package com.twitter.interaction\_graph.scio.agg\_direct\_interactions

import com.spotify.scio.ScioContext

import com.twitter.beam.io.dal.DAL

import com.twitter.beam.io.dal.DAL.DiskFormat

import com.twitter.beam.io.fs.multiformat.PathLayout

import com.twitter.beam.io.fs.multiformat.WriteOptions

import com.twitter.beam.job.ServiceIdentifierOptions

import com.twitter.interaction\_graph.scio.common.UserUtil

import com.twitter.interaction\_graph.thriftscala.Edge

import com.twitter.interaction\_graph.thriftscala.Vertex

import com.twitter.scio\_internal.job.ScioBeamJob

import com.twitter.statebird.v2.thriftscala.Environment

import org.joda.time.Interval

object InteractionGraphAggDirectInteractionsJob

extends ScioBeamJob[InteractionGraphAggDirectInteractionsOption] {

override protected def configurePipeline(

scioContext: ScioContext,

pipelineOptions: InteractionGraphAggDirectInteractionsOption

): Unit = {

@transient

implicit lazy val sc: ScioContext = scioContext

implicit lazy val dateInterval: Interval = pipelineOptions.interval

val dalEnvironment: String = pipelineOptions

.as(classOf[ServiceIdentifierOptions])

.getEnvironment()

val dalWriteEnvironment = if (pipelineOptions.getDALWriteEnvironment != null) {

pipelineOptions.getDALWriteEnvironment

} else {

dalEnvironment

}

val source = InteractionGraphAggDirectInteractionsSource(pipelineOptions)

val rawUsers = source.readCombinedUsers()

val safeUsers = UserUtil.getValidUsers(rawUsers)

val rawFavorites = source.readFavorites(dateInterval)

val rawPhotoTags = source.readPhotoTags(dateInterval)

val tweetSource = source.readTweetSource(dateInterval)

val (vertex, edges) = InteractionGraphAggDirectInteractionsUtil.process(

rawFavorites,

tweetSource,

rawPhotoTags,

safeUsers

)

vertex.saveAsCustomOutput(

"Write Vertex Records",

DAL.write[Vertex](

InteractionGraphAggDirectInteractionsVertexDailyScalaDataset,

PathLayout.DailyPath(

pipelineOptions.getOutputPath + "/aggregated\_direct\_interactions\_vertex\_daily"),

dateInterval,

DiskFormat.Parquet,

Environment.valueOf(dalWriteEnvironment),

writeOption =

WriteOptions(numOfShards = Some((pipelineOptions.getNumberOfShards / 8.0).ceil.toInt))

)

)

edges.saveAsCustomOutput(

"Write Edge Records",

DAL.write[Edge](

InteractionGraphAggDirectInteractionsEdgeDailyScalaDataset,

PathLayout.DailyPath(

pipelineOptions.getOutputPath + "/aggregated\_direct\_interactions\_edge\_daily"),

dateInterval,

DiskFormat.Parquet,

Environment.valueOf(dalWriteEnvironment),

writeOption = WriteOptions(numOfShards = Some(pipelineOptions.getNumberOfShards))

)

)

}

}