package com.twitter.search.earlybird;

import scala.Option;

import com.google.common.annotations.VisibleForTesting;

import com.twitter.decider.Decider;

public final class RecentTweetRestriction {

private static final String RECENT\_TWEETS\_THRESHOLD = "recent\_tweets\_threshold";

private static final String QUERY\_CACHE\_UNTIL\_TIME = "query\_cache\_until\_time";

@VisibleForTesting

public static final int DEFAULT\_RECENT\_TWEET\_SECONDS = 15;

private RecentTweetRestriction() {

}

/\*\*

\* Returns the point in time (in seconds past the unix epoch) before which all tweets will be

\* completely indexed. This is required by some clients, because they rely on Earlybird monotonically

\* indexing tweets by ID and that tweets are completely indexed when they see them.

\*

\* @param lastTime The time at which the most recent tweet was indexed, in seconds since the unix

\* epoch.

\*/

public static int recentTweetsUntilTime(Decider decider, int lastTime) {

return untilTimeSeconds(decider, lastTime, RECENT\_TWEETS\_THRESHOLD);

}

/\*\*

\* Returns the point in time (in seconds past the unix epoch) before which all tweets will be

\* completely indexed. This is required by some clients, because they rely on Earlybird monotonically

\* indexing tweets by ID and that tweets are completely indexed when they see them.

\*

\* @param lastTime The time at which the most recent tweet was indexed, in seconds since the unix

\* epoch.

\*/

public static int queryCacheUntilTime(Decider decider, int lastTime) {

return untilTimeSeconds(decider, lastTime, QUERY\_CACHE\_UNTIL\_TIME);

}

private static int untilTimeSeconds(Decider decider, int lastTime, String deciderKey) {

int recentTweetSeconds = getRecentTweetSeconds(decider, deciderKey);

if (recentTweetSeconds == 0) {

return 0;

}

return lastTime - recentTweetSeconds;

}

private static int getRecentTweetSeconds(Decider decider, String deciderKey) {

Option<Object> deciderValue = decider.getAvailability(deciderKey);

if (deciderValue.isDefined()) {

return (int) deciderValue.get();

}

return DEFAULT\_RECENT\_TWEET\_SECONDS;

}

}