

ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA HỆ THỐNG THÔNG TIN

PHAN VỸ HÀO	19520524
NGUYỄN HOÀI NHÂN	19520200
NGUYỄN HOÀI BẢO	19520405
TRỊNH HUỲNH ĐĂNG	19521320

ĐỒ ÁN MÔN HỌC
MÔN HỌC: CƠ SỞ DỮ LIỆU PHÂN TÁN
LỚP: IS211.M11

KIẾN TRÚC CƠ SỞ DỮ LIỆU
REDIS

GIẢNG VIÊN HƯỚNG DẪN
ThS. Thái Bảo Trân
Nguyễn Minh Nhựt

TP. HỒ CHÍ MINH, 2021

ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA HỆ THỐNG THÔNG TIN

PHAN VỸ HÀO	19520524
NGUYỄN HOÀI NHÂN	19520200
NGUYỄN HOÀI BẢO	19520405
TRỊNH HUỲNH ĐĂNG	19521320

ĐỒ ÁN MÔN HỌC
MÔN HỌC: CƠ SỞ DỮ LIỆU PHÂN TÁN
LỚP: IS211.M11

KIẾN TRÚC CƠ SỞ DỮ LIỆU
REDIS

GIẢNG VIÊN HƯỚNG DẪN
ThS. Thái Bảo Trân
Nguyễn Minh Nhựt

TP. HỒ CHÍ MINH, 2021

This image shows a full page of white paper with horizontal dotted lines. The lines are evenly spaced and run across the width of the page, providing a guide for handwriting practice. There are no margins, text, or other markings on the page.

LỜI CẢM ƠN

Nhóm sinh viên bày tỏ lòng biết ơn sâu sắc đến hai vị giảng viên đáng kính là thạc sĩ Thái Bảo Trân và cử nhân Nguyễn Minh Nhựt. Nhờ vào lòng tâm huyết, kiến thức chuyên môn sâu sắc và sự cống hiến sự phạm tận tâm của thầy cô mà nhóm sinh viên mới có tự tin, kiến thức để hoàn thành đồ án lần này.

Cơ sở dữ liệu thuộc nhóm No-SQL là một nhóm cơ sở dữ liệu hiện đại, hấp dẫn. Nhóm chọn tìm hiểu về Redis nhằm có cái nhìn và kiến thức sâu sắc hơn về lĩnh vực này, cũng như thỏa mãn đam mê tìm kiếm kiến thức và bắt kịp xu hướng của ngành Công nghệ Thông tin.

Nhóm hoàn thành đồ án với tinh thần nghiêm túc, cẩn chu nhất có thể. Tuy vậy, những sai sót chắc chắn vẫn có thể xảy ra. Do đó, nhóm mong nhận được những góp ý, xây dựng sâu sắc nhất từ quý vị thầy cô, không những nhằm hoàn thiện kiến thức mà còn để hoàn thiện kỹ năng.

Xin chân thành cảm ơn quý Thầy Cô.

Mục lục

Tóm tắt nội dung	x
1 Tổng quan	1
1.1 Giới thiệu đề tài	1
1.1.1 Tổng quan	1
1.1.2 khái niệm NoSQL	2
1.2 Redis	3
1.2.1 Khái niệm	3
1.2.2 Lịch sử ra đời và nguồn gốc	3
1.2.3 Ưu và nhược điểm của Redis	4
1.2.4 Các kiểu dữ liệu	5
1.2.4.1 String	5
1.2.4.2 List	6
1.2.4.3 Set	9
1.2.4.4 Hashes	10
1.2.4.5 Sorted Sets	11
1.2.5 Redis Configuration	13
1.2.5.1 Tùy chỉnh tham số khi khởi động	14
1.2.5.2 Thay đổi config khi Redis server đang hoạt động	14
1.2.6 redis-cli	15
1.2.6.1 Chế độ command line	15
1.2.6.2 Interactive mode	18

2	Cài đặt Redis	21
2.1	Tài liệu hướng dẫn	21
2.2	Hướng dẫn cài đặt ở môi trường Windows	21
2.3	Cài đặt truy vấn phân tán cho Redis	26
3	Truy vấn dữ liệu trên môi trường phân tán	30
3.1	Mô tả bài toán	30
3.2	Cấu trúc cơ sở dữ liệu sử dụng	30
3.3	Các bước thực hiện	31
3.3.1	Insert dữ liệu	31
3.3.2	Truy vấn trên môi trường phân tán	33
3.3.2.1	Tương tác dữ liệu qua lại giữa 2 máy	33
3.3.2.2	Truy vấn phân tán	35

Danh sách hình vẽ

1.1	Nâng cấp (scale up) và dàn trải (scale out)	1
1.2	Các loại NoSQL chính	3
1.3	Ví dụ về các câu lệnh trong string	6
1.4	Ví dụ về các câu lệnh trong list-1	8
1.5	Ví dụ về các câu lệnh trong list-2	8
1.6	Ví dụ về các câu lệnh trong set	10
1.7	Ví dụ về các câu lệnh trong hash	11
1.8	Ví dụ về các câu lệnh trong sorted set	13
1.9	Minh họa câu lệnh Config Get/ Set.	15
1.10	Minh họa câu lệnh redis trong chế độ command line.	16
1.11	Cách thứ nhất để đọc đầu vào.	17
1.12	Cách thứ hai để đọc đầu vào.	17
1.13	Thực hiện một dòng lệnh nhiều lần.	18
1.14	Connect và reconnect trong chế độ Interactive mode.	19
1.15	Chế độ nhắc lệnh của Redis.	19
1.16	Redis help.	20
2.1	Kiểm tra tính năng Linus của Windows.	22
2.2	Kiểm tra tính năng Linus của Windows.	22
2.3	Mở hộp thoại PowerShell.	23
2.4	Cài đặt môi trường Ubuntu.	23
2.5	Kiểm tra cập nhật Ubuntu.	24

2.6	Cài đặt Redis.	24
2.7	Kiểm tra phiên bản Redis.	25
2.8	Khởi động Redis Server.	25
2.9	Truy cập vào server của Redis.	26
2.10	Kiểm tra và ràng buộc phiên bản 1 cho Ubuntu.	27
2.11	Lấy ra địa chỉ IP công khai của Redis.	28
2.12	Tắt tường lửa.	28
2.13	Ràng buộc địa chỉ IP, cho phép truy vấn từ xa.	29
2.14	Máy 2 truy vấn đến Redis ở máy 1.	29
3.1	Sorted set (zset) trong redis	30
3.2	Máy 1 tạo dữ liệu KHDL.	31
3.3	Máy 1 truy vấn dữ liệu tập trung.	32
3.4	Máy 2 tạo dữ liệu HTTT.	33
3.5	Máy 2 truy vấn dữ liệu tập trung.	33
3.6	Máy 2 truy vấn dữ liệu phân tán sang máy 1.	34
3.7	Máy 2 xóa một dữ liệu phân tán của máy 1.	34
3.8	Máy 2 thêm một dữ liệu phân tán vào KHDL của máy 1.	35
3.9	Cài đặt package Python trên Ubuntu.	36
3.10	Máy 2 hợp (union) dữ liệu ở cả hai máy.	36

Danh sách bảng

1.1	Các lệnh cơ bản của string	5
1.2	Các lệnh cơ bản của list	7
1.3	Các lệnh cơ bản của set	9
1.4	Các lệnh cơ bản của hash	10
1.5	Các lệnh cơ bản của sorted set	12

TÓM TẮT ĐỒ ÁN

Trong đồ án này, nhóm sinh viên thực hiện tìm hiểu về nền tảng lý thuyết, các câu lệnh, cấu trúc lưu trữ của **Redis**. Mục tiêu của đồ án là:

- Hiểu được cơ chế làm việc của một hệ cơ sở dữ liệu dạng No-SQL, cụ thể là Redis.
- Tìm hiểu lý thuyết, cách lưu trữ và phân phối dữ liệu của Redis, cả tập trung, từ xa và phân tán.
- Thông qua đó, nâng cao kỹ năng làm việc nhóm, tìm kiếm thông tin.

Đồ án của nhóm được chia làm 3 phần:

1. **Chương 1** bao gồm các thông tin về lịch sử, nguồn gốc ra đời của Redis, cũng như một số hệ thống kiểu dữ liệu, câu lệnh được Redis hỗ trợ, cùng một vài cách chỉnh sửa configuration thông dụng cơ bản.
2. **Chương 2** trình bày cách cài đặt Redis trên môi trường Windows. Đồng thời, chương 2 cũng hướng dẫn cài đặt Redis để truy vấn từ xa đến một server khác trên mạng Internet.
3. **Chương 3** trình bày một phiên làm việc thử nghiệm (demo) của nhóm sinh viên, sử dụng Redis trên môi trường Windows bằng cửa sổ PowerShell hoặc Ubuntu.

