package com.twitter.cr\_mixer.module

import com.google.inject.Provides

import com.google.inject.Singleton

import com.google.inject.name.Named

import com.twitter.conversions.DurationOps.\_

import com.twitter.cr\_mixer.model.ModuleNames

import com.twitter.cr\_mixer.similarity\_engine.SimilarityEngine.keyHasher

import com.twitter.finagle.memcached.{Client => MemcachedClient}

import com.twitter.finagle.stats.StatsReceiver

import com.twitter.hermit.store.common.ObservedCachedReadableStore

import com.twitter.hermit.store.common.ObservedMemcachedReadableStore

import com.twitter.hermit.store.common.ObservedReadableStore

import com.twitter.inject.TwitterModule

import com.twitter.relevance\_platform.common.injection.LZ4Injection

import com.twitter.relevance\_platform.common.injection.SeqObjectInjection

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

import com.twitter.storehaus.ReadableStore

import com.twitter.strato.client.Client

import com.twitter.topic\_recos.stores.CertoTopicTopKTweetsStore

import com.twitter.topic\_recos.thriftscala.TweetWithScores

object CertoStratoStoreModule extends TwitterModule {

@Provides

@Singleton

@Named(ModuleNames.CertoStratoStoreName)

def providesCertoStratoStore(

@Named(ModuleNames.UnifiedCache) crMixerUnifiedCacheClient: MemcachedClient,

stratoClient: Client,

statsReceiver: StatsReceiver

): ReadableStore[TopicId, Seq[TweetWithScores]] = {

val certoStore = ObservedReadableStore(CertoTopicTopKTweetsStore.prodStore(stratoClient))(

statsReceiver.scope(ModuleNames.CertoStratoStoreName)).mapValues { topKTweetsWithScores =>

topKTweetsWithScores.topTweetsByFollowerL2NormalizedCosineSimilarityScore

}

val memCachedStore = ObservedMemcachedReadableStore

.fromCacheClient(

backingStore = certoStore,

cacheClient = crMixerUnifiedCacheClient,

ttl = 10.minutes

)(

valueInjection = LZ4Injection.compose(SeqObjectInjection[TweetWithScores]()),

statsReceiver = statsReceiver.scope("memcached\_certo\_store"),

keyToString = { k => s"certo:${keyHasher.hashKey(k.toString.getBytes)}" }

)

ObservedCachedReadableStore.from[TopicId, Seq[TweetWithScores]](

memCachedStore,

ttl = 5.minutes,

maxKeys = 100000, // ~150MB max

cacheName = "certo\_in\_memory\_cache",

windowSize = 10000L

)(statsReceiver.scope("certo\_in\_memory\_cache"))

}

}