Python and SQL Interview Questions

# Python Questions

Question 1: Implement a Python function `merge\_dictionaries` that takes two dictionaries as input and merges them. The function should return a new dictionary with the combined keys and values. If the same key exists in both dictionaries, their values should be summed if they are integers or concatenated if they are strings.

def merge\_dictionaries(dict1, dict2):

merged\_dict = {}

# Merge keys and values from the first dictionary

for key, value in dict1.items():

if key in dict2:

# If the key exists in both dictionaries, handle based on value type

if isinstance(value, int) and isinstance(dict2[key], int):

# If both values are integers, sum them

merged\_dict[key] = value + dict2[key]

elif isinstance(value, str) and isinstance(dict2[key], str):

# If both values are strings, concatenate them

merged\_dict[key] = value + dict2[key]

else:

# Otherwise, keep the value from the first dictionary

merged\_dict[key] = value

else:

# If the key exists only in the first dictionary, add it to the merged dictionary

merged\_dict[key] = value

# Merge keys and values from the second dictionary

for key, value in dict2.items():

if key not in dict1:

# If the key exists only in the second dictionary, add it to the merged dictionary

merged\_dict[key] = value

return merged\_dict

dict1 = {'a': 1, 'b': 'hello', 'c': 3}

dict2 = {'b': 'world', 'c': 4, 'e': 'Python'}

result = merge\_dictionaries(dict1, dict2)

print(result)

**Output:**

{'a': 1, 'b': 'helloworld', 'c': 7, 'e': 'Python'}

Question 2: Write a Python function `filter\_primes` that takes a list of numbers as input and returns a new list containing only the prime numbers from the original list. Use any method you prefer to determine if a number is prime.

def is\_prime(n):

if n <= 1:

return False

elif n <= 3:

return True

elif n % 2 == 0 or n % 3 == 0:

return False

i = 5

while i \* i <= n:

if n % i == 0 or n % (i + 2) == 0:

return False

i += 6

return True

def filter\_primes(numbers):

return [num for num in numbers if iss\_prime(num)]

# Example usage:

numbers = [1, 271, 3, 4, 5, 6, 7, 8, 9, 10]

prime\_numbers = filter\_primes(numbers)

print(prime\_numbers)

**Output:**

[271, 3, 5, 7]

Question 3: Create a Python function `rotate\_matrix` that takes a 2D list (matrix) as input and rotates it 90 degrees clockwise. The function should return the rotated matrix. Ensure your solution handles matrices of non-square dimensions.

def rotate\_matrix(matrix):

# Get the number of rows and columns in the original matrix

num\_rows = len(matrix)

num\_cols = len(matrix[0]) if matrix else 0

# Create a new matrix with swapped dimensions

rotated\_matrix = [[0] \* num\_rows for \_ in range(num\_cols)]

# Populate the rotated matrix by rotating each element

for i in range(num\_rows):

for j in range(num\_cols):

rotated\_matrix[j][num\_rows - 1 - i] = matrix[i][j]

return rotated\_matrix

# Example usage:

matrix = [

[1, 2, 3],

[4, 5, 6],

[7, 8, 9],

[10, 11, 12]

]

rotated\_matrix = rotate\_matrix(matrix)

for row in rotated\_matrix:

print(row)

**Output:**

[10, 7, 4, 1]

[11, 8, 5, 2]

[12, 9, 6, 3]

Question 4: Write a Python function `find\_anagrams` that takes two strings as input and returns `True` if the strings are anagrams of each other, and `False` otherwise. An anagram is a word or phrase formed by rearranging the letters of a different word or phrase, typically using all the original letters exactly once.

def find\_anagrams(str1, str2):

# Remove spaces and convert strings to lowercase

str1 = str1.replace(" ", "").lower()

str2 = str2.replace(" ", "").lower()

# Check if the sorted characters of both strings are equal

return sorted(str1) == sorted(str2)

# Example usage:

string1 = "listen"

string2 = "silent"

print(find\_anagrams(string1, string2))

**Output:** True

# SQL Questions

Question 1: Write an SQL query to find the top 3 departments with the highest number of employees. Assume you have an `Employees` table with a `DepartmentID` column that links to a `Departments` table with `DepartmentID` and `DepartmentName` columns.

SELECT d.DepartmentName, COUNT(e.EmployeeID) AS EmployeeCount

FROM Departments d

JOIN Employees e ON d.DepartmentID = e.DepartmentID

GROUP BY d.DepartmentID, d.DepartmentName

ORDER BY EmployeeCount DESC

LIMIT 3;

Question 2: Construct an SQL query that calculates the total sales for each product category for the last quarter. Assume you have a `Sales` table that includes `ProductID`, `Amount`, and `SaleDate`, and a `Products` table with `ProductID`, `ProductName`, and `CategoryID`.

SELECT p.CategoryID, c.CategoryName, SUM(s.Amount) AS TotalSales

FROM Sales s

JOIN Products p ON s.ProductID = p.ProductID

JOIN Categories c ON p.CategoryID = c.CategoryID

WHERE s.SaleDate >= DATE\_SUB(CURRENT\_DATE(), INTERVAL 3 MONTH)

GROUP BY p.CategoryID, c.CategoryName;

Question 3: Formulate an SQL query to display the employee name and the name of their manager. Assume an `Employees` table that includes `EmployeeID`, `Name`, and `ManagerID`, where `ManagerID` is a foreign key that references `EmployeeID` within the same table. Ensure the query handles employees without a manager.

SELECT e.Name AS EmployeeName,

COALESCE(m.Name, 'No Manager') AS ManagerName

FROM Employees e

LEFT JOIN Employees m ON e.ManagerID = m.EmployeeID;