package com.twitter.search.earlybird.document;

import java.io.IOException;

import com.google.common.annotations.VisibleForTesting;

import com.google.common.base.Preconditions;

import org.apache.lucene.document.Document;

import com.twitter.decider.Decider;

import com.twitter.search.common.schema.SchemaDocumentFactory;

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

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

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

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

import com.twitter.search.common.schema.earlybird.EarlybirdCluster;

import com.twitter.search.common.schema.earlybird.EarlybirdFieldConstants;

import com.twitter.search.common.schema.earlybird.EarlybirdFieldConstants.EarlybirdFieldConstant;

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

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

import com.twitter.search.earlybird.exception.CriticalExceptionHandler;

/\*\*

\* Builds a Lucene Document from a ThriftIndexingEvent. A simplified version of

\* {@link ThriftIndexingEventDocumentFactory} that can be used for update events, which exclude

\* many fields that the tweet indexing events contain.

\*/

public class ThriftIndexingEventUpdateFactory extends DocumentFactory<ThriftIndexingEvent> {

private static final FieldNameToIdMapping ID\_MAPPING = new EarlybirdFieldConstants();

private final SchemaDocumentFactory schemaDocumentFactory;

private final EarlybirdCluster cluster;

private final Schema schema;

public ThriftIndexingEventUpdateFactory(

Schema schema,

EarlybirdCluster cluster,

Decider decider,

CriticalExceptionHandler criticalExceptionHandler) {

this(

schema,

ThriftIndexingEventDocumentFactory.getSchemaDocumentFactory(schema, cluster, decider),

cluster,

criticalExceptionHandler

);

}

@VisibleForTesting

protected ThriftIndexingEventUpdateFactory(

Schema schema,

SchemaDocumentFactory schemaDocumentFactory,

EarlybirdCluster cluster,

CriticalExceptionHandler criticalExceptionHandler) {

super(criticalExceptionHandler);

this.schema = schema;

this.schemaDocumentFactory = schemaDocumentFactory;

this.cluster = cluster;

}

@Override

public long getStatusId(ThriftIndexingEvent event) {

Preconditions.checkNotNull(event);

Preconditions.checkState(

event.isSetDocument(), "ThriftDocument is null inside ThriftIndexingEvent.");

ThriftDocument thriftDocument;

try {

// Ideally, we should not call getSchemaSnapshot() here. But, as this is called only to

// retrieve status id and the ID field is static, this is fine for the purpose.

thriftDocument = ThriftDocumentPreprocessor.preprocess(

event.getDocument(), cluster, schema.getSchemaSnapshot());

} catch (IOException e) {

throw new IllegalStateException("Unable to obtain tweet ID from ThriftDocument: " + event, e);

}

return ThriftDocumentUtil.getLongValue(

thriftDocument, EarlybirdFieldConstant.ID\_FIELD.getFieldName(), ID\_MAPPING);

}

@Override

protected Document innerNewDocument(ThriftIndexingEvent event) throws IOException {

Preconditions.checkNotNull(event);

Preconditions.checkNotNull(event.getDocument());

ImmutableSchemaInterface schemaSnapshot = schema.getSchemaSnapshot();

ThriftDocument document = ThriftDocumentPreprocessor.preprocess(

event.getDocument(), cluster, schemaSnapshot);

return schemaDocumentFactory.newDocument(document);

}

}