MODUL DATABASE NoSQL
Key Value Database dengan Redis

Bagian I : Instalasi dan Pengaturan Awal Redis

Pengantar

Dalam lab ini, Anda akan belajar cara menginstal dan melakukan pengaturan awal. Lab ini mencakup langkah-langkah penting untuk mengaktifkan Redis, termasuk memverifikasi instalasi dan memastikan server dapat diakses.

Lab ini memandu Anda melalui pembaruan daftar paket, pemasangan server Redis, dan verifikasi pemasangan dengan memeriksa status layanan Redis. Anda juga akan belajar cara memulai server Redis dan menguji konektivitasnya.

Ini adalah Lab Terpanduan, yang menyediakan instruksi langkah demi langkah untuk membantu Anda belajar dan berlatih. Ikuti instruksi dengan cermat untuk menyelesaikan setiap langkah dan mendapatkan pengalaman praktis.

Instal Redis dan Hubungkan ke Server

Pada langkah ini, kita akan menginstal Redis dan terhubung ke server Redis menggunakan alat baris perintah redis-cli. Redis adalah penyimpanan struktur data opensource berbasis memori, sering digunakan sebagai database, cache, dan broker pesan.

Pertama, mari perbarui daftar paket untuk memastikan kita memiliki versi terbaru perangkat lunak. Buka terminal .

Jalankan perintah berikut:

```
sudo apt update
```

Perintah ini memperbarui daftar paket yang tersedia. Anda akan melihat output yang menunjukkan bahwa daftar paket sedang diperbarui.

Selanjutnya, instal Redis menggunakan perintah apt install:

```
sudo apt install redis-server
```

Perintah ini akan menginstal server Redis dan ketergantungannya. Anda mungkin diminta untuk mengonfirmasi instalasi dengan mengetik y dan menekan Enter.

Setelah instalasi selesai, jalankan server Redis:

```
sudo service redis-server start
```

Sekarang, mari kita sambungkan ke server Redis menggunakan perintah redis-cli. Perintah ini membuka antarmuka baris perintah Redis, memungkinkan Anda berinteraksi dengan server Redis.

```
redis-cli
```

Anda akan melihat prompt yang terlihat seperti ini:

```
127.0.0.1:6379&gt;
```

Ini menunjukkan bahwa Anda sekarang terhubung ke server Redis dan dapat mulai menjalankan perintah. Tetaplah terhubung untuk langkah berikutnya.

Uji Koneksi dan Tetapkan Kunci

Sekarang setelah Anda terhubung ke server Redis menggunakan redis-cli, mari uji koneksi dan atur pasangan kunci-nilai sederhana.

Pertama, uji koneksi menggunakan perintah PING:

```
PING
```

Jika server Redis berfungsi dengan benar, Anda akan menerima respons berikut:

```
PONG
```

Ini menunjukkan bahwa server Redis aktif dan berfungsi dengan baik, dan Anda dapat berkomunikasi dengannya.

Selanjutnya, mari kita atur pasangan kunci-nilai. Perintah SET digunakan untuk mengatur nilai string dari sebuah kunci. Misalnya, mari kita atur kunci bernama mykey dengan nilai Hello Redis:

```
SET mykey "Hello Redis"
```

Anda seharusnya menerima respons berikut:

```
OK
```

Ini menunjukkan bahwa pasangan kunci-nilai telah berhasil ditetapkan.

Akhirnya, keluar dari redis-cli:

```
quit
```

Penting untuk keluar dari redis-cli agar perintah Anda tercatat, sebelum mengklik tombol "Lanjutkan".

Mendapatkan Nilai dari Sebuah Kunci

Pada langkah ini, kita akan mengambil nilai kunci yang telah kita atur pada langkah sebelumnya menggunakan perintah GET.

Pertama, sambungkan ke server Redis menggunakan perintah redis-cli:

```
redis-cli
```

Sekarang, mari ambil nilai kunci mykey menggunakan perintah GET:

```
GET mykey
```

Anda seharusnya menerima respons berikut:

```
"Hello Redis"
```

Ini adalah nilai yang kita tetapkan untuk kunci mykey pada langkah sebelumnya.

Akhirnya, keluar dari redis-cli:

```
quit
```

Pastikan untuk keluar dari redis-cli agar perintah Anda tercatat sebelum mengklik tombol "Lanjutkan".

Ringkasan

Dalam lab ini, Anda telah belajar cara menginstal dan melakukan pengaturan awal Redis pada VM LabEx. Anda memulai dengan memperbarui daftar paket dan menginstal server Redis. Anda kemudian terhubung ke server Redis menggunakan alat baris perintah redis-cli, menguji koneksi dengan perintah PING, menetapkan pasangan kunci-nilai menggunakan perintah SET, dan mengambil nilai kunci menggunakan perintah GET. Pastikan untuk keluar dari redis-cli setelah setiap langkah untuk memastikan perintah Anda tercatat.

Bagian 2 : Verifikasi Status Server Redis

Pengantar

Dalam tantangan ini, Anda akan mengatasi masalah konektivitas server Redis dengan memverifikasi statusnya. Tugas ini melibatkan penggunaan perintah rediscli untuk terhubung ke server Redis, lalu menggunakan perintah PING untuk memastikan server berjalan dan responsif. Untuk menyelesaikan tantangan ini, Anda perlu menjalankan redis-cli di terminal, mengeksekusi perintah PING di antarmuka Redis CLI, memverifikasi bahwa server merespons dengan PONG, dan kemudian keluar dari antarmuka menggunakan perintah quit. Proses verifikasi memeriksa keberadaan perintah PING dalam log riwayat perintah Redis.

Ini adalah Tantangan, yang berbeda dari Lab Terpanduan karena Anda perlu mencoba menyelesaikan tugas tantangan secara mandiri, bukan mengikuti langkahlangkah lab untuk belajar. Tantangan biasanya agak sulit. Jika Anda merasa kesulitan, Anda dapat berdiskusi dengan Labby atau memeriksa solusi. Data historis menunjukkan bahwa ini adalah tantangan tingkat pemula dengan tingkat kelulusan 97%. Tantangan ini telah menerima tingkat ulasan positif 100% dari peserta.

Verifikasi Status Server Redis

Aplikasi mengalami masalah konektivitas dengan server Redis. Tugas Anda adalah memverifikasi bahwa server Redis sedang berjalan dan responsif menggunakan perintah redis-cli.

Tugas

- Gunakan perintah redis-cli untuk melakukan ping ke server Redis dan pastikan server merespons dengan PONG.

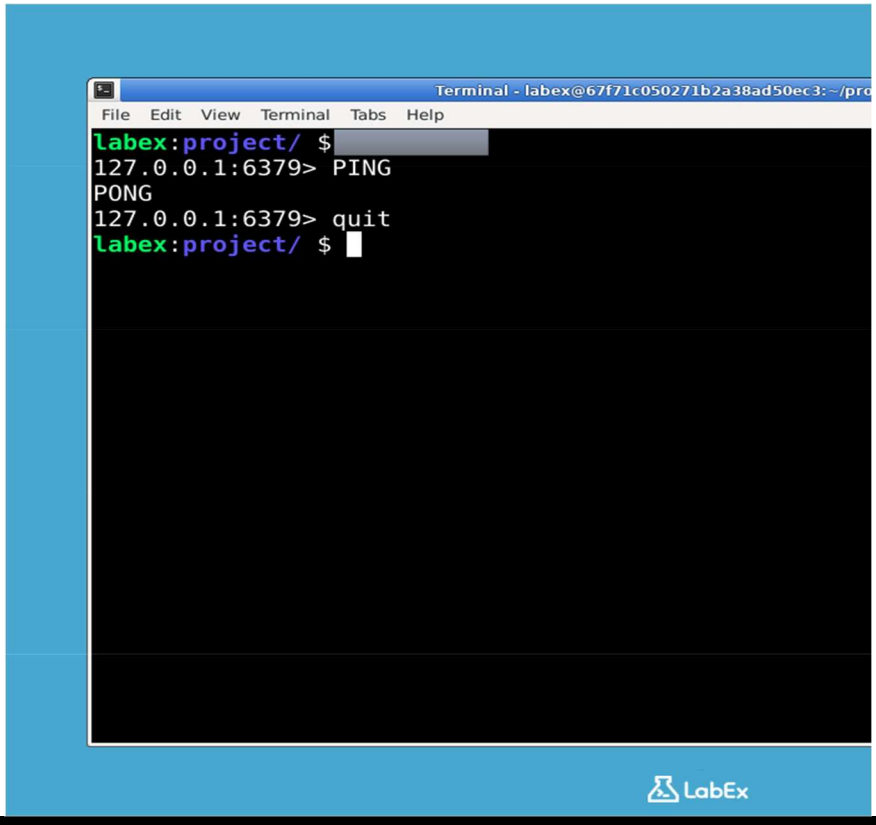
Persyaratan

1. Jalankan perintah redis-cli di terminal.
2. Di antarmuka redis-cli, jalankan perintah PING.
3. Pastikan server Redis merespons dengan PONG.
4. Keluar dari antarmuka redis-cli menggunakan perintah quit.

