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

import com.twitter.finagle.mtls.authentication.ServiceIdentifier

import com.twitter.frigate.common.store.strato.StratoStore

import com.twitter.simclusters\_v2.summingbird.common.Implicits.clustersWithScoreMonoid

import com.twitter.simclusters\_v2.summingbird.common.Implicits.clustersWithScoresCodec

import com.twitter.storehaus.algebra.MergeableStore

import com.twitter.simclusters\_v2.summingbird.common.ClientConfigs

import com.twitter.simclusters\_v2.summingbird.common.Implicits

import com.twitter.simclusters\_v2.thriftscala.ClustersWithScores

import com.twitter.simclusters\_v2.thriftscala.FullClusterIdBucket

import com.twitter.simclusters\_v2.thriftscala.MultiModelClustersWithScores

import com.twitter.simclusters\_v2.thriftscala.SimClusterEntity

import com.twitter.storehaus.Store

import com.twitter.storehaus\_internal.memcache.Memcache

import com.twitter.strato.client.Client

import com.twitter.summingbird.batch.BatchID

import com.twitter.summingbird\_internal.bijection.BatchPairImplicits

import com.twitter.util.Future

import com.twitter.strato.thrift.ScroogeConvImplicits.\_

object EntityClusterScoreReadableStore {

private[simclusters\_v2] final lazy val onlineMergeableStore: (

String,

ServiceIdentifier

) => MergeableStore[

((SimClusterEntity, FullClusterIdBucket), BatchID),

ClustersWithScores

] = { (path: String, serviceIdentifier: ServiceIdentifier) =>

Memcache

.getMemcacheStore[((SimClusterEntity, FullClusterIdBucket), BatchID), ClustersWithScores](

ClientConfigs.entityClusterScoreMemcacheConfig(path, serviceIdentifier)

)(

BatchPairImplicits.keyInjection[(SimClusterEntity, FullClusterIdBucket)](

Implicits.entityWithClusterInjection

),

clustersWithScoresCodec,

clustersWithScoreMonoid

)

}

}

object MultiModelEntityClusterScoreReadableStore {

private[simclusters\_v2] def MultiModelEntityClusterScoreReadableStore(

stratoClient: Client,

column: String

): Store[EntityClusterId, MultiModelClustersWithScores] = {

StratoStore

.withUnitView[(SimClusterEntity, Int), MultiModelClustersWithScores](stratoClient, column)

.composeKeyMapping(\_.toTuple)

}

case class EntityClusterId(

simClusterEntity: SimClusterEntity,

clusterIdBucket: Int) {

lazy val toTuple: (SimClusterEntity, Int) =

(simClusterEntity, clusterIdBucket)

}

}