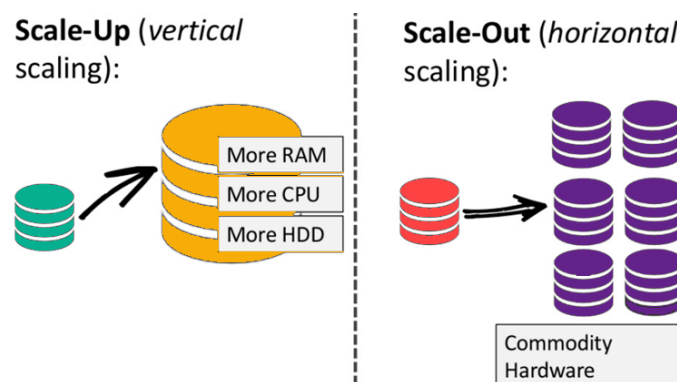
Chương 1

Tổng quan

1.1 Giới thiệu đề tài

1.1.1 Tổng quan

Các gã khổng lồ công nghệ Google, Facebook, Amazon phải xử lý lượng dữ liệu khổng lồ mỗi ngày trên các website dẫn đến việc nếu sử dụng một hệ quản trị cơ sở dữ liệu ràng buộc thì thời gian truy xuất sẽ rất lâu. Để giải quyết vấn đề này, nâng cấp (scale up) phần cứng lưu trữ thông tin là 1 cách hiệu quả nhưng tốn kém và sẽ không bao giờ đủ. Vậy nên, 1 phương án tối ưu không kém mà ít tốn chi phí hơn đó là dàn trải (scale out) dữ liệu phải xử lý ra cho nhiều máy chủ.



HÌNH 1.1: Nâng cấp (scale up) và dàn trải (scale out)

NoSQL có khả năng đàn trải (scale out) tốt hơn so với các hệ quản trị cơ sở dữ liệu ràng buộc truyền thống vì nó được thiết kế để ứng dụng trên website - nơi có lượng dữ liệu vô tận.

1.1.2 khái niệm NoSQL

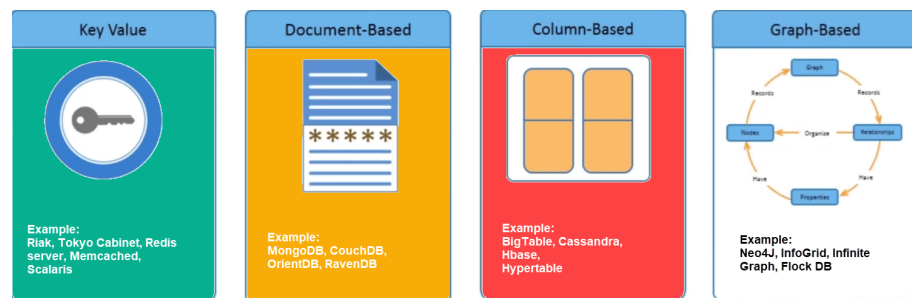
NoSQL là hệ quản trị cơ sở dữ liệu không ràng buộc (non-relational) thường được hiểu là "Not Only SQL" hoặc "Not SQL". Mục đích chính của việc sử dụng SQL là để xử lý Big data-lượng dữ liệu khổng lồ cần rất nhiều dung lượng lưu trữ hay các trang web tương tác với dữ liệu thời gian thực. Nó được giới thiệu lần đầu tiên vào năm 1998 khi Carlo Strozzi lần đầu tiên sử dụng khái niệm NoSQL để gọi cơ sở dữ liệu ràng buộc mã nguồn mở và nhẹ của ông.

Các tính năng chung của NoSQL gồm có không ràng buộc (non-relational), schema-free, API đơn giản và tính phân phối.

Nhìn chung, có thể chia NoSQL thành 4 loại chính:

- Key-value Pair Based
- Column-oriented Graph
- Graphs based
- Document-oriented

Không có một loại NoSQL nào trong số 4 loại trên tốt hơn 3 loại còn lại, tùy theo nhu cầu sử dụng mà chọn loại NoSQL phù hợp nhất. Trong báo cáo này chúng ta sẽ cùng tìm hiểu về 1 đại diện trong nhóm Key-value Pair Based là Redis.



HÌNH 1.2: Các loại NoSQL chính

1.2 Redis

1.2.1 Khái niệm

Redis (viết tắt của Remote Dictionary Server) là một mã nguồn mở được dùng để lưu trữ các dữ liệu có cấu trúc, có thể sử dụng như một bộ nhớ cache, database hay một message broker. Nó lưu trữ dữ liệu dưới dạng KEY-VALUE với nhiều tính năng được sử dụng rộng rãi, có thể hỗ trợ nhiều kiểu dữ liệu như: hashes, strings, lists, sets, sorted.

1.2.2 Lịch sử ra đời và nguồn gốc

Salvatore Sanfilippo khởi nghiệp tại Ý. Server của ông nhận lượng lớn thông tin từ nhiều trang web khác nhau thông qua JavaScript tracker. Những thông tin này gồm lưu trữ page view cho các trang, hiển thị theo thời gian thực cho user, kèm theo lưu trữ 1 lượng nhỏ lịch sử hiển thị của trang web. Khi lượng page view tăng vượt quá cao (hàng nghìn view trong 1 giây), Salvatore Sanfilippo thực sự bế tắc vì không thể tìm ra cách tiếp cận nào thực sự tối ưu cho việc thiết kế database của mình. Trong tình huống ấy, ông đã tạo ra Redis bằng ngôn ngữ Tcl và về sau ông dịch nó sang C. Sau vài tuần chạy thử thành công, ông quyết định công bố

source code, thông báo về dự án này trên Hacker News và nhanh chóng thu hút sự chú ý trong cộng đồng Ruby.

1.2.3 Ưu và nhược điểm của Redis

Ưu điểm của Redis:

- Redis có khả năng hỗ trợ thêm mới, thực hiện xóa, cập nhật dữ liệu rất nhanh chóng.
- Redis có thể lưu trữ được các dữ liệu ở dưới dạng KEY-VALUE.
- Hầu hết, các dữ liệu đều sẽ được lưu trữ ở trên RAM nên sẽ giúp cho quá trình truy xuất dữ liệu được thực hiện nhanh chóng hơn. Ngoài ra, bạn có thể thực hiện cấu hình để cho Redis thực hiện xóa dữ liệu trên ổ cứng được.
- Người dùng có thể thực hiện cấu hình cho key tự động để xóa đi một số khoảng thời gian nhất định.
- Có thể hỗ trợ cho nhiều loại kiểu dữ liệu khác nhau.
- Hỗ trợ Queue dựa vào cơ chế PUB/SUB nên chúng ta có thể dễ dàng sử dụng Redis để có thể làm hệ thống queue dành cho website xử lý theo tuần tự từng request. Ngoài ra, Redis còn cho phép người dùng sử dụng có thể linh hoạt hơn với nhiều kiểu dữ liệu làm việc khác nhau.
- Người dùng có thể để cho key tự động xóa trong một khoảng thời gian nhất định.
- Bạn có thể di chuyển dễ dàng key từ các cơ sở dữ liệu dạng này sang cơ sở dữ liệu dạng khác.
- Redis thường được trang bị với nhiều lệnh đặc biệt nên rất nhanh trong quá trình lấy cũng như ghi dữ liệu được dễ dàng hơn.

- Các tính năng Master - slave đều rất thích hợp cho những người dùng muốn gia tăng tính an toàn cho các dữ liệu, thu hẹp hoặc mở rộng cho các không gian lưu trữ Data.
- Người sử dụng có thể tìm kiếm dữ liệu một cách dễ dàng và nhanh chóng nhất.

Nhược điểm:

- Redis sử dụng RAM để lưu trữ cũng như truy xuất dữ liệu nên bộ dữ liệu càng lớn thì lượng RAM cần càng lớn.
- Không có ngôn ngữ truy vấn (chỉ có lệnh) và không hỗ trợ các lệnh liên quan tới tính toán.

1.2.4 Các kiểu dữ liệu

1.2.4.1 String

String là kiểu dữ liệu cơ bản nhất của Redis. Có 3 câu lệnh cơ bản với String, đó là GET, SET và DEL như trong bảng 1.1

Câu lệnh	Chức năng
GET key	Lấy giá trị value của key
SET key value	Thiết lập giá trị cho key
DEL key	Xóa key và giá trị tương ứng (Làm việc với tất cả kiểu dữ liệu, không chỉ string)

BẢNG 1.1: Các lệnh cơ bản của string

```
hoaibao1407@Admin:~$ redis-cli -a 123
127.0.0.1:6379> set baostring 123
OK
127.0.0.1:6379> get baostring
"123"
127.0.0.1:6379> set baostring 456
OK
127.0.0.1:6379> get baostring
"456"
127.0.0.1:6379> del baostring
(integer) 1
127.0.0.1:6379> get baostring
(nil)
```

HÌNH 1.3: Ví dụ về các câu lệnh trong string

1.2.4.2 List

List trong Redis là linked list, lưu trữ 1 danh sách có thứ tự (trước sau) của các string. Cách lưu trữ này giúp cho thời gian add thêm 1 phần tử vào đầu hoặc cuối list là hằng số, bất kể size của list là bao nhiêu. Lợi thế này cũng có 1 mặt trái là việc truy xuất đến phần tử theo index của linked list là lâu hơn rất nhiều so với array. Bảng 1.2 liệt kê các câu lệnh cơ bản khi làm việc với List.

