Data Engineer Challenge

You will have 3 days (72 hours) to submit the task. If you need an extension, please email us to explain the situation.

The goal of this home task is to assess the candidate's code style, problem solving methodology, and command of various skill sets.

All the tasks are simplified real world examples of what we do and are a good illustration of what could be your daily job.

Final submission must be one zip file containing all answers and words.

SQL

Please see http://sqlfiddle.com/#!17/3ff32 or https://dbfiddle.uk/?rdbms=postgres_9.6&fiddle=6eeee53c62aef4e84720cf4074a81e9a for database schema.

Given the transactions table and table containing exchange rates. Exchange rate timestamps are rounded to second, transactions are rounded up to millisecond. We have only data for one day, 1st of April, 2018. Please note there are no exchange rate from GBP to GBP as it is always 1

1. Write down a query that gives us a breakdown of spend in GBP by each user. Use the *exchange rate with the largest timestamp*.
2. **(If you consider yourself senior)** Write down the same query, but this time, use the latest *exchange rate smaller or equal than the transaction timestamp*. Solution should have the two columns: **user_id**, **total_spent_gbp**, *ordered* by **user_id**
3. **Bonus for postgres superstars:** Consider same schema, but now let's add some random data, to simulate real scale: http://sqlfiddle.com/#!17/c6a70 or https://dbfiddle.uk/?rdbms=postgres_9.6&fiddle=231257838892f0198c58bb5f46fb0d5d Write a solution for the previous task. Please ensure It executes within 5 seconds.

Readability of sql query will be evaluated as well.

Programming

The program must have the following:

1. Full solution written in Python 3.6
2. In case you are using any third-party variables, a requirements.txt files should be specified

3. Programs should get input from stdin and print results to stdout.
4. Any debugging or logging information should be printed to stderr
5. Use argparse, to specify parameters. --help should print out usage instructions.
6. Unit tests are a must.
7. Filename specified in the task description

Task 1:

Given an input as json array (each element is a flat dictionary) write a program that will parse this json, and return a nested dictionary of dictionaries of arrays, with keys specified in command line arguments and the leaf values as arrays of flat dictionaries matching appropriate groups

```
python nest.py nesting_level_1 nesting_level_2 … nesting_level_n
```

Example input for json can be found here: http://jsonformatter.org/74f158
When invoked like this:

```
cat input.json | python nest.py currency country city
```

the output should be something like this: http://jsonformatter.org/615048
Please note that the nesting keys should be stripped out from the dictionaries in the leaves.
Also please note that the program should support an arbitrary number of arguments, that is arbitrary level of nesting.

Task 2:
Create a REST service from the first task. Make sure your methods support basic auth.

Good luck!