package com.twitter.search.common.query;

import java.io.IOException;

import java.util.Set;

import org.apache.lucene.index.LeafReaderContext;

import org.apache.lucene.index.Term;

import org.apache.lucene.search.ConstantScoreScorer;

import org.apache.lucene.search.Explanation;

import org.apache.lucene.search.IndexSearcher;

import org.apache.lucene.search.Query;

import org.apache.lucene.search.Scorer;

import org.apache.lucene.search.ScoreMode;

import org.apache.lucene.search.Weight;

/\*\*

\* Lucene filter on top of a known docid

\*

\*/

public class DocIdFilter extends Query {

private final int docid;

public DocIdFilter(int docid) {

this.docid = docid;

}

@Override

public Weight createWeight(

IndexSearcher searcher, ScoreMode scoreMode, float boost) throws IOException {

return new Weight(this) {

@Override

public void extractTerms(Set<Term> terms) {

}

@Override

public Explanation explain(LeafReaderContext context, int doc) throws IOException {

Scorer scorer = scorer(context);

if ((scorer != null) && (scorer.iterator().advance(doc) == doc)) {

return Explanation.match(0f, "Match on id " + doc);

}

return Explanation.match(0f, "No match on id " + doc);

}

@Override

public Scorer scorer(LeafReaderContext context) throws IOException {

return new ConstantScoreScorer(this, 0.0f, scoreMode, new SingleDocDocIdSetIterator(docid));

}

@Override

public boolean isCacheable(LeafReaderContext ctx) {

return true;

}

};

}

@Override

public int hashCode() {

return docid;

}

@Override

public boolean equals(Object obj) {

if (!(obj instanceof DocIdFilter)) {

return false;

}

return docid == DocIdFilter.class.cast(obj).docid;

}

@Override

public String toString(String field) {

return "DOC\_ID\_FILTER[docId=" + docid + " + ]";

}

}