# 🚀 Apache Kafka KRaft Command Cheat Sheet

Essential commands for Kafka with KRaft mode (No Zookeeper!)

✨ **KRaft Mode - Kafka 3.3+**

> 💡 **What is KRaft?** KRaft (Kafka Raft) is Kafka's built-in consensus protocol that replaces the need for Apache Zookeeper. It's production-ready since Kafka 3.3.1 and is the future of Kafka cluster management.

## 🎯 KRaft Setup & Initialization

### Generate Cluster ID (Required for KRaft)

```
bin/kafka-storage.sh random-uuid
```

### Format Storage Directory with Cluster ID

```
bin/kafka-storage.sh format -t <cluster-id> -c
config/kraft/server.properties
```

### Format Storage for Controller Node

```
bin/kafka-storage.sh format -t <cluster-id> -c
config/kraft/controller.properties
```

### Format Storage for Broker Node

```
bin/kafka-storage.sh format -t <cluster-id> -c
```

```
config/kraft/broker.properties
```

## 📦 Kafka Server Management (KRaft Mode)

### Start Kafka in KRaft Mode (Combined Mode)

```
bin/kafka-server-start.sh config/kraft/server.properties
```

### Start Kafka Controller (Separate Mode)

```
bin/kafka-server-start.sh config/kraft/controller.properties
```

### Start Kafka Broker (Separate Mode)

```
bin/kafka-server-start.sh config/kraft/broker.properties
```

### Stop Kafka Server

```
bin/kafka-server-stop.sh
```

### Start Kafka in Background

```
bin/kafka-server-start.sh -daemon config/kraft/server.properties
```

## 🔍 Cluster Metadata & Information

### Display Cluster Metadata

```
bin/kafka-metadata-shell.sh --snapshot /tmp/kraft-combined-logs/__cluster_metadata-0/00000000000000000000.log
```

### List Brokers in Cluster

```
bin/kafka-broker-api-versions.sh --bootstrap-server localhost:9092
```

### Get Cluster ID

```
bin/kafka-cluster.sh cluster-id --bootstrap-server localhost:9092
```

## 📋 Topic Operations

### Create a Topic

```
bin/kafka-topics.sh --create --topic my-topic --bootstrap-server localhost:9092 --partitions 3 --replication-factor 1
```

### List All Topics

```
bin/kafka-topics.sh --list --bootstrap-server localhost:9092
```

### Describe a Topic

```
bin/kafka-topics.sh --describe --topic my-topic --bootstrap-server localhost:9092
```

### Describe All Topics

```
bin/kafka-topics.sh --describe --bootstrap-server localhost:9092
```

### Delete a Topic

```
bin/kafka-topics.sh --delete --topic my-topic --bootstrap-server localhost:9092
```

### Alter Topic Partitions (Increase Only)

```
bin/kafka-topics.sh --alter --topic my-topic --partitions 5 --bootstrap-server localhost:9092
```

### Alter Topic Configuration

```
bin/kafka-configs.sh --alter --entity-type topics --entity-name my-topic --add-config retention.ms=86400000 --bootstrap-server localhost:9092
```

## 📤 Producer Operations

### Console Producer

```
bin/kafka-console-producer.sh --topic my-topic --bootstrap-server localhost:9092
```

### Producer with Key-Value Pairs

```
bin/kafka-console-producer.sh --topic my-topic --bootstrap-server localhost:9092 --property "parse.key=true" --property "key.separator=:"
```

### Producer with Acknowledgments

```
bin/kafka-console-producer.sh --topic my-topic --bootstrap-server localhost:9092 --producer-property acks=all
```

### Produce from File

```
bin/kafka-console-producer.sh --topic my-topic --bootstrap-server localhost:9092 < input-file.txt
```

# 📥 Consumer Operations

### Console Consumer (from latest)

```
bin/kafka-console-consumer.sh --topic my-topic --bootstrap-server localhost:9092
```

### Consumer from Beginning

```
bin/kafka-console-consumer.sh --topic my-topic --from-beginning --bootstrap-server localhost:9092
```

