Single-Source Data Engineering System Design Interview Guide (with Monitoring)

This document is a complete interview-ready reference for Senior, Staff, and Principal Data Engineer roles (Atlassian, SEEK, FAANG). It covers scalability, reliability, maintainability, efficiency, optimisation, explicit trade-offs, and now includes a full production monitoring and troubleshooting playbook mapped to Databricks, Spark, and Delta Lake.

10. Monitoring & Troubleshooting – How Real Systems Are Operated

Core Monitoring Dimensions

Every pipeline stage must be monitored across freshness, volume, quality, performance, and cost. Only monitoring success/failure is insufficient at scale.

Ingestion Monitoring

Monitor freshness lag, throughput, partition arrival, schema drift, and small file counts. If ingestion is slow, separate read vs compute vs write bottlenecks using Spark UI.

When Ingestion Runs Slow – What To Do

Check Spark stages for skew or spills, enforce schema, compact files, and fix partitioning before scaling compute.

Transformation Monitoring

Track runtime per stage, shuffle volume, skew indicators, and spill metrics. Most slowdowns are caused by skewed joins or excessive shuffles.

When Transformation Runs Slow – What To Do

Prune columns early, push filters down, broadcast only truly small dimensions, handle skew with salting or pre-aggregation.

Backfill & Idempotency Monitoring

Monitor merge file counts, rewritten partitions, and backfill duration. Slow backfills usually indicate missing partition pruning.

Storage Health Monitoring

Track file counts, average file size, partition size distribution, and OPTIMIZE cadence. Small files and over-partitioning are common long-term degradation causes.

Data Quality Monitoring

Monitor null rates, constraint violations, schema drift, and outliers. Gate publishing on hard checks and alert on soft checks.

Observability & SLAs

Monitor freshness SLAs, lag detectors, skew detectors, and lineage impact. Observability is how failures are detected before users are impacted.

Spoken Interview Answer

When a job runs slow, I first determine whether it is data growth, skew, or orchestration delay. I use Spark UI to isolate the expensive stage, fix query shape and data layout, and only then scale compute if needed.