Seekho Bigdata
Data is the New Oil

PySpark Scenario-Based Interview Questions (Complete Notes Series)

DAY 12 — File Formats, Partitioning & Bucketing

Karthik Kondpak
9989454737

📘 PySpark Scenario-Based Interview Questions (Complete Notes Series)

DAY 12 — File Formats, Partitioning & Bucketing (Hive-Style Optimization)

Concepts Covered Today

- File formats: CSV vs JSON vs Parquet vs ORC
- Partitioning (when & how)
- Over-partitioning problem
- Bucketing (Hive-style optimization)
- Partitioning vs Bucketing (interview favourite)

Scenario

You are building a **data lake for an Indian e-commerce company**.

Table: orders

- 1+ billion records
- Queries mostly filter by **order_date, city**
- Frequent joins on **customer_id**

✅ Question 1: Which File Format Should You Choose and Why?

🔹 Interview Question

Why is Parquet preferred over CSV in Spark?

Correct Answer (Short & Strong)

- Columnar format
- Predicate pushdown supported
- Compression built-in
- Faster IO & lower storage cost

File Format Comparison

Format	Use Case	Interview Note
CSV	Raw ingestion	No schema, slow
JSON	Nested data	Semi-structured
Parquet	Analytics	Best choice
ORC	Hive workloads	Similar to Parquet

✅ Question 2: Writing Data in Parquet

PySpark Code

```
orders_ df.wri te.mo de("o verwri te").pa rquet( "/dat a/ord e
rs")
```

Question 3: Partitioning — MOST USED OPTIMIZATION

🔹 Scenario

Queries filter by `order_date`daily.

PySpark Solution

```
orders_df.write \
    .partitionBy("order_date") \
    .mode("overwrite") \
    .parquet("/data/orders")
```

Why Partitioning Helps

- Partition pruning
- Reads only required folders
- Huge scan reduction

Question 4: Over-Partitioning Problem

🔹 Interview Trap

Partitioning by `order_id`or `customer_id`.

❌ Why Wrong?

- Millions of small files
- Metadata overhead

- Slow queries

Question 5: Bucketing (Hive-Style Optimization)

🔹 Scenario

Frequent joins on `customer_id`.

PySpark Solution

```
orders_df.write \
    .bucketBy(32, "customer_id") \
    .sortBy("customer_id") \
    .mode("overwrite") \
    .saveAsTable("orders_bucketed")
```

📝 Why Bucketing Helps

- Reduces shuffle during joins
- Improves join performance

Partitioning vs Bucketing (MOST ASKED)

Feature	Partitioning	Bucketing
Storage	Directory-based	File-based
Use case	Filtering	Joins
Cardinality	Low	High

Seekho Bigdata

Data is the New Oil

Let's build your Data Engineering journey together!

📩 Call us directly at: 9989454737

🌐 https://seekhobigdata.com/