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

import java.io.IOException;

import java.util.Set;

import java.util.concurrent.ConcurrentHashMap;

import com.google.common.collect.Sets;

import com.twitter.search.common.schema.base.Schema;

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

public class OptimizedDocValuesManager extends DocValuesManager {

public OptimizedDocValuesManager(Schema schema, int segmentSize) {

super(schema, segmentSize);

}

public OptimizedDocValuesManager(DocValuesManager docValuesManager,

DocIDToTweetIDMapper originalTweetIdMapper,

DocIDToTweetIDMapper optimizedTweetIdMapper) throws IOException {

super(docValuesManager.schema, docValuesManager.segmentSize);

Set<ColumnStrideIntViewIndex> intViewIndexes = Sets.newHashSet();

for (String fieldName : docValuesManager.columnStrideFields.keySet()) {

ColumnStrideFieldIndex originalColumnStrideField =

docValuesManager.columnStrideFields.get(fieldName);

if (originalColumnStrideField instanceof ColumnStrideIntViewIndex) {

intViewIndexes.add((ColumnStrideIntViewIndex) originalColumnStrideField);

} else {

ColumnStrideFieldIndex optimizedColumnStrideField =

originalColumnStrideField.optimize(originalTweetIdMapper, optimizedTweetIdMapper);

columnStrideFields.put(fieldName, optimizedColumnStrideField);

}

}

// We have to process the ColumnStrideIntViewIndex instances after we process all other CSFs,

// because we need to make sure we've optimized the CSFs for the base fields.

for (ColumnStrideIntViewIndex intViewIndex : intViewIndexes) {

String fieldName = intViewIndex.getName();

columnStrideFields.put(fieldName, newIntViewCSF(fieldName));

}

}

private OptimizedDocValuesManager(

Schema schema,

int segmentSize,

ConcurrentHashMap<String, ColumnStrideFieldIndex> columnStrideFields) {

super(schema, segmentSize, columnStrideFields);

}

@Override

protected ColumnStrideFieldIndex newByteCSF(String field) {

return new OptimizedColumnStrideByteIndex(field, segmentSize);

}

@Override

protected ColumnStrideFieldIndex newIntCSF(String field) {

return new OptimizedColumnStrideIntIndex(field, segmentSize);

}

@Override

protected ColumnStrideFieldIndex newLongCSF(String field) {

return new OptimizedColumnStrideLongIndex(field, segmentSize);

}

@Override

protected ColumnStrideFieldIndex newMultiIntCSF(String field, int numIntsPerField) {

return new OptimizedColumnStrideMultiIntIndex(field, segmentSize, numIntsPerField);

}

@Override

public DocValuesManager optimize(DocIDToTweetIDMapper originalTweetIdMapper,

DocIDToTweetIDMapper optimizedTweetIdMapper) throws IOException {

return this;

}

@Override

public FlushHandler getFlushHandler() {

return new OptimizedFlushHandler(this);

}

public static class OptimizedFlushHandler extends FlushHandler {

public OptimizedFlushHandler(Schema schema) {

super(schema);

}

private OptimizedFlushHandler(DocValuesManager docValuesManager) {

super(docValuesManager);

}

@Override

protected DocValuesManager createDocValuesManager(

Schema schema,

int maxSegmentSize,

ConcurrentHashMap<String, ColumnStrideFieldIndex> columnStrideFields) {

return new OptimizedDocValuesManager(schema, maxSegmentSize, columnStrideFields);

}

}

}