### Consumer with Group ID

```
bin/kafka-console-consumer.sh --topic my-topic --group my-consumer-group --bootstrap-server localhost:9092
```

### Consumer with Key and Value

```
bin/kafka-console-consumer.sh --topic my-topic --bootstrap-server localhost:9092 --property print.key=true --property key.separator=":"
```

### Consumer with Timestamp and Offset

```
bin/kafka-console-consumer.sh --topic my-topic --bootstrap-server localhost:9092 --property print.timestamp=true --property print.offset=true
```

### Consumer with Headers

```
bin/kafka-console-consumer.sh --topic my-topic --bootstrap-server localhost:9092 --property print.headers=true
```

### Consumer from Specific Partition

```
bin/kafka-console-consumer.sh --topic my-topic --partition 0 --bootstrap-server localhost:9092
```

## 👥 Consumer Group Management

### List Consumer Groups

```
bin/kafka-consumer-groups.sh --list --bootstrap-server localhost:9092
```

### Describe Consumer Group

```
bin/kafka-consumer-groups.sh --describe --group my-consumer-group --bootstrap-server localhost:9092
```

### Describe Consumer Group (with Members)

```
bin/kafka-consumer-groups.sh --describe --group my-consumer-group --members --bootstrap-server localhost:9092
```

### Describe Consumer Group (with State)

```
bin/kafka-consumer-groups.sh --describe --group my-consumer-group --state --bootstrap-server localhost:9092
```

### Delete Consumer Group

```
bin/kafka-consumer-groups.sh --delete --group my-consumer-group --bootstrap-server localhost:9092
```

### Reset Offsets to Earliest

```
bin/kafka-consumer-groups.sh --reset-offsets --to-earliest --execute --topic my-topic --group my-consumer-group --bootstrap-server localhost:9092
```

### Reset Offsets to Latest

```
bin/kafka-consumer-groups.sh --reset-offsets --to-latest --execute --topic my-topic --group my-consumer-group --bootstrap-server localhost:9092
```

### Reset Offsets to Specific Offset

```
bin/kafka-consumer-groups.sh --reset-offsets --to-offset 100 --execute --topic my-topic --group my-consumer-group --bootstrap-server localhost:9092
```

### Reset Offsets by Duration (e.g., 1 hour back)

```
bin/kafka-consumer-groups.sh --reset-offsets --by-duration PT1H --execute --topic my-topic --group my-consumer-group --bootstrap-server localhost:9092
```

### Reset Offsets to Datetime

```
bin/kafka-consumer-groups.sh --reset-offsets --to-datetime 2024-01-01T00:00:00.000 --execute --topic my-topic --group my-consumer-group --bootstrap-server localhost:9092
```

## ⚙️ Configuration Management

### List Topic Configurations

```
bin/kafka-configs.sh --describe --entity-type topics --entity-name my-topic --bootstrap-server localhost:9092
```

### Add Topic Configuration

```
bin/kafka-configs.sh --alter --entity-type topics --entity-name my-topic --add-config retention.ms=604800000,segment.ms=3600000 --bootstrap-server localhost:9092
```

### Remove Topic Configuration

```
bin/kafka-configs.sh --alter --entity-type topics --entity-name my-topic --delete-config retention.ms --bootstrap-server localhost:9092
```

### Broker Configuration

```
bin/kafka-configs.sh --describe --entity-type brokers --entity-name 0 --bootstrap-server localhost:9092
```

### Dynamic Broker Configuration

```
bin/kafka-configs.sh --alter --entity-type brokers --entity-name 0 --add-config log.retention.hours=168 --bootstrap-server localhost:9092
```

## 🔍 Performance Testing & Benchmarking

### Producer Performance Test

```
bin/kafka-producer-perf-test.sh --topic my-topic --num-records 1000000 --record-size 1024 --throughput -1 --producer-props bootstrap.servers=localhost:9092
```

### Producer Performance Test (with acks)

```
bin/kafka-producer-perf-test.sh --topic my-topic --num-records 100000 --record-size 1024 --throughput 10000 --producer-props bootstrap.servers=localhost:9092 acks=all
```

