package com.twitter.interaction\_graph.scio.agg\_address\_book

import com.spotify.scio.ScioContext

import com.spotify.scio.values.SCollection

import com.twitter.addressbook.matches.thriftscala.UserMatchesRecord

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

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

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

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

import com.twitter.beam.job.ServiceIdentifierOptions

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

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

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

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

import java.time.Instant

import org.joda.time.Interval

object InteractionGraphAddressBookJob extends ScioBeamJob[InteractionGraphAddressBookOption] {

override protected def configurePipeline(

scioContext: ScioContext,

pipelineOptions: InteractionGraphAddressBookOption

): Unit = {

@transient

implicit lazy val sc: ScioContext = scioContext

implicit lazy val dateInterval: Interval = pipelineOptions.interval

implicit lazy val addressBookCounters: InteractionGraphAddressBookCountersTrait =

InteractionGraphAddressBookCounters

val interactionGraphAddressBookSource = InteractionGraphAddressBookSource(pipelineOptions)

val addressBook: SCollection[UserMatchesRecord] =

interactionGraphAddressBookSource.readSimpleUserMatches(

dateInterval.withStart(dateInterval.getStart.minusDays(3))

)

val (vertex, edges) = InteractionGraphAddressBookUtil.process(addressBook)

val dalEnvironment: String = pipelineOptions

.as(classOf[ServiceIdentifierOptions])

.getEnvironment()

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

pipelineOptions.getDALWriteEnvironment

} else {

dalEnvironment

}

vertex.saveAsCustomOutput(

"Write Vertex Records",

DAL.writeSnapshot[Vertex](

InteractionGraphAggAddressBookVertexSnapshotScalaDataset,

PathLayout.DailyPath(pipelineOptions.getOutputPath + "/address\_book\_vertex\_daily"),

Instant.ofEpochMilli(dateInterval.getEndMillis),

DiskFormat.Parquet,

Environment.valueOf(dalWriteEnvironment),

writeOption =

WriteOptions(numOfShards = Some((pipelineOptions.getNumberOfShards / 16.0).ceil.toInt))

)

)

edges.saveAsCustomOutput(

"Write Edge Records",

DAL.writeSnapshot[Edge](

InteractionGraphAggAddressBookEdgeSnapshotScalaDataset,

PathLayout.DailyPath(pipelineOptions.getOutputPath + "/address\_book\_edge\_daily"),

Instant.ofEpochMilli(dateInterval.getEndMillis),

DiskFormat.Parquet,

Environment.valueOf(dalWriteEnvironment),

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

)

)

}

}