package com.twitter.unified\_user\_actions.enricher.partitioner

import com.twitter.unified\_user\_actions.enricher.internal.thriftscala.EnrichmentEnvelop

import com.twitter.unified\_user\_actions.enricher.internal.thriftscala.EnrichmentIdType

import com.twitter.unified\_user\_actions.enricher.internal.thriftscala.EnrichmentInstruction

import com.twitter.unified\_user\_actions.enricher.internal.thriftscala.EnrichmentInstruction.NotificationTweetEnrichment

import com.twitter.unified\_user\_actions.enricher.internal.thriftscala.EnrichmentInstruction.TweetEnrichment

import com.twitter.unified\_user\_actions.enricher.internal.thriftscala.EnrichmentKey

import com.twitter.unified\_user\_actions.enricher.partitioner.DefaultPartitioner.NullKey

import com.twitter.unified\_user\_actions.thriftscala.Item

import com.twitter.unified\_user\_actions.thriftscala.NotificationContent

object DefaultPartitioner {

val NullKey: Option[EnrichmentKey] = None

}

class DefaultPartitioner extends Partitioner {

override def repartition(

instruction: EnrichmentInstruction,

envelop: EnrichmentEnvelop

): Option[EnrichmentKey] = {

(instruction, envelop.uua.item) match {

case (TweetEnrichment, Item.TweetInfo(info)) =>

Some(EnrichmentKey(EnrichmentIdType.TweetId, info.actionTweetId))

case (NotificationTweetEnrichment, Item.NotificationInfo(info)) =>

info.content match {

case NotificationContent.TweetNotification(content) =>

Some(EnrichmentKey(EnrichmentIdType.TweetId, content.tweetId))

case NotificationContent.MultiTweetNotification(content) =>

// we scarify on cache performance in this case since only a small % of

// notification content will be multi-tweet types.

Some(EnrichmentKey(EnrichmentIdType.TweetId, content.tweetIds.head))

case \_ => NullKey

}

case \_ => NullKey

}

}

}