# Project 5: HBase FreqIndexBuilder
# Cloud Computing
# Spring 2017

Professor Judy Qiu

## Goal

Write an HBase FreqIndexBuilder program to build an inverted index table which has the unique term's occurrences in all documents from the clueWeb09 dataset. Each row record of columnfamily "frequencies" is unique, where the rowkey is the unique term stored in byte format, column name is the documentId that contains this term, and value is the term frequency shown per document. Note that each row has multiple columns. The result must be loaded to HBase clueWeb09IndexTable. Figure 1 shows the schema of clueWeb09IndexTable.

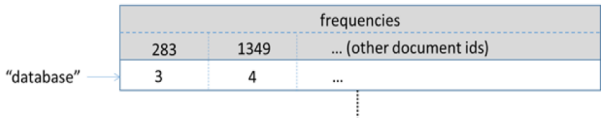


Figure 1: clueWeb09IndexTable table schema for storing term frequencies and their related documentId

## Deliverables

Zip your source code, results and report in a file named username_project5.zip. Submit this file to the Canvas submission page.

- Complete source code
- A written report describing the main steps

## Evaluation

The point total for this project is 3, where the distribution is as follows:

- Completeness of your code and output (2 points)
- Correctness of written report (1 points)

## Introduction

HBase FreqIndexBuilder is an advanced WordCount program which counts the number of occurrences of each word in a given text input dataset and also stores the related document name (identification number) as HBase inverted index records. These Inverted indicesfor text data are built for supporting efficient searches in a huge set of text data.

# What is Inverted Index?

Figure 2 shows an example of an inverted index. For a given set of documents, each composed of a series of terms (words), it records the following information: for each term, which subset of documents contains it in their texts.

To build these inverted indices, we reuse the ClueWeb09 dataset from before, which was created to support research on information retrieval and related human language technologies. The dataset is used by several tracks of the TREC conference. New inverted index table schemas are designed as shown in Figure 1.

In the clueWeb09IndexTable table each term will have the same structure, with term as rowkey, values contained in documentId, and the occurrence of the term within this document shown. Our goal is to write an HBase program which generates an inverted index table by extracting the information from the ClueWeb09 dataset.

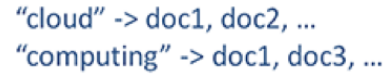


Figure 2: A sample inverted index

# Mapper and Main Program

Now we are going to implement the HBase FreqIndexBuilder program. As opposed to WordCount, our implementation only consists of two main parts:

- Mapper
- Main program

This type of application is called ?Map-Only? parallel application.

## Mapper

A Mapper overrides the “map” function from the Class “org.apache.hadoop.hbase.mapreduce.TableMapper <Text, LongWritable>”, which provides <key, value>pairs as the input. A Mapper implementation may output <key,value>pairs using the provided Context. <key, value>of this map function is <rowkey, content>, where the key is the rowkey of an HBase record related to a specified URI, and the content is the stored text of that URI. Your Map task should output <word, <docId, frequency>>for each word in the content of text.

### Pseudocode

```
void Map(key, value) {
    for each word x in the content of a hbase record:
    context.write(x, );
}
```

### Detailed implementation

```
public static class FibMapper extends TableMapper<ImmutableBytesWritable, Writable> {
  @Override
  protected void map(ImmutableBytesWritable rowKey, Result result, Context context) throws
    IOException, InterruptedException {
    byte[] docIdBytes = rowKey.get();
    byte[] contentBytes = result.getValue(Constants.CF_DETAILS_BYTES, Constants.
    QUAL_CONTENT_BYTES);
    String content = Bytes.toString(contentBytes);
```

```
7
8       // TODO: write your implementation for getting the term frequencies from each document,
9       // and generating Put objects for clueWeb09IndexTable.
10              // Hint: use the "getTermFreqs" function to count the frequencies of terms in
        content.
11              // The schema of the clueWeb09IndexTable is:
12              // row key: term, column family: "frequencies", qualifier: document Id, cell
        value: term frequency in the corresponding document
13              // Check iu.pti.hbaseapp.Constants for useful constant values.
14
15
16    }
17  }
```

### Main Program

Again, the main function has been provided as standard initialization, and you may modify it to fit your own style. See the examples of using TableMapReduceUtil.initTableMapperJob and TableMapReduceUtil.initTableReducerJob.

## Edit, compile and run your code

The sketch code is stored within the provided VirtualBox image. You can use linux text editor vi/vim to add your code.

```
1 $ cd /root/MoocHomeworks/HBaseInvertedIndexing/
2 $ vim src/iu/pti/hbaseapp/clueweb09/FreqIndexBuilderClueWeb09.java
3 $ cd /root/MoocHomeworks/HBaseInvertedIndexing/
4 $ ./compileAndExecFreqIndexBuilderClueWeb.sh
```

## View the result

The result is generated as /root/hbaseMoocAntProject/output/project2.txt.

```
1 $ cd /root/MoocHomeworks/HBaseInvertedIndexing/
2 $ cat output/project2.txt
```