REFERENCIA RÁPIDA DE REDIS

Cadenas, listas, conjuntos, hashes, pub/sub, persistencia

Conexión

CLI

```
redis-cli
redis-cli -h 127.0.0.1 -p 6379
redis-cli -a password -n 2
redis-cli --tls -u rediss://user:pass@host:6380
```

Conexión con Driver (Python)

```
import redis
r = redis.Redis(host='localhost', port=6379, db=0)
r.set('key', 'value')
print(r.get('key'))
```

Información del Servidor

```
PING            -- returns PONG
INFO server     -- server details
INFO memory     -- memory usage
DBSIZE          -- number of keys in current db
```

Cadenas

Operaciones Básicas

```
SET name "Alice"
GET name
SET counter 100
MSET a 1 b 2 c 3
MGET a b c
```

Operaciones Numéricas

```
INCR counter        -- 101
INCRBY counter 10   -- 111
DECR counter        -- 110
DECRBY counter 5    -- 105
INCRBYFLOAT price 2.5
```

Comandos de Cadenas

`SET key val`	Establecer valor de cadena
`GET key`	Obtener valor de cadena
`SETNX key val`	Establecer solo si la clave no existe
`SETEX key sec val`	Establecer con expiración en segundos
`APPEND key val`	Agregar al valor existente
`STRLEN key`	Longitud del valor de cadena

Listas

Operaciones de Lista

```
LPUSH queue "first"
RPUSH queue "last"
LRANGE queue 0 -1    -- all elements
LPOP queue
RPOP queue
```

Comandos de Lista

`LPUSH / RPUSH`	Insertar al inicio / final de la lista
`LPOP / RPOP`	Extraer del inicio / final
`LRANGE key start stop`	Obtener rango de elementos
`LLEN key`	Longitud de la lista
`LINDEX key idx`	Elemento en el índice
`LREM key count val`	Eliminar count ocurrencias de val
`BLPOP key timeout`	Extracción bloqueante (para colas)

Conjuntos y Conjuntos Ordenados

Operaciones de Conjuntos

```
SADD tags "python" "redis" "docker"
SMEMBERS tags
SISMEMBER tags "python"   -- 1 (true)
SREM tags "docker"
SCARD tags                -- count
```

Operaciones de Conjuntos

```
SUNION set1 set2        -- union
SINTER set1 set2        -- intersection
SDIFF set1 set2         -- difference
```

Operaciones de Conjuntos Ordenados

```
ZADD leaderboard 100 "Alice" 85 "Bob"
ZRANGE leaderboard 0 -1 WITHSCORES
ZREVRANGE leaderboard 0 2
ZSCORE leaderboard "Alice"
ZRANK leaderboard "Alice"    -- 0-based rank
```

Comandos de Conjuntos Ordenados

`ZADD key score member`	Agregar miembro con puntuación
`ZRANGE key start stop`	Rango por posición (menor a mayor)
`ZREVRANGE key start stop`	Rango por posición (mayor a menor)
`ZINCRBY key incr member`	Incrementar puntuación del miembro
`ZRANGEBYSCORE key min max`	Rango por valor de puntuación
`ZCARD key`	Número de miembros

Hashes

Operaciones de Hash

```
HSET user:1 name "Alice" age 30
HGET user:1 name
HGETALL user:1
HMSET user:2 name "Bob" age 25
HMGET user:1 name age
```

Comandos de Hash

`HSET key field val`	Establecer campo del hash
`HGET key field`	Obtener campo del hash
`HGETALL key`	Obtener todos los campos y valores
`HDEL key field`	Eliminar campo del hash
`HEXISTS key field`	Verificar existencia del campo
`HINCRBY key field n`	Incrementar valor del campo
`HKEYS key`	Todos los nombres de campos
`HLEN key`	Número de campos

Claves y Expiración

Comandos de Claves

`KEYS pattern`	Buscar claves que coincidan con el patrón (lento)
`SCAN cursor MATCH pat`	Iterar claves de forma incremental (seguro)
`EXISTS key`	Verificar si la clave existe
`DEL key`	Eliminar clave
`TYPE key`	Obtener tipo de dato de la clave
`RENAME key newkey`	Renombrar una clave

Comandos de Expiración

```
EXPIRE key 3600       -- expire in 1 hour
PEXPIRE key 5000      -- expire in 5000 ms
TTL key               -- seconds until expiry
PTTL key              -- ms until expiry
PERSIST key           -- remove expiry
```

Patrones de Claves

```
SET session:abc123 "data" EX 1800
-- EX = seconds, PX = milliseconds
-- NX = only if not exists
-- XX = only if exists
SET lock:order42 "owner" NX EX 10
```

Pub/Sub

Pub/Sub Básico

```
-- Subscriber (terminal 1)
SUBSCRIBE news alerts

-- Publisher (terminal 2)
PUBLISH news "Breaking: Redis 8 released"
```

Suscripción por Patrón

```
PSUBSCRIBE news.*
-- matches news.tech, news.sports, etc.
```

Comandos Pub/Sub

`SUBSCRIBE channel`	Escuchar mensajes en el canal
`PUBLISH channel msg`	Enviar mensaje al canal
`PSUBSCRIBE pattern`	Suscribir a patrón
`UNSUBSCRIBE channel`	Dejar de escuchar
`PUBSUB CHANNELS`	Listar canales activos

Transacciones

MULTI / EXEC

```
MULTI
SET balance:1 900
SET balance:2 1100
EXEC        -- executes atomically
```

Bloqueo Optimista

```
WATCH balance:1
val = GET balance:1   -- read current
MULTI
SET balance:1 (val - 100)
EXEC
-- EXEC returns nil if balance:1 changed
```

Comandos de Transacciones

`MULTI`	Iniciar bloque de transacción
`EXEC`	Ejecutar comandos en cola
`DISCARD`	Descartar comandos en cola
`WATCH key`	Vigilar clave para cambios (bloqueo optimista)
`UNWATCH`	Olvidar todas las claves vigiladas

Persistencia

Snapshots RDB

```
SAVE            -- synchronous snapshot
BGSAVE          -- background snapshot
LASTSAVE        -- timestamp of last save
```

AOF (Archivo Solo de Adición)

`appendonly yes`	Habilitar AOF en redis.conf
`appendfsync always`	Fsync en cada escritura (más seguro, más lento)
`appendfsync everysec`	Fsync una vez por segundo (recomendado)
`appendfsync no`	Dejar que el OS decida (más rápido, más riesgoso)

Comandos de Persistencia

```
CONFIG GET save
CONFIG SET save "900 1 300 10"
-- snapshot if 1 change in 900s or 10 in 300s
BGREWRITEAOF    -- rewrite AOF in background
```

Patrones Comunes

Bloqueo Distribuido

```
SET lock:resource "owner-id" NX EX 30
-- NX = acquire only if not held
-- EX 30 = auto-release after 30s
DEL lock:resource  -- explicit release
```

Limitador de Velocidad

```
key = "rate:user:42"
INCR key
EXPIRE key 60   -- 60-second window
-- reject if GET key > max_requests
```

Patrón de Caché

```
val = GET "cache:user:1"
if val is nil:
  val = fetch_from_db(1)
  SET "cache:user:1" val EX 300
```

Almacenamiento de Sesiones

```
HSET sess:abc uid 42 role "admin"
EXPIRE sess:abc 1800   -- 30 min TTL
HGETALL sess:abc
```