05 DBFS & Delta Lake Concepts

DBFS & Delta Lake Concepts

Basic Level Questions (1-8)

1. What is DBFS (Databricks File System) and how does it differ from traditional file systems?

What to look for: Understanding of distributed file system, abstraction layer over cloud storage, FUSE mount capabilities, and integration with Databricks workspace.

2. Explain the different mount points and directories available in DBFS.

What to look for: `/mnt`, `/tmp`, `/FileStore`, `/databricks-datasets`, and their specific purposes and access patterns.

3. What is Delta Lake and what problems does it solve in data lakes?

What to look for: ACID transactions, schema enforcement, time travel, data versioning, and reliability improvements over traditional data lakes.

4. How do you create and read a Delta table in Databricks?

What to look for: Basic Delta table creation syntax, reading operations, and understanding of Delta format structure.

5. What are the key features that make Delta Lake different from Parquet files?

What to look for: Transaction log, ACID properties, schema evolution, time travel, and data reliability features.

6. How do you mount external storage (Azure Blob Storage, S3) to DBFS?

What to look for: Mount commands, authentication methods, and best practices for secure mounting.

7. What is the Delta transaction log and why is it important?

What to look for: JSON transaction logs, metadata tracking, atomicity guarantees, and change tracking mechanisms.

8. Explain the concept of time travel in Delta Lake and provide a basic example.

What to look for: Version history, timestamp querying, VERSION AS OF and TIMESTAMP AS OF syntax.

Intermediate Level Questions (9-17)

9. How would you implement incremental data loading using Delta Lake?

What to look for: MERGE operations, upsert patterns, change data capture, and efficient incremental processing strategies.

10. Explain the different types of operations available in Delta Lake (INSERT, UPDATE, DELETE, MERGE).

What to look for: SQL and DataFrame API syntax, performance considerations, and use case scenarios for each operation.

11. How do you handle schema evolution in Delta Lake tables?

What to look for: Schema enforcement, schema evolution modes, column addition/modification, and backward compatibility.

12. What are Delta Lake optimizations and how do you implement them?

What to look for: OPTIMIZE command, Z-ORDER clustering, file compaction, and performance tuning strategies.

13. How would you implement data retention and cleanup policies for Delta tables?

What to look for: VACUUM command, retention periods, storage optimization, and compliance requirements.

14. Explain how to use Delta Lake's change data feed (CDF) feature.

What to look for: Change tracking, CDC implementation, downstream processing, and real-time analytics use cases.

15. How do you monitor and troubleshoot Delta Lake table performance issues?

What to look for: Table statistics, file size analysis, query performance metrics, and optimization techniques.

16. What are the best practices for partitioning Delta Lake tables?

What to look for: Partition strategy, performance implications, query optimization, and maintenance considerations.

17. How would you implement data quality checks and constraints in Delta Lake?

What to look for: Check constraints, data validation, schema enforcement, and error handling strategies.

Advanced/Difficult Level Questions (18-25)

18. Design a medallion architecture (Bronze, Silver, Gold) using Delta Lake with proper data governance.

What to look for: Layered data architecture, data quality progression, governance policies, and pipeline orchestration.

19. How would you implement a multi-table transaction system using Delta Lake?

What to look for: Transaction coordination, consistency guarantees, rollback mechanisms, and distributed transaction patterns.

20. Explain how you would handle concurrent writes and conflict resolution in Delta Lake.

What to look for: Optimistic concurrency control, conflict detection, retry mechanisms, and isolation levels.

21. How would you design a disaster recovery strategy for Delta Lake tables across multiple regions?

What to look for: Cross-region replication, backup strategies, failover mechanisms, and data consistency guarantees.

22. Implement a solution for handling late-arriving data and out-of-order events in Delta Lake.

What to look for: Watermarking strategies, event time processing, data reconciliation, and consistency maintenance.

