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

import java.io.IOException;

import java.util.Objects;

import org.apache.lucene.index.LeafReaderContext;

import org.apache.lucene.index.NumericDocValues;

import com.twitter.search.common.metrics.SearchRateCounter;

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

import com.twitter.search.common.schema.earlybird.EarlybirdFieldConstants.EarlybirdFieldConstant;

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

import com.twitter.search.earlybird.common.userupdates.UserScrubGeoMap;

import com.twitter.search.earlybird.index.TweetIDMapper;

/\*\*

\* Filter that can be used with searches over geo field postings lists in order to filter out tweets

\* that have been geo scrubbed. Determines if a tweet has been geo scrubbed by comparing the

\* tweet's id against the max scrubbed tweet id for that tweet's author, which is stored in the

\* UserScrubGeoMap.

\*

\* See: go/realtime-geo-filtering

\*/

public class UserScrubGeoFilter implements FilteredQuery.DocIdFilterFactory {

private UserScrubGeoMap userScrubGeoMap;

private final SearchRateCounter totalRequestsUsingFilterCounter =

SearchRateCounter.export("user\_scrub\_geo\_filter\_total\_requests");

public static FilteredQuery.DocIdFilterFactory getDocIdFilterFactory(

UserScrubGeoMap userScrubGeoMap) {

return new UserScrubGeoFilter(userScrubGeoMap);

}

public UserScrubGeoFilter(UserScrubGeoMap userScrubGeoMap) {

this.userScrubGeoMap = userScrubGeoMap;

totalRequestsUsingFilterCounter.increment();

}

@Override

public FilteredQuery.DocIdFilter getDocIdFilter(LeafReaderContext context) throws IOException {

// To determine if a given doc has been geo scrubbed we need two pieces of information about the

// doc: the associated tweet id and the user id of the tweet's author. We can get the tweet id

// from the TweetIDMapper for the segment we are currently searching, and we can get the user id

// of the tweet's author by looking up the doc id in the NumericDocValues for the

// FROM\_USER\_ID\_CSF.

//

// With this information we can check the UserScrubGeoMap to find out if the tweet has been

// geo scrubbed and filter it out accordingly.

final EarlybirdIndexSegmentAtomicReader currTwitterReader =

(EarlybirdIndexSegmentAtomicReader) context.reader();

final TweetIDMapper tweetIdMapper =

(TweetIDMapper) currTwitterReader.getSegmentData().getDocIDToTweetIDMapper();

final NumericDocValues fromUserIdDocValues = currTwitterReader.getNumericDocValues(

EarlybirdFieldConstant.FROM\_USER\_ID\_CSF.getFieldName());

return (docId) -> fromUserIdDocValues.advanceExact(docId)

&& !userScrubGeoMap.isTweetGeoScrubbed(

tweetIdMapper.getTweetID(docId), fromUserIdDocValues.longValue());

}

@Override

public String toString() {

return "UserScrubGeoFilter";

}

@Override

public boolean equals(Object obj) {

if (!(obj instanceof UserScrubGeoMap)) {

return false;

}

UserScrubGeoFilter filter = UserScrubGeoFilter.class.cast(obj);

// filters are considered equal as long as they are using the same UserScrubGeoMap

return Objects.equals(userScrubGeoMap, filter.userScrubGeoMap);

}

@Override

public int hashCode() {

return userScrubGeoMap == null ? 0 : userScrubGeoMap.hashCode();

}

}