Contoh

Output perintah PING yang berhasil:

```
127.0.0.1:6379&gt; PING
PONG
```



127.0.0.1:6379> quitRUN

Tips

- Perintah redis-cli terhubung ke server Redis.
- Perintah PING memeriksa apakah server merespons.
- Pastikan server Redis sedang berjalan sebelum mencoba terhubung.

Penting: Keluar dari antarmuka redis-cli sebelum mengklik tombol "Lanjutkan".

Ringkasan

Dalam tantangan ini, tujuan utamanya adalah memverifikasi status dan responsivitas server Redis. Hal ini melibatkan penggunaan perintah redis-cli untuk terhubung ke server Redis, lalu menjalankan perintah PING di dalam antarmuka redis-cli. Respons PONG yang berhasil menunjukkan bahwa server Redis sedang berjalan dan responsif.
Poin pembelajaran utama adalah memahami cara menggunakan alat redis-cli untuk berinteraksi dengan server Redis dan bagaimana perintah PING berfungsi sebagai pemeriksaan kesehatan dasar. Tantangan ini menekankan pentingnya memverifikasi konektivitas server saat mengatasi masalah aplikasi yang terkait dengan Redis.

Bagian 3 : Konfigurasi Batas Memori Maksimum Redis

Pengantar

Dalam tantangan ini, Anda akan mengonfigurasi batas memori maksimum Redis untuk mencegah kehilangan data akibat kehabisan memori. Sebagai administrator sistem, tugas Anda adalah terhubung ke server Redis menggunakan redis-cli dan menggunakan perintah CONFIG SET untuk membatasi penggunaan memori Redis menjadi 200MB.
Tantangan ini mengharuskan Anda untuk mengatur parameter maxmemory menjadi

200MB di lingkungan redis-cli , lalu keluar.

Ini adalah Tantangan, yang berbeda dari Lab Terbimbing karena Anda perlu mencoba menyelesaikan tugas tantangan secara mandiri, bukan mengikuti langkahlangkah lab untuk belajar. Tantangan biasanya agak sulit. Jika Anda merasa kesulitan, Anda dapat berdiskusi dengan Labby atau memeriksa solusi. Data historis menunjukkan bahwa ini adalah tantangan tingkat pemula dengan tingkat kelulusan 100%. Tantangan ini telah menerima tingkat ulasan positif 98% dari peserta.

Konfigurasi Batas Memori Maksimum Redis

Server Redis Anda mendekati batas kapasitas memorinya, berisiko kehilangan data. Sebagai administrator sistem, konfigurasikan maxmemory menjadi 200MB untuk mencegah crash dan memastikan integritas data. Tugas

- Gunakan perintah CONFIG SET untuk membatasi penggunaan memori Redis menjadi 200MB.

Persyaratan

1. Hubungkan ke server Redis menggunakan perintah redis-cli.
2. Gunakan perintah CONFIG SET untuk mengatur parameter maxmemory menjadi 200MB.
3. Setelah menjalankan perintah CONFIG SET, keluar dari redis-cli

menggunakan perintah exit .

Contoh

Setelah berhasil mengatur parameter maxmemory, Anda dapat memeriksa konfigurasi menggunakan perintah CONFIG GET.

```
127.0.0.1:6379&gt; CONFIG GET maxmemory

1) "maxmemory"

2) "200mb"
```

```
127.0.0.1:6379&gt;                    exitRUN
```

Ini menunjukkan bahwa parameter `maxmemory` telah berhasil disetel menjadi 200MB.

Tips

- Gunakan perintah `redis-cli` untuk berinteraksi dengan server Redis.
- Perintah `CONFIG SET` digunakan untuk mengubah parameter konfigurasi Redis.
- Pastikan untuk menyertakan satuan `mb` saat mengatur parameter `maxmemory`.
- Pastikan untuk keluar dari `redis-cli` setelah mengatur parameter.

Ringkasan

Dalam tantangan ini, tugasnya adalah mengatur pengaturan `maxmemory` Redis menjadi 200MB untuk mencegah kehilangan data akibat server mendekati kapasitas memorinya. Hal ini melibatkan terhubung ke server Redis menggunakan `redis-cli` dan menjalankan perintah `CONFIG SET maxmemory 200mb`.

Poin pembelajaran utama meliputi penggunaan `redis-cli` untuk berinteraksi dengan server Redis, pemahaman perintah `CONFIG SET` untuk mengubah parameter konfigurasi Redis, dan ingat untuk menyertakan satuan `mb` saat mengatur parameter `maxmemory`.

Bagian 4 : Pengantar Struktur Data Redis

Pengantar

Dalam lab ini, Anda akan menjelajahi struktur data Redis dasar dan cara berinteraksi dengannya menggunakan alat baris perintah redis-cli. Lab ini berfokus pada latihan praktis untuk membantu Anda memahami cara menyimpan dan mengambil data di Redis.

Anda akan memulai dengan bekerja dengan String, belajar cara menetapkan, mengambil, memeriksa keberadaan, dan menghapus nilai string. Selanjutnya, Anda akan beralih ke List, menggunakan perintah seperti LPUSH dan LRANGE. Kemudian, Anda akan mengelola Set dengan SADD dan SMEMBERS. Terakhir, Anda akan menjelajahi Hash menggunakan HSET dan HGET. Pengalaman praktis ini akan memberikan dasar yang kokoh untuk menggunakan Redis dalam berbagai aplikasi.

Ini adalah Lab Terpanduan, yang menyediakan instruksi langkah demi langkah untuk membantu Anda belajar dan berlatih. Ikuti instruksi dengan cermat untuk menyelesaikan setiap langkah dan mendapatkan pengalaman praktis. Data historis menunjukkan bahwa ini adalah lab tingkat pemula dengan tingkat penyelesaian 96%. Lab ini telah menerima tingkat ulasan positif 100% dari peserta.

Bekerja dengan String untuk Data Sederhana

Pada langkah ini, kita akan menjelajahi cara menggunakan Redis untuk menyimpan dan mengambil data string sederhana. Redis sering digunakan sebagai cache atau penyimpanan kunci-nilai sederhana, dan string adalah tipe data dasar yang ditawarkannya.

Pertama, mari kita terhubung ke server Redis menggunakan alat baris perintah redis-cli. Buka terminal di VM LabEx. Anda seharusnya sudah berada di direktori ~/project.

Ketik perintah berikut untuk terhubung ke server Redis:

```
redis-cli
```

Anda akan melihat prompt yang terlihat seperti ini:

```
127.0.0.1:6379&gt;
```

Ini menandakan bahwa Anda sekarang terhubung ke server Redis.

Sekarang, mari kita atur nilai string sederhana. Kita akan menggunakan perintah SET. Perintah SET membutuhkan dua argumen: kunci dan nilai. Mari kita atur kunci bernama mykey dengan nilai Hello Redis:

```
SET mykey "Hello Redis"
```

Anda seharusnya melihat output berikut:

```
OK
```

Ini berarti nilai telah berhasil ditetapkan.

Sekarang, mari kita ambil nilai tersebut menggunakan perintah GET. Perintah GET membutuhkan satu argumen: kunci. Mari kita ambil nilai dari mykey:

```
GET mykey
```

Anda seharusnya melihat output berikut:

```
"Hello Redis"
```

Ini membuktikan bahwa kita telah berhasil menyimpan dan mengambil nilai string di Redis.

Mari kita coba contoh lain. Kali ini, mari kita simpan angka sebagai string.

```
SET counter 100

GET counter
```

Anda seharusnya melihat:

```
"100"
```

Redis menganggap ini sebagai string, meskipun sebenarnya mewakili angka.

Anda juga dapat menggunakan perintah EXISTS untuk memeriksa apakah suatu kunci ada.

```
EXISTS mykey
```

Anda harus melihat:

```
(bilangan bulat) 1
```

Hal ini menunjukkan bahwa kunci mykey ada. Jika kunci tidak ada, perintah akan mengembalikan (bilangan bulat) 0.

Akhirnya, mari kita hapus kunci menggunakan perintah DEL.

```
DEL mykey
```

Anda seharusnya melihat:

```
(bilangan bulat) 1
```

Hal ini menunjukkan bahwa kunci mykey telah dihapus dengan sukses.

Sekarang, jika Anda mencoba mendapatkan nilai mykey lagi:

```
GET mykey
```

Anda harus melihat:

```
(nil)
```

Ini mengonfirmasi bahwa kunci telah dihapus.

Pastikan untuk keluar dari redis-cli agar perintah Anda tercatat. Ketik:

```
exit
```

Ini akan mengembalikan Anda ke prompt terminal biasa.

