package com.twitter.simclusters\_v2.summingbird.stores

import com.twitter.bijection.Injection

import com.twitter.bijection.scrooge.CompactScalaCodec

import com.twitter.simclusters\_v2.thriftscala.{ClustersUserIsKnownFor, ModelVersion}

import com.twitter.storage.client.manhattan.kv.ManhattanKVClientMtlsParams

import com.twitter.storehaus.ReadableStore

import com.twitter.storehaus\_internal.manhattan.{Athena, ManhattanRO, ManhattanROConfig}

import com.twitter.storehaus\_internal.util.{ApplicationID, DatasetName, HDFSPath}

import com.twitter.util.Future

object UserKnownForReadableStore {

private val dataSetNameDec11 = "simclusters\_v2\_known\_for\_20m\_145k\_dec11"

private val dataSetNameUpdated = "simclusters\_v2\_known\_for\_20m\_145k\_updated"

private val dataSetName2020 = "simclusters\_v2\_known\_for\_20m\_145k\_2020"

private def buildForModelVersion(

appId: String,

storeName: String,

mhMtlsParams: ManhattanKVClientMtlsParams

): ReadableStore[Long, ClustersUserIsKnownFor] = {

implicit val keyInjection: Injection[Long, Array[Byte]] = Injection.long2BigEndian

implicit val knownForCodec: Injection[ClustersUserIsKnownFor, Array[Byte]] =

CompactScalaCodec(ClustersUserIsKnownFor)

ManhattanRO.getReadableStoreWithMtls[Long, ClustersUserIsKnownFor](

ManhattanROConfig(

HDFSPath(""), // not needed

ApplicationID(appId),

DatasetName(storeName),

Athena

),

mhMtlsParams

)

}

def get(appId: String, mhMtlsParams: ManhattanKVClientMtlsParams): UserKnownForReadableStore = {

val dec11Store = buildForModelVersion(appId, dataSetNameDec11, mhMtlsParams)

val updatedStore = buildForModelVersion(appId, dataSetNameUpdated, mhMtlsParams)

val version2020Store = buildForModelVersion(appId, dataSetName2020, mhMtlsParams)

UserKnownForReadableStore(dec11Store, updatedStore, version2020Store)

}

def getDefaultStore(mhMtlsParams: ManhattanKVClientMtlsParams): UserKnownForReadableStore =

get("simclusters\_v2", mhMtlsParams)

}

case class Query(userId: Long, modelVersion: ModelVersion = ModelVersion.Model20m145kUpdated)

/\*\*

\* Mainly used in debuggers to fetch the top knownFor clusters across different model versions

\*/

case class UserKnownForReadableStore(

knownForStoreDec11: ReadableStore[Long, ClustersUserIsKnownFor],

knownForStoreUpdated: ReadableStore[Long, ClustersUserIsKnownFor],

knownForStore2020: ReadableStore[Long, ClustersUserIsKnownFor])

extends ReadableStore[Query, ClustersUserIsKnownFor] {

override def get(query: Query): Future[Option[ClustersUserIsKnownFor]] = {

query.modelVersion match {

case ModelVersion.Model20m145kDec11 =>

knownForStoreDec11.get(query.userId)

case ModelVersion.Model20m145kUpdated =>

knownForStoreUpdated.get(query.userId)

case ModelVersion.Model20m145k2020 =>

knownForStore2020.get(query.userId)

case c =>

throw new IllegalArgumentException(

s"Never heard of $c before! Is this a new model version?")

}

}

}