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

import com.twitter.inject.Test

import com.twitter.unified\_user\_actions.enricher.EnricherFixture

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 org.scalatest.prop.TableDrivenPropertyChecks

class DefaultPartitionerTest extends Test with TableDrivenPropertyChecks {

test("default partitioner should work") {

new EnricherFixture {

val partitioner = new DefaultPartitioner

val instructions = Table(

("instruction", "envelop", "expected"),

// tweet info

(

TweetEnrichment,

EnrichmentEnvelop(1L, mkUUATweetEvent(123L), tweetInfoEnrichmentPlan),

Some(EnrichmentKey(EnrichmentIdType.TweetId, 123L))),

// notification tweet info

(

NotificationTweetEnrichment,

EnrichmentEnvelop(2L, mkUUATweetNotificationEvent(234L), tweetNotificationEnrichmentPlan),

Some(EnrichmentKey(EnrichmentIdType.TweetId, 234L))),

// notification with multiple tweet info

(

NotificationTweetEnrichment,

EnrichmentEnvelop(

3L,

mkUUAMultiTweetNotificationEvent(22L, 33L),

tweetNotificationEnrichmentPlan),

Some(EnrichmentKey(EnrichmentIdType.TweetId, 22L))

) // only the first tweet id is partitioned

)

forEvery(instructions) {

(

instruction: EnrichmentInstruction,

envelop: EnrichmentEnvelop,

expected: Some[EnrichmentKey]

) =>

val actual = partitioner.repartition(instruction, envelop)

assert(expected === actual)

}

}

}

test("unsupported events shouldn't be partitioned") {

new EnricherFixture {

val partitioner = new DefaultPartitioner

val instructions = Table(

("instruction", "envelop", "expected"),

// profile uua event

(

TweetEnrichment,

EnrichmentEnvelop(1L, mkUUAProfileEvent(111L), tweetInfoEnrichmentPlan),

NullKey),

// unknown notification (not a tweet)

(

NotificationTweetEnrichment,

EnrichmentEnvelop(1L, mkUUATweetNotificationUnknownEvent(), tweetInfoEnrichmentPlan),

NullKey),

)

forEvery(instructions) {

(

instruction: EnrichmentInstruction,

envelop: EnrichmentEnvelop,

expected: Option[EnrichmentKey]

) =>

val actual = partitioner.repartition(instruction, envelop)

assert(expected === actual)

}

}

}

}