Kafka Day 9 – Interview Revision Notes

1. Core Consumer Tuning Configs

Config	Purpose	When to Tune
`fetch.min.bytes`	Broker waits for enough data	Low-volume topics
`fetch.max.wait.ms`	Max wait for above	Tune for latency control
`fetch.max.bytes`	Upper limit of data per fetch	For large messages
`max.poll.records`	Poll batch size	Boost throughput if lagging
`max.poll.interval.ms`	Max processing time	Increase if consumer is slow
`session.timeout.ms`	Consumer liveness timeout	Tune for GC or network delay
`heartbeat.interval.ms`	Heartbeat frequency	Should be < session timeout

---

2. Rebalance Optimization

- **Static Membership (`group.instance.id`)**: Avoids rebalances during restart.

Cooperative Sticky Assignor: Enables incremental rebalance.
Config:

properties
CopyEdit
`partition.assignment.strategy=org.apache.kafka.clients.consumer.CooperativeStickyAssignor`

-

---

3. assign() vs subscribe()

- **assign()**: Manual partition control. No auto rebalancing.

- **subscribe()**: Kafka handles dynamic partition assignment and rebalancing.

---

4. Offset Commit Strategies

- **Auto-commit**: Risky, easy to lose messages if crash happens after commit.
- **Manual commit**: Safer. Commit **after processing**.
- **Sync vs Async**: Sync = safe but slow, Async = fast but needs retry logic.

---

5. Lag Debugging Scenario

Lag spikes daily at 6 PM — Steps:

- Check:
    - Consumer throughput, GC logs
    - Broker CPU/Disk I/O
    - Downstream system (DB/API) performance
    - Any rebalances happening?
- Fix:
    - Tune poll settings
    - Use static membership
    - Scale consumers
    - Optimize processing time

---

6. Monitoring Tools

- **Prometheus + Grafana** (for metrics)

- **Burrow** (lag per partition)
- **JMX** (heap, GC, thread pools)
- **Confluent Control Center**