Command	Ý nghĩa
LPOP key	Lấy phần tử ở đầu danh sách và xóa
RPOP key	Lấy giá trị ở cuối danh sách và xóa
LPUSH key value1 value2 ...	Thêm value1 value2... vào đầu danh sách
RPUSH key value1 value2 ...	Thêm value1 value2 ... vào cuối danh sách
LRANGE key start stop	Lấy các phần tử trong list từ vị trí start đến vị trí stop
LINSERT key BEFORE value1 value2	Thêm phần tử value2 vào trước phần tử value1 trong danh sách
LINSERT key AFTER value1 value2	Thêm phần tử value2 vào sau phần tử value1 trong danh sách
LSET key index value	Đặt giá trị value cho phần tử tại index
LREM key count value	Xóa những giá trị value trong key, count là số lượng value cần xóa.

BẢNG 1.2: Các lệnh cơ bản của list

```
127.0.0.1:6379> lpush baolist 123
(integer) 2
127.0.0.1:6379> lpush baolist 456
(integer) 3
127.0.0.1:6379> lrange baolist 0 -1
1) "456"
2) "123"
3) "123"
127.0.0.1:6379> del baolist
(integer) 1
127.0.0.1:6379> lrange baolist 0 -1
(empty list or set)
127.0.0.1:6379> clear
127.0.0.1:6379> lpush baolist 123
(integer) 1
127.0.0.1:6379> lpush baolist 456
(integer) 2
127.0.0.1:6379> lrange baolist 0 -1
1) "456"
2) "123"
127.0.0.1:6379> rpush baolist 789
(integer) 3
127.0.0.1:6379> lrange baolist 0 -1
1) "456"
2) "123"
3) "789"
127.0.0.1:6379> linsert baolist before 1 22
(integer) -1
```

HÌNH 1.4: Ví dụ về các câu lệnh trong list-1

```
127.0.0.1:6379> linsert baolist before 1 22
(integer) -1
127.0.0.1:6379> lrange baolist 0 -1
1) "456"
2) "123"
3) "789"
127.0.0.1:6379> linsert baolist before 123 22
(integer) 4
127.0.0.1:6379> lrange baolist 0 -1
1) "456"
2) "22"
3) "123"
4) "789"
127.0.0.1:6379> lset baolist 1 223
OK
127.0.0.1:6379> lrange baolist 0 -1
1) "456"
2) "223"
3) "123"
4) "789"
127.0.0.1:6379> lrem baolist 1 223
(integer) 1
127.0.0.1:6379> lrange baolist 0 -1
1) "456"
2) "123"
3) "789"
```

HÌNH 1.5: Ví dụ về các câu lệnh trong list-2

1.2.4.3 Set

Set trong Redis khá giống với list, nhưng khác 1 điều là các phần tử trong set không được sắp xếp theo thứ tự nào cả. Tuy nhiên, Redis đã tăng performance khi làm việc với set bằng cách sử dụng 1 bảng băm (hash table) để lưu trữ các phần tử của set. Hiểu đơn giản thì mỗi item được add vào set sẽ là 1 key trong bảng băm, còn value thì không có. Việc làm này giúp theo tác truy xuất dữ liệu trên SET nhanh hơn nhiều (do tận dụng ưu thế về tốc độ tìm kiếm trên bảng băm), nhất là khi muốn đảm bảo không bị trùng lặp phần tử trong set. Bảng 1.3 liệt kê các câu lệnh cơ bản khi làm việc với Set.

Command	Ý nghĩa
SADD key value1 value2 ..	Thêm các giá trị value1 value2 ... vào tập hợp
SCARD key	Lấy số lượng phần tử trong tập hợp
SMEMBERS key	Lấy các phần tử trong tập hợp
SISMEMBER key value	Check xem 1 phần tử có tồn tại trong set hay không (trả về 1 nếu tồn tại)
SREM key value	Xóa đi value trong set (trả về 1 nếu tồn tại)

BẢNG 1.3: Các lệnh cơ bản của set

```
127.0.0.1:6379> sadd baoset 1 2 3 4
(integer) 4
127.0.0.1:6379> sadd baoset 1
(integer) 0
127.0.0.1:6379> smembers baoset
1) "1"
2) "2"
3) "3"
4) "4"
127.0.0.1:6379> srem baoset 1
(integer) 1
127.0.0.1:6379> smembers baoset
1) "2"
2) "3"
3) "4"
127.0.0.1:6379> del baoset
(integer) 1
127.0.0.1:6379> smembers baoset
(empty list or set)
```

HÌNH 1.6: Ví dụ về các câu lệnh trong set

1.2.4.4 Hashes

Không giống như LIST và SET lưu trữ 1 tập dữ liệu là các string, HASH lưu trữ tập các map của key và value. Key vẫn là string, còn value có thể là string hoặc số. Nếu là số thì chúng ta có thể làm các thao tác tăng, giảm giá trị 1 cách đơn giản. HASH được coi là 1 mô hình thu nhỏ của Redis, khi dữ liệu được tổ chức dạng key-value. Bảng 1.4 liệt kê các câu lệnh cơ bản khi làm việc với Hash.

Command	Ý nghĩa
HSET key field value	Đặt giá trị cho field là value trong hash
HGET key field	Lấy giá trị của field trong hash
HDEL key field1 field2 ...	xóa field1, field2 ... trong hash
HGETALL key	Lấy tất cả các field và value của nó trong hash

BẢNG 1.4: Các lệnh cơ bản của hash

```
127.0.0.1:6379> hset baohash hash1 1
(integer) 1
127.0.0.1:6379> hset baohash hash1 2
(integer) 0
127.0.0.1:6379> hset baohash hash2 2
(integer) 1
127.0.0.1:6379> hget baohash hash2
"2"
127.0.0.1:6379> hgetall baohash
1) "hash1"
2) "2"
3) "hash2"
4) "2"
127.0.0.1:6379> hdel baohash hash1
(integer) 1
127.0.0.1:6379> hgetall baohash
1) "hash2"
2) "2"
```

HÌNH 1.7: Ví dụ về các câu lệnh trong hash

1.2.4.5 Sorted Sets

Sorted Set (ZSET) là 1 phiên bản đầy đủ của set, khi mà phần value của item được thiết lập, và bắt buộc là 1 số (float number) được gọi là score. Ở điểm này thì zset khá giống với hash khi lưu trữ 1 cặp key, value (trong zset gọi là member và score). Và vì là “sorted”, nên các cặp member-score được add vào sorted set sẽ được sắp xếp theo thứ tự của các score, nếu score trùng nhau thì tiếp tục sắp xếp theo member. Ngoài ra cũng cần chú ý là không cho phép 2 phần tử khác nhau của zset có member trùng nhau. Bảng 1.5 liệt kê các câu lệnh cơ bản khi làm việc với sorted set.

Command	Ý nghĩa
ZADD key score1 value1 score2 value2 ..	Thêm các phần tử value1 value2 vào sorted set với độ ưu tiên tương ứng là score1 và score2
ZCARD key	Lấy số lượng phần tử trong sorted set
ZRANGE key start stop	Lấy các phần tử trong tập hợp từ start đến stop
ZRANGE key start stop WITHSCORES	Lấy các phần tử trong tập hợp từ start đến stop kèm theo giá trị score của chúng
ZRANGEBYSCORE key min max WITHSCORES	Lấy các phần tử trong tập hợp có score từ min đến max kèm theo giá trị score của chúng (kết quả xếp tăng dần)
ZREVRANGEBYSCORE key max min WITH- SCORES	Lấy các phần tử trong tập hợp có score từ max đến min kèm theo giá trị score của chúng (kết quả xếp giảm dần)
ZSCORE key member	Lấy giá trị score của member
ZRANK key member	Lấy vị trí của member trong sorted set
ZCOUNT key score1 score2	Đếm số member có score tương ứng trong đoạn score1 đến score2
ZREM key member	Xóa phần tử member trong key
ZREMRANGEBYSCORE key min max	Xóa phần tử trong tập hợp có score trong khoảng min tới max
ZREMRANGEBYRANK key start stop	Xóa các phần tử trong tập hợp từ start đến stop

BẢNG 1.5: Các lệnh cơ bản của sorted set

```
127.0.0.1:6379> zadd baoss 9 bao 8 hao 7 nhan 7 dang 6 trung
(integer) 5
127.0.0.1:6379> zrange baoss 0 -1
1) "trung"
2) "dang"
3) "nhan"
4) "hao"
5) "bao"
127.0.0.1:6379> zrange baoss 0 -1 withscores
1) "trung"
2) "6"
3) "dang"
4) "7"
5) "nhan"
6) "7"
7) "hao"
8) "8"
9) "bao"
10) "9"
127.0.0.1:6379> zrangebyscore baoss 7 8 withscores
1) "dang"
2) "7"
3) "nhan"
4) "7"
5) "hao"
6) "8"
127.0.0.1:6379> zscore baoss dang
"7"
127.0.0.1:6379> zrank baoss trung
(integer) 0
127.0.0.1:6379> zcount baoss 7 8
(integer) 3
127.0.0.1:6379> zrem baoss trung
(integer) 1
127.0.0.1:6379> zrange baoss 0 -1 withscores
1) "dang"
2) "7"
3) "nhan"
4) "7"
5) "hao"
6) "8"
7) "bao"
8) "9"
```

HÌNH 1.8: Ví dụ về các câu lệnh trong sorted set

1.2.5 Redis Configuration

Chỉ cần cài đặt Redis là có thể sử dụng Redis ngay. Đó là vì Redis có sẵn một hệ các thông số mặc định được cài sẵn vào một tập tin tên là **redis.conf**. Tập tin này chứa một loạt các lệnh có định dạng đơn giản.

keyword argument1 argument2 ... argumentN

Ví dụ một dòng lệnh:

tcp-keepalive 300

1.2.5.1 Tùy chỉnh tham số khi khởi động

Từ phiên bản Redis 2.6 trở lên, người dùng có thể khởi động Redis với tham số tùy chọn. Tham số truyền vào giống y chang như trong file config, tuy nhiên chú ý là **keyword** được gắn với **-**. Điều này sẽ khởi tạo trong bộ nhớ tạm một file config tạm thời, với tham số là kết quả phép hợp giữa tham số truyền vào ở câu lệnh và tham số mặc định của Redis.

