**Interview Question Answers**

SQL & Python

04 October 2022

# Document Information

## Table of Contents

[Document Information 2](#_Toc115857704)

[Table of Contents 2](#_Toc115857705)

[Revision History 2](#_Toc115857706)

[Introduction 3](#_Toc115857707)

[Question 1 - Python (5 marks) 4](#_Toc115857708)

[Question 2 - Python (7.5 Marks) 5](#_Toc115857709)

[Question 3 - SQL (5 Marks) 8](#_Toc115857710)

[Question 4 - SQL (7.5 Marks) 10](#_Toc115857711)

[Question 5 - Hybrid (10 Marks) 12](#_Toc115857712)

## Revision History

|  |  |  |
| --- | --- | --- |
| **Version** | **Details** | **Date** |
| 0.01 | Draft Creation | 05/10/22 |
| 0.02 | Solution Review | 07/10/22 |

# Introduction

Below is a set of questions that will test your skillsets of both when approaching both Python and SQL problems.

Unless specified otherwise,

* You are free to use any python packages you see fit
* You can use any flavour of SQL you choose

The aim will be too complete is **two** or **three** of the questions, you can choose any 2/3 but you must choose **1 SQL** question and **1 python**. Optionally we'd like you to choose the hybrid one two as the question can be solved using either SQL/Python

The Idea of these questions are not to quiz you on your quickfire ability to remember syntax for specific languages, we are more interested in how you approach solving certain problems and would prefer that you accompany each solution with description as to why you solved it in that manner.

Aim to finish your solutions in **~ 30 minutes**

# Question 1 - Python (5 marks)

Below is a function that attempts to calculate a players expected rating score as part of an upcoming chess tournament’s ranking algorithm, this algorithm is based off one part of [ELO rating system](https://en.wikipedia.org/wiki/Elo_rating_system) by Arpad Elo.



The formula that this python function attempts to replicate can be mathematically expressed as:

Where:

RB = The players opponent

RA = The player whom the score Is being estimated for

Your task is re-writing this function with the aim of improving quality and robustness.

**Solution Guideline:** The provided python function does successfully compile. So, an Input of **x = 99** & **y = 101,** will result In the following:

And the output should be **~0.4971218004251891**

The goal of this task should be for candidates to evaluate existing working code and Improve It with core design principles. Some Items to look out for.

* Code formatting
* Reduce number of lines
* Using appropriate variable and function naming conventions
* Error handling? (What would happen If RA contained a non-numeric value?)

**Example Solution:**

****

After examination of the provided function In task one I Identified the following Issues with the code:

* Poor Code formatting
* Lack of Human readability (Lack of function and variable names)
* Ineffective use of variables
* Lack of Error handling

In the function I have developed, I have attempted to address these Issues by making the following changes:

* Given the Function and Argument readable names
* Used consistent code formatting
* Added expected argument and return data types to the function
* Condensed the equation into a single line
* Added a simple exception to catch function errors

# Question 2 - Python (7.5 Marks)

Given a single string variable `input\_sentance`, create a function which returns a dictionary that counts the occurrence of words in any given input (exclude special characters where you deem necessary).

In your answer provide justification on how you approached this question and outline any error handling or robustness you included.

* Please note this solution should be completed using python3 & that you only have access to the **re** package

**Example:**



**Solution Guideline:** The example output provided in this question Is one such Input + return that we may see, but not necessarily the only correct solution.Some Ideas on what we are looking for.

* How Capital Letters are handled
* Some special characters can change a word meaning (Rabbit's vs Rabbits)

Other Examples could look like:

****



**Example Solution (BASIC):**

****

This solution would provide an answer similar to that provided In the example output In the question. It will remove capital letters and apostrophes "Rabbit's" > "Rabbits". However correct answers could keep these and be correct.

Looking for Candidates to explain their output and outline why their method Is valuable over alternatives.

# Question 3 - SQL (5 Marks)

Consider the following tables:



Each record in the `sales` table represents the total ($) sale amount a clerk made on any particular transaction. A clerk can make multiple sales per day. `clerk\_id` contains the `id` of the associated clerk.

Each record in the `clerks` table represents an employee’s information `id` & name.

It’s the end of the month (*March 1999*), so you would like to decide the employee of the month based on whomever has the highest total sales amount for this month only. However, we want to ensure that the full name is included in the results set

Write a SQL Query (use any flavour of SQL) that best represents this result and explain why you chose this as the best output.

