package com.twitter.tweetypie.storage

import com.twitter.finagle.stats.StatsReceiver

import com.twitter.stitch.Stitch

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

import com.twitter.tweetypie.storage.TweetUtils.collectWithRateLimitCheck

import com.twitter.tweetypie.storage\_internal.thriftscala.StoredTweet

import com.twitter.tweetypie.thriftscala.Tweet

import com.twitter.util.Time

object AddTweetHandler {

private[storage] type InternalAddTweet = (

Tweet,

ManhattanOperations.Insert,

Scribe,

StatsReceiver,

Time

) => Stitch[Unit]

def apply(

insert: ManhattanOperations.Insert,

scribe: Scribe,

stats: StatsReceiver

): TweetStorageClient.AddTweet =

tweet => doAddTweet(tweet, insert, scribe, stats, Time.now)

def makeRecords(

storedTweet: StoredTweet,

timestamp: Time

): Seq[TweetManhattanRecord] = {

val core = CoreFieldsCodec.fromTweet(storedTweet)

val packedCoreFieldsBlob = CoreFieldsCodec.toTFieldBlob(core)

val coreRecord =

TweetManhattanRecord(

TweetKey.coreFieldsKey(storedTweet.id),

ManhattanValue(TFieldBlobCodec.toByteBuffer(packedCoreFieldsBlob), Some(timestamp))

)

val otherFieldIds =

TweetFields.nonCoreInternalFields ++ TweetFields.getAdditionalFieldIds(storedTweet)

val otherFields =

storedTweet

.getFieldBlobs(otherFieldIds)

.map {

case (fieldId, tFieldBlob) =>

TweetManhattanRecord(

TweetKey.fieldKey(storedTweet.id, fieldId),

ManhattanValue(TFieldBlobCodec.toByteBuffer(tFieldBlob), Some(timestamp))

)

}

.toSeq

otherFields :+ coreRecord

}

private[storage] val doAddTweet: InternalAddTweet = (

tweet: Tweet,

insert: ManhattanOperations.Insert,

scribe: Scribe,

stats: StatsReceiver,

timestamp: Time

) => {

assert(tweet.coreData.isDefined, s"Tweet ${tweet.id} is missing coreData: $tweet")

val storedTweet = StorageConversions.toStoredTweet(tweet)

val records = makeRecords(storedTweet, timestamp)

val inserts = records.map(insert)

val insertsWithRateLimitCheck =

Stitch.collect(inserts.map(\_.liftToTry)).map(collectWithRateLimitCheck).lowerFromTry

Stats.updatePerFieldQpsCounters(

"addTweet",

TweetFields.getAdditionalFieldIds(storedTweet),

1,

stats

)

insertsWithRateLimitCheck.unit.onSuccess { \_ => scribe.logAdded(storedTweet) }

}

}