Menggunakan Daftar dengan LPUSH dan LRANGE

Pada langkah ini, kita akan menjelajahi cara menggunakan daftar Redis untuk menyimpan dan mengambil kumpulan data yang terurut. Daftar Redis diimplementasikan sebagai daftar tertaut, yang membuatnya efisien untuk menambahkan dan menghapus elemen dari awal atau akhir daftar. Kita akan fokus pada perintah LPUSH dan LRANGE.

Kita akan terus menggunakan alat baris perintah redis-cli. Jika Anda belum terhubung, buka terminal di VM LabEx dan ketik:

```
redis-cli
```

Sekarang, mari kita buat daftar dan tambahkan beberapa elemen ke dalamnya menggunakan perintah LPUSH. LPUSH menambahkan elemen ke bagian kiri (awal) daftar. Perintah LPUSH menerima dua atau lebih argumen: kunci daftar dan nilai yang akan ditambahkan. Mari kita buat daftar bernama mylist dan tambahkan nilai item1, item2, dan item3:

```
LPUSH mylist item1
```

Anda seharusnya melihat output berikut:

```
(integer) 1
```

Ini berarti satu elemen telah ditambahkan ke daftar. Nilai kembalian LPUSH adalah panjang daftar setelah operasi.

Sekarang, mari tambahkan item lainnya:

```
LPUSH mylist item2

(bilangan bulat) 2

LPUSH mylist item3

(bilangan bulat) 3
```

Sekarang, mari kita ambil elemen-elemen daftar menggunakan perintah LRANGE. LRANGE mengembalikan rentang elemen dari daftar. Perintah LRANGE membutuhkan tiga argumen: kunci daftar, indeks awal, dan indeks akhir. Indeks menggunakan basis nol, jadi elemen pertama berada di indeks 0. Untuk mengambil semua elemen daftar, kita dapat menggunakan indeks awal 0 dan indeks akhir -1.

```
LRANGE mylist 0 -1
```

Anda seharusnya melihat output berikut:

```
"item3"

"item2"

"item1"
```

Perhatikan bahwa elemen-elemen dikembalikan dalam urutan terbalik dari saat kita menambahkannya, karena LPUSH menambahkan elemen ke awal daftar.

Mari tambahkan beberapa item lagi ke daftar:

```
LPUSH mylist item4

LPUSH mylist item5
```

Sekarang, mari kita ambil 3 elemen pertama dari daftar (indeks 0 hingga 2):

```
LRANGE mylist 0 2
```

Anda seharusnya melihat:

```
"item5"

"item4"

"item3"
```

Anda juga dapat menggunakan indeks negatif untuk mengakses elemen dari akhir daftar. Misalnya, untuk mengambil elemen terakhir dari daftar, Anda dapat menggunakan indeks -1:

```
LRANGE mylist -1 -1
```

Anda akan melihat:

```
1) "item1"
```

Pastikan untuk keluar dari redis-cli agar perintah Anda tercatat. Ketik:

```
exit
```

Ini akan mengembalikan Anda ke prompt terminal biasa.

Mengelola Set dengan SADD dan SMEMBERS

Pada langkah ini, kita akan menjelajahi cara menggunakan set Redis untuk menyimpan dan mengelola kumpulan elemen unik yang tidak terurut. Set Redis berguna untuk tugas seperti melacak pengunjung unik, menyimpan tag, atau mengelola hubungan antara objek. Kita akan fokus pada perintah SADD dan SMEMBERS.

Kita akan terus menggunakan alat baris perintah redis-cli. Jika Anda belum terhubung, buka terminal di VM LabEx dan ketik:

```
redis-cli
```

Sekarang, mari kita buat sebuah set dan tambahkan beberapa anggota ke dalamnya menggunakan perintah SADD. SADD menambahkan satu atau lebih anggota ke dalam sebuah set. Perintah SADD membutuhkan dua atau lebih argumen: kunci set dan anggota yang akan ditambahkan. Mari kita buat sebuah set bernama myset dan tambahkan anggota member1, member2, dan member3:

```
SADD myset member1
```

Anda seharusnya melihat output berikut:

```
(integer) 1
```

Ini berarti satu anggota telah ditambahkan ke himpunan. Nilai kembalian SADD adalah jumlah anggota yang ditambahkan ke himpunan (tidak termasuk anggota yang sudah ada sebelumnya).

Sekarang, mari tambahkan item lainnya:

```
SADD myset member2

(bilangan bulat) 1

SADD myset member3

(bilangan bulat) 1
```

Sekarang, mari kita ambil anggota himpunan menggunakan perintah SMEMBERS. SMEMBERS mengembalikan semua anggota himpunan. Perintah SMEMBERS membutuhkan satu argumen: kunci himpunan.

```
SMEMBERS myset
```

Anda akan melihat output berikut (urutan anggota mungkin berbeda, karena himpunan tidak terurut):

```
"member3"

"member2"

"member1"
```

Mari kita coba menambahkan anggota duplikat ke dalam himpunan:

SADD myset member1 Anda

seharusnya melihat:

```
(bilangan bulat) 0
```

Ini menunjukkan bahwa tidak ada anggota baru yang ditambahkan, karena member1 sudah ada dalam himpunan.

Mari kita tambahkan beberapa anggota lagi ke dalam himpunan:

```
SADD myset member4

SADD myset member5
```

Sekarang, mari kita ambil kembali semua anggota:

```
SMEMBERS myset
```

Anda seharusnya melihat sesuatu seperti:

```
"member5"

"member4"

"member3"
```

```
"member2"

"member1"
```

Urutan mungkin berbeda.

Pastikan untuk keluar dari redis-cli agar perintah Anda tercatat. Ketik:

```
exit
```

Ini akan mengembalikan Anda ke prompt terminal biasa.

Menjelajahi Hashes dengan HSET dan HGET

Pada langkah ini, kita akan menjelajahi cara menggunakan Redis hashes untuk menyimpan dan mengambil kumpulan pasangan bidang-nilai. Redis hashes berguna untuk mewakili objek dengan atribut multiple. Kita akan fokus pada perintah HSET dan HGET.

Kita akan terus menggunakan alat baris perintah redis-cli. Jika Anda belum terhubung, buka terminal di VM LabEx dan ketik:

```
redis-cli
```

Sekarang, mari kita buat hash dan tambahkan beberapa bidang dan nilai ke dalamnya menggunakan perintah HSET. HSET menetapkan nilai bidang dalam hash. Perintah HSET menerima tiga argumen: kunci hash, bidang, dan nilai. Mari kita buat hash bernama myhash dan tetapkan bidang field1 ke nilai value1:

```
HSET myhash field1 value1
```

Anda seharusnya melihat output berikut:

```
(integer) 1
```

Ini berarti bidang baru telah ditambahkan ke hash. Nilai kembalian HSET adalah 1 jika bidang baru dalam hash dan 0 jika bidang sudah ada dan nilainya diperbarui.

Sekarang, mari kita tambahkan bidang lain:

```
HSET myhash field2 value2

(bilangan bulat) 1
```

Sekarang, mari kita ambil nilai bidang menggunakan perintah HGET. HGET mengambil nilai bidang dalam hash. Perintah HGET membutuhkan dua argumen: kunci hash dan bidang. Mari kita ambil nilai field1 dalam myhash:

```
HGET myhash field1
```

Anda seharusnya melihat output berikut:

```
"value1"
```

Ini membuktikan bahwa kita telah berhasil menyimpan dan mengambil pasangan bidangnilai dalam hash.

Mari kita coba memperbarui nilai bidang yang sudah ada:

```
HSET myhash field1 newvalue1
```

Anda seharusnya melihat:

```
(integer) 0
```

Ini menunjukkan bahwa bidang tersebut sudah ada, dan nilainya telah diperbarui.

Sekarang, mari kita ambil nilai field1 lagi:

```
HGET myhash field1  Anda harus melihat:

"newvalue1"
```

Ini mengonfirmasi bahwa nilainya telah diperbarui.

Mari kita tambahkan beberapa bidang lagi ke hash:

```
HSET myhash field3 value3

HSET myhash field4 value4
```

Pastikan untuk keluar dari redis-cli agar perintah Anda tercatat. Ketik:

```
exit
```

Ini akan mengembalikan Anda ke prompt terminal biasa.

Ringkasan

Dalam lab ini, kita menjelajahi struktur data dasar Redis, dimulai dengan string. Kita belajar cara terhubung ke server Redis menggunakan redis-cli, lalu menggunakan perintah SET untuk menyimpan nilai string yang terkait dengan kunci. Kita kemudian mengambil nilai-nilai tersebut menggunakan perintah GET. Kita juga melihat bagaimana Redis menangani angka yang disimpan sebagai string.