### Consumer Performance Test

```
bin/kafka-consumer-perf-test.sh --topic my-topic --messages 1000000 --bootstrap-server localhost:9092
```

### Consumer Performance Test (with Group)

```
bin/kafka-consumer-perf-test.sh --topic my-topic --messages 100000 --bootstrap-server localhost:9092 --group perf-test-group
```

## 🔄 Partition & Replica Management

### Generate Partition Reassignment JSON

```
bin/kafka-reassign-partitions.sh --generate --topics-to-move-json-file topics.json --broker-list "0,1,2" --bootstrap-server localhost:9092
```

### Execute Partition Reassignment

```
bin/kafka-reassign-partitions.sh --execute --reassignment-json-file reassignment.json --bootstrap-server localhost:9092
```

### Verify Partition Reassignment

```
bin/kafka-reassign-partitions.sh --verify --reassignment-json-file reassignment.json --bootstrap-server localhost:9092
```

### Preferred Replica Election (All Topics)

```
bin/kafka-leader-election.sh --election-type PREFERRED --all-topic-partitions --bootstrap-server localhost:9092
```

### Unclean Leader Election

```
bin/kafka-leader-election.sh --election-type UNCLEAN --all-topic-partitions --bootstrap-server localhost:9092
```

# 🔧 ACL & Security

### Add Read ACL for User

```
bin/kafka-acls.sh --add --allow-principal User:alice --operation Read --topic my-topic --bootstrap-server localhost:9092
```

### Add Write ACL for User

```
bin/kafka-acls.sh --add --allow-principal User:bob --operation Write --topic my-topic --bootstrap-server localhost:9092
```

### Add Consumer Group ACL

```
bin/kafka-acls.sh --add --allow-principal User:alice --operation Read --group my-consumer-group --bootstrap-server localhost:9092
```

### List All ACLs

```
bin/kafka-acls.sh --list --bootstrap-server localhost:9092
```

### List ACLs for Specific Topic

```
bin/kafka-acls.sh --list --topic my-topic --bootstrap-server localhost:9092
```

### Remove ACL

```
bin/kafka-acls.sh --remove --allow-principal User:alice --operation Read --topic my-topic --bootstrap-server localhost:9092
```

# 📊 Monitoring & Log Analysis

### Dump Log Segment

```
bin/kafka-dump-log.sh --files /tmp/kraft-combined-logs/my-topic-0/00000000000000000000.log --print-data-log
```

### Dump Log with Offsets

```
bin/kafka-dump-log.sh --files /tmp/kraft-combined-logs/my-topic-0/00000000000000000000.log --offsets-decoder
```

### Get Log End Offset

```
bin/kafka-run-class.sh kafka.tools.GetOffsetShell --topic my-topic --time -1 --bootstrap-server localhost:9092
```

### Get Log Start Offset

```
bin/kafka-run-class.sh kafka.tools.GetOffsetShell --topic my-topic --time -2 --bootstrap-server localhost:9092
```

## 🛠️ Advanced Operations

### Mirror Maker 2 (Replication)

```
bin/connect-mirror-maker.sh config/connect-mirror-maker.properties
```

### Kafka Streams Reset Tool

```
bin/kafka-streams-application-reset.sh --application-id my-streams-app --bootstrap-servers localhost:9092 --input-topics my-input-topic
```

### Delete Records Before Offset

```
bin/kafka-delete-records.sh --bootstrap-server localhost:9092 --offset-json-file offsets-to-delete.json
```

### 📌 Important Notes:

- Replace `localhost:9092` with your actual Kafka broker address
- KRaft mode requires Kafka 3.3.1 or higher for production use
- Always generate and use a unique cluster ID when formatting storage
- The `--bootstrap-server` flag is used instead of `--zookeeper`

💻 **Windows Users:** Replace `.sh` with `.bat` and use backslashes (\) instead of forward slashes (/) in file paths.

### 🎯 Quick Start with KRaft:

1. Generate cluster ID: `kafka-storage.sh random-uuid`
2. Format storage: `kafka-storage.sh format -t <cluster-id> -c config/kraft/server.properties`
3. Start Kafka: `kafka-server-start.sh config/kraft/server.properties`