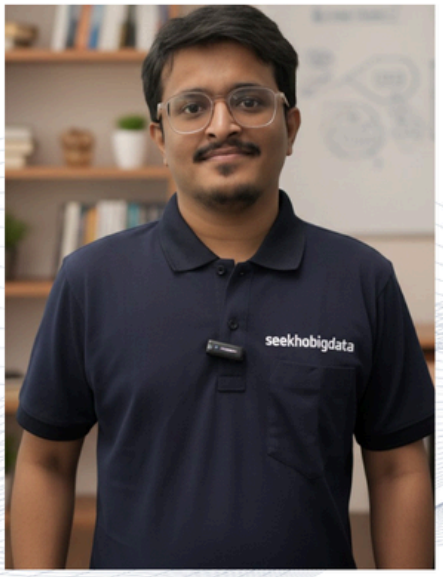
Seekho Bigdata
Data is the New Oil

PySpark Scenario-Based Interview Questions

DAY 1 – DataFrame Basics & Filtering (Foundation Level)



Karthik Kondpak
9989454737

PySpark Scenario-Based Interview Questions

DAY 1 — DataFrame Basics & Filtering (Foundation Level)

✅ Question 1: Filter Active Customers Based on City

🔹 Scenario

You work for an Indian e-commerce company. The business team wants a list of **active customers** who belong to **Bangalore or Hyderabad**.

Sample Data: `customers_df`

customer_id	customer_name	city	status
101	Rahul	Bangalore	Active
102	Priya	Mumbai	Inactive
103	Aman	Hyderabad	Active
104	Neha	Delhi	Active

Expected Output

Only customers who are **Active** and from **Bangalore or Hyderabad**.

PySpark Solution

```
from pyspark.sql.functions import col

result_df = customers_df.filter(
    (col("status") == "Active") &
    (col("city").isin("Bangalore", "Hyderabad"))
)

result_df.show()
```

Explanation

- filter() is used to apply row-level conditions isin() is
- efficient for checking multiple values Logical AND (&)
- ensures both conditions are satisfied

✅ Question 2: Calculate Total Sales Per Customer

🔹 Scenario

An Indian retail company wants to calculate the **total purchase amount per customer** for reporting.

Sample Data: `orders_df`

order_id	customer_id	order_amount
1	101	2000
2	101	1500
3	102	3000
4	103	2500

🧪 PySpark Solution

```
from pyspark.sql.functions import sum

sales_df = orders_df.groupBy("customer_id") \
    .agg(sum("order_amount").alias("total_sales"))

sales_df.show()
```

📝 Explanation

- `groupBy()` groups records per customer
- `sum()` aggregates order values
- `alias()` renames the aggregated column

✅ Question 3: Identify High-Value Customers

🔹 Scenario

The marketing team wants customers whose **total purchase value exceeds ₹4000**.

🧪 PySpark Solution

```
high_value_df = sales_df.filter(col("total_sales")>
4000)

high_value_df.show()
```

📝 Explanation

- Filtering is applied after aggregation
- Helps identify premium customers for offers

✅ Question 4: Add Order Category Column

🔹 Scenario

Classify orders into **Low**, **Medium**, and **High** value categories:

- < 2000 → Low
- 2000–3000 → Medium
- > 3000 → High

🧪 PySpark Solution

```
from pyspark.sql.functions import when

categorized_df = orders_df.withColumn(
    "order_category",
    when(col("order_amount") < 2000, "Low")
    .when(col("order_amount").between(2000, 3000),
```

```
"Medium")
    .otherwise("High")
)

categorized_df.show()
```

📝 Explanation

- `withColumn()` addsderived columns
- `when().otherwise()` implements conditional logic

✅ Question 5: Remove Duplicate Customer Records

🔹 Scenario

Due to data ingestion issues, duplicate customer records exist.

Remove duplicates based on customer_id.

🧪 PySpark Solution

```
clean_df =
customers_df.dropDuplicates(["customer_id"])

clean_df.show()
```

📝 Explanation

- dropDuplicates() ensures unique customer records
- Commonly used in raw ingestion pipelines

Seekho Bigdata

Data is the New Oil

Let's build your Data Engineering journey together!

📩 Call us directly at: 9989454737

https://seekhobigdata.com/