Selain itu, kita belajar cara membuat dan memanipulasi Daftar menggunakan LPUSH dan LRANGE, mengelola Kumpulan menggunakan SADD dan SMEMBERS, serta menjelajahi Hash menggunakan HSET dan HGET. Perintah-perintah ini memungkinkan Anda menyimpan dan mengambil berbagai jenis data di Redis, menjadikannya alat yang serbaguna dan dapat diandalkan ( ) untuk berbagai aplikasi. Ingatlah untuk keluar dari redis-cli setelah setiap langkah untuk memastikan perintah Anda tercatat untuk verifikasi.

Bagian 4: Operasi Dasar Key-Value di Redis

Pengantar

Dalam lab ini, Anda akan mempelajari operasi dasar kunci-nilai di Redis. Kita akan menggunakan antarmuka baris perintah redis-cli untuk berinteraksi dengan server Redis dan melakukan operasi dasar seperti menetapkan, mengambil, memeriksa keberadaan, menghapus, dan menetapkan waktu kedaluwarsa untuk kunci. Pada akhir lab ini, Anda akan memiliki pemahaman yang kuat tentang cara menggunakan Redis sebagai penyimpanan data sederhana.

Menyetel dan Mengambil Pasangan Kunci-Nilai

Pada langkah ini, kita akan fokus pada operasi inti penambahan dan pengambilan pasangan kunci-nilai di Redis. Ini merupakan dasar untuk menggunakan Redis sebagai penyimpanan data.

Redis menyimpan data sebagai pasangan kunci-nilai, mirip dengan kamus. Kunci adalah identifikasi unik, dan nilai adalah data yang terkait dengan kunci tersebut.

1. Menghubungkan ke Redis:

Buka terminal di VM LabEx. Anda seharusnya sudah berada di direktori ~/project. Hubungkan ke server Redis menggunakan perintah redis-cli:

```
redis-cli
```

Anda akan melihat prompt Redis: 127.0.0.1:6379>. Ini menandakan koneksi yang berhasil ke server Redis.

2. Menetapkan Pasangan Kunci-Nilai:

Mari kita atur pasangan kunci-nilai menggunakan perintah SET. Kita akan mengatur kunci mykey ke nilai myvalue.

```
SET mykey myvalue    Redis akan merespons dengan:
OK
```

Ini mengonfirmasi bahwa pasangan kunci-nilai telah disimpan dengan sukses.

3. Mendapatkan Nilai dari Sebuah Kunci:

Untuk mengambil nilai yang terkait dengan sebuah kunci, gunakan perintah GET. Mari kita ambil nilai dari mykey:

```
GET mykey
```

Redis akan merespons dengan:

```
"myvalue"
```

Ini menunjukkan bahwa kita telah berhasil mengambil nilai yang terkait dengan kunci mykey.

4. Menetapkan Pasangan Kunci-Nilai Lain:

Mari kita atur pasangan kunci-nilai lain dengan kunci dan nilai yang berbeda. Kali ini, kita akan menggunakan user:1001 sebagai kunci dan John sebagai nilai.

```
SET user:1001 John
```

Redis akan merespons dengan:

```
OK
```

5. Mendapatkan Nilai dari Kunci Baru:

Sekarang, mari kita ambil nilai dari kunci user:1001:

```
GET user:1001
```

Redis akan merespons dengan:

```
"John"
```

Anda telah berhasil mengatur dan mengambil pasangan kunci-nilai di Redis.

6. Keluar dari Redis CLI:

Penting untuk keluar dari Redis CLI setelah setiap langkah agar perintah tercatat dengan benar. Ketik:

```
exit
```

Memeriksa Keberadaan Kunci

Pada langkah ini, kita akan belajar cara memeriksa apakah sebuah kunci ada di Redis menggunakan perintah EXISTS. Hal ini berguna untuk menentukan apakah sebuah kunci telah disimpan sebelum mencoba mengambil nilainya atau melakukan operasi lain.

1. Terhubung ke Redis:

Hubungkan ke server Redis menggunakan perintah redis-cli:

```
redis-cli
```

Anda akan melihat prompt Redis: 127.0.0.1:6379>.

2. Periksa Keberadaan Kunci yang Sudah Ada:

Pada langkah sebelumnya, kita telah menetapkan kunci mykey. Mari periksa apakah kunci tersebut ada menggunakan perintah EXISTS:

```
EXISTS mykey
```

Redis akan merespons dengan:

```
(integer) 1
```

Respons 1 menunjukkan bahwa kunci mykey ada di Redis.

3. Periksa Keberadaan Kunci yang Tidak Ada:

Sekarang, mari kita periksa apakah kunci yang belum kita buat ada. Misalnya, mari kita periksa kunci bernama nonexistentkey:

```
EXISTS nonexistentkey Redis akan merespons dengan:
(integer) 0
```

Respons 0 menunjukkan bahwa kunci nonexistentkey tidak ada di Redis.

4. Keluar dari Redis CLI:

Keluar dari Redis CLI untuk memastikan perintah tercatat:

```
exit
```

Menghapus Kunci

Pada langkah ini, kita akan belajar cara menghapus kunci dari Redis menggunakan perintah DEL. Hal ini penting untuk mengelola data dan menghapus entri yang sudah tidak digunakan atau tidak diinginkan.

1. Terhubung ke Redis:

Hubungkan ke server Redis menggunakan perintah redis-cli:

```
redis-cli
```

Anda akan melihat prompt Redis: 127.0.0.1:6379>.

2. Menghapus Kunci yang Sudah Ada:

Pada langkah sebelumnya, kita telah menetapkan kunci mykey. Mari hapus kunci tersebut menggunakan perintah DEL:

```
DEL mykey
Redis akan merespons dengan:
(integer) 1
```

Respons (integer) 1 menunjukkan bahwa satu kunci telah dihapus dengan sukses.

3. Menghapus Kunci yang Tidak Ada:

Mari kita coba menghapus kunci yang tidak ada, seperti nonexistentkey:

```
DEL nonexistentkey Redis akan merespons dengan:
(bilangan bulat) 0
```

Respons (integer) 0 menunjukkan bahwa tidak ada kunci yang dihapus (karena kunci tersebut tidak ada).

4. Menghapus Beberapa Kunci:

Perintah DEL juga dapat digunakan untuk menghapus beberapa kunci sekaligus. Mari kita hapus kunci user:1001 yang kita buat sebelumnya, dan juga mencoba menghapus nonexistentkey lagi dalam perintah yang sama:

```
DEL user:1001 nonexistentkey Redis akan merespons dengan:
(bilangan bulat) 1
```

Respons (integer) 1 menunjukkan bahwa satu kunci telah dihapus dengan sukses (user:1001), dan upaya untuk menghapus nonexistentkey diabaikan karena kunci tersebut tidak ada.

5. Keluar dari Redis CLI:

Keluar dari Redis CLI:

```
exit
```

Menetapkan Waktu Kadaluwarsa Kunci

Pada langkah ini, kita akan belajar cara mengatur waktu kadaluwarsa untuk sebuah kunci di Redis menggunakan perintah EXPIRE dan perintah SET dengan parameter EX. Hal ini berguna untuk menghapus data secara otomatis setelah periode tertentu, seperti data sesi atau cache sementara.

1. Hubungkan ke Redis:

Hubungkan ke server Redis menggunakan perintah redis-cli:

```
redis-cli
Anda akan melihat prompt Redis: 127.0.0.1:6379&gt;.
```

2. Menetapkan Pasangan Kunci-Nilai dengan Waktu Kedaluwarsa (Metode 1: SET dengan parameter EX):

Redis memungkinkan Anda untuk menetapkan pasangan kunci-nilai dengan waktu kedaluwarsa dalam satu perintah menggunakan parameter EX. Mari kita tetapkan kunci sessionkey ke nilai sessionvalue dengan waktu kedaluwarsa 15 detik:

```
SET sessionkey sessionvalue EX 15
```

Redis akan merespons dengan:

```
OK
```

Perintah ini menetapkan pasangan kunci-nilai dan waktu kedaluwarsa dalam satu operasi, yang lebih efisien daripada menggunakan perintah terpisah.

3. Periksa TTL dari Pasangan Kunci yang Ditetapkan dengan EX:

Mari periksa sisa waktu hidup (TTL) untuk sessionkey:

```
TTL sessionkey
```

Redis akan merespons dengan jumlah detik yang tersisa hingga kunci kadaluwarsa (misalnya, (bilangan bulat) 14). Nilai ini akan sedikit kurang dari 15 karena waktu yang telah berlalu sejak kunci ditetapkan.