Ví dụ câu lệnh dưới đây khởi động một phiên Redis mới ở port 6380 là một bản sao của phiên Redis đang hoạt động ở 127.0.0.1 với port 6379.

```
./redis-server -port 6380 -replicaof 127.0.0.1 6379
```

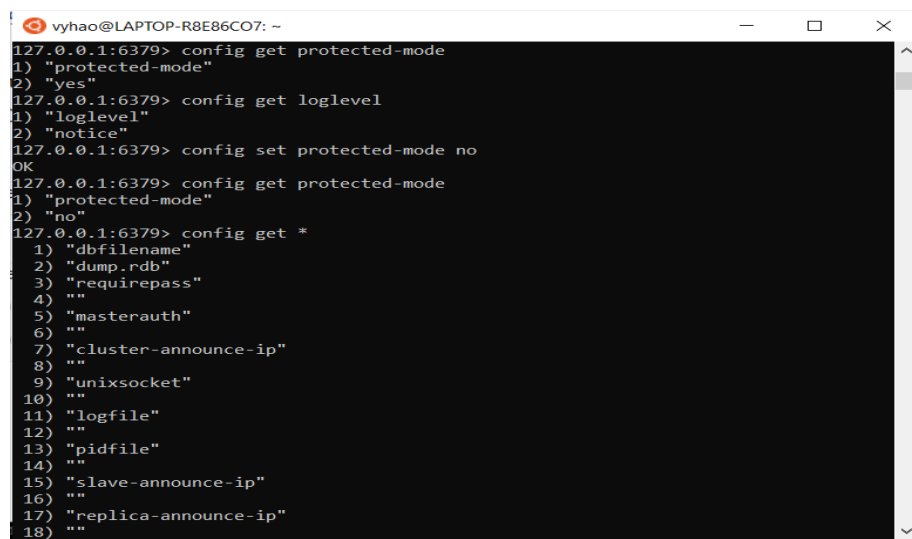
1.2.5.2 Thay đổi config khi Redis server đang hoạt động

Redis cho phép người dùng thay đổi thông số configuration mà không cần khởi động lại hệ thống hoặc tạo mới một phiên Redis bằng cặp lệnh đặc biệt: **CONFIG SET** và **CONFIG GET**.

Việc dùng cặp lệnh trên không tác động gì vào file config của hệ thống. Những thông số bị thay đổi bằng cặp lệnh trên chỉ mang tính tạm thời trong một phiên làm việc. Do đó, trong lần khởi động tiếp theo, Redis vẫn sẽ sử dụng các thông số trong file config.

Cách dùng cặp lệnh trên là:

- `CONFIG GET <keyword>`
- `CONFIG SET <keyword>`
- Để lấy ra toàn bộ thông số configuration mà Redis đang dùng, chúng ta dùng dấu `*` thay cho **<keyword>**, tức là `CONFIG GET *`
- Ví dụ minh họa ở hình 1.9.

A screenshot of a terminal window titled 'vyhao@LAPTOP-R8E86CO7: ~'. The terminal shows a series of Redis configuration commands and their outputs. The commands are: 'config get protected-mode', 'config get loglevel', 'config set protected-mode no', and 'config get *'. The outputs are: 'protected-mode' (yes/no), 'loglevel' (notice), 'protected-mode' (no), and a list of configuration keys including dbfilename, dump.rdb, requirepass, masterauth, cluster-announce-ip, unixsocket, logfile, pidfile, slave-announce-ip, and replica-announce-ip.

```
vyhao@LAPTOP-R8E86CO7: ~  
127.0.0.1:6379> config get protected-mode  
1) "protected-mode"  
2) "yes"  
127.0.0.1:6379> config get loglevel  
1) "loglevel"  
2) "notice"  
127.0.0.1:6379> config set protected-mode no  
OK  
127.0.0.1:6379> config get protected-mode  
1) "protected-mode"  
2) "no"  
127.0.0.1:6379> config get *  
1) "dbfilename"  
2) "dump.rdb"  
3) "requirepass"  
4) ""  
5) "masterauth"  
6) ""  
7) "cluster-announce-ip"  
8) ""  
9) "unixsocket"  
10) ""  
11) "logfile"  
12) ""  
13) "pidfile"  
14) ""  
15) "slave-announce-ip"  
16) ""  
17) "replica-announce-ip"  
18) ""
```

HÌNH 1.9: Minh họa câu lệnh Config Get/ Set.

1.2.6 redis-cli

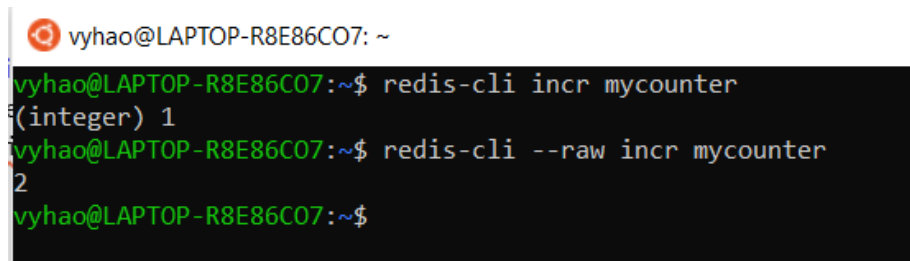
redis-cli là một giao diện dòng lệnh của Redis cho phép người dùng truyền câu lệnh, tương tác và nhận phản hồi ngay lập tức từ terminal.

redis-cli có hai chế độ lệnh khác nhau:

1. **Interactive mode** (chế độ tương tác): chế độ này cho phép người dùng liên tục nhập lệnh và nhận phản hồi ngay tức khắc. Chế độ này chỉ dừng lại khi nhận lệnh *"exit"* từ phía người dùng.
2. **Chế độ command line**: Chế độ này nhận lệnh như một tham số truyền vào cùng lệnh *redis-cli*, sau đó thực thi và trả về output dạng chuẩn.

1.2.6.1 Chế độ command line

Trong chế độ này, để nhận một phản hồi từ server, chúng ta đơn giản chỉ cần truyền câu lệnh trên cùng một dòng với **redis-cli**, ví dụ như minh họa trong hình 1.10.



```
vyhao@LAPTOP-R8E86C07: ~  
vyhao@LAPTOP-R8E86C07:~$ redis-cli incr mycounter  
(integer) 1  
vyhao@LAPTOP-R8E86C07:~$ redis-cli --raw incr mycounter  
2  
vyhao@LAPTOP-R8E86C07:~$
```

HÌNH 1.10: Minh họa câu lệnh redis trong chế độ command line.

Trong hình 1.10, dòng đầu tiên gọi thực hiện tăng giá trị biến *mycounter* mặc định trong hệ thống thêm 1, kết quả trả về là "1". Mặc định giá trị trả về luôn là string trong dấu ngoặc kép vì nó được ghi vào command line, dù giá trị bên trong dấu " có thể là string, array, list, NULL, error,... Để lấy ra giá trị nguyên bản của câu lệnh, chúng ta cần thêm tham số **-raw**, như dòng thứ hai trong hình 1.10.

Truy cập cơ sở dữ liệu với host, port.

- Mặc định, nếu người dùng chỉ gọi lệnh **redis-cli**, Redis sẽ kết nối đến server tại địa chỉ 127.0.0.1 port 6379.
- Để truy cập đến một cơ sở dữ liệu tại một địa chỉ khác, người dùng phải xác định địa chỉ host name bằng **-h**, xác định địa chỉ port (nếu khác mặc định) bằng **-p**. Ví dụ:

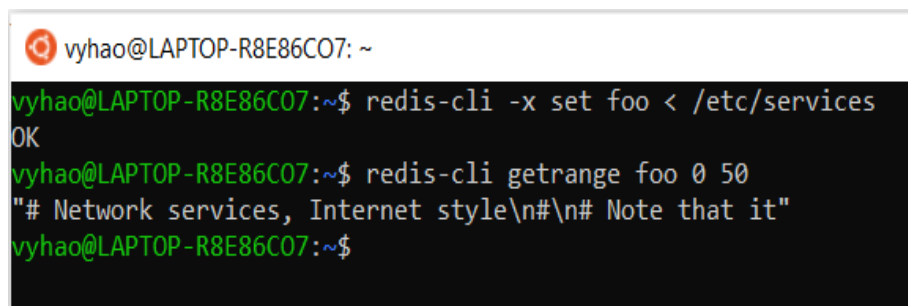
```
$ redis-cli -h redis15.localnet.org -p 6390 ping  
PONG
```

- Nếu cơ sở dữ liệu được khóa bằng mật khẩu, người dùng cần xác thực bằng câu lệnh **-a <password>**.

```
$ redis-cli -a myUnguessablePazzzzzword123 ping  
PONG
```

Đọc đầu vào từ chương trình khác: Có hai cách.

