99chairs coding challenge

The goal of this assignment is to create a command line application that will be capable of processing CSV files. The tools/gems chosen to fulfil this assignment do not matter, as long as the user gets the desired results after calling the script. The only constraint is that the script must be written in ruby language. The same script should be able to perform all tasks whereby the user of the script is able to select which task he or she wants to execute.

You **do not** have to accomplish the whole assignment. It is better, if you only accomplish **Task 1** and test your code, then if you start all of the tasks and do not finish any. Through this assignment you should show us, how you organize your code and how you identify possible abstractions.

Please include a short description on how to setup/start the service as well.

Task 1
taks1.csv file consists of 3 columns: “id”, “first_name” and “last_name”. Your task is to aggregate identical first names and print the summary to the user. Only names that occur more than once in the “first_name” column should be printed to the stdout. Please make sure that the results are printed in form of a table, e.g.

```
first_name | count
-----------+-------
Dan        | 2
Phil       | 3
```

Please also make sure that the user will be able to use an option to write aggregated data in a file in the CSV format.

Task 2
taks1.csv and task2.csv both include an “id” column. Your task is to print the outcome of the following SQL query without importing data into the RDBMS:

SELECT task2.first_name FROM task1 LEFT JOIN task2 ON task1.id = task2.id;

Your task is to join the rows from both files using the “id” field and print the outcome as if it would be a query against an RDBMS.

Please make sure that the results are printed in form of a table, e.g.

```
first_name
-----------
```

```
Dan
NULL
Phil
```

Please also make sure that the user will be able to use an option to write aggregated data in a file in the CSV format.

Task 3
After successfully merging the data it is time to allow external services to access it. In this part we need the result of following query:

SELECT task1.id, task1.first_name, task1.last_name FROM task1 INNER JOIN task2 ON task1.id = task2.id;

to be available to the external services through the HTTP protocol as a JSON string. Clients should be able to locate the resource using the value from “id” column. The server should return a response in following format:

```
{
     “data”: {
          “id”: 1234,
          “first_name”: “John”,
          “last_name”: “Doe”
     }
}
```