Data Engineering Interview Questions

Ankita Gulati

Shubh Goyal

Job Details

- **Position:** Senior Data Engineer
- **Experience:** 5+ years
- **Location:** Bangalore
- **Work mode:** Office
- **Compensation:** ₹50+ LPA
- **Total Rounds:** 6
- **Top Required Skills:**

1. SQL
2. PySpark / Python
3. Cloud Data Engineering
4. ETL / Data Modeling
5. Big Data & Streaming
6. System Design

Ankita Gulati

Shubh Goyal

Round 1
Data Structures, Algorithm & SQL

1. Explain the difference between OLTP and OLAP systems and give two examples of when you'd use each.
2. What are the differences between ETL and ELT? When would you use ELT over ETL?
3. Describe star and snowflake schemas — when would you choose one over the other?
4. Write a production-ready SQL query (single statement) that removes duplicate rows from a table while keeping the most recent record per user_id (give column assumptions).
5. Using window functions, write a query that returns the latest event per user and flags ties. (Use ROW_NUMBER() or RANK() as appropriate.)
6. Explain partition pruning and predicate pushdown. How do they improve query performance in systems like Presto/Hive/BigQuery?

Ankita Gulati Shubh Goyal

7. How do indexes, partitions, and distributed joins each affect query execution time? Show one example of a performance improvement you would expect from each.

8. When should you use a Common Table Expression (CTE) vs. a derived table (subquery) vs. a temporary table? Give a short example for readability and performance trade-offs.

9. Explain sessionization: how would you compute sessions from a clickstream table ?

10. Describe cohort analysis at a high level and write the SQL approach to compute weekly cohorts and retention rates.

11. Explain how you would deduplicate large data sets in a distributed environment (e.g., Spark or Hive). Mention memory, shuffle, and partitioning considerations.

12. “Find the first occurrence of 8 in an array” — implement an optimized C++ solution and explain its time and space complexity.

Ankita Gulati

Shubh Goyal

Round 2
Techno-Managerial

1. Implement a two-pointer solution for: given a sorted array and a target sum, return indices of the two numbers that add to the target. Explain complexity.
2. Binary search variants: show how to find the first index of a value ≥ x in a sorted array (explain edge cases).
3. Sliding window: Given an array of positive integers, implement an optimized algorithm to find the length of the smallest subarray with sum ≥ S.
4. Dynamic Programming: implement subset-sum (or knapsack) in O(n·W) and explain how to optimize space to O(W).
5. Graphs: implement BFS and DFS; then explain and implement a shortest-path algorithm for a weighted DAG (topological shortest path).
6. Advanced DP: provide an example of interval DP (e.g., optimal matrix chain multiplication) and explain state transitions.

Ankita Gulati

Shubh Goyal

7. Implement an algorithm to generate all unique permutations of a list with duplicates and explain pruning strategies.
8. Geometry: given points, implement a convex hull algorithm and explain complexity.
9. Convert “find first occurrence of 8 in optimized way c++” into a general interview prompt: given an unsorted array with possible duplicates, design an approach to find the first occurrence of a value with optimal expected time — discuss indexing or hashing tradeoffs.
10. Coding follow-up (system-style): how would you adapt your algorithm to run in parallel or on streamed chunks (explain synchronization/sharding issues)?

Ankita Gulati

Shubh Goyal

Round 3
Advanced SQL & Data Analysis

1. Write a single SQL query to compute rolling 7-day active users per day for the past 90 days (assume event_date, user_id). Explain how you would optimize for a very large table.
2. Write a query that computes daily cohort retention and also returns a “cohort-size-adjusted” retention metric (explain the math).
3. Use window functions to compute the nth highest sale per product and explain the difference between RANK() and DENSE_RANK().
4. Write a SQL query using ROW_NUMBER() to deduplicate records keeping the latest updated_at timestamp.
5. Show how to compute approximate percentiles (p50, p95, p99) in SQL for large datasets; discuss trade-offs between exact and approximate methods.
6. Recursive CTEs: write a query that flattens a hierarchical manager-report table into an employee → top-level manager mapping.

Ankita Gulati Shubh Goyal

7. Explain materialized views and when to use them; write an example that pre-aggregates hourly metrics for faster dashboard queries.

8. Query plan interpretation: given a sample execution plan (describe typical operators — filter, scan, hash join, nested loop), explain steps you would take to reduce costly shuffles or nested loops.

9. Schema-design question: when would you choose columnar storage (Parquet/ORC) vs. row-based (Avro/JSON)? Mention compression and read patterns.

10. Write an optimized join strategy combining a very small lookup table and a large fact table — show both broadcast (map-side) and partitioned join patterns.