4. Menetapkan Pasangan Kunci-Nilai (Metode 2: Menggunakan Perintah EXPIRE Secara Terpisah):

Sebagai alternatif, Anda dapat menetapkan pasangan kunci-nilai terlebih dahulu, lalu menetapkan waktu kadaluwarsanya secara terpisah. Mari kita tetapkan kunci tempkey ke nilai tempvalue:

```
SET tempkey tempvalue
```

Redis akan merespons dengan:

```
OK
```

5. Menetapkan Waktu Kedaluwarsa Menggunakan Perintah EXPIRE:

Sekarang, mari kita atur waktu kedaluwarsa 10 detik untuk tempkey menggunakan perintah EXPIRE:

```
EXPIRE tempkey 10
```

Redis akan merespons dengan:

```
(bilangan bulat) 1
```

Respons (integer) 1 menunjukkan bahwa waktu kedaluwarsa telah berhasil diatur.

6. Periksa Waktu Hidup Tersisa (TTL):

Untuk memeriksa sisa waktu hidup (TTL) untuk tempkey, gunakan perintah TTL:

```
TTL tempkey
```

Redis akan merespons dengan jumlah detik yang tersisa hingga kunci kadaluwarsa (misalnya, (bilangan bulat) 9). Nilai tersebut mungkin sedikit kurang dari 10 karena waktu yang telah berlalu sejak pengaturan kadaluwarsa. Jika kunci tidak ada atau tidak memiliki kadaluwarsa, TTL akan mengembalikan -2 atau -1 masing-masing.

7. Keluar dari Redis CLI:

Keluar dari Redis CLI:

```
exit
```

Ringkasan

Dalam lab ini, Anda telah mempelajari operasi dasar kunci-nilai di Redis menggunakan antarmuka baris perintah redis-cli. Anda belajar cara terhubung ke server Redis dan menggunakan perintah SET untuk menyimpan data sebagai pasangan kunci-nilai. Anda juga berlatih mengambil nilai menggunakan perintah GET. Selain itu, Anda telah mempelajari cara memeriksa keberadaan kunci menggunakan perintah EXISTS, menghapus kunci menggunakan perintah DEL, dan menetapkan waktu kadaluwarsa untuk kunci menggunakan perintah SET dengan parameter EX serta perintah EXPIRE. Perintah SET dengan parameter EX sangat berguna karena memungkinkan Anda menetapkan pasangan kunci-nilai dengan waktu kadaluwarsa dalam satu operasi yang efisien. Ini adalah fondasi untuk menggunakan Redis sebagai penyimpanan data yang sederhana dan efisien.

Bagian 5: Pengelolaan Data Dasar di Redis

Pengantar

Dalam lab ini, Anda akan menjelajahi teknik pengelolaan data dasar di Redis. Anda akan memulai dengan mempelajari cara menambah dan mengurangi nilai numerik menggunakan perintah atomik INCR dan DECR, yang cocok untuk penghitung dan pembatas laju. Anda akan terhubung ke server Redis menggunakan redis-cli, menetapkan nilai awal, lalu menambah dan mengurangi nilai tersebut, dan memverifikasi hasilnya dengan perintah GET.
Selain itu, Anda akan belajar cara mengambil semua kunci yang disimpan di Redis

menggunakan perintah KEYS.
Ini adalah Lab Terpanduan, yang menyediakan instruksi langkah demi langkah untuk membantu Anda belajar dan berlatih. Ikuti instruksi dengan cermat untuk menyelesaikan setiap langkah dan mendapatkan pengalaman praktis. Data historis menunjukkan bahwa ini adalah lab tingkat pemula dengan tingkat penyelesaian 100%. Lab ini telah menerima tingkat ulasan positif 100% dari peserta.

Menambah dan Mengurangi Nilai dengan INCR/DECR

Pada langkah ini, Anda akan belajar cara menambah dan mengurangi nilai numerik yang disimpan di Redis menggunakan perintah INCR dan DECR. Perintah ini bersifat atomik, artinya dijamin akan dieksekusi tanpa gangguan dari klien lain, sehingga cocok untuk tugas seperti penghitung dan pembatas kecepatan.
Pertama, mari kita sambungkan ke server Redis menggunakan perintah redis-cli di terminal:

```
redis-cli
```

Sekarang, mari kita atur kunci bernama mycounter dengan nilai awal 10. Ini akan membuat pasangan kunci-nilai di database Redis. Kunci adalah mycounter, dan nilainya adalah 10.

```
SET mycounter 10
```

Anda seharusnya melihat output berikut, yang mengonfirmasi bahwa kunci telah ditetapkan dengan sukses:

```
OK
```

Perintah INCR menambah nilai kunci sebesar 1. Mari kita tambahkan nilai mycounter:

```
INCR mycounter
```

Outputnya akan menjadi nilai yang ditambah:

```
(integer) 11
```

Untuk memverifikasi nilai, Anda dapat menggunakan perintah GET:

```
GET mycounter
```

Hasilnya seharusnya:

```
"11"
```

Perintah DECR mengurangi nilai kunci sebesar 1. Mari kita kurangi nilai mycounter:

```
DECR mycounter
```

Hasilnya akan menjadi nilai yang dikurangi:

```
(integer) 10
```

Lagi, verifikasi nilai dengan GET:

```
GET mycounter
```

Hasilnya seharusnya:

```
"10"
```

Jika kunci tidak ada, `INCR` dan `DECR` menganggapnya seolah-olah berisi nilai 0. Mari coba menambah nilai kunci yang tidak ada bernama `newcounter`:

```
INCR newcounter
```

Hasilnya akan menjadi:

```
(integer) 1
```

Sekarang, periksa nilai `newcounter`:

```
GET newcounter
```

Hasilnya harus:

```
"1"
```

Demikian pula, mengurangi nilai kunci yang tidak ada akan dianggap sebagai 0 dan dikurangi menjadi -1.

```
DECR anothercounter
```

Hasilnya akan menjadi:

```
(bilangan bulat) -1
```

```
GET anothercounter
```

Hasilnya seharusnya:

```
"-1"
```

Akhirnya, keluar dari `redis-cli`:

```
exit
```

Penting untuk keluar dari `redis-cli` setelah menyelesaikan perintah agar riwayat perintah tercatat dengan benar.

Mendapatkan Semua Kunci dengan Perintah KEYS

Pada langkah ini, Anda akan belajar cara mengambil semua kunci yang disimpan di Redis menggunakan perintah `KEYS`. Meskipun `KEYS` berguna untuk pengembangan dan debugging, perintah ini umumnya tidak disarankan untuk lingkungan produksi dengan dataset besar karena dapat memblokir server saat mengiterasi semua kunci. Pertama, sambungkan ke server Redis menggunakan `redis-cli`:

```
redis-cli
```

Pada langkah sebelumnya, Anda telah membuat beberapa kunci: `mycounter`, `newcounter`, dan `anothercounter`. Mari tambahkan beberapa kunci lagi untuk membuat contoh ini lebih menarik.

```
SET user:1000:name "John"

SET user:1000:age 30

SET user:1001:name "Jane"
```

Sekarang, gunakan perintah `KEYS` dengan pola `*` untuk mengambil semua kunci dalam database:

```
KEYS *
```

Hasilnya akan berupa daftar semua kunci:

```
1) "anothercounter"

2) "user:1000:age"

3) "user:1001:name"

4) "mycounter"

5) "newcounter"
```

```
6) "pengguna:1000:nama"
```

Urutan kunci dapat bervariasi.
Anda juga dapat menggunakan pola untuk mengambil kunci yang sesuai dengan pola tertentu. Misalnya, untuk mengambil semua kunci yang dimulai dengan `user:`, gunakan perintah berikut:

```
KEYS user:*
```

Hasilnya akan menjadi:

```
1) "user:1000:age"
2) "user:1001:name"
3) "user:1000:name"
```

Contoh lain, untuk mengambil semua kunci yang mengandung `counter`, gunakan perintah berikut:

```
KEYS *counter*
```

Hasilnya akan menjadi:

```
1) "anothercounter"
2) "mycounter"
3) "newcounter"
```

Perlu diingat bahwa penggunaan `KEYS *` pada database besar dapat mempengaruhi kinerja. Untuk lingkungan produksi, pertimbangkan untuk menggunakan `SCAN` sebagai gantinya, yang mengiterasi melalui keyspace secara non-blocking.
Akhirnya, keluar dari `redis-cli`:

```
exit
```

Periksa Tipe Data dengan TYPE

Pada langkah ini, Anda akan belajar cara memeriksa tipe data dari sebuah kunci yang disimpan di Redis menggunakan perintah `TYPE`. Redis mendukung berbagai tipe data, termasuk string, daftar, himpunan, himpunan terurut, dan hash. Memahami tipe data dari sebuah kunci sangat penting untuk melakukan operasi yang sesuai padanya.
Pertama, sambungkan ke server Redis menggunakan `redis-cli`:

```
redis-cli
```

Pada langkah sebelumnya, Anda telah membuat beberapa kunci dengan nilai yang berbeda. Mari kita periksa tipe datanya.

