Seekho Bigdata
Data is the New Oil

DAY 3 — Scenario-Based PySpark

Window Functions (Ranking, Running Totals, Dedup)

Karthik Kondpak
9989454737

DAY 3 — Window Functions (Ranking, Running Totals, Dedup)

✅ Question 1: Rank Employees by Salary Within Each Department

Scenario

You work for an Indian IT company (Infosys-like).

HR wants to **rank employees by salary within each department**.

Sample Data

emp_id	emp_name	dept	salary
1	Rahul	IT IT HR	90000
2	Priya	HR	120000
3	Amit	Finance	70000
4	Neha		70000
5	Suresh		110000

Expected Output

emp_name	dept	salary	rank
Priya	IT	120000	1
Rahul	IT	90000	2
Amit	HR	70000	1
Neha	HR	70000	1

Suresh	Finance	110000	1

PySpark Solution

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

window_spec =
Window.partitionBy("dept").orderBy(col("salary").desc
())

result = df.withColumn("rank",
rank().over(window_spec))
result.show()
```

✅ Question 2: Find Top 2 Highest Paid Employees per Department

Scenario

Management wants to identify **top 2 highest paid employees in each department**.

Expected Output

Only top 2 employees per department.

PySpark Solution

```
from pyspark.sql.functions import dense_rank

window_spec =
Window.partitionBy("dept").orderBy(col("salary").d
esc())

result = (
    df.withColumn("drank",
dense_rank().over(window_spec))
      .filter(col("drank") <= 2)
)

result.show()
```

Question 3: Running Total of Daily Sales Per City

Scenario

You work for an Indian retail chain.

Business wants **running (cumulative) sales per city by date**.

Sample Data

city	date	sales
Mumbai	2024-01-01	5000
Mumbai	2024-01-02	7000

Mumbai	2024-01-03	3000
Delhi	2024-01-01	4000
Delhi	2024-01-02	6000

Expected Output

city	date	sales	running_total
Mumbai	2024-01-01	5000	5000
Mumbai	2024-01-02	7000	12000
Mumbai	2024-01-03	3000	15000
Delhi	2024-01-01	4000	4000
Delhi	2024-01-02	6000	10000

PySpark Solution

```
from pyspark.sql.functions import sum

window_spec = (
    Window.partitionBy("city")
          .orderBy("date")
          .rowsBetween(Window.unboundedPreceding,
Window.currentRow)
)

result = df.withColumn("running_total",
sum("sales").over(window_spec))
result.show()
```

✅ Question 4: Deduplicate Customer Records (Keep Latest Record)

Scenario

Due to multiple data ingestions, customer records are duplicated.

You need to **keep only the latest record per customer**.

Sample Data

customer_id	name	update_time
1	Rahul	2024-01-01
1	Rahul	2024-01-05
2	Priya	2024-01-03

Expected Output

Latest record per customer.

Solution

```
from pyspark.sql.functions import row_number

window_spec =
Window.partitionBy("customer_id").orderBy(col("update_
time").desc())

result = (
    df.withColumn("rn",
```

```
row_number().over(window_spec))
      .filter(col("rn") == 1)
      .drop("rn")
)

result.show()
```

✅ Question 5: Identify Salary Increase Compared to Previous Month

Scenario

HR wants to check **whether employee salary increased compared to the previous month**.

Sample Data

emp_id	month	salary
1	2024-01	60000
1	2024-02	65000
1	2024-03	65000

Expected Output

emp_id	month	salary	prev_salary	increased
1	2024-01	60000	NULL	false
1	2024-02	65000	60000	true
1	2024-03	65000	65000	false

PySpark Solution

```
from pyspark.sql.functions import lag, when


window_spec =
Window.partitionBy("emp_id").orderBy("month")

result = df \
    .withColumn("prev_salary",
lag("salary").over(window_spec)) \
    .withColumn(
        "increased",
        when(col("salary") > col("prev_salary"),
True).otherwise(False)
    )

result.show()
```

Seekho Bigdata

Data is the New Oil

Let's build your Data Engineering journey together!

Call us directly at: 9989454737

https://seekhobigdata.com/