package com.twitter.search.core.earlybird.index.column;

import java.io.IOException;

import com.twitter.search.common.util.io.flushable.DataDeserializer;

import com.twitter.search.common.util.io.flushable.DataSerializer;

import com.twitter.search.common.util.io.flushable.FlushInfo;

import com.twitter.search.common.util.io.flushable.Flushable;

import com.twitter.search.core.earlybird.index.DocIDToTweetIDMapper;

public class OptimizedColumnStrideByteIndex extends ColumnStrideFieldIndex implements Flushable {

private final byte[] values;

public OptimizedColumnStrideByteIndex(String name, int maxSize) {

super(name);

values = new byte[maxSize];

}

public OptimizedColumnStrideByteIndex(

ColumnStrideByteIndex columnStrideByteIndex,

DocIDToTweetIDMapper originalTweetIdMapper,

DocIDToTweetIDMapper optimizedTweetIdMapper) throws IOException {

super(columnStrideByteIndex.getName());

int maxDocId = optimizedTweetIdMapper.getPreviousDocID(Integer.MAX\_VALUE);

values = new byte[maxDocId + 1];

int docId = optimizedTweetIdMapper.getNextDocID(Integer.MIN\_VALUE);

while (docId != DocIDToTweetIDMapper.ID\_NOT\_FOUND) {

int originalDocId = originalTweetIdMapper.getDocID(optimizedTweetIdMapper.getTweetID(docId));

setValue(docId, columnStrideByteIndex.get(originalDocId));

docId = optimizedTweetIdMapper.getNextDocID(docId);

}

}

private OptimizedColumnStrideByteIndex(String name, byte[] values) {

super(name);

this.values = values;

}

@Override

public void setValue(int docID, long value) {

this.values[docID] = (byte) value;

}

@Override

public long get(int docID) {

return values[docID];

}

@Override

public FlushHandler getFlushHandler() {

return new FlushHandler(this);

}

public static final class FlushHandler extends Flushable.Handler<OptimizedColumnStrideByteIndex> {

private static final String NAME\_PROP\_NAME = "fieldName";

public FlushHandler() {

super();

}

public FlushHandler(OptimizedColumnStrideByteIndex objectToFlush) {

super(objectToFlush);

}

@Override

protected void doFlush(FlushInfo flushInfo, DataSerializer out) throws IOException {

OptimizedColumnStrideByteIndex columnStrideByteIndex = getObjectToFlush();

flushInfo.addStringProperty(NAME\_PROP\_NAME, columnStrideByteIndex.getName());

out.writeByteArray(columnStrideByteIndex.values);

}

@Override

protected OptimizedColumnStrideByteIndex doLoad(FlushInfo flushInfo, DataDeserializer in)

throws IOException {

byte[] values = in.readByteArray();

return new OptimizedColumnStrideByteIndex(

flushInfo.getStringProperty(NAME\_PROP\_NAME), values);

}

}

}