Pertama, mari kita periksa tipe data kunci `mycounter`:

```
TYPE mycounter
```

Outputnya akan menjadi:

```
string
```

Hal ini menunjukkan bahwa `mycounter` disimpan sebagai string, meskipun nilainya berupa angka. Redis secara otomatis mengonversi nilai numerik menjadi string saat menggunakan `perintah SET`.
Selanjutnya, mari kita periksa tipe data dari kunci `user:1000:name`:

```
TYPE user:1000:name
```

Outputnya akan menjadi:

```
string
```

Ini juga menunjukkan bahwa `user:1000:name` disimpan sebagai string.
Sekarang, mari kita periksa tipe data dari kunci yang tidak ada, seperti `nonexistentkey`:

```
TIPE nonexistentkey
```

Hasilnya akan menjadi:

```
none
```

Ini menunjukkan bahwa kunci tersebut tidak ada dalam database.
Untuk lebih memperjelas jenis data, mari kita buat daftar:

```
LPUSH mylist "item1"

LPUSH mylist "item2"
```

Sekarang, periksa tipe data `mylist`:

```
TYPE mylist
```

Hasilnya akan menjadi:

```
list
```

Ini membuktikan bahwa `mylist` disimpan sebagai daftar.
Demikian pula, Anda dapat membuat jenis data lain seperti himpunan, himpunan terurut, dan hash, lalu gunakan perintah `TYPE` untuk memverifikasi jenisnya.
Misalnya:

```
SADD myset "member1"

SADD myset "member2"

TYPE myset
```

Outputnya akan menjadi:

```
set

ZADD mysortedset 1 "element1"

ZADD mysortedset 2 "element2"

TIPE mysortedset
```

Hasilnya akan menjadi:

```
zset

HSET myhash field1 "value1"

HSET myhash field2 "value2"

Tipe myhash
```

Hasilnya akan menjadi:

```
hash
```

Akhirnya, keluar dari `redis-cli`:

```
exit
```

Hapus Data dengan FLUSHDB

Pada langkah ini, Anda akan belajar cara menghapus semua data dari database Redis yang saat ini dipilih menggunakan perintah `FLUSHDB`. Perintah ini berguna untuk mereset database selama pengembangan atau pengujian. **Gunakan perintah ini dengan hati-hati di lingkungan produksi, karena akan menghapus permanen semua data di database.**

Pertama, sambungkan ke server Redis menggunakan `redis-cli`:

```
redis-cli
```

Sebelum menghapus database, mari verifikasi bahwa ada kunci yang ada. Gunakan perintah `KEYS *`:

```
KEYS *
```

Anda seharusnya melihat daftar kunci yang Anda buat pada langkah sebelumnya, seperti `mycounter`, `newcounter`, `user:1000:name`, dan `mylist`.
Sekarang, jalankan perintah `FLUSHDB`:

```
FLUSHDB
```

Outputnya akan menjadi:

```
OK
```

Ini menunjukkan bahwa database telah berhasil dibersihkan.
Untuk memverifikasi bahwa database sekarang kosong, gunakan perintah `KEYS *` lagi:

```
KEYS *
```

Hasilnya akan berupa daftar kosong:

```
(daftar kosong)
```

Hal ini membuktikan bahwa semua kunci telah dihapus dari basis data.
Penting untuk memahami bahwa `FLUSHDB` hanya menghapus database yang saat ini dipilih. Redis mendukung beberapa database (bernomor 0 hingga 15 secara default). Jika Anda ingin menghapus semua database, Anda dapat menggunakan perintah `FLUSHALL`. Namun, kami tidak akan menggunakan `FLUSHALL` dalam lab ini untuk menghindari kehilangan data yang tidak disengaja.
Akhirnya, keluar dari `redis-cli`:

```
exit
```

Ringkasan

Dalam lab ini, Anda mempelajari teknik pengelolaan data dasar di Redis. Secara khusus, Anda berlatih menambah dan mengurangi nilai numerik menggunakan perintah atomik `INCR` dan `DECR`, yang berguna untuk mengimplementasikan penghitung dan pembatas kecepatan. Anda juga belajar bahwa jika kunci tidak ada, perintah ini menganggapnya sebagai nilai 0.
Selain itu, Anda diperkenalkan dengan perintah `KEYS` untuk mengambil semua kunci yang disimpan di Redis. Meskipun berguna untuk pengembangan dan debugging, perintah ini umumnya tidak direkomendasikan untuk lingkungan produksi dengan dataset besar. Terakhir, Anda belajar cara memeriksa tipe data dan membersihkan database.

Bagian 6: Pengelolaan Kunci Lanjutan Redis

Pengantar

Dalam lab ini, Anda akan menjelajahi teknik pengelolaan kunci lanjutan di Redis. Anda akan memulai dengan mempelajari cara mengganti nama kunci menggunakan perintah RENAME, terhubung ke server Redis melalui redis-cli, menetapkan pasangan kunci-nilai, dan kemudian mengganti namanya, sambil memverifikasi perubahan sepanjang proses.

Selanjutnya, Anda akan mempelajari cara memindahkan kunci antara database Redis yang berbeda menggunakan perintah MOVE. Lab ini akan memandu Anda melalui proses memindahkan kunci dari satu database ke database lain dalam instance Redis. Lab ini juga mencakup pengaturan dan pengambilan beberapa kunci, serta iterasi kunci secara efisien.

Ini adalah Lab Terpanduan, yang menyediakan instruksi langkah demi langkah untuk membantu Anda belajar dan berlatih. Ikuti instruksi dengan cermat untuk menyelesaikan setiap langkah dan mendapatkan pengalaman praktis. Data historis menunjukkan bahwa ini adalah lab tingkat pemula dengan tingkat penyelesaian 83%. Lab ini telah menerima tingkat ulasan positif 100% dari peserta.

Mengganti Nama Kunci dengan RENAME

Pada langkah ini, Anda akan belajar cara mengganti nama kunci di Redis menggunakan perintah RENAME. Mengganti nama kunci dapat berguna untuk berbagai alasan, seperti memperbaiki kesalahan ketik, mengatur ulang data Anda, atau memperbarui nama kunci untuk mencerminkan perubahan dalam aplikasi Anda.

Pertama, mari kita terhubung ke server Redis menggunakan antarmuka baris perintah Redis (redis-cli). Buka terminal di direktori ~/project Anda dan ketik perintah berikut:

```
redis-cliRUN
```

Anda akan melihat prompt Redis: 127.0.0.1:6379>.

Sekarang, mari kita atur pasangan kunci-nilai yang dapat kita ganti namanya.

Gunakan perintah SET untuk membuat kunci bernama mykey dengan nilai myvalue:

```
SET mykey myvalueRUN
```

Anda seharusnya melihat output: OK.

Untuk memverifikasi bahwa kunci telah ditetapkan dengan benar, gunakan perintah GET:

```
GET mykeyRUN
```

Anda seharusnya melihat output: "myvalue".
Sekarang, mari kita ganti nama kunci mykey menjadi newkey menggunakan perintah RENAME:

```
RENAME mykey newkeyRUN
```

Anda seharusnya melihat output: OK.
Untuk memverifikasi bahwa kunci telah diganti namanya, coba dapatkan nilai dari kunci lama (mykey):

```
GET mykeyRUN
```

Anda seharusnya melihat output: (nil), yang berarti kunci tersebut tidak lagi ada. Sekarang, coba ambil nilai kunci baru (newkey):

```
GET newkeyRUN
```

Anda seharusnya melihat output: "myvalue", yang mengonfirmasi bahwa kunci telah berhasil diganti namanya.
Akhirnya, mari kita ganti nama kunci newkey kembali menjadi mykey untuk langkah selanjutnya:

```
RENAME newkey mykeyRUN
```

Anda harus melihat output: OK.
Pastikan kunci telah diganti namanya kembali:

```
GET mykeyRUN
```

Anda harus melihat output: "myvalue".
Ingat untuk keluar dari redis-cli dengan mengetik exit atau menekan Ctrl+D. Hal ini memastikan perintah Anda tercatat dengan benar.

```
exitRUN
```

Anda telah berhasil mengganti nama kunci di Redis menggunakan perintah RENAME.