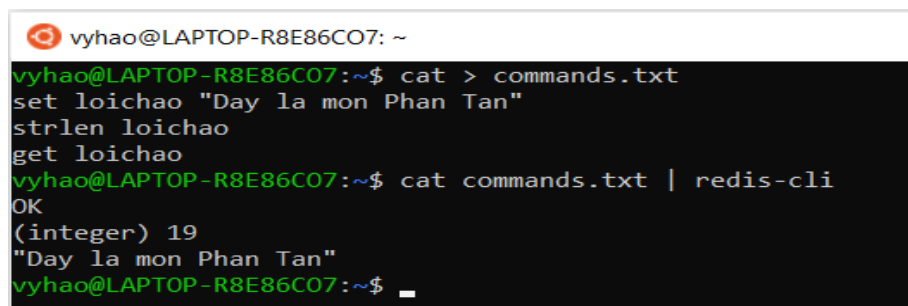
1. **Sử dụng -x:** Giả sử người dùng set một key **foo** có nội dung của file **/etc/services**, người dùng thực hiện gần như hình 1.11.



```
vyhao@LAPTOP-R8E86C07: ~  
vyhao@LAPTOP-R8E86C07:~$ redis-cli -x set foo < /etc/services  
OK  
vyhao@LAPTOP-R8E86C07:~$ redis-cli getrange foo 0 50  
"# Network services, Internet style\n#\n# Note that it"  
vyhao@LAPTOP-R8E86C07:~$
```

HÌNH 1.11: Cách thứ nhất để đọc đầu vào.

2. **Truyền vào một dãy lệnh viết sẵn trong một file text:** Thực hiện như hình 1.12.



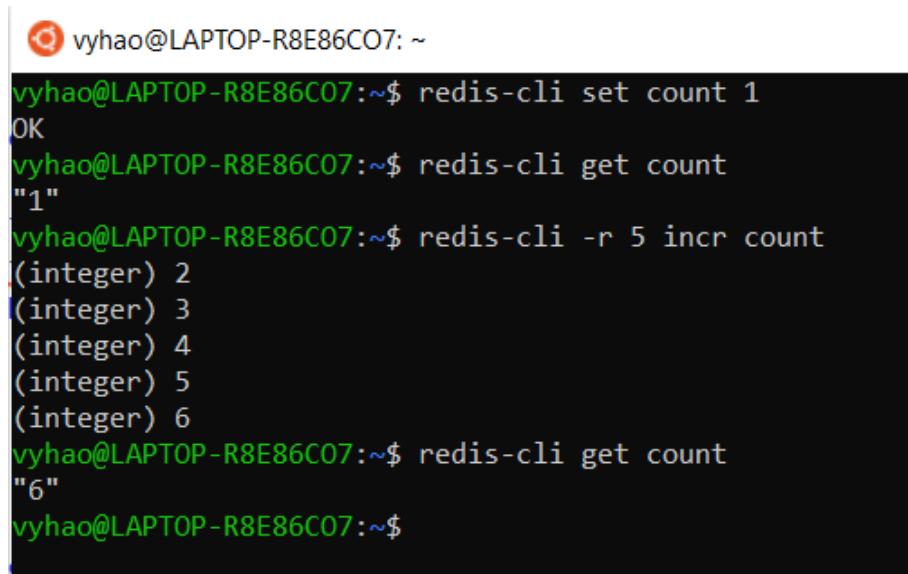
```
vyhao@LAPTOP-R8E86C07: ~  
vyhao@LAPTOP-R8E86C07:~$ cat > commands.txt  
set loichao "Day la mon Phan Tan"  
strlen loichao  
get loichao  
vyhao@LAPTOP-R8E86C07:~$ cat commands.txt | redis-cli  
OK  
(integer) 19  
"Day la mon Phan Tan"  
vyhao@LAPTOP-R8E86C07:~$
```

HÌNH 1.12: Cách thứ hai để đọc đầu vào.

Thực thi liên tục một câu lệnh

- Điều này cho phép theo dõi hoặc tự động thực hiện một công việc nào đó trong Redis.

- Việc này có thể thông qua hai tham số **-r <count>** (thực thi lệnh bao nhiêu lần) và **-i <delay>** (giữa hai lần thực thi cách nhau bao nhiêu giây), ví dụ hình 1.13.
- Để thực thi lệnh mãi mãi, đặc biệt cần thiết trong những tác vụ theo dõi, người dùng cần truyền tham số **-r -1** thay cho **-r <count>**.



```
vyhao@LAPTOP-R8E86C07: ~  
vyhao@LAPTOP-R8E86C07:~$ redis-cli set count 1  
OK  
vyhao@LAPTOP-R8E86C07:~$ redis-cli get count  
"1"  
vyhao@LAPTOP-R8E86C07:~$ redis-cli -r 5 incr count  
(integer) 2  
(integer) 3  
(integer) 4  
(integer) 5  
(integer) 6  
vyhao@LAPTOP-R8E86C07:~$ redis-cli get count  
"6"  
vyhao@LAPTOP-R8E86C07:~$
```

HÌNH 1.13: Thực hiện một dòng lệnh nhiều lần.

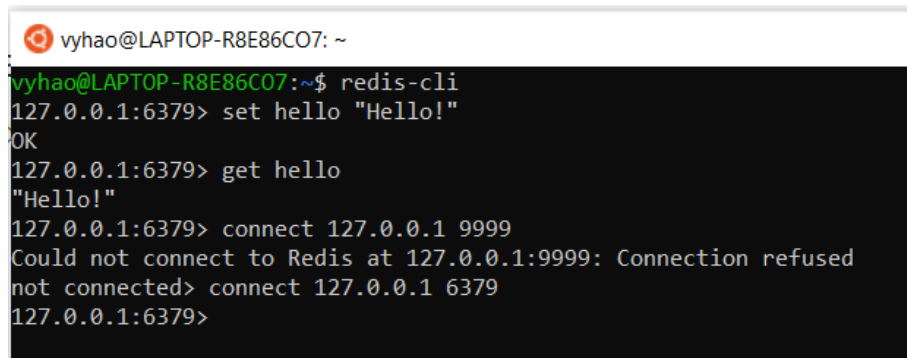
1.2.6.2 Interactive mode

Đa số người dùng Redis sẽ chuộng giao diện này. Chỉ cần người dùng nhập lệnh **redis-cli** là có thể vào chế độ này. Ở đây, người dùng có thể liên tiếp nhập các câu lệnh và nhận phản hồi ngay lập tức, như một cách tương tác (interact) trong thời gian thực.

Trong chế độ này, người dùng có thể dùng lệnh **connect** để kết nối đến một server khác ở một port khác mà người dùng chỉ định, hoặc lệnh **debug restart** để khởi động lại kết nối, như hình 1.14.

```
connect <host> <port>
```

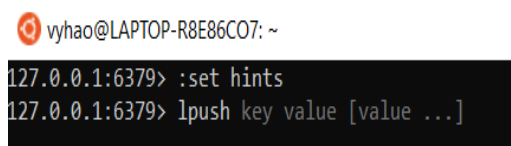
debug restart



```
vyhao@LAPTOP-R8E86C07: ~  
vyhao@LAPTOP-R8E86C07:~$ redis-cli  
127.0.0.1:6379> set hello "Hello!"  
OK  
127.0.0.1:6379> get hello  
"Hello!"  
127.0.0.1:6379> connect 127.0.0.1 9999  
Could not connect to Redis at 127.0.0.1:9999: Connection refused  
not connected> connect 127.0.0.1 6379  
127.0.0.1:6379>
```

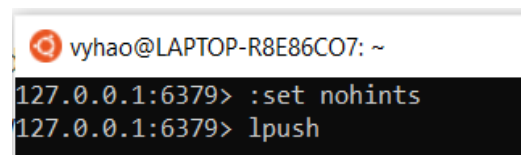
HÌNH 1.14: Connect và reconnect trong chế độ Interactive mode.

Trong chế độ này, người dùng còn có thể cài đặt chế độ nhắc lệnh **set hint** hoặc không nhắc lệnh **set nohint** như hình 1.15.



```
vyhao@LAPTOP-R8E86C07: ~  
127.0.0.1:6379> :set hints  
127.0.0.1:6379> lpush key value [value ...]
```

(A) Set hints.



```
vyhao@LAPTOP-R8E86C07: ~  
127.0.0.1:6379> :set nohints  
127.0.0.1:6379> lpush
```

(B) Set nohints.

