Kafka Interview - Day 4 Revision Notes (May 1)

1. What is CooperativeStickyAssignor and how does it help?

It enables incremental rebalancing by only revoking partitions that need to be reassigned. This reduces downtime and consumer lag during rebalancing.

2. What is the trade-off of increasing session.timeout.ms?

Higher session timeout reduces unnecessary rebalances but delays detection of dead consumers. It's a balance between stability and fault detection.

3. How does compression affect producer/consumer performance?

Compression reduces payload size and improves throughput, but adds CPU overhead. Choose codecs based on workload: Snappy for speed, GZIP for size.

4. How is exactly-once semantics achieved in Kafka?

By enabling idempotent producers, transactions, and committing consumer offsets within the same transaction boundary using Kafka's transactional APIs.

5. How do you monitor and troubleshoot Kafka consumer lag?

Use tools like Burrow or Prometheus. Track metrics like `consumer_lag`, `records-lag-max`, and alert when thresholds are breached.

6. How to avoid partition skew or uneven data distribution?

Choose the right assignor (e.g., StickyAssignor), use good partitioning keys, and monitor throughput per partition to avoid hotspots.

7. What is ISR and how does it ensure durability?

ISR (In-Sync Replicas) are brokers that are fully caught up with the leader. Kafka only acknowledges writes when min.insync.replicas are met to ensure durability.

8. How do you ensure consistency with multiple consumer groups?

Use committed offsets, exactly-once delivery (if needed), and ensure schema validation to maintain consistency across multiple consumers.

9. How do you achieve fault tolerance in Kafka architecture?

Use replication, enable retries with backoff, monitor ISR, and have high availability consumers and producers to handle broker or consumer failure.

10. How to handle out-of-order messages in Kafka?

Use partitioning keys to enforce ordering within partitions. For full ordering, use external sequencing or deduplication logic downstream.

11. How to reduce end-to-end latency in Kafka pipelines?

Use compression, batch records, reduce linger.ms, avoid excessive rebalancing, and fine-tune producer/consumer fetch configurations.

12. How do you monitor Kafka broker performance?

Track metrics like `under_replicated_partitions`, `active_controller_count`, CPU usage, GC time, and use Grafana or Confluent Control Center.