08 Delta Lake & Data Management

Delta Lake Architecture, Medallion Pattern, Schema Evolution, Time Travel & ACID Transactions

Basic Level Questions (1-10)

1. What is Delta Lake and how does it differ from traditional data lake storage formats? *Expected Answer: Delta Lake is an open-source storage framework that brings ACID transactions, scalable metadata handling, and unified streaming/batch processing to data lakes. Unlike traditional formats, it provides transactional guarantees and versioning.*

2. Explain the Bronze, Silver, and Gold layers in the medallion architecture. *Expected Answer: Bronze (raw data ingestion), Silver (cleaned/validated data), Gold (business-ready aggregated data). Each layer represents increasing data quality and business value.*

3. What are ACID transactions and why are they important in data engineering? *Expected Answer: Atomicity, Consistency, Isolation, Durability. Ensures data integrity, prevents partial writes, maintains consistency across concurrent operations.*

4. What is schema evolution in Delta Lake? *Expected Answer: Ability to change table schema over time without breaking existing queries or data. Supports adding columns, changing data types (with compatibility), and renaming columns.*

5. What is time travel in Delta Lake and what are its use cases? *Expected Answer: Ability to query historical versions of data using version numbers or timestamps. Use cases include data recovery, auditing, reproducible ML experiments, and debugging.*

6. How does Delta Lake handle concurrent writes to the same table? *Expected Answer: Uses optimistic concurrency control with transaction logs. Each write operation creates a new version, and conflicts are resolved through retry mechanisms.*

7. What file formats does Delta Lake use internally? *Expected Answer: Parquet files for data storage, JSON files for transaction logs, and checkpoint files for metadata optimization.*

8. What is the transaction log in Delta Lake? *Expected Answer: A series of JSON files that record every change made to the table, enabling ACID transactions, time travel, and metadata management.*

9. How do you create a Delta table? *Expected Answer: Using DataFrame.write.format("delta").save() or SQL CREATE TABLE USING DELTA statements.*

10. What are the benefits of using the medallion architecture pattern? *Expected Answer: Separates concerns, enables incremental data processing, provides data quality gates, supports different consumer needs, and enables easier debugging and maintenance.*

Intermediate Level Questions (11-20)

11. How would you implement schema evolution to add a new column to an existing Delta table with historical data? *Expected Answer: Use ALTER TABLE ADD COLUMN or mergeSchema option during writes. Historical data will have null values for new columns, and the schema change is tracked in the transaction log.*

12. Explain the concept of checkpointing in Delta Lake and when it occurs. *Expected Answer: Checkpoint files consolidate transaction log entries to improve read performance. Occurs automatically every 10 commits by default, can be configured or triggered manually.*

13. How do you handle schema enforcement in Delta Lake? *Expected Answer: Delta Lake enforces schema by default, rejecting incompatible writes. Can be controlled using mergeSchema option or schema evolution settings.*

14. What is the difference between MERGE, INSERT, and UPSERT operations in Delta Lake? *Expected Answer: INSERT adds new records, MERGE conditionally inserts/updates/deletes based on* *conditions, UPSERT is a specific type of merge that inserts new records or updates existing ones.*

15. How would you implement a slowly changing dimension (SCD) Type 2 pattern using Delta Lake? *Expected Answer: Use MERGE operations with conditions to insert new records for changes and update existing records with end dates. Leverage Delta Lake's time travel for historical tracking.*

16. Explain how to optimize Delta Lake tables for better query performance. *Expected Answer: Use OPTIMIZE command for file compaction, Z-ORDERING for data clustering, proper partitioning strategies, and regular VACUUM operations to remove old files.*

17. How do you handle late-arriving data in the medallion architecture? *Expected Answer: Design idempotent pipelines, use watermarking strategies, implement backfill processes, and leverage Delta Lake's time travel for data correction.*

18. What are the different ways to read historical data using time travel? *Expected Answer: VERSION AS OF, TIMESTAMP AS OF syntax in SQL, or using versionAsOf and timestampAsOf options in DataFrame reads.*

19. How would you implement data quality checks between Bronze and Silver layers? *Expected Answer: Use Delta Lake constraints, implement validation logic, use expectations frameworks, and leverage MERGE operations with quality conditions.*

20. Explain the role of the Delta Lake transaction protocol. *Expected Answer: Defines how readers and writers interact with Delta tables, ensures ACID properties, handles concurrent access, and maintains consistency across operations.*

Advanced/Difficult Level Questions (21-30)

21. Design a real-time streaming architecture using Delta Lake that handles both batch and streaming data with exactly-once processing guarantees. *Expected Answer: Use structured streaming with checkpointing, implement idempotent operations, use Delta Lake's ACID properties, design proper watermarking strategies, and implement comprehensive monitoring.*

22. How would you implement a multi-table transaction across different Delta tables while maintaining ACID properties? *Expected Answer: Use application-level transaction coordination, implement compensating transactions, leverage Delta Lake's atomic operations, and design proper rollback mechanisms.*

23. Explain how you would handle schema evolution conflicts when multiple teams are writing to the same Delta table concurrently. *Expected Answer: Implement schema governance policies, use schema registry, establish team coordination protocols, implement automated conflict resolution, and use feature flags for schema changes.*

24. Design a data lineage and impact analysis solution for a complex medallion architecture with 100+ Delta tables. *Expected Answer: Implement metadata tracking, use Delta Lake's transaction log for lineage, create dependency graphs, implement automated impact analysis, and integrate with data catalog solutions.*

25. How would you implement a disaster recovery strategy for Delta Lake tables with RPO/RTO requirements? *Expected Answer: Use cross-region replication, implement automated backup strategies, leverage Delta Lake's time travel for point-in-time recovery, design failover procedures, and implement monitoring for data consistency.*

26. Explain how you would optimize a medallion architecture for cost while maintaining performance requirements. *Expected Answer: Implement intelligent partitioning, use appropriate storage tiers, optimize file sizes, implement data lifecycle policies, use spot instances for processing, and implement usage-based scaling.*

27. How would you handle schema evolution in a scenario where downstream consumers cannot handle breaking changes? *Expected Answer: Implement schema versioning, maintain backward compatibility, use schema evolution policies, implement gradual rollout strategies, and provide migration tools for consumers.*

28. Design a solution for handling PII data in Delta Lake while maintaining compliance with data privacy regulations. *Expected Answer: Implement column-level encryption, use Delta Lake's time travel for audit trails, implement data masking, design proper access controls, and implement right-to-be-forgotten capabilities.*

29. How would you implement a metadata-driven medallion architecture that can automatically adapt to new data sources? *Expected Answer: Create configuration-driven pipelines, implement schema inference, use Delta Lake's schema evolution, design template-based transformations, and implement automated testing frameworks.*

30. Explain how you would troubleshoot and resolve a scenario where Delta Lake time travel queries are performing poorly. *Expected Answer: Analyze transaction log size, implement checkpoint optimization, review file structure, optimize query patterns, implement proper indexing strategies, and consider table maintenance procedures.*