HÌNH 1.15: Chế độ nhắc lệnh của Redis.

Redis hỗ trợ lệnh **help** để người dùng xem lại cách dùng một lệnh nào đó khi cần.

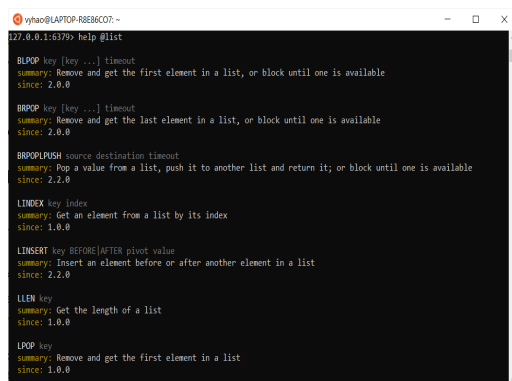
- Để xem hướng dẫn của một nhóm lệnh, thực hiện lệnh để xem toàn bộ lệnh trong nhóm đó.

help @<category>

Trong đó, các category bao gồm generic, list, set, sorted_set, hash, pubsub, transactions, connection, server, scripting, hyperloglog. Ví dụ hình 1.16a.

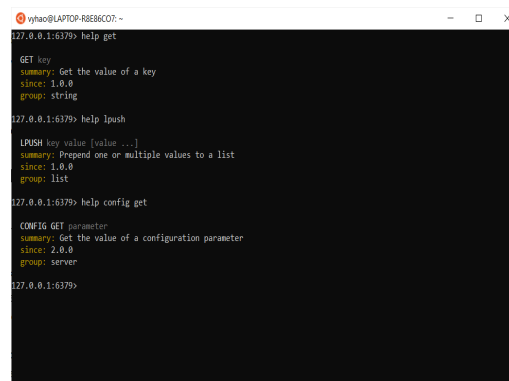
- Để xem nhắc lệnh về một câu lệnh cụ thể, gọi lệnh:

`help <commandname>`



```
vyhao@LAPTOP-R8E86C07: ~  
127.0.0.1:6379> help @list  
  
BLPOP key [key ...] timeout  
summary: Remove and get the first element in a list, or block until one is available  
since: 2.0.0  
  
BRPOP key [key ...] timeout  
summary: Remove and get the last element in a list, or block until one is available  
since: 2.0.0  
  
BRPOPLPUSH source destination timeout  
summary: Pop a value from a list, push it to another list and return it; or block until one is available  
since: 2.2.0  
  
LINDEX key index  
summary: Get an element from a list by its index  
since: 1.0.0  
  
LINSERT key BEFORE|AFTER pivot value  
summary: Insert an element before or after another element in a list  
since: 2.2.0  
  
LLEN key  
summary: Get the length of a list  
since: 1.0.0  
  
LPOP key  
summary: Remove and get the first element in a list  
since: 1.0.0
```

(A) Lệnh help nhóm lệnh.



```
vyhao@LAPTOP-R8E86C07: ~  
127.0.0.1:6379> help get  
  
GET key  
summary: Get the value of a key  
since: 1.0.0  
group: String  
  
127.0.0.1:6379> help lpush  
  
LPUSH key value [value ...]  
summary: Prepend one or multiple values to a list  
since: 1.0.0  
group: list  
  
127.0.0.1:6379> help config get  
  
CONFIG GET parameter  
summary: Get the value of a configuration parameter  
since: 2.0.0  
group: server  
  
127.0.0.1:6379>
```

(B) Lệnh help lệnh cụ thể.

HÌNH 1.16: Redis help.

Chương 2

Cài đặt Redis

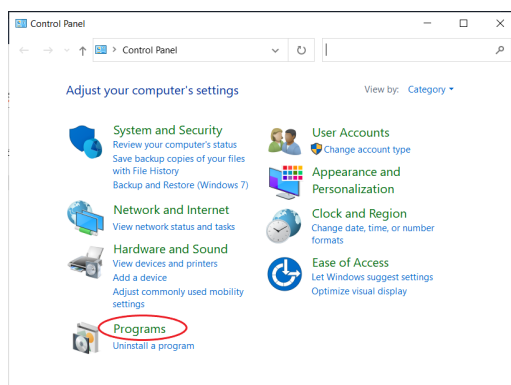
2.1 Tài liệu hướng dẫn

Redis biên soạn và hướng dẫn cách cài đặt dành cho Windows ở đường link: [Running Redis on Windows 10](#).

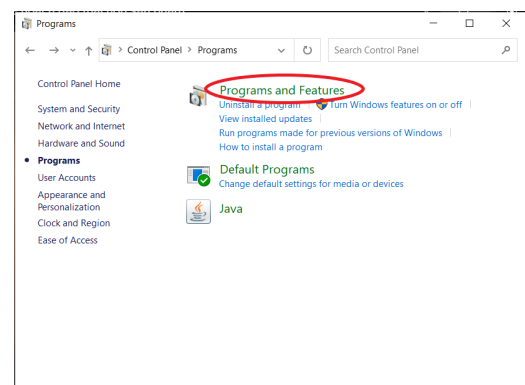
2.2 Hướng dẫn cài đặt ở môi trường Windows

1. Kiểm tra tính năng của hệ điều hành có hỗ trợ Linux hay chưa bằng cách:
 - Mở cửa sổ **Control Panel** của hệ điều hành Windows.
 - Truy cập vào mục **Program** (hình 2.1a). Sau đó truy cập **Program and Features** (hình 2.1b).
 - Chọn bên góc phải **Turn Windows features on or off**. Lúc này một hộp thoại nhỏ sẽ hiện ra. (hình 2.2a)
 - Tìm kiếm mục **Windows Subsystem for Linux** đã xuất hiện và tick chọn chưa (hình 2.2b). Nếu đã có thì tiến đến bước tiếp theo.

- Nếu chưa xuất hiện mục Subsystem ở bước trên, thì thực hiện mở PowerShell của Windows (hình 2.3), và gõ lệnh `wsl --install` để cài đặt. Đối với những máy không tương thích với việc cài đặt bằng câu lệnh trên thì có thể cài đặt wsl thủ công tại trang hướng dẫn: Manual installation steps for older versions of WSL.

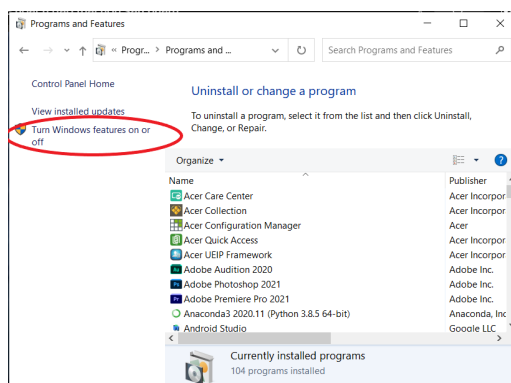


(A) Truy cập Programs trong Control Panel.

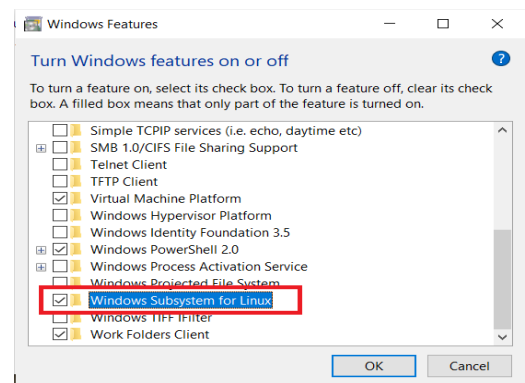


(B) Truy cập Programs and Features.

HÌNH 2.1: Kiểm tra tính năng Linux của Windows.



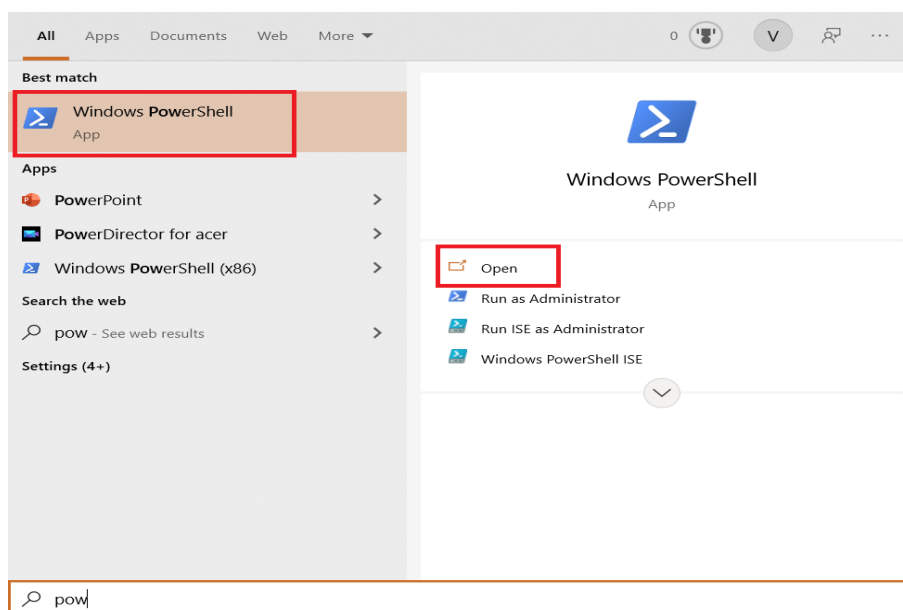
(A) Truy cập Turn Windows features on or off.



