Most Frequently Asked Pyspark Dataframe Query (Scenario Based)

1.Here is the sample boilerplate code and a dataframe for your reference.

- **Code for creating sparksession**:

```
from pyspark.sql import SparkSession
import getpass
username = getpass.getuser()
spark= SparkSession. \
builder. \
config('spark.ui.port','0'). \
config("spark.sql.warehouse.dir", f"/user/{username}/warehouse"). \
enableHiveSupport(). \
master('yarn'). \
getOrCreate()
```

- **Here is the sample Dataframe to work with:**

```
from pyspark.sql import SparkSession

from pyspark.sql import Row

from pyspark.sql.types import StructType, StructField, IntegerType, StringType, DoubleType

from pyspark.sql.functions import *


# Create the SparkSession

spark = SparkSession.builder.getOrCreate()


# Define the data

data = [Row(1, "John", 30, "Sales", 50000.0),

        Row(2, "Alice", 28, "Marketing", 60000.0),

        Row(3, "Bob", 32, "Finance", 55000.0),

        Row(4, "Sarah", 29, "Sales", 52000.0),
```

```
    Row(5, "Mike", 31, "Finance", 58000.0)
]

# Define the schema
    schema = StructType([
        StructField("id", IntegerType(), nullable=False),
        StructField("name", StringType(), nullable=False),
        StructField("age", IntegerType(), nullable=False),
        StructField("department", StringType(), nullable=False),
        StructField("salary", DoubleType(), nullable=False)
        ])

# Create the DataFrame
    employeeDF = spark.createDataFrame(data, schema)

# Show the DataFrame
    employeeDF.show()
```

Sample Data:

```
+---+-----+---+----------+-------+
| id| name|age|department| salary|
+---+-----+---+----------+-------+
|  1| John| 30|     Sales|50000.0|
|  2|Alice| 28| Marketing|60000.0|
|  3|  Bob| 32|   Finance|55000.0|
|  4|Sarah| 29|     Sales|52000.0|
|  5| Mike| 31|   Finance|58000.0|
+---+-----+---+----------+-------+
```

Question 1:

Calculate the average salary for each department:

```
##Calculate the average salary for each department
```

```
avgSalbyDept = employeeDF.groupBy("department").agg(avg("salary").alias("average_sal_by_dept"))
```

```
avgSalbyDept.show()
```

```
+----------+------------------+
|department|average_sal_by_dept|
+----------+------------------+
|     Sales|           51000.0|
|   Finance|           56500.0|
| Marketing|           60000.0|
+----------+------------------+
```

Question 2:

Add a new column named "bonus" that is 10% of the salary for all employees.

```
##Add a new column named "bonus" that is 10% of the salary for all employees.
```

```
employeeDF = employeeDF.selectExpr('*', 'salary * 0.1 as bonus')
```

```
employeeDF.show()
```

```
+---+-----+---+----------+-------+------+
| id| name|age|department| salary| bonus|
+---+-----+---+----------+-------+------+
|  1| John| 30|     Sales|50000.0|5000.0|
|  2|Alice| 28| Marketing|60000.0|6000.0|
|  3|  Bob| 32|   Finance|55000.0|5500.0|
|  4|Sarah| 29|     Sales|52000.0|5200.0|
|  5| Mike| 31|   Finance|58000.0|5800.0|
+---+-----+---+----------+-------+------+
```

Question 3: Group the data by department and find the employee with the highest salary in each department

```
##Group the data by department and find the employee with the highest salary in each department
```

```
from pyspark.sql.window import Window
from pyspark.sql.functions import row_number
```

```
windowSpec  = Window.partitionBy("department").orderBy("salary")
```

```
highestSalByDept=employeeDF.withColumn("row_number",row_number().over(windowSpec))
```

```
highestSalByDept.filter (highestSalByDept.row_number==1).show()
```

```
+---+-----+---+----------+-------+------+----------+
| id| name|age|department| salary| bonus|row_number|
+---+-----+---+----------+-------+------+----------+
|  1| John| 30|     Sales|50000.0|5000.0|         1|
|  3|  Bob| 32|   Finance|55000.0|5500.0|         1|
|  2|Alice| 28| Marketing|60000.0|6000.0|         1|
+---+-----+---+----------+-------+------+----------+
```

Question 4: Find the top 3 departments with the highest total salary.

```
top3dept = employeeDF.groupBy("department").agg(sum("salary").alias("totalSalBydept")).orderBy(desc("totalSalBydept"))
```

```
top3dept.show()
```

```
+----------+-------------+
|department|totalSalBydept|
+----------+-------------+
|   Finance|     113000.0|
|     Sales|     102000.0|
| Marketing|      60000.0|
+----------+-------------+
```

Question5:

Find the top most department having highest salary

```
##Find the top most department having the highest total salary.
```

```
employee_CTE = employeeDF.groupBy("department").agg(sum("salary").alias("total_sal"))
```

```
windowSpec = Window.orderBy(employee_CTE["total_sal"].desc())
```

```
 department_rnk = employee_CTE.withColumn("rn", row_number().over(windowSpec))
```

```
result = department_rnk.filter(department_rnk["rn"] == 1).select("department")
```

```
result.show()
```

```
+----------+
|department|
+----------+
|   Finance|
+----------+
```

