package com.twitter.graph\_feature\_service.scalding.adhoc

import com.twitter.bijection.Injection

import com.twitter.frigate.common.constdb\_util.Injections

import com.twitter.ml.api.Feature.Discrete

import com.twitter.ml.api.{DailySuffixFeatureSource, DataSetPipe, RichDataRecord}

import com.twitter.scalding.\_

import com.twitter.scalding\_internal.job.TwitterExecutionApp

import java.nio.ByteBuffer

import java.util.TimeZone

object RandomRequestGenerationJob {

implicit val timeZone: TimeZone = DateOps.UTC

implicit val dateParser: DateParser = DateParser.default

val timelineRecapDataSetPath: String =

"/atla/proc2/user/timelines/processed/suggests/recap/data\_records"

val USER\_ID = new Discrete("meta.user\_id")

val AUTHOR\_ID = new Discrete("meta.author\_id")

val timelineRecapOutPutPath: String = "/user/cassowary/gfs/adhoc/timeline\_data"

implicit val inj: Injection[Long, ByteBuffer] = Injections.long2Varint

def run(

dataSetPath: String,

outPutPath: String,

numOfPairsToTake: Int

)(

implicit dateRange: DateRange,

uniqueID: UniqueID

): Execution[Unit] = {

val NumUserAuthorPairs = Stat("NumUserAuthorPairs")

val dataSet: DataSetPipe = DailySuffixFeatureSource(dataSetPath).read

val userAuthorPairs: TypedPipe[(Long, Long)] = dataSet.records.map { record =>

val richRecord = new RichDataRecord(record, dataSet.featureContext)

val userId = richRecord.getFeatureValue(USER\_ID)

val authorId = richRecord.getFeatureValue(AUTHOR\_ID)

NumUserAuthorPairs.inc()

(userId, authorId)

}

userAuthorPairs

.limit(numOfPairsToTake)

.writeExecution(

TypedTsv[(Long, Long)](outPutPath)

)

}

}

/\*\*

\* ./bazel bundle graph-feature-service/src/main/scalding/com/twitter/graph\_feature\_service/scalding/adhoc:all

\*

\* oscar hdfs --screen --user cassowary --tee gfs\_log --bundle gfs\_random\_request-adhoc \

--tool com.twitter.graph\_feature\_service.scalding.adhoc.RandomRequestGenerationApp \

-- --date 2018-08-11 \

--input /atla/proc2/user/timelines/processed/suggests/recap/data\_records \

--output /user/cassowary/gfs/adhoc/timeline\_data

\*/

object RandomRequestGenerationApp extends TwitterExecutionApp {

import RandomRequestGenerationJob.\_

override def job: Execution[Unit] = Execution.withId { implicit uniqueId =>

Execution.getArgs.flatMap { args: Args =>

implicit val dateRange: DateRange = DateRange.parse(args.list("date"))(timeZone, dateParser)

run(

args.optional("input").getOrElse(timelineRecapDataSetPath),

args.optional("output").getOrElse(timelineRecapOutPutPath),

args.int("num\_pairs", 3000)

)

}

}

}