1-**Python :**

1. How can you replace string spaces with a given character in Python?

a special character can be replaced using replace() method :

string.replace(old, new[, count])

2. How do you identify and deal with missing values? (Identifying missing values)

- Pandas provides convenient functions to handle missing values in tabular data. The isnull() function returns a boolean mask indicating missing values, and the fillna() or dropna() function can be used to fill missing values with a specified value or strategy.

- The numpy library provides functions to work with numerical data. The isnan() function can be used to identify missing values in numeric arrays, and the nan_to_num() function can be used to replace missing values with a specified value.

3. What is the difference between merge, join, and concatenate?

merge() and join() in pandas are used to combine data from different sources based on common columns or their indexes., while concatenate() in numpy is used to combine data along a particular axis.

4. Why use else in the try/except construct in Python?

To executed when no exception occurs in the try block. To perform some actions only if the try block runs successfully without any exceptions.

5. What is the Python "with" statement designed for?

It simplifies the management of common resources like file streams and make the code more readable.

6. What is monkey patching in Python?

modifying or updating a piece of code or class or any module at the runtime to change the behavior or working of a class/ module at the runtime without changing the whole python code.

7. Explain List, Dictionary, and Tuple comprehensions with examples.

- List comprehension is a concise way to create lists , create a new list by specifying an expression and an optional condition. Here's an example:

```
numbers = [1, 2, 3, 4, 5]

squares = [x**2 for x in numbers if x % 2 == 0]

print(squares)  # Output: [4, 16]
```

- **Dictionary comprehension** creates dictionaries or create a new dictionary by specifying key-value pairs and an optional condition. Here's an example:

```
numbers = [1, 2, 3, 4, 5]

squares_dict = {x: x**2 for x in numbers if x % 2 == 0}

print(squares_dict)  # Output: {2: 4, 4: 16}
```

- Tuple comprehension does not exist in Python because Tuples are immutable.

8. What is the difference between a mutable data type and an immutable data type?

Mutable data types can be modified after they are created, while immutable data types cannot be modified.

9. What is `__init__()` in Python?

The __init__() method is the constructor.

(used to initialize the object's attributes or perform any other setup that is required)

10. Given a positive integer num, write a function that returns True if num is a perfect square else False.

```
import math

def is_perfect_square(num):

    #check if the square root is an integer

      sqrt = math.sqrt(num)

      return sqrt == int(sqrt)
```

11. Given an integer n, return the number of trailing zeroes in n factorial n!.

```
def trailing_zeroes(n):

   count = 0

   while n >= 5:

      n //= 5

      count += n

   return count
```

12. Find the missing number in the array. You have been provided with a list of positive integers from 1 to n. All the numbers from 1 to n are present except one number x, and you must find x

```
def find_missing_number(nums):
    n=nums[-1]
    values = (i for i in range(1, n+1) if i not in nums)
    print(*values)
```

2- SQL (PostgreSQL) :

13. Given two tables, `orders` and `customers`, with relevant fields, write an SQL query to retrieve all orders along with the corresponding customer names.

```
SELECT orders.order_id, orders.order_date, customers.customer_name

FROM orders

JOIN customers ON orders.customer_id = customers.customer_id;
```

14. Using a subquery, find the second-highest salary from a table named `employees`.

```
SELECT MAX(salary)

FROM employees

WHERE salary < (SELECT MAX(salary) FROM employees);
```

15. Explain what indexes are in the context of a database. How can you create an index on a specific column to improve query performance?

Indexes in a database are data structures that improve the speed of data retrieval operations on database tables. They are created on specific columns of a table to allow the database management system to quickly locate and access the rows that satisfy the conditions specified in a query. To create an index on a specific column :

```
CREATE INDEX idx_column_name ON table_name (column_name);
```

16. Describe the concept of database normalization. Provide an example of transforming an unnormalized table into third normal form (3NF).

