Kafka interview guide

Your complete **Kafka Interview Guide** with 100+ questions has been created in the document on the right side.
If you want additions like:

✅ More tricky scenario-based questions
✅ Real-world architecture diagrams
✅ MCQs with answers
✅ Cheat sheets (commands, configs, tuning parameters)
✅ Separate guides for Producer, Consumer, Broker, Topic, Partition
✅ Advanced Kafka Streams / Connect interview sections

Just tell me — I can update or extend the document anytime!

Kafka Interview Guide

Apache Kafka Interview Guide (100+ Questions)

Below is a structured guide with 4 sets, each containing 25 Kafka interview questions. Each question includes an explanation or answer.

Set 1: Core Kafka Concepts (25 Questions)

1. **What is Apache Kafka?** Kafka is a distributed event-streaming platform used for building real-time data pipelines and streaming applications.
2. **What is a Topic?** A category or stream name to which records are stored.
3. **What is a Partition?** A topic is split into partitions for parallelism and scalability.

4. **What is an Offset?** A unique ID assigned to each message within a partition.
5. **What is a Consumer Group?** A group of consumers working together to consume messages from a topic.
6. **What is a Producer?** An application that publishes messages to Kafka topics.
7. **What is a Broker?** A Kafka server storing data and serving client requests.
8. **What is Zookeeper?** Used by Kafka (pre-Kraft) for broker coordination.
9. **What is the use of Kafka Controller?** Manages partition leadership and replication.
10. **What is Message Retention?** How long Kafka stores messages (time-based or size-based).
11. **What is Log Compaction?** Kafka keeps only the latest value for each key.
12. **What is a Replica?** A copy of partition data stored across brokers.
13. **What is ISR (In-Sync Replica)?** A set of replicas that are fully caught up with the leader.
14. **What is ACK in Kafka?** Defines how many replicas must acknowledge a write.
15. **Difference between `acks=0`, `acks=1`, `acks=all`**

- 0 → worst durability, fastest
- 1 → leader-only acknowledgment
- all → safest, slowest

16. **What is Kafka Streams?** A library for building streaming applications.
17. **What is Exactly-Once Semantics?** Kafka guarantees a message is processed exactly once.
18. **What is Consumer Lag?** Difference between last produced and last consumed message.
19. **What are the main components of Kafka?** Producers, Consumers, Brokers, Topics, Partitions.

20. **What is a Dead Letter Queue?** A topic for failed or unprocessable messages.
21. **What is Rebalancing?** Redistribution of partitions across consumers.
22. **What is Sticky Partitioning?** Producer sends messages to the same partition until batch full.
23. **What are Records?** The actual key-value messages stored in Kafka.
24. **What is Kafka Connect?** A tool to transfer data between Kafka and external systems.
25. **Use of Schema Registry** Manages schema versions for Kafka messages.

Set 2: Message Consumption & Consumer Errors (25 Questions)

1. **How does a consumer read messages?** It polls data from Kafka using `poll()` API.
2. **What happens when two consumers are in the same consumer group?** Kafka divides partitions between them.
3. **What if partitions < consumers?** Extra consumers remain idle.
4. **What if partitions > consumers?** Some consumers handle multiple partitions.
5. **Why messages remain unconsumed?** Consumer down, partition mismatch, lag, wrong group ID.
6. **What causes consumer lag?** Slow processing, network issues, insufficient consumers.
7. **Fix consumer lag** Increase partitions, scale consumers, optimize processing.
8. **What is commit offset?** Marks messages as processed.
9. **What is auto commit?** Kafka automatically commits offsets.
10. **When does auto commit fail?** If consumer crashes before commit.
11. **Manual commit advantages** Control over message acknowledgement.

12. **What happens if consumer fails after processing but before commit?** Message will be reprocessed → at-least-once.
13. **What is at-most-once processing?** Messages may be lost.
14. **What is at-least-once?** Messages may be duplicated.
15. **Errors:** `OffsetOutOfRangeException` Offset deleted due to retention. Fix → reset offset = earliest/latest.
16. `RebalanceInProgressException` Occurs during consumer group rebalance. Fix → handle commit in try/catch.
17. `CommitFailedException` Commit attempted after rebalance. Fix → retry commit.
18. `SerializationException` Invalid message format. Fix → correct serializer/deserializer.
19. `TimeoutException` **while consuming** Slow broker or network. Fix → increase `poll.timeout`.
20. **What is max.poll.interval.ms?** Max time between polls.
21. **What happens when max.poll.interval exceeded?** Kafka removes consumer from group.
22. **What is max.poll.records?** Max messages returned per poll.
23. **What is heartbeat interval?** Prevents consumer removal from group.
24. **Why consumer stuck in rebalancing?** Slow heartbeat, overloaded consumer.
25. **Fix rebalancing loop** Tune: heartbeat, max.poll.interval, session.timeout.

Set 3: Partitions, Scaling & Producer Errors (25 Questions)

1. **How are partitions assigned?** Round-robin, sticky, or custom partitioner.
2. **What happens when you increase partitions?** Parallelism increases but message order breaks.
3. **Can you decrease partitions?** No → irreversible.

