Seekho Bigdata
Data is the New Oil

PySpark Scenario-Based Interview Questions

DAY 2 — Joins in PySpark

Karthik Kondpak
9989454737

PySpark Scenario-Based Interview Questions

📅 DAY 2 — Joins in PySpark

Types of Joins Covered Today

- Inner Join
- Left Join
- Right Join
- Full Outer Join
- Anti Join
- Semi Join
- Join Optimization Basics

✅ Question 1: Fetch Customer Order Details (INNER JOIN)

🔹 Scenario

An Indian e-commerce company wants to generate a report showing **customer name, city, and order amount** only for customers who have placed orders.

📊 Sample Data

`customers_df`

customer_id	customer_name	city
101	Rahul	Bangalore
102	Priya	Mumbai
103	Aman	Hyderabad

`orders_df`

order_id	customer_id	order_amount
1	101	2000
2	101	1500
3	103	3000

🧪 PySpark Solution

```
joined_df = customers_df.join(
    orders_df,
    customers_df.customer_id ==
orders_df.customer_id,
    "inner"
)

joined_df.select("customer_name", "city",
"order_amount").show()
```

📝 Explanation

- inner join returns records present in **both tables**
- Most commonly used join in reporting pipelines

✅ Question 2: List All Customers Even If No Orders (LEFT JOIN)

🔹 Scenario

The business wants to identify **customers who have not placed any orders yet**.

🧪 PySpark Solution

```
left_join_df = customers_df.join(
    orders_df,
    "customer_id",
    "left"
)

left_join_df.show()
```

📝 Explanation

- Left join keeps **all records from the left table**
- Unmatched records from the right table appear as `NULL`

- Useful for churn and engagement analysis

✅ Question 3: Identify Customers Without Orders (LEFT ANTI JOIN)

🔹 Scenario

Marketing wants to target customers who **never placed any order**.

🧪 PySpark Solution

```
no_order_df = customers_df.join(
    orders_df,
    "customer_id",
    "left_anti"
)

no_order_df.show()
```

📝 Explanation

- left_anti returns records from left table **with no match** in right table
- Very efficient compared to filtering NULLs

✅ Question 4: Get Customers Who Have Placed Orders (LEFT SEMI JOIN)

🔹 Scenario

Operations team needs only the **list of customers who have at least one order**, without order details.

🧪 PySpark Solution

```
ordered_customers_df = customers_df.join(
    orders_df,
    "customer_id",
    "left_semi"
)
```

```
ordered_customers_df.show()
```

📝 Explanation

- left_semi returns only columns from the left table
- Acts like an efficient EXISTS condition

✅ Question 5: Combine Customer and Order Data Completely (FULL OUTER JOIN)

🔹 Scenario

Audit team wants to see **all customers and all orders**, even if data is missing on either side.

🧪 PySpark Solution

```
full_df = customers_df.join(
    orders_df,
    "customer_id",
    "full"
```

```
)

full_df.show()
```

📝 Explanation

- Full join keeps **all records from both tables**
- Missing matches are filled with NULL

Question 6: Optimize Join Using Broadcast Join

🔹 Scenario

customers_df is small (50K records) and `orders_df` is very large (500M records). Optimize the join.

🧪 PySpark Solution

```
from pyspark.sql.functions import broadcast

optimized_df = orders_df.join(
```

```
    broadcast(customers_df),
    "customer_id",
    "inner"
)

optimized_df.show()
```

📝 Explanation

- Broadcast avoids shuffle by sending small table to all executors
- Extremely important for performance-related interviews

Let's build your Data Engineering journey together!

Call us directly at: 9989454737

https://seekhobigdata.com/