Persistent Systems

Data Engineer Interview (Round 2)

Real Scenario-Based Questions

1 Question: PySpark Coding — Remove Duplicates While Keeping Latest Record

What they asked me

"You have customer records with multiple entries. Deduplicate and keep the latest by timestamp."

What I said

I used **Window + row_number**:

```
from pyspark.sql import Window
import pyspark.sql.functions as F

w = Window.partitionBy("customer_id").orderBy(F.desc("updated_at"))

df_final = df.withColumn("rn", F.row_number().over(w)) \
             .filter("rn = 1") \
             .drop("rn")
```

Tips

- Persistent expects **Window functions**, not groupBy + join (slow).
- Mention "Distributed + scalable solution".

---

2 Question: Implement SCD Type-2 in Delta Lake

What they asked me

"How will you implement SCD2 using Delta Lake?"

What I said

I explained MERGE logic:

```
MERGE INTO dim_customer t
USING updates s
ON t.customer_id = s.customer_id
WHEN MATCHED AND t.hash <> s.hash THEN
  UPDATE SET t.is_current = false, t.end_date = current_date()
WHEN NOT MATCHED THEN
  INSERT *
```

Tips

- Always mention **hash comparison** to detect changes.
- Delta MERGE = expected answer.

---

3 Question: PySpark Optimization Scenario

What they asked me

"Your join is extremely slow. What steps would you take to optimize it?"

What I said

I used 5 optimizations:

1. **Broadcast small dimension table**
2. `df.join(F.broadcast(dim), "id")`
3. Repartition by join key
4. Cache reused DataFrames
5. Avoid wide transformations early
6. Select only needed columns

Tips

- Start with **broadcast join**.
- Persistent LOVES Spark optimization.

---

4 Question: ADF Debug — Pipeline is failing randomly

What they asked me

"A pipeline runs sometimes but fails at other times. How will you debug?"

What I said

Step-by-step approach:

1. Check **Activity logs** → identify which activity fails
2. Verify **Linked Service** Authentication
3. Validate **dataset schema drift**
4. Check **mapping data flow partitions**
5. Check **integration runtime** capacity
6. Review **source-side throttling** (SQL / Salesforce API limits)

Tips

- Mention "Integration Runtime throttling" — common in Persistent projects.

---

5 Question: SQL — Find Second Highest Salary

What they asked me

"Write SQL to fetch the 2nd highest salary without using MAX twice."

What I said

```
SELECT salary
FROM (
        SELECT salary,
                DENSE_RANK() OVER (ORDER BY salary DESC) AS rnk
        FROM employees
) t
WHERE rnk = 2;
```

Tips

- Window functions preferred.
- Avoid `LIMIT 1 OFFSET 1` as main answer.

---

6 Question: Data Partitioning Strategy

What they asked me

"How do you decide partitioning for a Delta table?"

What I said

I consider:

- High-cardinality (NOT preferred for partitions)
- Query patterns (date-based)
- File size expectations
- Avoid over-partitioning

Example:
Partition by `order_date` (year, month) but NOT by `customer_id`.

Tips

- Always warn about "too many partitions → small file problem".

---

7 Question: Data Quality Framework in Databricks

What they asked me

"How do you apply DQ in your Silver zone?"

What I said

DQ Rules applied:

- Null checks
- Duplicate checks
- Pattern checks
- Range checks
- Reference integrity validation

Invalid rows stored in:
`/silver/errors/<table>/<date>/`

Tips

- Persistent LOVES error folder + replay answers.

---

8 Question: ADF – How do you implement Incremental Copy?

What they asked me

"You need to move only changed records into ADLS. How do you design this in ADF?"

What I said

1. Maintain `last_watermark` in metadata table
2. Parameterize Copy Activity
3. Use dynamic query:
4. `SELECT * FROM table WHERE last_updated > @watermark`
5. Update watermark after success
6. Process in Databricks with MERGE

Tips

- Don't say "full load daily" — they reject immediately.

---

9 Question: Handle Small File Problem in ADLS

What they asked me

"How do you solve the small file problem?"

What I said

Solutions:

- `df.repartition(10)` or `.coalesce(10)`
- Enable **Auto Optimize + Auto Compaction**
- Use Delta **OPTIMIZE ZORDER**

```
OPTIMIZE silver.orders ZORDER BY (order_date)
```

Tips

- ZORDER is a Persistent favorite keyword.

---

10 Question: Join Two Delta Tables Efficiently (PySpark)

What they asked me

"Your join on two Delta tables is slow. What will you check?"

What I said

1. Is one table small? → Broadcast
2. Is data skewed? → Repartition or salt
3. Are both tables partitioned properly?
4. Select only needed columns
5. Enable AQE (Adaptive Query Execution)

```
spark.conf.set("spark.sql.adaptive.enabled", True)
```

Tips

- Mention **AQE** → they love updated Spark knowledge.