData.getDocIDToTweetIDMapper();

Set<Integer> set = Sets.newHashSet();

for (long tweetID : tweetIDs) {

int docID = docIdToTweetIdMapper.getDocID(tweetID);

if (docID != DocIDToTweetIDMapper.ID\_NOT\_FOUND) {

set.add(docID);

}

}

if (set.isEmpty()) {

return DocIdSetIterator.empty();

}

int[] docIDs = new int[set.size()];

int i = 0;

for (int docID : set) {

docIDs[i++] = docID;

}

Arrays.sort(docIDs);

return new IntArrayDocIdSetIterator(docIDs);

}

};

}

@Override

public int hashCode() {

return tweetIDs.hashCode();

}

@Override

public boolean equals(Object obj) {

if (!(obj instanceof TweetIDQuery)) {

return false;

}

return tweetIDs.equals(TweetIDQuery.class.cast(obj).tweetIDs);

}

@Override

public String toString(String field) {

return "TWEET\_ID\_QUERY: " + tweetIDs;

}

}