**Elementor - Data Engineer home assignment**

Hi,

In Elementor we’re leaders in providing tools for building the web.

This assignment is designed for senior data engineers - we count on you to prioritize.

We expect you to write the solution in python and SQL (if you choose otherwise - pls. explain).

In the end of this exercise, we expect to get in a **git repository**:

1. Short design of YOUR solution and how would you implement it on AWS.
2. Working solution (and instruction how to install and run it).
3. Suggestions for improvements or changes.

There is no one right solution - **pls. limit it to the given time**.

Total time: 3-5 hours.

The evaluation has 2 parts

1. Coding assignment
2. SQL questions
3. **Coding assignment:**

**The story:**

We have a list of websites and would like to know if these sites are at risk or safe.

The site [VirusTotal](https://en.wikipedia.org/wiki/VirusTotal) can give information about the safety level of a website (and files etc.)

Our goal is to create a database holding risk/safe information about all our websites.

We would like to avoid extra costs of querying. When information has been received for a website, avoid sending requests again (if the data is up-to-date). Data is considered as up to date if the query result is not older than the last 30 minutes. The ultimate (but optional) product would be an API that reaches out VirusTotal and updates the database.

**Data Source:**

1. List of sites for classification (DS1):
   1. Type: CSV
   2. Description: site url
   3. Location:<https://elementor-pub.s3.eu-central-1.amazonaws.com/Data-Enginner/Challenge1/request1.csv>
   4. Example for row in file:

[www.elementor.com](http://www.elementor.com)

1. [VirusTotal](https://en.wikipedia.org/wiki/VirusTotal) - a site that aggregate many antivirus and online scan products data (DS2)
   1. Type - API
   2. Description: Analyze suspicious files and URLs to detect types of malware, automatically share them with the security community
   3. Source: <https://www.virustotal.com/>

**The Task:**

The system input would get a CSV (DS1). It shall start running on a local folder e.g. “/usr/sites”, classifying the sites in the files.The classification will be done against DS2 (unless updated result is already in the db). You should save in the database of your choice both tracking of the requests and the results from site. Scratch task - write API for this functionality.

**Detailed requirements:**

1. For each **URL,** check and save the following:

The “Site risk” field will be determined by the following logic:

* + 1. In case that in the 'result' tag there are more than:

One (1) malicious or one phishing or one malware: the site will be categorized as “risk” else “safe”

Example :

* Site “fourthgate.org/Yryzvt” - will be categorized as “risk”.
* Site: [www.elementor.com](http://www.elementor.com) - will be categorized as “safe”
  + 1. We would also like to keep the Total voting of each site

for example for [www.elementor.com](http://www.elementor.com): 'clean': 70, 'unrated': 8

* + 1. We would like to keep the classification based on ‘categories’ tag.

for example for [www.elementor.com](http://www.elementor.com): 'marketing': 1, 'mobile communications': 1, 'information technology': 2

1. As stated before - perform check only if the results you have for site check is not “fresh”.
2. create table if not exists public.traces  
   (  
    "RecordId" serial not null  
    constraint traces\_pk  
    primary key,  
    "TraceDateTime" timestamp,  
    "APIname" text,  
    "Trace-Id" serial,  
    "Status" integer,  
    "Message" text,  
    "RequestJSON" text,  
    "ResponseJSON" text  
   );  
     
   alter table public.traces  
    owner to postgres;

create function *trace\_add\_new*(p\_api\_name text, p\_status integer, p\_request\_json text)  
 returns TABLE("RecordId" integer, "Trace-Id" text)  
 language sql  
as  
$$  
  
INSERT INTO public.traces ("TraceDateTime", "APIname", "Status", "RequestJSON")  
values (NOW(), p\_api\_name,  
 p\_status, p\_request\_json)  
RETURNING "RecordId","Trace-Id";  
  
$$;  
  
alter function *trace\_add\_new*(text, integer, text) owner to postgres;

create procedure *log\_trace*(p\_api\_name text, p\_status integer, p\_message text, p\_request\_json text, p\_response\_json text)  
 language sql  
as  
$$  
INSERT INTO public.traces ("TraceDateTime", "APIname", "Status", "Message", "RequestJSON", "ResponseJSON")  
VALUES (NOW(), p\_api\_name, p\_status, p\_message, p\_request\_json, p\_response\_json);  
  
$$;  
  
alter procedure *log\_trace*(text, integer, text, text, text) owner to postgres;

**2. SQL Questions :**

1. Employee question - you have the following tables:

* **employees**: employee\_id, first\_name, last\_name, hire\_date, salary, manager\_id, department\_id
* **departments**: department\_id, department\_name, location\_id

We would like to know for each department top earning employee, salary, difference

from the second earning employee.

select t1.d1,sum(coalesce(t1.s1,0)-coalesce(t1.s2,0))  
from  
(  
  
select d.department\_id d1,max(salary) as s1,null as s2  
from employees  
inner join departments d on employees.department\_id = d.department\_id  
group by d.department\_id  
  
union all  
  
SELECT d.department\_id,null as s1,MAX(salary) AS s2  
FROM employees  
inner join departments d on employees.department\_id = d.department\_id  
WHERE salary not IN  
(SELECT MAX(salary)  
FROM employees  
 group by department\_id  
)  
group by d.department\_id)T1  
GROUP BY t1.d1  
  
  
  
SELECT d.department\_id,MAX(salary) AS salary  
FROM employees  
inner join departments d on employees.department\_id = d.department\_id  
WHERE salary not IN  
(SELECT MAX(salary)  
FROM employees  
 group by department\_id  
)  
group by d.department\_id

1. Site visiting question - you have the following tables:

* **site\_visitors** : date, site, number of visitors
* **promotion dates :** start\_date, end\_date, site, promotion\_code

We would like to know what percent of the site traffic was on promotion dates

select t1.site,  
 cast(sum(coalesce(t1.p1, 0))/sum(coalesce(t1.p2, 0))\*100 as double precision)  
from  
(  
 (  
 SELECT p1.site, sum(number\_of\_visitors) as p1,null as p2  
FROM promotion p1  
inner join site\_visitors sv on p1.site = sv.site  
WHERE not(start\_date<=sv.date and end\_date>=sv.date)  
group by number\_of\_visitors,p1.site)  
  
  
union  
 (SELECT p2.site, null as p1,sum(number\_of\_visitors)as p2  
 FROM promotion p2  
 inner join site\_visitors sv on p2.site = sv.site  
 --WHERE start\_date<=sv.date and end\_date>=sv.date  
 group by number\_of\_visitors,p2.site) ) t1  
  
GROUP BY t1.site

Good Luck !

Don’t hesitate email any questions to: [dataexe@elementor.com](mailto:dataexe@elementor.com)