package com.twitter.search.earlybird\_root;

import javax.inject.Inject;

import javax.inject.Singleton;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import com.twitter.search.common.decider.SearchDecider;

/\*\*

\* Controls fractions of requests that are sent out to each tier.

\*/

@Singleton

public class EarlybirdTierThrottleDeciders {

private static final Logger LOG =

LoggerFactory.getLogger(EarlybirdTierThrottleDeciders.class);

private static final String TIER\_THROTTLE\_DECIDER\_KEY\_FORMAT =

"percentage\_to\_hit\_cluster\_%s\_tier\_%s";

private final SearchDecider decider;

/\*\*

\* Construct a decider using the singleton decider object injected by Guice for the

\* specified tier.

\* See {@link com.twitter.search.common.root.SearchRootModule#provideDecider()}

\*/

@Inject

public EarlybirdTierThrottleDeciders(SearchDecider decider) {

this.decider = decider;

}

/\*\*

\* Return the throttle decider key for the specified tier.

\*/

public String getTierThrottleDeciderKey(String clusterName, String tierName) {

String deciderKey = String.format(TIER\_THROTTLE\_DECIDER\_KEY\_FORMAT, clusterName, tierName);

if (!decider.getDecider().feature(deciderKey).exists()) {

LOG.warn("Decider key {} not found. Will always return unavailable.", deciderKey);

}

return deciderKey;

}

/\*\*

\* Check whether a request should be sent to the specified tier.

\*/

public Boolean shouldSendRequestToTier(final String tierDarkReadDeciderKey) {

return decider.isAvailable(tierDarkReadDeciderKey);

}

}