package com.twitter.search.earlybird\_root;

import java.util.List;

import javax.inject.Inject;

import com.twitter.finagle.Filter;

import com.twitter.finagle.Service;

import com.twitter.search.common.schema.earlybird.EarlybirdCluster;

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

import com.twitter.search.earlybird\_root.common.EarlybirdFeatureSchemaMerger;

import com.twitter.search.earlybird\_root.common.EarlybirdRequestContext;

import com.twitter.search.earlybird\_root.mergers.EarlybirdResponseMerger;

import com.twitter.search.earlybird\_root.mergers.TierResponseAccumulator;

import com.twitter.util.Function;

import com.twitter.util.Future;

/\*\*

\* Filter used to merge results from multiple tiers

\*/

public class MultiTierResultsMergeFilter extends

Filter<EarlybirdRequestContext, EarlybirdResponse,

EarlybirdRequestContext, List<Future<EarlybirdResponse>>> {

private final EarlybirdFeatureSchemaMerger featureSchemaMerger;

@Inject

public MultiTierResultsMergeFilter(EarlybirdFeatureSchemaMerger featureSchemaMerger) {

this.featureSchemaMerger = featureSchemaMerger;

}

@Override

public Future<EarlybirdResponse> apply(

final EarlybirdRequestContext request,

Service<EarlybirdRequestContext, List<Future<EarlybirdResponse>>> service) {

return service.apply(request).flatMap(Function.func(responses -> merge(request, responses)));

}

private Future<EarlybirdResponse> merge(

EarlybirdRequestContext requestContext,

List<Future<EarlybirdResponse>> responses) {

// For multi-tier response merging, the number of partitions do not have meaning because

// the response is not uniformly partitioned anymore. We pass Integer.MAX\_VALUE for stats

// counting purpose.

EarlybirdResponseMerger merger = EarlybirdResponseMerger.getResponseMerger(

requestContext,

responses,

new TierResponseAccumulator(),

EarlybirdCluster.FULL\_ARCHIVE,

featureSchemaMerger,

Integer.MAX\_VALUE);

return merger.merge();

}

}