Streaming Consumer Requirements – Interview Cheat Sheet

What you own as the consumer team

• Confirm the event contract (event_id, event_time, entity key, schema_version, CDC ordering keys).

• Design for delivery semantics (assume at-least-once; implement idempotent sinks and dedup).

• Define ordering assumptions (ordering per partition only; align partition key with correctness needs).

• Handle late/out-of-order data (watermarks, bounded state, backfill/replay strategy).

• Enforce schema + evolution (parse with expected schema; quarantine invalid; contract tests).

• Ensure reliable progress tracking (durable checkpoints/offset commits; safe restarts).

• Scale consumption (parallelism vs partitions; autoscaling; backpressure awareness).

• Observability + SLAs (freshness, lag, DLQ/error rate, completeness, duplicate rate).

Minimum event fields to request

Field	Why it matters
event_id	Deduplication, idempotent writes, replay safety
event_time	Event-time processing, watermarking, business correctness
entity_key (e.g., order_id/user_id)	Partitioning and ordering per entity
schema_version	Safe schema evolution and compatibility checks
source_system / event_type	Routing, lineage, analytics segmentation
op + sequence (CDC only)	Correct merges (I/U/D) and deterministic ordering

Key SLIs to monitor (tie to SLAs)

• Freshness: now – max(event_time_processed)

• Consumer lag / processing delay (broker lag, micro-batch delay)

• Error/DLQ rate (invalid records per minute)

• Completeness (expected vs actual volume where definable)

• Duplicate rate (dedup hits / total events)

• Pipeline success rate (streaming uptime and batch completions)

Ownership split (platform/producer vs consumer team)

Owned by platform/producer (mostly)	Owned by consumer team (you)
Broker durability/replication, retention, partition provisioning, producer acks/retries	Idempotent processing, dedup, event-time correctness, late data h
Publishing schema versions / contracts (producer)	Schema enforcement gates, quarantine/DLQ, contract tests in CI

Publishing partition key strategy (often platform)	Scaling consumers/executors, managing lag, observability + alerti

30-second spoken answer (memorize)

“As the consumer team, I first confirm the event contract: unique event_id, event_time, partition key per entity, schema_version, and CDC ordering keys if applicable. I assume at-least-once delivery, so I ensure idempotent writes and deduplication. I define lateness tolerance and handle out-of-order events using event-time processing and watermarks, with a replay/backfill plan for extreme lateness. I store durable checkpoints for safe restarts and scale parallelism with partitions while monitoring lag. Finally, I enforce schema compatibility and quarantine bad records, and I monitor freshness, lag, and DLQ rate against SLAs.”