23. How would you optimize Delta Lake for streaming workloads with high throughput requirements?

What to look for: Structured streaming integration, micro-batch optimization, checkpoint management, and latency considerations.

24. Design a data lineage and audit trail system using Delta Lake's features.

What to look for: Transaction history tracking, metadata management, compliance auditing, and governance frameworks.

25. How would you implement a cost-effective archival strategy for Delta Lake tables?

What to look for: Lifecycle management, storage tiering, compression strategies, and access pattern optimization.

Technical Deep-Dive Scenarios

Scenario A: Performance Optimization

"Your Delta table queries are running slowly despite having 10TB of data. Walk me through your optimization approach."

What to look for:

- File size analysis and compaction strategies
- Z-ORDER clustering implementation
- Partition strategy evaluation
- Query pattern analysis
- Statistics management

Scenario B: Data Corruption Recovery

"A Delta table got corrupted due to a failed write operation. How would you recover the data?"

What to look for:

- Time travel capabilities
- Transaction log analysis
- Rollback procedures
- Data validation techniques
- Prevention strategies

Scenario C: Real-Time Analytics

"Design a real-time analytics solution that processes streaming data into Delta Lake with sub-second latency requirements."

What to look for:

- Structured streaming configuration
- Micro-batch optimization
- Change data feed utilization
- Performance tuning strategies

Hands-On Code Examples

26. DBFS Operations

```
# Review and explain this DBFS code:
dbutils.fs.ls("/mnt/datalake/")
dbutils.fs.cp("/mnt/source/file.parquet", "/mnt/target/")
dbutils.fs.mount(
    source="abfss://container@storage.dfs.core.windows.net/",
    mount_point="/mnt/datalake",
    extra_configs={"fs.azure.account.key.storage.dfs.core.windows.net": "key"}
)
```

27. Delta Lake Operations

```
-- Optimize this Delta Lake query:
MERGE INTO target_table t
USING source_table s ON t.id = s.id
WHEN MATCHED THEN UPDATE SET *
WHEN NOT MATCHED THEN INSERT *;

-- How would you improve performance?
```

28. Schema Evolution

```
# Handle this schema evolution scenario:
# Original schema: id, name, age
# New schema: id, name, age, email, phone
# How do you safely evolve the schema?
```

Troubleshooting Questions

29. Common Issues

- "How would you debug a Delta table that's not updating despite successful MERGE operations?"
- "What steps would you take if VACUUM is running too slowly?"
- "How do you handle schema conflicts during Delta table writes?"

30. Performance Problems

- "Your Delta table scans are taking too long. What's your diagnostic approach?"
- "How would you identify and fix small file problems in Delta tables?"
- "What causes Delta Lake write operations to be slow, and how do you optimize them?"

Integration and Advanced Concepts

31. Multi-Format Integration

- "How would you migrate from Parquet to Delta Lake with zero downtime?"
- "Explain your approach to reading Delta tables from external systems."

32. Governance and Security

- "How do you implement row-level security with Delta Lake?"
- "Design a data governance framework for Delta Lake tables."

33. Cost Optimization

- "What strategies would you use to reduce Delta Lake storage costs?"
- "How do you balance performance and cost in Delta Lake implementations?"

Real-World Problem Solving

Scenario D: Migration Challenge

"You need to migrate 100+ Parquet-based pipelines to Delta Lake. What's your strategy?"

Scenario E: Compliance Requirements

"Your Delta Lake solution needs to comply with GDPR right-to-be-forgotten requirements. How do you implement this?"

Scenario F: High Concurrency

"Multiple teams need to write to the same Delta table simultaneously. How do you handle this?"

Follow-Up Questions:

- "How would you handle Delta table corruption in production?"
- "What's your approach to testing Delta Lake operations before deployment?"
- "How do you monitor Delta Lake table health and performance?"
- "Explain your strategy for managing Delta Lake table lifecycles."
- "How would you implement automated optimization for Delta tables?"