Memindahkan Kunci Antara Database dengan MOVE

Pada langkah ini, Anda akan belajar cara memindahkan kunci dari satu database Redis ke database lain menggunakan perintah MOVE. Redis mendukung beberapa database logis dalam satu instance. Secara default, terdapat 16 database, bernomor dari 0 hingga 15. Perintah MOVE memungkinkan Anda memindahkan kunci dari database yang saat ini dipilih ke database lain.
Pertama, pastikan Anda terhubung ke server Redis menggunakan antarmuka baris perintah Redis (redis-cli). Buka terminal di direktori ~/project Anda dan ketik perintah berikut:

```
redis-cliRUN
```

Anda akan melihat prompt Redis: 127.0.0.1:6379>. Secara default, Anda terhubung ke database 0.
Kami sudah memiliki kunci bernama mykey dengan nilai myvalue di database 0 dari langkah sebelumnya. Mari kita verifikasi ini:

```
GET mykeyRUN
```

Anda seharusnya melihat output: "myvalue".
Sekarang, mari pindahkan kunci mykey dari database 0 ke database 1 menggunakan perintah MOVE:

```
MOVE mykey 1RUN
```

Anda seharusnya melihat output: (integer) 1, yang berarti kunci telah dipindahkan dengan sukses.
Untuk memverifikasi bahwa kunci telah dipindahkan, coba ambil nilai kunci mykey di database 0:

```
GET mykeyRUN
```

Anda seharusnya melihat output: (nil), yang berarti kunci tersebut tidak lagi ada di database 0.
Sekarang, beralih ke database 1 menggunakan perintah SELECT:

```
SELECT 1RUN
```

Anda seharusnya melihat output: OK.
Sekarang, coba dapatkan nilai kunci mykey di database 1:

```
GET mykeyRUN
```

Anda seharusnya melihat output: "myvalue", yang menunjukkan bahwa kunci telah berhasil dipindahkan ke database 1.
Akhirnya, mari pindahkan kunci mykey kembali ke database 0 untuk langkah selanjutnya. Pertama, kembali ke database 0:

```
SELECT 0RUN
```

Anda seharusnya melihat output: OK.
Sekarang, pindahkan kunci mykey dari database 1 ke database 0:

```
MOVE mykey 0RUN
```

Anda seharusnya melihat output: (error) ERR objek sumber dan tujuan sama.
Kesalahan ini terjadi karena Anda menjalankan perintah SELECT 0 dan kemudian MOVE mykey 0 dalam sesi yang sama. Perintah MOVE tidak diizinkan untuk memindahkan kunci ke database yang sama dengan database tempat kunci tersebut saat ini berada.
Untuk memindahkan kunci ke database yang berbeda, Anda perlu memilih database tujuan terlebih dahulu, lalu gunakan perintah MOVE.
Misalnya, pilih database 1 terlebih dahulu:

```
SELECT 1RUN
```

Anda seharusnya melihat output: OK.
Sekarang, pindahkan kunci mykey dari database 1 ke database 0:

```
PINDAHKAN 0RUN
mykey
```

Anda akan melihat output: (bilangan bulat) 1, yang berarti kunci telah dipindahkan dengan sukses.
Sekarang, kembali ke database 0:

```
SELECT 0RUN
```

Verifikasi bahwa kunci telah kembali ke database 0:

```
GET mykeyRUN
```

Anda seharusnya melihat output: "myvalue".
Ingat untuk keluar dari redis-cli dengan mengetik exit atau menekan Ctrl+D. Hal ini memastikan perintah Anda tercatat dengan benar.

```
exitRUN
```

Anda telah berhasil memindahkan kunci antara database Redis menggunakan perintah MOVE.

Menetapkan Beberapa Kunci dengan MSET

Pada langkah ini, Anda akan belajar cara menetapkan beberapa kunci beserta nilainya dalam satu perintah menggunakan perintah MSET di Redis. Ini lebih efisien daripada menggunakan beberapa perintah SET, karena mengurangi jumlah perjalanan bolak-balik ke server Redis.
Pertama, pastikan Anda terhubung ke server Redis menggunakan antarmuka baris perintah Redis (redis-cli). Jika Anda belum terhubung, buka terminal di direktori ~/project Anda dan ketik perintah berikut:

```
redis-cliRUN
```

Anda akan melihat prompt Redis: 127.0.0.1:6379>.
Kami sudah memiliki kunci bernama mykey dengan nilai myvalue di database 0 dari langkah sebelumnya. Sekarang, mari kita tambahkan dua kunci lagi, key1 dan key2, dengan nilai value1 dan value2 masing-masing, menggunakan perintah MSET:

```
MSET key1 value1 key2 value2RUN
```

Anda seharusnya melihat output: OK, yang berarti kunci-kunci tersebut telah berhasil ditetapkan.
Untuk memverifikasi bahwa kunci-kunci telah dibuat dengan benar, gunakan perintah GET untuk setiap kunci:

```
GET key1RUN
```

Anda seharusnya melihat output: "value1".

```
GET kunci2RUN
```

Anda seharusnya melihat output: "value2".

Sekarang, mari kita atur tiga kunci, termasuk mykey, menggunakan MSET:

```
MSET mykey newvalue key3 value3 key4 value4RUN
```

Anda seharusnya melihat output: OK.
Verifikasi nilai-nilai:

```
GET mykeyRUN
```

Anda seharusnya melihat output: "newvalue".

```
GET key3RUN
```

Anda seharusnya melihat output: "value3".

```
GET key4RUN
```

Anda seharusnya melihat output: "value4".
Ingat untuk keluar dari redis-cli dengan mengetik exit atau menekan Ctrl+D. Hal ini memastikan bahwa perintah Anda tercatat dengan benar.

```
exitRUN
```

Anda telah berhasil menetapkan beberapa kunci menggunakan perintah MSET.

Mengambil Beberapa Kunci dengan MGET

Pada langkah ini, Anda akan belajar cara mengambil nilai dari beberapa kunci dalam satu perintah menggunakan perintah MGET di Redis. Hal ini lebih efisien daripada menggunakan beberapa perintah GET, karena mengurangi jumlah perjalanan bolakbalik ke server Redis.
Pertama, pastikan Anda terhubung ke server Redis menggunakan antarmuka baris perintah Redis (redis-cli). Buka terminal di direktori ~/project Anda dan ketik perintah berikut:

```
redis-cliRUN
```

Anda akan melihat prompt Redis: 127.0.0.1:6379>.
Dari langkah sebelumnya, kita memiliki kunci dan nilai berikut:

- mykey: newvalue
- key1: value1

- key2: value2 □ key3: value3

□ key4 : value4

Sekarang, mari kita ambil nilai dari key1, key2, dan mykey menggunakan perintah MGET:

```
MGET key1 key2 mykeyRUN
```

Anda seharusnya melihat output:

```
1) "value1"

2) "value2"

3) "newvalue" RUN
```

Output ini menampilkan nilai-nilai kunci sesuai urutan yang ditentukan dalam perintah MGET.
Sekarang, mari kita ambil nilai dari key3, key4, dan kunci yang tidak ada key5 menggunakan perintah MGET:

```
MGET key3 key4 key5Jalankan
```

Anda seharusnya melihat output:

```
1) "value3"

2) "value4"

3) (nil) RUN
```

Perhatikan bahwa nilai untuk kunci yang tidak ada key5 adalah (nil).
Ingat untuk keluar dari redis-cli dengan mengetik exit atau menekan Ctrl+D. Hal ini memastikan perintah Anda tercatat dengan benar.

```
exitRUN
```

Anda telah berhasil mengambil beberapa kunci menggunakan perintah `MGET`.

Mengiterasi Kunci secara Efisien dengan SCAN

Pada langkah ini, Anda akan belajar cara mengiterasi kunci di Redis secara efisien menggunakan perintah `SCAN`. Berbeda dengan perintah `KEYS`, yang dapat memblokir server saat digunakan pada basis data besar, `SCAN` adalah iterator berbasis kursor yang mengambil kunci secara bertahap, sehingga meminimalkan dampak pada kinerja.

Pertama, pastikan Anda terhubung ke server Redis menggunakan antarmuka baris perintah Redis (`redis-cli`). Buka terminal di direktori `~/project` Anda dan ketik perintah berikut:

```
redis-cliRUN
```

Anda akan melihat prompt Redis: `127.0.0.1:6379&gt;`.

Dari langkah-langkah sebelumnya, kita memiliki kunci-kunci berikut:

- `mykey`
- `key1`
- `key2`
- `key3`
- `key4`

Perintah `SCAN` memerlukan nilai kursor awal, yang biasanya `0`. Perintah ini mengembalikan nilai kursor baru untuk iterasi berikutnya dan daftar kunci yang ditemukan pada iterasi saat ini. Mari kita mulai iterasi dengan kursor `0`:

```
SCAN 0RUN
```

Anda seharusnya melihat output serupa dengan ini:

```
1) "0"
2) 1) "key2"
   2) "key3"
   3) "key4"
```

```
4) "key1"
5) "mykey"
```
Jalankan

Elemen pertama dalam output ("0") adalah nilai kursor. Elemen kedua adalah array kunci yang ditemukan dalam iterasi ini. Karena nilai kursor yang dikembalikan adalah 0, hal ini menunjukkan bahwa iterasi telah selesai dan kita telah mengambil semua kunci dalam satu kali pemindaian.

Anda juga dapat menggunakan opsi MATCH untuk menyaring kunci berdasarkan pola. Misalnya, untuk menemukan semua kunci yang dimulai dengan "key", gunakan perintah berikut:

```
SCAN 0 MATCH key*
```
RUN

Anda seharusnya melihat output serupa dengan ini:

```
1) "0"
2) 1) "key2"
   2) "key3"
   3) "key4"
   4) "key1"
```
Jalankan

Kursor berada pada "0", yang berarti iterasi telah selesai, dan kita dapat melihat bahwa hanya kunci yang sesuai dengan pola "key*" yang dikembalikan (kecuali "mykey").

