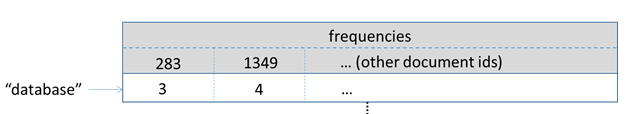
**Project Introduction:**

The goal of this project is to implement HBase and hadoop Map function to build an inverted index table “clueWeb09IndexTable” which schema is shown in figure 1 below. This table has the unique term occurrences in all the documents from the clueWeb09 data set. Each row record of column family **“frequencies”** is unique, and the rowkey is the unique term stored in byte format. The column name is the **“documentId”** and the value is the term frequency.

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

Hbase FrequencyIndexBuilder is an advance program that counts the number of occurrence of each word in a data set. It also stores the related document name as inverted index records.

## Inverted Index:

Each given documents consists of a set of terms. The inverted index table is constructed such that for each term show the occurrences within the documents (See figure 2 below for example of inverted index). To build the indices, we use the clueWeb09 which was introduced in the previous project to extract the information used by the Hbase program.

**Figure 2:** Example – Inverted Index

“Indiana” -> Doc1, Doc4, Doc 6, ...  
“University” -> Doc1, Doc2, Doc10, …

## Mapper Class:

For this project we implemented Map-only paradigm, in other words no reduce function needed.

### Mapper Function:

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.

**The Mapper Class:**

In the mapper class we were supposed to output <word, <docId, frequency>> pairs of each term words in each row of the data set using the “*getTermFreqs*” as show in the code below.  
*HashMap<String, Integer> resultData = getTermFreqs(content);*  
the for loop loops through resultData rows outputting the <word, <docId, frequency>> pairs for each word in the content of text as shown in the following source code.

*for (String key: resultData.keySet(){  
byte freq[] = Bytes.toBytes(resultData.get(key)+"");  
byte wKey[] = Bytes.toBytes(key);  
Put put = new Put(wKey);*

The following code puts the word, docId and frequency into the hashmap *put.add(Constants.CF\_FREQUENCIES.getBytes(), docIdBytes, freq);*

The following code outputs the <word, <docId, frequency>>  
*context.write(null, put);  
}*

The output is stored in HBase table clueWeb09IndexTable.