Ankita Gulati

Shubh Goyal

Round 4
Foundational Statistics

1. Given a wide table with missing values, write a Python (pandas) function to impute_missing by column using a combination of median for numeric columns and mode for categoricals. Explain OOM mitigation strategies for large tables.
2. Implement feature engineering steps in Python for a time-series ML model: lag features, rolling means, and time-since-last-event. Show vectorized solution and explain why it's preferred over apply.
3. Data reshaping: show how to pivot and melt dataframes for a model input and explain memory-efficient approaches for very large data (e.g., chunking, dtypes downcasting).
4. Write a generator-based Python function that streams CSV rows, applies lightweight transformations, and yields compressed Parquet file parts for downstream processing.

Ankita Gulati Shubh Goyal

5. Explain how you would find and fix a hotspot in Python ETL that spends most time in deserialization. Mention tools (profiler, memory profiler) and strategies (C extensions, multiprocessing, PySpark).

6. When and how would you use multithreading vs. multiprocessing vs. async IO in an ETL task? Give a code sketch for each case.

7. Unit/integration testing: write a pytest unit test for a transformation function and describe how you'd mock external data sources.

8. Data validation: design a function that asserts data contracts (column presence, data types, row-count thresholds) and integrates with CI/CD.

9. Advanced Python coding prompt (live): given a log file feed, implement an efficient deduplication + last-seen aggregator that uses bounded memory (explain algorithm/design).

10. Explain how you'd implement idempotent transformations in your Python ETL jobs.

Ankita Gulati

Shubh Goyal

Round 5
Data Structures & Algorithms

1. What are the differences between RDD, DataFrame, and Dataset in Spark? When would you use each?
2. Explain Catalyst optimizer and Tungsten — how do they help performance?
3. repartition() vs coalesce() — describe semantics and give concrete use cases.
4. Explain broadcast joins vs shuffle joins and when to use each; include cost trade-offs.
5. How do you detect and fix data skew in a Spark job? Provide at least three mitigation strategies.
6. Explain checkpointing and write-ahead logs — when are they required in streaming jobs?
7. Spark streaming: explain event-time processing, watermarks, and late-arriving data handling. Provide a code sketch for windowed aggregations with allowed lateness.
8. How do you design Spark jobs to be fault-tolerant and to provide exactly-once semantics (if possible)? Discuss idempotency and external sinks.

Ankita Gulati Shubh Goyal

9. Partitioning strategy: how would you choose partition keys for a fact table intended for daily batch processing and frequent user-level queries?
10. Implement a Spark job that merges an hourly batch into a daily partitioned table while ensuring schema evolution and manifest updates.

Ankita Gulati

Shubh Goyal

Round 6
Data Design & Architecture

1. Explain Kafka topics, partitions, consumer groups, offsets, and how these concepts impact throughput and scaling.
2. Kafka design question: how would you design topic partitioning for clickstream ingestion at 500k events/sec? Explain producers, consumers, and partition key trade-offs.
3. Compare Kinesis, Kafka (MSK), and SQS — when would you pick each for ingestion?
4. Change Data Capture (CDC): explain Debezium and how you would use CDC to populate a data warehouse like Snowflake or BigQuery. Discuss schema evolution and ordering guarantees.
5. Describe exactly-once, at-least-once, and at-most-once delivery semantics and give example architectures that achieve or approximate each.
6. Stream processing frameworks: compare Kafka Streams, Flink, and Spark Structured Streaming for stateful aggregations and low-latency processing.

Ankita Gulati Shubh Goyal

7. Design challenge: build a fault-tolerant, consumer-backpressure-safe pipeline from Kafka → stream processor → datastore, and explain how you'd handle reprocessing after failures.

8. Monitoring: what metrics do you track for Kafka clusters, stream processors, and consumers? How would you alert on consumer lag?

9. Security: explain how to secure Kafka (TLS, ACLs, encryption-at-rest) and considerations for multi-tenant clusters.

10. Implement a consumer that reads from Kafka, deduplicates events by event_id within a 24-hour window, and writes to a sink with idempotent writes.

11. Design an end-to-end clickstream ingestion and analytics pipeline that supports both near-real-time dashboards (≤ 30s latency) and batch analytics (daily aggregates). Provide architecture diagram, component choices, fault-tolerance, and monitoring.

12. Design a CDC system to stream changes from MySQL/Postgres into Snowflake (or BigQuery). Explain ordering, transactional guarantees, schema changes, and how you'd ensure idempotent writes.

Ankita Gulati **Shubh Goyal**

Thank You

Best of luck with your upcoming interviews — you've got this!

Ankita Gulati

Shubh Goyal