**Map(String docName, String record)**{

String[] tokens = record.split(",");

for(String s:tokens){

if(s.length()>=3){

emit(s,1);

}

}

emit("total-number-of-all-records",1);

//docHash is a hash map used to maintain documents that have been counted. Explained in detail below

if(!docHash.containsKey(docName)){

emit("total-number-of-all-documents",1);

docHash.put(docName,true);

}

}

**Reduce(String word, Iterable<Integer> values )**{

int sum = 0;

for(Integer n : values){

sum += n;

}

emit(word,sum);

}

Map Function: Here map function accepts document name and record as an input. In map function we split the record on “,” and save the tokens. For each token we check if length is >= 3 and if yes, then emit(token,1).

After this for loop is executed completely we emit word “total-number-of-all-records” with value as 1. This is done so that for each record this word is emitted. We also use a global Hash Map docHash.

**HashMap<String,Boolean> docHash = new HashMap<String,Boolean>();**

This hashmap is initially empty and it saves the document name as and when the records containing them are passed to the map function. Thus we check if docName for current record is it already present in the hashmap. For the first record of that document HashMap will not have this docName and then we will emit “total-number-of-all-documents” as a word with value 1 and add this docName to the Hash map.

Thus when next record of same document is passed to the map function, it will check if document name already exists in hash map. As it exists It won’t do anything for document name.

Reduce function accepts word and its value as input and will count the Iterable values for that particular word and final sum will be generated.

Input to Mappers:

“Doc1”, “a,crazy,fox,jumped”

“Doc1”, “crazy,fox,jumped,high”

“Doc2”, “a,crazy,crazy,fox,jumped”

“Doc2”,” fox,jumped”

“Doc3”, “a,crazy,fox,jumped”

“Doc3”, “da,fox,jumped,high”

Output of Mappers:

|  |  |
| --- | --- |
| Word | Value |
| Crazy | 1 |
| Fox | 1 |
| Jumped | 1 |
| total-number-of-all-records | 1 |
| total-number-of-all-documents | 1 |
|  |  |
| Crazy | 1 |
| Fox | 1 |
| Jumped | 1 |
| High | 1 |
| total-number-of-all-records | 1 |
|  |  |
| Crazy | 1 |
| Crazy | 1 |
| Fox | 1 |
| Jumped | 1 |
| total-number-of-all-records | 1 |
| total-number-of-all-documents | 1 |
|  |  |
| Fox | 1 |
| Jumped | 1 |
| total-number-of-all-records | 1 |
|  |  |
| Crazy | 1 |
| Fox | 1 |
| Jumped | 1 |
| total-number-of-all-records | 1 |
| total-number-of-all-documents | 1 |
|  |  |
| Fox | 1 |
| Jumped | 1 |
| High | 1 |
| total-number-of-all-records | 1 |
|  |  |

Input to Reducers:

Shuffle and Sort takes the output from mappers, sorts it and aggregates the values by keys.

|  |  |
| --- | --- |
| Word | Iterable |
|  |  |
| Crazy | [1,1,1,1,1] |
| Fox | [1,1,1,1,1,1] |
| Jumped | [1,1,1,1,1,1] |
| High | [1,1] |
| total-number-of-all-records | [1,1,1,1,1,1] |
| total-number-of-all-documents | [1,1,1] |
|  |  |

Output of Reducers

|  |  |
| --- | --- |
| Crazy | 5 |
| Fox | 6 |
| Jumped | 6 |
| High | 2 |
| total-number-of-all-records | 6 |
| total-number-of-all-documents | 3 |
|  |  |