Anda juga dapat menggunakan opsi COUNT untuk memberi petunjuk kepada Redis tentang jumlah elemen yang ingin Anda ambil dalam satu panggilan. Perlu diingat bahwa ini hanya petunjuk, dan Redis mungkin mengembalikan lebih banyak atau lebih sedikit elemen. Misalnya:

```
SCAN 0 COUNT 3
```
RUN

Anda seharusnya melihat output serupa dengan ini:

```
1) "6"
2) 1) "key2"
   2) "key3"
   3) "key4"
```

Jalankan

Dalam hal ini, kita mendapatkan nilai kursor baru "6", yang menunjukkan bahwa masih ada kunci yang perlu dipindai. Anda akan melanjutkan iterasi menggunakan nilai kursor baru ini hingga menerima kursor "0".

Ingat untuk keluar dari `redis-cli` dengan mengetik `exit` atau menekan `Ctrl+D`. Hal ini memastikan bahwa perintah Anda tercatat dengan benar.

```
exit
```

Jalankan

Anda telah berhasil mengiterasi melalui kunci di Redis menggunakan perintah `SCAN`.

Ringkasan

Dalam lab ini, Anda belajar cara melakukan operasi manajemen kunci lanjutan di Redis. Secara khusus, Anda menggunakan perintah `RENAME` untuk mengganti nama kunci, mengubah identifikasinya sambil mempertahankan nilainya yang terkait. Anda juga berlatih memverifikasi penggantian nama yang berhasil dengan mengambil nilai menggunakan nama kunci lama dan baru, memastikan keberadaan kunci dan nilainya setelah operasi. Akhirnya, Anda mengganti nama kunci kembali ke nama aslinya untuk langkah-langkah selanjutnya.

Lab ini juga memperkenalkan konsep basis data Redis dan perintah `MOVE`, yang memungkinkan Anda memindahkan kunci antara basis data logis ini dalam satu instance Redis. Anda belajar cara menggunakan perintah `MSET` untuk menetapkan beberapa kunci sekaligus secara efisien dan perintah `MGET` untuk mengambil nilainya. Akhirnya, Anda menjelajahi perintah `SCAN`, alat yang kuat untuk mengiterasi melalui kunci dalam basis data besar tanpa memblokir server. Ingatlah untuk selalu keluar dari `redis-cli` setelah setiap langkah untuk memastikan perintah Anda tercatat untuk verifikasi.

Bagian 7: Transaksi Redis

Pengantar

Dalam lab ini, Anda akan menjelajahi transaksi Redis. Transaksi Redis memungkinkan Anda menjalankan sekelompok perintah sebagai operasi tunggal dan atomik, memastikan konsistensi data. Anda akan belajar cara memulai transaksi dengan perintah `MULTI`, mengantre perintah, dan kemudian menjalankan atau membatalkan transaksi. Anda akan berlatih menetapkan kunci, mengambil nilai, dan meningkatkan penghitung dalam transaksi.

Memulai Transaksi Redis

Pada langkah ini, Anda akan belajar cara memulai transaksi di Redis menggunakan perintah `MULTI`. Transaksi Redis memastikan bahwa serangkaian perintah dieksekusi sebagai satu unit atomik. Artinya, semua perintah berhasil atau tidak ada yang berhasil, sehingga menjamin integritas data.

Pertama, sambungkan ke server Redis menggunakan perintah `redis-cli` di terminal Anda:

```
redis-cliRUN
```

Sekarang Anda berada di lingkungan `redis-cli`, Anda dapat memulai transaksi. Ketik perintah berikut dan tekan Enter:

```
MULTIRUN
```

Anda seharusnya melihat output berikut:

```
OKRUN
```

Ini menunjukkan bahwa Redis telah masuk ke mode transaksi. Perintah apa pun yang Anda masukkan selanjutnya akan diantrekan dan dieksekusi bersama-sama saat Anda menggunakan perintah `EXEC`.

Mari kita antre perintah pertama kita. Kita akan menetapkan kunci bernama `mykey` dengan nilai `myvalue`. Ketik perintah berikut dan tekan Enter:

```
SET mykey "myvalue"RUN
```

Outputnya seharusnya:

```
QUEUEDRUN
```

Ini menunjukkan bahwa perintah SET telah berhasil ditambahkan ke antrian transaksi. Perintah ini tidak akan dieksekusi hingga kita secara eksplisit memerintahkan Redis untuk melakukannya.
Tetap buka lingkungan redis-cli untuk langkah berikutnya.

Menambahkan Perintah Lain ke Antrian dan Menjalankan Transaksi

Pada langkah ini, Anda akan menambahkan perintah tambahan ke antrian transaksi dan kemudian menjalankan seluruh transaksi menggunakan perintah EXEC.
Sekarang, mari antre perintah untuk mengambil nilai mykey:

```
GET mykeyRUN
```

Anda seharusnya melihat:

```
QUEUEDRUN
```

Selanjutnya, mari tambahkan perintah lain untuk menetapkan kunci yang berbeda, anotherkey, dengan nilai anothervalue:

```
SET anotherkey "anothervalue" RUN
```

Outputnya seharusnya:

```
QUEUEDRUN
```

Akhirnya, mari kita antre perintah INCR untuk menambah nilai counter bernama mycounter. Jika mycounter tidak ada, Redis akan membuatnya dan menginisialisasinya menjadi 0 sebelum menambah nilainya:

```
INCR mycounterRUN
```

Anda seharusnya melihat:

```
QUEUEDRUN
```

Anda telah mengantrekan beberapa perintah dalam transaksi. Untuk mengeksekusi semuanya sekaligus, gunakan perintah EXEC:

```
EXECRUN
```

Outputnya seharusnya mirip dengan ini:

```
1) OK
```

```
2) "myvalue"

3) OK

4) (bilangan bulat) 1
```

Mari kita uraikan output tersebut:

- `1) OK`: Hasil perintah `SET mykey "myvalue"`.
- `2) "myvalue"`: Hasil perintah `GET mykey`.
- `3) OK`: Hasil perintah `SET anotherkey "anothervalue"`.
- `4) (bilangan bulat) 1`: Hasil perintah `INCR mycounter`.

Semua perintah dalam transaksi dieksekusi secara atomik.
Tetap buka lingkungan `redis-cli` untuk langkah berikutnya.

Memverifikasi Eksekusi Transaksi

Pada langkah ini, Anda akan memverifikasi bahwa perintah dalam transaksi sebelumnya dieksekusi dengan benar dengan mengambil nilai kunci yang telah Anda tetapkan.
Untuk memeriksa nilai, gunakan perintah `GET` untuk setiap kunci:

```
GET mykey

Dapatkan kunci lain

GET mycounterRUN
```

Anda seharusnya melihat output berikut:

```
"myvalue"

"nilaiLain"

"1"Jalankan
```

Ini menunjukkan bahwa transaksi telah dieksekusi dengan sukses, dan nilai-nilai kunci telah diperbarui sesuai harapan.
Tetap buka lingkungan `redis-cli` untuk langkah berikutnya.

Membatalkan Transaksi dengan DISCARD

Pada langkah ini, Anda akan belajar cara membatalkan transaksi Redis menggunakan perintah `DISCARD`. Hal ini berguna jika Anda memutuskan tidak ingin menjalankan perintah yang telah antri.
Pertama, sambungkan ke server Redis dan mulai transaksi baru:

```
MULTIRUN
```

Sekarang, mari antrekan beberapa perintah:

```
SET mykey "newvalue"

INCR mycounterRUN
```

Anda seharusnya melihat respons `QUEUED` untuk setiap perintah.
Sekarang, alih-alih menjalankan transaksi, mari kita batalkan. Gunakan perintah `DISCARD`:

```
DISCARDRUN
```

Anda seharusnya melihat:

```
OKRUN
```

Ini mengonfirmasi bahwa transaksi telah dibatalkan, dan semua perintah yang antri telah dihapus. Untuk memverifikasi, Anda dapat memeriksa nilai `mykey` dan `mycounter`. Mereka seharusnya tidak diperbarui.

Pastikan untuk keluar dari lingkungan `redis-cli` agar perintah tercatat:

```
exitRUN
```

Ringkasan

Dalam lab ini, Anda telah belajar cara menggunakan transaksi Redis untuk menjalankan beberapa perintah secara atomik. Anda berlatih memulai transaksi dengan `MULTI`, mengantre perintah, menjalankan transaksi dengan `EXEC`, dan membatalkan transaksi dengan `DISCARD`. Pengetahuan ini sangat penting untuk menjaga konsistensi data di Redis saat melakukan operasi yang kompleks.