1. ***Describe the design you used for Part 1 and 3. This should include the keys and values you used.***

Part 1:

This is the implementation of MapReduce to find the number of Starbucks located in each city. There are total 3 code file i.e. **mapper.py, reducer.py** and **query.py** in this section.

The mapper code reads the given input file 'starbucks-locations-sort.csv' from Hadoop file system and passes the output to the reducer. The reducer then counts the number of Starbucks for each city. Hadoop takes care of the shuffle and sort.

After that we write the reducer output to a file 'cityInformation'. The query reads this file and returns the number of Starbucks located in that city given by the user.

Part 3:

In this part we implemented Inverted Index using MapReduce. There are total 3 code file i.e. **mapper.py, reducer.py** and **query.py** similar as Part1.

The mapper code reads the given input file 'movies.dat' from Hadoop file system and passes the output to the reducer. The reducer then build the Inverted Index where each term is represented by a genre and each posting list corresponds to the movie list for that particular genre. Hadoop takes care of the shuffle and sort.

After that we write the reducer output to a file 'invertedIndex'. The query reads this file and returns the list of movies for a given genre. Here, the query can also be a Boolean search query with either only AND or only OR.

2. ***What else would you have done in the inverted index implementation, given more time, energy, resources, etc.?***

There is nothing more that can be done in terms of design and implementation of the inverted index. Though, I must admit the query part is the most tricky in this section, specially the Boolean search.

I have implemented this using the concepts of set theory where AND denotes intersection and OR denotes union of two sets. But I think, this approach is more time consuming because we have to scan the entire list of movies and that might be challenging for large datasets.

Given more time, energy and resources I would like to modify this approach in order to include some optimization technique where the processing could be in the increasing order of list size.

3. ***How difficult was it to implement the inverted index? How difficult would it be to implement another task, given this experience? What would be straightforward? What would take more time?***

Implementing the Inverted index in Part 2 & 3 was straightforward. The Boolean query implementation in Part 2 and Part 3 was a very difficult and tricky.

Implementing another similar task should be easier, but in programming you never know!!!

Setting up the environment (VM, Hadoop and Python) would be straightforward.
I would certainly spend much more time in the Boolean Query part.