(B) Truy cập Programs and Features.

HÌNH 2.2: Kiểm tra tính năng Linux của Windows.

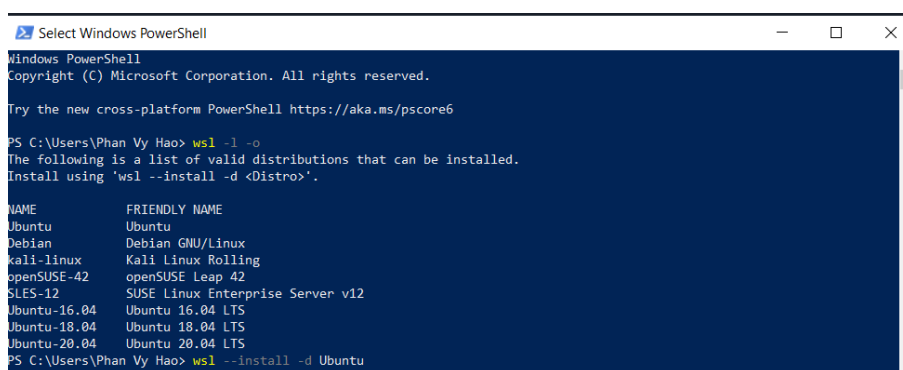
2. Mở hộp thoại PowerShell (hình 2.3).



HÌNH 2.3: Mở hộp thoại PowerShell.

3. Gõ lệnh `wsl -l -o` để xem các phiên bản Ubuntu có thể cài đặt được (hình 2.4). Sau đó, chọn một phiên bản và tiến hành cài đặt bằng câu lệnh:

`wsl --install -d <phiên bản>`

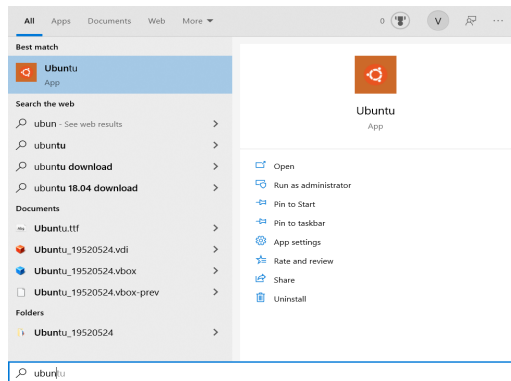


HÌNH 2.4: Cài đặt môi trường Ubuntu.

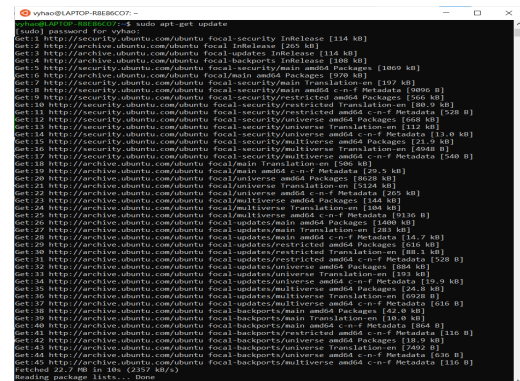
4. Lúc này một phiên bản Ubuntu đã được cài đặt. Tại thanh tìm kiếm ở taskbar, tìm kiếm mở phiên bản Ubuntu đã cài (hình 2.5a). Lúc này một

hộp thoại sẽ hiện ra. Gõ câu lệnh sau để kiểm tra cập nhật của các package của Ubuntu (hình 2.5b).

```
sudo apt-get update
```



(A) Mở phiên bản Ubuntu đã cài.

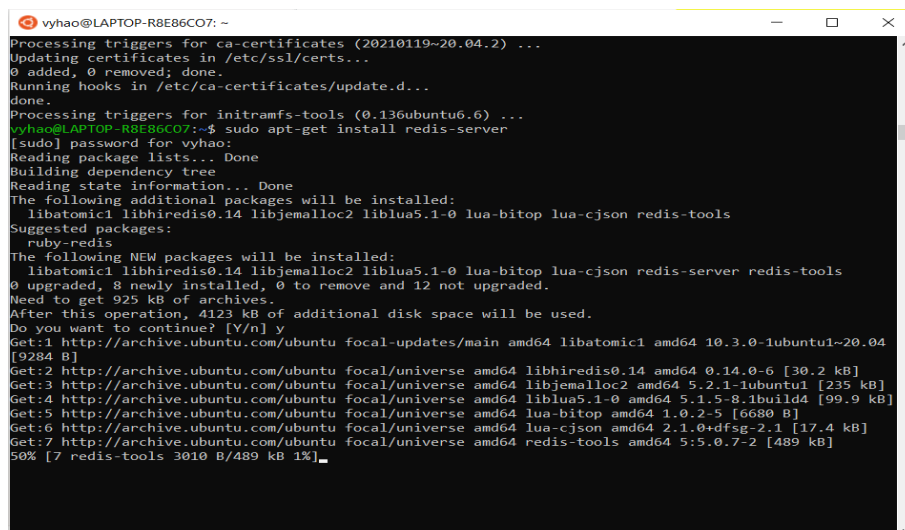


(B) Câu lệnh sudo apt-get update.

HÌNH 2.5: Kiểm tra cập nhật Ubuntu.

5. Nhập lệnh `sudo apt-get upgrade` để nâng cấp những gói tin (package) của Ubuntu.

6. Nhập lệnh `sudo apt-get install redis-server` để cài đặt Redis (hình 2.6).



HÌNH 2.6: Cài đặt Redis.

7. Nhập lệnh `redis-server --version` để kiểm tra phiên bản Redis đã cài (hình 2.7).

```
Processing triggers for libc-bin (2.31-0ubuntu9.2) ...  
vyhao@LAPTOP-R8E86C07:~$ redis-server --version  
Redis server v=5.0.7 sha=00000000:0 malloc=jemalloc-5.2.1 bits=64 build=636cde3b5c7a3923
```

HÌNH 2.7: Kiểm tra phiên bản Redis.

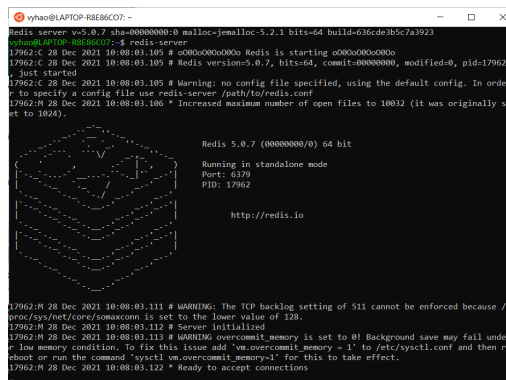
8. Nhập lệnh `redis-server` để khởi động Redis server (hình 2.8a). Nhớ cấp quyền truy cập tường lửa khi Redis yêu cầu (hình 2.8).

Ngoài ra còn có lệnh để khởi động lại Redis server:

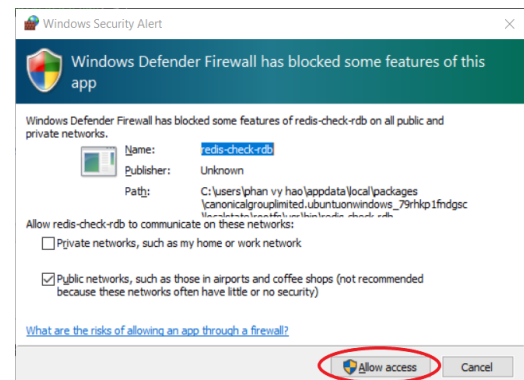
```
sudo service redis-server restart
```

Lệnh để tắt Redis server:

```
sudo service redis-server stop
```



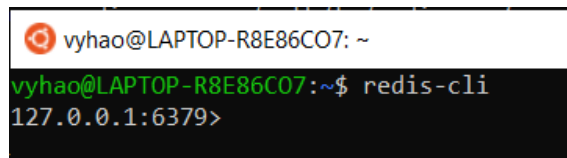
(A) Nhập lệnh `redis-server`.



(B) Cấp quyền cho Redis.

HÌNH 2.8: Khởi động Redis Server.

9. Mở một cửa sổ **Ubuntu** khác và nhập lệnh `redis-cli` để truy cập vào server của Redis để tương tác (hình 2.9).



HÌNH 2.9: Truy cập vào server của Redis.