SQL for this problem:

```
WITH employee_CTE AS (
   SELECT department, SUM(salary) AS total_sal FROM employee GROUP BY department
), department_rnk AS (
   SELECT department, ROW_NUMBER() OVER (ORDER BY total_sal DESC) AS rn FROM
employee_CTE
)
SELECT department FROM department_rnk WHERE rn = 1
```

Question 6:

Filter the DataFrame to keep only employees aged 30 or above and working in the "Sales" department

```
above30Sales = employeeDF.filter((employeeDF.age >= 30) & (employeeDF.department == "Sales"))
```

```
above30Sales.show()
```

```
+---+----+---+----------+-------+------+
| id|name|age|department| salary| bonus|
+---+----+---+----------+-------+------+
|  1|John| 30|     Sales|50000.0|5000.0|
+---+----+---+----------+-------+------+
```

Question 7:

Calculate the difference between each employee's salary and the average salary of their respective department

```
#Calculate the difference between each employee's salary and the average salary of their respective department.
```

```
windowSpec = Window.partitionBy("department")
```

```
employeeDF = employeeDF.withColumn("avg_sal_by_dpt", avg(col("salary")).over(windowSpec))
```

```
from pyspark.sql.functions import col
diffSalDf = employeeDF.withColumn("diff_sal_by_dept", col("salary") - col("avg_sal_by_dpt"))
```

```
diffSalDf.show()
```

```
+---+-----+---+----------+-------+--------------+----------------+
| id| name|age|department| salary|avg_sal_by_dpt|diff_sal_by_dept|
+---+-----+---+----------+-------+--------------+----------------+
|  1| John| 30|     Sales|50000.0|       51000.0|         -1000.0|
|  4|Sarah| 29|     Sales|52000.0|       51000.0|          1000.0|
|  3|  Bob| 32|   Finance|55000.0|       56500.0|         -1500.0|
|  5| Mike| 31|   Finance|58000.0|       56500.0|          1500.0|
|  2|Alice| 28| Marketing|60000.0|       60000.0|             0.0|
+---+-----+---+----------+-------+--------------+----------------+
```

8. Calculate the sum of salaries for employees whose names start with the letter "J".

```
##Calculate the sum of salaries for employees whose names start with the letter "J".
```

```
sumSalaries = employeeDF.filter(col("name").startswith("J")).agg(sum(col("salary")).alias("total_salary"))
```

```
sumSalaries.show()
```

```
+------------+
|total_salary|
+------------+
|     50000.0|
+------------+
```

9. Sort the DataFrame based on the "age" column in ascending order and then by "salary" column in descending order

```
##Sort the DataFrame based on the "age" column in ascending order and then by "salary" column in descending order
```

```
sortedDF = employeeDF.orderBy("age", desc("salary"))
```

```
sortedDF.show()
```

```
+---+-----+---+----------+-------+--------------+
| id| name|age|department| salary|avg_sal_by_dpt|
+---+-----+---+----------+-------+--------------+
|  2|Alice| 28| Marketing|60000.0|       60000.0|
|  4|Sarah| 29|     Sales|52000.0|       51000.0|
|  1| John| 30|     Sales|50000.0|       51000.0|
|  5| Mike| 31|   Finance|58000.0|       56500.0|
|  3|  Bob| 32|   Finance|55000.0|       56500.0|
+---+-----+---+----------+-------+--------------+
```

10. Replace the department name "Finance" with "Financial Services" in the DataFrame:

```
##Replace the department name "Finance" with "Financial Services" in the DataFrame
```

```
updatedDF = employeeDF.withColumn("department", when(col("department") == "Finance", "Financial Servic
            .otherwise(col("department")))
```

```
updatedDF.show()
```

```
+---+-----+---+------------------+-------+--------------+
| id| name|age|        department| salary|avg_sal_by_dpt|
+---+-----+---+------------------+-------+--------------+
|  4|Sarah| 29|             Sales|52000.0|       51000.0|
|  1| John| 30|             Sales|50000.0|       51000.0|
|  3|  Bob| 32|Financial Services|55000.0|       56500.0|
|  5| Mike| 31|Financial Services|58000.0|       56500.0|
|  2|Alice| 28|         Marketing|60000.0|       60000.0|
+---+-----+---+------------------+-------+--------------+
```

11. Calculate the percentage of total salary each employee contributes to their respective department.

```
##Calculate the percentage of total salary each employee contributes to their respective department.
```

```
windowSpec = Window.partitionBy("department")
employeeDF = employeeDF.withColumn("total_salary_dept", sum("salary").over(windowSpec))
```

```
percentageContribution = (col("salary") / col("total_salary_dept")) * 100
```

```
employeeDF = employeeDF.withColumn("percentage_contribution", round(percentageContribution, 2))
```

```
employeeDF.show()
```

```
+---+-----+---+----------+-------+-------------+-----------------+----------------------+
| id| name|age|department| salary|avg_sal_by_dpt|total_salary_dept|percentage_contribution|
+---+-----+---+----------+-------+-------------+-----------------+----------------------+
|  4|Sarah| 29|     Sales|52000.0|      51000.0|         102000.0|                 50.98|
|  1| John| 30|     Sales|50000.0|      51000.0|         102000.0|                 49.02|
|  3|  Bob| 32|   Finance|55000.0|      56500.0|         113000.0|                 48.67|
|  5| Mike| 31|   Finance|58000.0|      56500.0|         113000.0|                 51.33|
|  2|Alice| 28| Marketing|60000.0|      60000.0|          60000.0|                 100.0|
+---+-----+---+----------+-------+-------------+-----------------+----------------------+
```