Shipnow Back-end interview

Exercises 1 and 2 can be solved with any programming language (please no pseudo-code).

Exercise 3 can be solved in any SQL flavor.

In exercise 4 there is no need for coding. If there is any missing/confusing requirement, please do mention it and make your own assumptions. This is an open-ended question, there is no right or wrong answer.

If you have any doubts, please do not hesitate to contact us at: developers@shipnow.com.ar

---

1. *Anagrams*

   An anagram is a type of word play, the result of rearranging the letters of a word or phrase to produce a new word or phrase, using all the original letters exactly once; for example, orchestra can be rearranged into carthorse.

Implement a function def anagrams(word, possible_anagrams) that receives 2 parameters: a string(word) and an array of strings(possible_anagrams); and returns an array with all the anagrams for that word.

For example: Given 'horse' and ['heros', 'horse', 'shore', 'standard'] as arguments, the output should be: ['heros', 'horse', 'shore']

2. Frog Jump

   A small frog wants to get to the other side of the road. The frog is currently located at position X and wants to get to a position greater than or equal to Y. The small frog always jumps a fixed distance, D. Count the minimal number of jumps that the small frog must perform to reach its target.

Write a function def frog_jump(x, y, d) that, given three integers X , Y and D , returns the minimal number of jumps from position X to a position equal to or greater than Y .

For example: Given: X = 10 , Y = 85 and D = 30 , the function should return 3, because the frog will be positioned as follows:

● After the first jump, at position 40 (10 + 30)

● After the second jump, at position 70 (10 + 30 + 30)

● After the third jump, at position 100 (10 + 30 + 30 + 30)

Assume that: X , Y and D are integers within the range (1..1,000,000,000) and X ≤ Y .

3. Department Salaries

You are given two tables, department and employee , with the following structure:

```
create table department (
  dept_id integer not null,
  dept_name varchar(30) not null,
  dept_location varchar(30) not
null,
  unique(dept_id)
);
```

```
create table employee (
  emp_id integer not null,
  emp_name varchar(50) not null,
  dept_id integer not null,
  salary integer not null,
  unique(emp_id)
);
```

Each record in the table department represents a department which might hire some employees. Each record in the table employee represents an employee who works for one of the departments from the table department.

Write an SQL query that returns a table comprising all the departments (dept_id) in the table department that hire at least one employee, the number of people they employ and the sum of salaries in each department. The table should be ordered by dept_id (in increasing order).

For example:

Department Table:

dept_id	dept_name	dept_location
10	Accounts	Delhi
20	Marketing	Delhi
30	Production	Hyderabad
40	IT	Warsaw
50	Sales	Bengaluru

Employee Table:

emp_id	emp_name	dept_id	salary
1	Jojo	20	5000
2	Popat Lal	30	15000
3	Santa Singh	40	25000
4	Banta Singh	20	7500
5	Sohan Lal	20	15000
6	Kk	10	12000
7	Bob	20	35000
8	John	30	25000
9	Smith	40	5000

Your query should return:

dept_id	count	sum_of_salary
10	1	12000
20	4	62500
30	2	40000
40	2	30000

4. URL Shortener

A URL shortening service, is a web service that provides short aliases for redirection of long URLs. For example, https://medium.com/shipnow/por-que-tener-varias-opciones-de-envio-eeafa099fd71 is long and hard to remember, a URL shortener can create a alias for it (http://bit.ly/2g4hx7h). If you click the alias, it'll redirect you to the original URL.

We need you to design (there is no need to code anything) a URL shortener with these specifications:

● It doesn't need to have a web interface, a JSON API is fine.

● The API is open, there is no need for authentication / user accounts / developer keys.

● Users should be able to choose their shortlink. For example, they might want http://sho.rt/my-short-link.

● Short Links should expire after 6 months after it's created.

5. And Another Thing...

It's Saturday, the night is coming and Chano knows it. Today is the day that he will finally perform "The Golden Mile": a challenge where he has to drink a beer in each of the best 20 breweries in Buenos Aires City. But such an epic challenge does not combine well with driving a car, so he wants to make sure not to being involved in a car crash again. Therefore, your mission will be:

a) Write an algorithm that finds the optimal path for Chano to reach all the breweries, plus, find out the amount of time that it would take him to do so.

b) Explain what criteria you used to consider that the path is optimal (and anything else you think is important to clarify about the algorithm).

Some clarifications:

● The location of the 20 breweries in a json is attached in separated file.

● Consider a constant speed of 60 km / h for Chano’s car, and ignore the time he spends at each brewery.

● Consider as if the distance between brewery and brewery is Euclidean, that is, the road is a straight line.

● You can define in advance the brewery where the challenge will start, but the rest of the route must be found by your algorithm.