package com.twitter.search.earlybird.archive;

import java.io.IOException;

import java.util.Date;

import com.google.common.base.Predicate;

import com.google.common.base.Predicates;

import com.twitter.search.common.partitioning.base.Segment;

import com.twitter.search.common.partitioning.base.TimeSlice;

import com.twitter.search.common.schema.thriftjava.ThriftIndexingEvent;

import com.twitter.search.common.util.io.recordreader.RecordReader;

import com.twitter.search.earlybird.archive.ArchiveTimeSlicer.ArchiveTimeSlice;

import com.twitter.search.earlybird.document.DocumentFactory;

import com.twitter.search.earlybird.document.TweetDocument;

public class ArchiveSegment extends Segment {

private final ArchiveTimeSlice archiveTimeSlice;

public static final Predicate<Date> MATCH\_ALL\_DATE\_PREDICATE = input -> true;

// Constructor used for indexing an archive segment

public ArchiveSegment(ArchiveTimeSlice archiveTimeSlice,

int hashPartitionID,

int maxSegmentSize) {

super(new TimeSlice(archiveTimeSlice.getMinStatusID(hashPartitionID),

maxSegmentSize, hashPartitionID,

archiveTimeSlice.getNumHashPartitions()),

archiveTimeSlice.getEndDate().getTime());

this.archiveTimeSlice = archiveTimeSlice;

}

/\*\*

\* Constructor used for loading a flushed segment. Only be used by SegmentBuilder; Earlybird

\* does not use this.

\*/

ArchiveSegment(long timeSliceId,

int maxSegmentSize,

int partitions,

int hashPartitionID,

Date dataEndDate) {

super(new TimeSlice(timeSliceId, maxSegmentSize, hashPartitionID, partitions),

dataEndDate.getTime());

// No archive timeslice is needed for loading.

this.archiveTimeSlice = null;

}

/\*\*

\* Returns the tweets reader for this segment.

\*

\* @param documentFactory The factory that converts ThriftDocuments to Lucene documents.

\*/

public RecordReader<TweetDocument> getStatusRecordReader(

DocumentFactory<ThriftIndexingEvent> documentFactory) throws IOException {

return getStatusRecordReader(documentFactory, Predicates.<Date>alwaysTrue());

}

/\*\*

\* Returns the tweets reader for this segment.

\*

\* @param documentFactory The factory that converts ThriftDocuments to Lucene documents.

\* @param filter A predicate that filters tweets based on the date they were created on.

\*/

public RecordReader<TweetDocument> getStatusRecordReader(

DocumentFactory<ThriftIndexingEvent> documentFactory,

Predicate<Date> filter) throws IOException {

if (archiveTimeSlice != null) {

return archiveTimeSlice.getStatusReader(this, documentFactory, filter);

} else {

throw new IllegalStateException("ArchiveSegment has no associated ArchiveTimeslice."

+ "This ArchiveSegment can only be used for loading flushed segments.");

}

}

public Date getDataEndDate() {

return archiveTimeSlice == null

? new Date(getDataEndDateInclusiveMillis()) : archiveTimeSlice.getEndDate();

}

public ArchiveTimeSlice getArchiveTimeSlice() {

return archiveTimeSlice;

}

@Override

public String toString() {

return super.toString() + " " + archiveTimeSlice.getDescription();

}

}