Database normalization is the process of organizing data in a database to eliminate redundancy and improve data integrity. It involves decomposing a table into multiple tables and defining relationships between them , Examples:

<u>**Un-N**</u> (<u>D#</u>,D name, Location,M_name,M_id, tel, <u>C#</u>,C_name,date_comp,nature_comp)

<u>**1NF**</u> **Deps**(D#,D name, Location,M_name,M_id, tel, C#,date_comp,nature_comp)
Customers (C#,C_name)

2NF **Deps**(D#,D name, Location,M_name,M_id,C#, tel)
Customers (C#,C_name)
Comp(C#, D#,date_comp,nature_comp)

3NF **Deps**(D#,D name, Location,M_id, C#, tel)
Customers (C#,C_name)
Comp(C#, D#,date_comp,nature_comp)
Mang (M_name,M_id)

==

Un-N (emp_no, name, department, manager ,project_id,location,weeks_on_project, Proj-Start-Date)

1NF (emp_no(PK) , name , department , dep_manager ,weeks_on_project)
(project_id(PK), Proj-Start-Date , location ,emp_no(FK))

2NF (emp_no , name , department , dep_manage , location)
(project_id ,weeks_on_project,emp_no)
(project_id , Proj-Start-Date)

3NF (emp_no , name , department , location)
(department , dep_manage)
(project_id ,weeks_on_project,emp_no)
(project_id , Proj-Start-Date)

17. Write an SQL query to calculate the average, maximum, and minimum order amounts from an `order_details` table.

```
SELECT AVG(amount) , MAX(amount) , MIN(amount)

FROM order_details;
```

18. Define window functions in SQL and provide an example of using the `ROW_NUMBER()` function to paginate results from a table.

Window functions in SQL perform calculations across a set of rows that are related to the current row.

ROW_NUMBER() function to paginate results:

```
SELECT ROW_NUMBER() OVER (ORDER BY column_name) AS row_number, column_name

FROM table_name

OFFSET 0 LIMIT 10;
```

19. Explain the purpose of Common Table Expressions (CTEs) in SQL. Create a CTE that calculates the total revenue for each month over a year from an `invoices` table.

CTEs are temporary named result sets that can be referenced within a SQL statement. They are primarily used to simplify complex queries, improve query readability, and avoid redundant code.

Recursive queries, also known as recursive common table expressions (CTEs)

```
WITH monthly_revenue AS (
    SELECT EXTRACT(MONTH FROM invoice_date) AS month, SUM(amount) AS total_revenue
    FROM invoices
    WHERE EXTRACT(YEAR FROM invoice_date) = 2023
    GROUP BY month
)
SELECT month, total_revenue
FROM monthly_revenue;
```

20. Discuss what recursive queries are and when they might be used. Write a recursive SQL query to find all the ancestors of a given employee in an organizational hierarchy table.

Used to perform operations that involve recursion. A recursive query allows you to repeatedly query a table based on the results of a previous iteration. It is useful when dealing with tree-like data structures, such as file systems.

```
WITH RECURSIVE Ancestors AS (
    SELECT EmployeeID, ManagerID
    FROM Employee
    WHERE EmployeeID = <given_employee_id>

    UNION ALL

    SELECT e.EmployeeID, e.ManagerID
    FROM Employee e
    JOIN Ancestors a ON e.EmployeeID = a.ManagerID
)
SELECT EmployeeID
FROM Ancestors;
```

22. How can stored procedures improve code organization and reusability in SQL? Provide an example of creating a stored procedure that inserts a new customer into a table.

procedures provide a way to group SQL statements into a reusable and modular unit.

```
CREATE PROCEDURE InsertCustomer
   @FirstName VARCHAR(50),
   @LastName VARCHAR(50),
   @Email VARCHAR(100)
AS
BEGIN
   INSERT INTO Customers (FirstName, LastName, Email)
   VALUES (@FirstName, @LastName, @Email)
END;
```