4. **What is message ordering guarantee?** Kafka guarantees ordering only within a partition.
5. **How to ensure ordering?** Use key-based partitioning.
6. **Producer** `BufferExhaustedException` Insufficient buffer size.
7. **Producer** `TimeoutException` Brokers overloaded.
8. **What is idempotent producer?** Prevents duplicates.
9. **Use of** `enable.idempotence=true` Guarantees exactly-once for producers.
10. **What is transactional producer?** Used for atomic multi-partition writes.
11. **What is linger.ms?** Delay before sending batch to accumulate messages.
12. **What is batch.size?** Max message batch size.
13. **What is compression.type?** Snappy, gzip, zstd reduce payload size.
14. **What is replication.factor?** Number of replicas for durability.
15. **What if replication factor > brokers?** Topic creation fails.
16. **Leader election in Kafka** Controller assigns partition leaders.
17. **Under-replicated partition?** Some replicas not in sync.
18. **Fix under-replicated partitions** Check broker down, network issues.
19. **Unclean leader election** Allows out-of-sync replica to become leader → data loss.
20. **min.insync.replicas** Minimum replicas required for write.
21. **Producer** `RecordTooLargeException` Message size exceeds limit.
22. **Fix message too large** Increase `max.request.size`, `message.max.bytes`.
23. **What is retention.ms?** Time-based message retention.
24. **What is retention.bytes?** Size-based log retention.
25. **What is segment.ms?** Time to roll log segment.

Set 4: Real-World Scenarios & Tricky Questions (25 Questions)

1. **Two consumers reading same topic but receiving same messages?** They have different consumer groups.
2. **Two consumers, same group, same partition?** Impossible; one partition assigned to only one consumer.
3. **Consumer not receiving messages after restart** Committed offset points to latest.
4. **How to reprocess all messages?** Reset offset → earliest.
5. **How to handle poison messages?** Send to Dead Letter Queue.
6. **Kafka losing messages?** Possible if:

- acks=0 or acks=1
- unclean leader election
- low replication

7. **How to guarantee no message loss?** acks=all, replication>=3, idempotent producer.
8. **Why unconsumed messages increase?** Consumer lag.
9. **Consumer processed message but crashed before commit** Reprocessed.
10. **Producer retry logic** Uses exponential backoff.
11. **What if producer retries cause duplicates?** Enable idempotence.
12. **Partition leader down** ISR replica becomes new leader.
13. **What if no ISR available?** Cluster unavailable.
14. **How Kafka achieves scalability?** Partitioning + replication.
15. **Kafka as queue vs. pub/sub** Queue → one consumer per partition
    Pub/Sub → multiple consumer groups
16. **Kafka vs RabbitMQ** Kafka → streaming RabbitMQ → messaging
17. **Kafka vs ActiveMQ** Kafka for high throughput.
18. **What is backpressure?** Consumer slower than producer.
19. **Fix backpressure** Scale consumers, increase partitions.
20. **What is watermarking in Kafka Streams?** Marks event time progress.
21. **What is windowing?** Aggregations over time intervals.

22. **What is retention vs compaction?** Retention deletes old data; compaction keeps latest.
23. **Can Kafka lose data after commit?** Rare, unless disk corruption.
24. **What is replication throttle?** Limits replication bandwidth.
25. **Kafka exactly-once in distributed system** Uses transactions + idempotent writes.

Additional Sections Added (Cheat-sheets, Component Guides, Advanced Topics, Tricky Scenarios)

Cheat Sheet — Commands, Useful Scripts & Quick Ops

Common CLI commands (Kafka binary distribution)

- List topics: `kafka-topics.sh --bootstrap-server <broker:9092> --list`
- Describe topic: `kafka-topics.sh --bootstrap-server <broker:9092> --describe --topic <topic-name>`
- Create topic: `kafka-topics.sh --bootstrap-server <broker:9092> --create --topic <topic-name> --partitions <N> --replication-factor <R>`
- Delete topic: `kafka-topics.sh --bootstrap-server <broker:9092> --delete --topic <topic-name>`
- Produce to console: `kafka-console-producer.sh --broker-list <broker:9092> --topic <topic-name>`
- Consume from console: `kafka-console-consumer.sh --bootstrap-server <broker:9092> --topic <topic-name> --from-beginning`

- Consumer group list: `kafka-consumer-groups.sh --bootstrap-server <broker:9092> --list`
- Consumer group describe: `kafka-consumer-groups.sh --bootstrap-server <broker:9092> --describe --group <group-id>`
- Reset offsets: `kafka-consumer-groups.sh --bootstrap-server <broker:9092> --group <group-id> --reset-offsets --to-earliest --topic <topic> --execute`
- Reassign partitions (create JSON then run): `kafka-reassign-partitions.sh --bootstrap-server <broker:9092> --reassignment-json-file reassignment.json --execute`
- Preferred leader election: `kafka-preferred-replica-election.sh --bootstrap-server <broker:9092>`
- Tool to delete records (log-retention workaround): `kafka-delete-records.sh --bootstrap-server <broker:9092> --offset-json-file offsets.json --execute`

Quick config lookups

- Broker config: `kafka-configs.sh --bootstrap-server <broker:9092> --entity-type brokers --entity-name <broker-id> --describe`
- Topic-level config: `kafka-configs.sh --bootstrap-server <broker:9092> --entity-type topics --entity-name <topic-name> --describe`

Done — I've added the cheat-sheets, separate component guides (Producer/Consumer/Broker/Topic/Partition), advanced Kafka Streams & Connect sections, and 20 extra tricky scenarios to the **Kafka Interview Guide** document.

Would you like me to:

- export the document to PDF, or
- add sample `server.properties` / `producer.properties` files and runnable code snippets for Producer/Consumer examples?