Seekho Bigdata
Data is the New Oil

DAY 2 — Scenario-Based PySpark Joins

Topic: Customer – Order – Payment Joins

Karthik Kondpak
9989454737

DAY 2 — Scenario-Based PySpark Joins

Topic:Customer –Order –PaymentJoins(RealIndianUse-Cases)

Question 1: Customer Order Details (INNER JOIN)

Scenario

You work for an Indian e-commerce platform (Flipkart-like).

The analytics team wants **customer name, order_id, and order amount** for **only customers who placed orders**.

Sample Data

customers

customer_id	customer_name	city
1	Rahul	Pune
2	Priya	Mumbai
3	Amit	Delhi

orders

order_id	customer_id	amount
101	1	12000
102	2	8000

103	1	15000

Expected Output

customer_name	order_id	amount
Rahul	101	12000
Rahul	103	15000
Priya	102	8000

PySpark Solution

```
result = customers_df.join(

    orders_df,
    customers_df.customer_id ==
orders_df.customer_id,
    "inner"
).select("customer_name", "order_id", "amount")

result.show()
```

Question 2: Customers Without Any Orders (LEFT ANTI JOIN)

Scenario

The marketing team wants to target **customers who never placed any order**.

Expected Output

customer_id

	customer_name	city
3	Amit	Delhi

PySpark Solution

```
result = customers_df.join(

    orders_df,
    customers_df.customer_id ==
orders_df.customer_id,
    "left_anti"
)

result.show()
```

Question 3: Orders With or Without Customers (LEFT JOIN)

Scenario

Due to data quality issues, some orders might exist without customer records.

The audit team wants **all orders**, even if customer info is missing.

Expected Output

```
Orders appear even if customer_name is NULL.
```

PySpark Solution

```
result = orders_df.join(

    customers_df,
    orders_df.customer_id ==
customers_df.customer_id,
    "left"
).select("order_id", "customer_name", "amount")

result.show()
```

Question 4: Order Payment Status (MULTI-TABLE JOIN)

Scenario

You receive payment data from a separate system.

Business wants to know **customer name, order amount, payment status**.

payments

payment_id	order_id	status
9001	101	SUCCESS
9002	102	FAILED
9003	104	SUCCESS

Expected Output

Only orders that have payment records.

PySpark Solution

```
result = customers_df \

    .join(orders_df, "customer_id") \
    .join(payments_df, "order_id") \
    .select("customer_name", "order_id", "amount",
"status")

result.show()
```

✅ Question 5: Orders Without Successful Payment (LEFT JOIN + FILTER)

Scenario

The finance team wants to identify **orders that are unpaid or payment failed**.

Expected Output

Orders with `FAILED` payment or **no payment record**.

PySpark Solution

```
result = orders_df.join(

    payments_df,
    "order_id",
    "left"
).filter(
    (payments_df.status != "SUCCESS")|
(payments_df.status.isNull())
)

result.show()
```

Let's build your Data Engineering journey together!

📩 Call us directly at: 9989454737

🌐 https://seekhobigdata.com/