package com.twitter.simclustersann.modules

import com.google.inject.Provides

import com.twitter.decider.Decider

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

import com.twitter.finagle.stats.StatsReceiver

import com.twitter.inject.TwitterModule

import com.twitter.representation\_manager.StoreBuilder

import com.twitter.representation\_manager.config.{

DefaultClientConfig => RepresentationManagerDefaultClientConfig

}

import com.twitter.representation\_manager.thriftscala.SimClustersEmbeddingView

import com.twitter.simclusters\_v2.common.SimClustersEmbedding

import com.twitter.simclusters\_v2.stores.SimClustersEmbeddingStore

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

import com.twitter.simclusters\_v2.thriftscala.EmbeddingType.\_

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

import com.twitter.simclusters\_v2.thriftscala.ModelVersion.\_

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

import com.twitter.storehaus.ReadableStore

import com.twitter.strato.client.{Client => StratoClient}

import javax.inject.Singleton

object EmbeddingStoreModule extends TwitterModule {

val TweetEmbeddings: Set[SimClustersEmbeddingView] = Set(

SimClustersEmbeddingView(LogFavLongestL2EmbeddingTweet, Model20m145kUpdated),

SimClustersEmbeddingView(LogFavLongestL2EmbeddingTweet, Model20m145k2020)

)

val UserEmbeddings: Set[SimClustersEmbeddingView] = Set(

// KnownFor

SimClustersEmbeddingView(FavBasedProducer, Model20m145kUpdated),

SimClustersEmbeddingView(FavBasedProducer, Model20m145k2020),

SimClustersEmbeddingView(FollowBasedProducer, Model20m145k2020),

SimClustersEmbeddingView(AggregatableLogFavBasedProducer, Model20m145k2020),

// InterestedIn

SimClustersEmbeddingView(UnfilteredUserInterestedIn, Model20m145k2020),

SimClustersEmbeddingView(

LogFavBasedUserInterestedMaxpoolingAddressBookFromIIAPE,

Model20m145k2020),

SimClustersEmbeddingView(

LogFavBasedUserInterestedAverageAddressBookFromIIAPE,

Model20m145k2020),

SimClustersEmbeddingView(

LogFavBasedUserInterestedBooktypeMaxpoolingAddressBookFromIIAPE,

Model20m145k2020),

SimClustersEmbeddingView(

LogFavBasedUserInterestedLargestDimMaxpoolingAddressBookFromIIAPE,

Model20m145k2020),

SimClustersEmbeddingView(

LogFavBasedUserInterestedLouvainMaxpoolingAddressBookFromIIAPE,

Model20m145k2020),

SimClustersEmbeddingView(

LogFavBasedUserInterestedConnectedMaxpoolingAddressBookFromIIAPE,

Model20m145k2020),

SimClustersEmbeddingView(UserNextInterestedIn, Model20m145k2020),

SimClustersEmbeddingView(LogFavBasedUserInterestedInFromAPE, Model20m145k2020)

)

@Singleton

@Provides

def providesEmbeddingStore(

stratoClient: StratoClient,

memCachedClient: MemcachedClient,

decider: Decider,

stats: StatsReceiver

): ReadableStore[SimClustersEmbeddingId, SimClustersEmbedding] = {

val rmsStoreBuilder = new StoreBuilder(

clientConfig = RepresentationManagerDefaultClientConfig,

stratoClient = stratoClient,

memCachedClient = memCachedClient,

globalStats = stats,

)

val underlyingStores: Map[

(EmbeddingType, ModelVersion),

ReadableStore[SimClustersEmbeddingId, SimClustersEmbedding]

] = {

val tweetEmbeddingStores: Map[

(EmbeddingType, ModelVersion),

ReadableStore[SimClustersEmbeddingId, SimClustersEmbedding]

] = TweetEmbeddings

.map(embeddingView =>

(

(embeddingView.embeddingType, embeddingView.modelVersion),

rmsStoreBuilder

.buildSimclustersTweetEmbeddingStoreWithEmbeddingIdAsKey(embeddingView))).toMap

val userEmbeddingStores: Map[

(EmbeddingType, ModelVersion),

ReadableStore[SimClustersEmbeddingId, SimClustersEmbedding]

] = UserEmbeddings

.map(embeddingView =>

(

(embeddingView.embeddingType, embeddingView.modelVersion),

rmsStoreBuilder

.buildSimclustersUserEmbeddingStoreWithEmbeddingIdAsKey(embeddingView))).toMap

tweetEmbeddingStores ++ userEmbeddingStores

}

SimClustersEmbeddingStore.buildWithDecider(

underlyingStores = underlyingStores,

decider = decider,

statsReceiver = stats

)

}

}