redis-cli Redis command line

START HERE

`redis-cli`
Open interactive prompt

`INFO memory`
Server stats section

`redis-cli -h <host> -p <port>`
Connect to a server

`HELP <command>`
Built-in syntax help

`PING`
Check the connection

`SET <key> <value>`
Store a string

`GET <key>`
Read it back

`redis-cli --scan --pattern '<pat>'`
List keys safely

Input args, stdin, prompt → RESP → **redis-cli** modes and flags → TCP / TLS → **Server** db 0-15, cluster → typed reply → **Output** tty, raw, csv, json

CONCEPTS

Every redis-cli call sends one command and prints the typed reply. On a terminal you see (integer) or quotes; piped output switches to raw automatically.

Keys live in numbered databases (-n or SELECT). Cluster mode has only database 0.

Never run KEYS * or FLUSHALL on a production instance: KEYS blocks the server, FLUSHALL deletes every database. Use --scan and think twice.

CONNECT

`redis-cli -h <host> -p <port>`
Connect to host and port

`redis-cli -u redis://<user>:<pass>@<host>:<port>/<db>`
Connect with a URI
Use the rediss:// scheme for TLS; user and db are optional

`redis-cli -n <db>`
Start in database number

`redis-cli -s <socket>`
Connect over a Unix socket

`redis-cli --user <name> --askpass`
Log in, prompt for password

CONNECT (cont.)

`REDISCLI_AUTH=<pass> redis-cli`
Pass password via environment
Avoids the password showing in the process list, unlike -a

`redis-cli -a <pass>`
Authenticate with a password
Prints a warning; silence with --no-auth-warning

`redis-cli --tls --cacert <ca.crt>`
Connect over TLS

`redis-cli -c`
Follow cluster redirects

RUN AND SCRIPT

`redis-cli <command> [args]`
Run one command and exit

`redis-cli --raw <command>`
Plain output without type labels

`redis-cli --csv <command>`
Reply as CSV

`redis-cli --json <command>`
Reply as JSON
Defaults to RESP3 replies; add -2 for RESP2

`redis-cli -x SET <key> < <file>`
Use stdin as last argument

`redis-cli < <commands.txt>`
Run commands from a file

RUN AND SCRIPT (cont.)

`redis-cli -r <n> -i <sec> <command>`
Repeat a command
-r -1 repeats forever

`redis-cli -e <command>`
Exit non-zero on error reply

`redis-cli --pipe < <file>`
Bulk-load raw protocol

PROMPT

`HELP <command>`
Show syntax of a command

`HELP @<group>`
List commands of a group
Groups include @string, @list, @hash, @set, @sorted_set, @stream

`SELECT <db>`
Switch database

`quit / exit`
Leave the prompt

KEYS

`EXISTS <key>`
Does the key exist

`DEL <key>`
Delete a key

`TYPE <key>`
Show the key's type

`EXPIRE <key> <seconds>`
Set a time to live

`TTL <key>`
Seconds left (-1 none, -2 gone)

KEYS (cont.)

`SCAN <cursor> MATCH <pat> COUNT <n>`
Iterate keys in steps
Repeat with the returned cursor until it is 0

STRINGS AND COUNTERS

`SET <key> <value>`
Store a string

`GET <key>`
Read a string

`SET <key> <value> EX <seconds> NX`
Set with TTL if missing
A simple lock or cache-fill pattern

`INCR <key>`
Add 1 to a counter

HASHES AND LISTS

`HSET <key> <field> <value>`
Set hash fields

`HGET <key> <field>`
Read one field

`HGETALL <key>`
Read all fields

`LPUSH <key> <value>`
Push on the left

`RPUSH <key> <value>`
Push on the right

`LRANGE <key> 0 -1`
Read the whole list

SETS AND SORTED SETS

`SADD <key> <member>`
Add set members

`SMEMBERS <key>`
List set members

`SISMEMBER <key> <member>`
Test membership

`ZADD <key> <score> <member>`
Add with a score

`ZRANGE <key> 0 -1 WITHSCORES`
Read by rank

STREAMS AND PUB/SUB

`XADD <key> * <field> <value>`
Append a stream entry

`PUBLISH <channel> <message>`
Send a message

`SUBSCRIBE <channel>`
Listen on a channel

INSPECT AND MONITOR

`INFO`
Full server report

`INFO <section>`
One section (memory, stats, ...)

`SLOWLOG GET 10`
Latest slow commands

INSPECT AND MONITOR (cont.)

`MONITOR`
Stream every command
Costly under load; stop with Ctrl+C

`redis-cli --stat`
Rolling server stats

`redis-cli --bigkeys`
Largest key per type

`redis-cli --scan --pattern '<pat>'`
List matching keys

`redis-cli --latency`
Measure round-trip time

RECIPES

Delete keys by pattern safely

```
redis-cli --scan --pattern 'session:*' | head
# looks right? then delete in batches (repeat -h/-a/-n flags on both calls)
redis-cli --scan --pattern 'session:*' | xargs -d '\n' -n 100 redis-cli UNLINK
```

Bulk load from a command file

```
# data.txt holds one command per line
redis-cli < data.txt
# or raw protocol for millions of rows
redis-cli --pipe < data.resp
```

Back up and count a prefix

```
redis-cli --rdb backup.rdb
redis-cli --scan --pattern 'user:*' | wc -l
redis-cli --json HGETALL user:1 | jq
```