Introduction

SpiceHeads frequently visit our websites from browsers on a variety of devices (mobile, tablet, laptop, etc.). These websites may also be on different domains. Sometimes they log in and sometimes they browse anonymously. In order to better understand their needs and provide an enhanced experience, it is helpful to uniquely identify a given person across all their browsing history. To reach this goal, we have different identifiers that we use for each of these websites. Some of these identifiers (let's call these 'local ids') are used across different websites and there are also ways to obtain a match between one identifier and another. We then use this match data to assign a 'person id' to each one of the local ids.

Requirements

For this challenge, you will be writing code to assign person ids given match data for three types of local ids. Your code should:

1. Be able to process a CSV file where the columns are:

local id 1, local id 2, local id 3

Note:
  a. If a column is empty, then there is no match data for that local id.
  b. A row can contain a single id (i.e. there was no match).
  c. The data for a single person id is not all grouped together.
  d. For the sake of simplicity, you can assume that the first row 'happened before' the second row, etc.

2. Assign a person id to each of the local ids. Given three systems in which we have local ids, here are three examples of a series of match data in which each should result in a single person id, respectively.

local id type 1	local id type 2	local id type 3
abc	def	
	def	ghi

local id type 1	local id type 2	local id type 3
jkl		
	mno	
		pqr

```
       jkl        |                    |        pqr
                  |         mno        |        pqr


local id type 1   |  local id type 2   |  local id type 3
-------------------------------------------------------
       stu        |                    |
       stu        |         vxy        |
       stu        |         z12        |
       stu        |                    |        345
```

Note that this is not an exhaustive set of examples of how matches can be structured.

3. Output the person id for each local id. If you incorporate a database or datastore within your code, please include examples of how to query for each local id. In the above examples, we could issue the following queries and expect the matching results:

```
{ 'id_type': 1, 'id': 'abc' } => '1'
{ 'id_type': 2, 'id': 'def' } => '1'
{ 'id_type': 3, 'id': 'ghi' } => '1'

{ 'id_type': 1, 'id': 'jkl' } => '2'
{ 'id_type': 2, 'id': 'mno' } => '2'
{ 'id_type': 3, 'id': 'prq' } => '2'


{ 'id_type': 1, 'id': 'stu' } => '3'
{ 'id_type': 2, 'id': 'vxy' } => '3'
{ 'id_type': 2, 'id': 'z12' } => '3'
{ 'id_type': 3, 'id': '345' } => '3'
```

4. Be able to work 'at scale'. This can mean that you need to process a very large amount of data at once. This can also mean that you need to process a moderate amount of data very quickly. Ideally you can do both. Please discuss how your code meets at least one of these definitions and if it cannot meet both requirements, how and why that tradeoffs were made.

Guidelines

1. Spend less than 4 hours on this challenge. If you feel like you have not completed in that time, please discuss what you have accomplished and how you would finish the rest of the challenge.

2. If you use any external resources or frameworks that require setup, please include any instructions on how to set them up or an instance we can login to.
3. Please include build and execution instructions for your code.