**Solution Guideline:**

The idea behind this task is to leave the solution without an exact design, this would allow candidates to present what they think Is the best solution and then discuss why.

Example: ID probably should be included, because what if two employees have the same first + last name? Should the Number of transactions also be returned? To Include what the value of each transaction on average is, per clerk.

Example:

**Given the following Input:**

****

**Example Output Sets:**

**(1)**

|  |  |  |
| --- | --- | --- |
| **Clerk Id** | **Clerk Full Name** | **Total Sales Amount ($)** |
| 3 | Jackson Smith | $ 199 |
| 1 | John Doe | $ 99.5 |
| 6 | James Smith | $ 50.5 |
| 2 | Jane Doe | $ 47.5 |
| 5 | John Doe | $ 30.5 |

**OR**

**(2)**

|  |  |  |
| --- | --- | --- |
| **Clerk Id** | **Clerk Full Name** | **Total Sales Amount (Monthly)** |
| 3 | Jackson Smith | $ 199 |

**OR**

**(3)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Clerk Id** | **Clerk Full Name** | **Total Sales Amount (Monthly)** | **NO# of Transactions** |
| 3 | Jackson Smith | $ 199 | 2 |

**Example Solution (MySQL):**

**(1)**

****

**OR**

**(2)**

****

These three solutions all provide the correct answer however differ In what context a candidate deems necessary to provide. The Idea Is to have candidates explore why they chose the method they used.

**OR**

(3)



# Question 4 - SQL (7.5 Marks)

Consider the following tables:



“Happy” is a small party business that has attempted to build their own database to store employee information as they grow. This attempt resulted in the creation of the `happy\_org` table shown above.

After using the database, they have found limitations and would like you to assist them in redesigning it. Some of the problems they face include but are not limited to.

* If updated, Happy has no way to track historical entries. Previous “Job Title”, “Salary”, “Name”
* Some employee’s names are too long and need to be abbreviated.
* Two of their employees share the same name and cause confusion for the admin team.

Describe and/or show how you would redesign this database to create a more robust solution for Happy Org.

**Solution Guidelines:**

In this question, candidates can present their answer as they see fit, text, UML Diagram, SQL.

The Idea is to have candidates design a new solution above the poorly designed one above to create a more robust solution and explain why.

In the question, some problems have been highlighted but not all. (What If an employee, leaves the org and returns, how should that be handled? What If a position title Is updated?)

A very basic solution could look as follows:

Chart

Description automatically generated

This could also be displayed as a SQL Table creation.

# Question 5 - Hybrid (10 Marks)

Consider that you have a set of data that captures metrics on both past and currently running business-critical SQL queries.

**[query\_ID | status| start\_datetime | end\_datetime]**

Your objective in this task is to build a solution that can detect if a currently running query is running longer than expected.

* Both `start\_datetime` & `end\_datetime` will be in a UNIX epoch time format
* The `query\_id` is the unique identifier of a query
* The `status` column will be either `completed` or `active`. You can assume that all `active` events are currently running.
* All `active` queries will have no value in the `end\_datetime` field until it moves into `completed`

**You may use either Python or any flavour of SQL in this task (Assume the data is either a DataFrame or a SQL Table, we really want to see how you approach the problem over your SQL or Python skills)**

**Solution Guidelines:** Due to the nature of this question It Is quite hard to build an exact solution example that you can validate a candidates answer with 100% accuracy. In It current state this question requires the use of a **.now()** which will be dependant on the time of execution depending on the data. This use case mimics a monitoring system.

However, there are a few Items that could be blockers for some candidates which you should keep an eye out for in their code.

* Conversion of epoch to standardized time, with use of certain libraries, you should be able to relative time difference without conversion.

**Solution:**

**Python (Easy Method):**



**SQL (Postgre)**

****

**SQL(MySQL)**

****

* Will need to measure the runtime using something similar to a **now()** function and comparing It against the start date of an active query.

**Solution:**

**Python:**

****

**SQL MySQL**

****

* Multiple Methods can be used for outlier detection (Median Absolute Deviation, IQR, average ()) Aim is to get the justification on why. `Could data skew if there was a deadlock during a query causing it to run 500x its normal runtime?

**Python - Average Method (Simple Solution)**

df**[**'runtime'**]** **>=** df**[**'runtime'**].**mean**()** **\*** 1.5

**SQL (MySQL) - IQR Method (Robust Solution)**

****

**Example Data:**

**Python:**

****