package com.twitter.home\_mixer.product.scored\_tweets.feature\_hydrator.real\_time\_aggregates

import com.twitter.product\_mixer.core.feature.Feature

import com.twitter.product\_mixer.core.feature.datarecord.DataRecordInAFeature

import com.twitter.product\_mixer.core.feature.featuremap.FeatureMap

import com.twitter.product\_mixer.core.feature.featuremap.FeatureMapBuilder

import com.twitter.product\_mixer.core.functional\_component.feature\_hydrator.QueryFeatureHydrator

import com.twitter.product\_mixer.core.pipeline.PipelineQuery

import com.twitter.product\_mixer.core.util.OffloadFuturePools

import com.twitter.stitch.Stitch

trait BaseRealTimeAggregateQueryFeatureHydrator[K]

extends QueryFeatureHydrator[PipelineQuery]

with BaseRealtimeAggregateHydrator[K] {

val outputFeature: DataRecordInAFeature[PipelineQuery]

override def features: Set[Feature[\_, \_]] = Set(outputFeature)

override lazy val statScope: String = identifier.toString

def keysFromQueryAndCandidates(

query: PipelineQuery

): Option[K]

override def hydrate(

query: PipelineQuery

): Stitch[FeatureMap] = OffloadFuturePools.offloadFuture {

val possiblyKeys = keysFromQueryAndCandidates(query)

fetchAndConstructDataRecords(Seq(possiblyKeys)).map { dataRecords =>

FeatureMapBuilder()

.add(outputFeature, dataRecords.head)

.build()

}

}

}