10. Hoàn thành cài đặt Redis trên môi trường Windows.

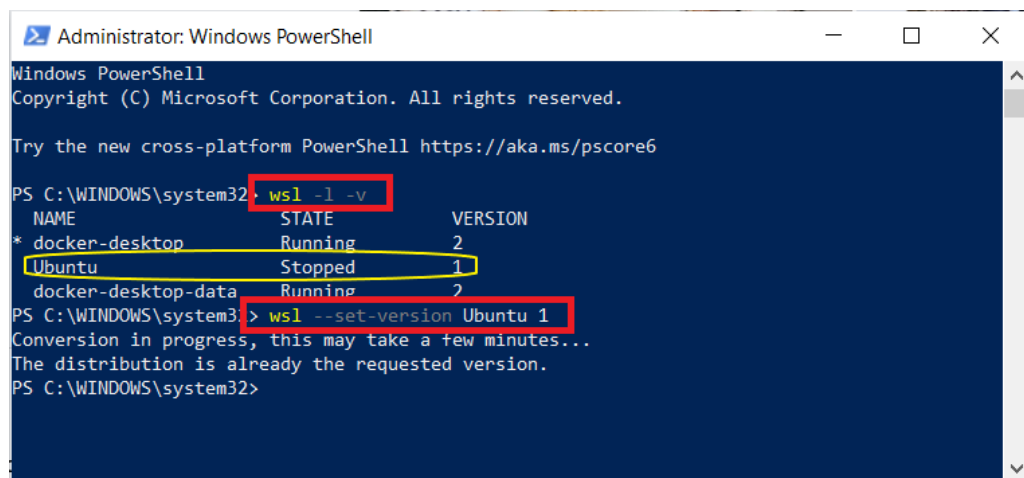
2.3 Cài đặt truy vấn phân tán cho Redis

Nhóm sử dụng hệ điều hành *Windows* để thực hiện cài đặt các bước truy vấn phân tán cũng như thêm, xóa, sửa, truy vấn cơ sở dữ liệu phân tán.

Gọi máy chứa cơ sở dữ liệu là **Máy 1**, máy truy vấn đến máy 1 là **Máy 2**.

1. Dùng lệnh `wsl -l -v` hoặc `wsl --list --verbose` để kiểm tra phiên bản Ubuntu đã cài (hình 2.10). Để thực hiện các câu lệnh phía sau, máy tính cần cài đặt phiên bản Ubuntu 1.
2. Nếu quan sát thấy trong danh sách, xuất hiện Ubuntu phiên bản 2 trở lên thì dùng câu lệnh sau để ràng buộc *Ubuntu* về phiên bản 1 (hình 2.10).

```
wsl --set-version <Distribution> 1
```



```
Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\WINDOWS\system32> wsl -l -v
NAME                STATE              VERSION
* docker-desktop    Running            2
Ubuntu               Stopped            1
docker-desktop-data Running            2
PS C:\WINDOWS\system32> wsl --set-version Ubuntu 1
Conversion in progress, this may take a few minutes...
The distribution is already the requested version.
PS C:\WINDOWS\system32>
```

HÌNH 2.10: Kiểm tra và ràng buộc phiên bản 1 cho Ubuntu.

3. Tại máy 1, lấy địa chỉ IP công khai (Public IP address) của Redis Server.

- Để truy vấn đến một cơ sở dữ liệu đặt tại một máy ở xa (remote query), hai máy cần liên lạc lẫn nhau thông qua địa chỉ IP công khai.
- Hiện tại bất cứ một thiết bị có kết nối Internet nào đều có địa chỉ IP thuộc 1 trong 2 loại: IP công khai hoặc IP riêng tư. Trong đó, IP công khai là loại IP dành cho các thiết bị liên lạc với phần còn lại của thế giới Internet, trong khi IP riêng tư là địa chỉ IP được gán cho thiết bị trong phạm vi một cục router.
- Các nhóm địa chỉ IP sau thuộc nhóm IP riêng tư:
 - Từ 10.0.0.0 đến 10.255.255.255.
 - Từ 172.16.0.0 đến 172.31.255.255.
 - Từ 192.168.0.0 đến 192.168.255.255.
- Các địa chỉ IP không thuộc các nhóm trên thuộc về nhóm IP công khai.
- Để lấy địa chỉ IP công khai của Redis server, mở cửa sổ **Ubuntu**, nhập lệnh `ifconfig`. Địa chỉ IP công khai sẽ nằm ở mục **inet** (hình 2.11).

```
vyhao@LAPTOP-R8E86C07: ~
vyhao@LAPTOP-R8E86C07:~$ ifconfig
eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.56.1 netmask 255.255.255.0 broadcast 192.168.56.255
    ether 0a:00:27:00:00:1a (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

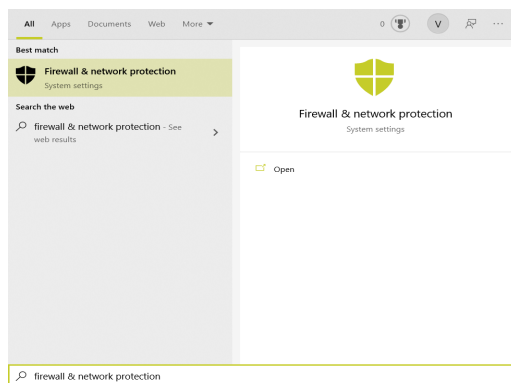
eth3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 26.167.167.11 netmask 255.0.0.0 broadcast 26.255.255.255
    ether 02:50:33:44:17:d8 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

eth4: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.20.240.1 netmask 255.255.240.0 broadcast 172.20.255.255
    ether 00:15:5d:e1:50:ff (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

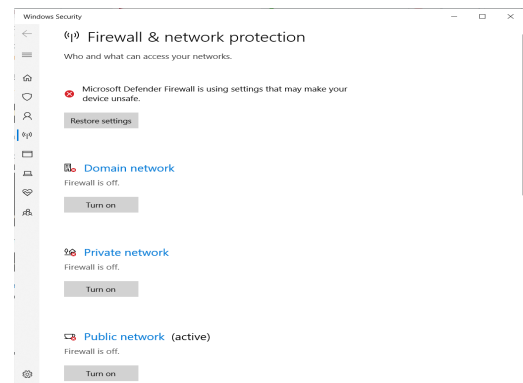
eth7: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 65535
    inet 10.157.0.6 netmask 255.255.255.255 broadcast 10.157.0.6
```

HÌNH 2.11: Lấy ra địa chỉ IP công khai của Redis.

4. Tại cả hai máy, mở cài đặt **Firewall & Network protection** của Windows (hình 2.12a) và tắt hệ thống tường lửa, kết quả như hình 2.12b.



(A) Mở cài đặt tường lửa của Windows.



(B) Tắt tường lửa.

HÌNH 2.12: Tắt tường lửa.

5. Ở Máy 1, tại cửa sổ **Ubuntu** (mở bằng cách như hình 2.5a), mở file config của Redis bằng câu lệnh:

```
sudo nano /etc/redis/redis.conf
```

- Sau đó, tìm và chỉnh sửa dòng **bind 127.0.0.1 ::1** thành **bind 127.0.0.1 <IP công khai>** (như hình 2.13a).
- Chỉnh sửa dòng **protected-mode yes** thành **protected-mode no** (hình 2.13b).

```

GNU nano 4.8 /etc/redis/redis.conf
# Following bind directive, that will force Redis to listen only into
# the IPv4 loopback interface address (this means Redis will be able to
# accept connections only from clients running into the same computer it
# is running).
# IF YOU ARE SURE YOU WANT YOUR INSTANCE TO LISTEN TO ALL THE INTERFACES
# JUST COMMENT THE FOLLOWING LINE.
# bind 127.0.0.1 26.167.167.11
#
# Protected mode is a layer of security protection, in order to avoid that
# Redis instances left open on the internet are accessed and exploited.
#
# When protected mode is on and if:
#
# 1) The server is not binding explicitly to a set of addresses using the
#    "bind" directive.
# 2) No password is configured.
#
# The server only accepts connections from clients connecting from the
# IPv4 and IPv6 loopback addresses 127.0.0.1 and ::1, and from Unix domain
# sockets.
#
# By default protected mode is enabled. You should disable it only if
# you are sure you want clients from other hosts to connect to Redis
# even if no authentication is configured, nor a specific set of interfaces
# are explicitly listed using the "bind" directive.
protected-mode no
#
# Accept connections on the specified port, default is 6379 (IANA #815344).
# If port 0 is specified Redis will not listen on a TCP socket.
port 6379
#
# TCP listen() backlog.
#
# In high requests-per-second environments you need an high backlog in order
# to avoid slow clients connections issues. Note that the Linux kernel
# will silently truncate it to the value of /proc/sys/net/core/somaxconn so
# make sure to raise both the value of somaxconn and tcp_max_syn_backlog
# in order to get the desired effect.
tcp-backlog 511
#
# Unix socket.
#
# Specify the path for the Unix socket that will be used to listen for
#
# Get Help Write Out Where Is Cut Text Justify Cur Pos
# Exit Read File Replace Paste Text To Spell Go To Line

```

(A) Ràng buộc IP công khai.

```

GNU nano 4.8 /etc/redis/redis.conf
# The server only accepts connections from clients connecting from the
# IPv4 and IPv6 loopback addresses 127.0.0.1 and ::1, and from Unix domain
# sockets.
#
# By default protected mode is enabled. You should disable it only if
# you are sure you want clients from other hosts to connect to Redis
# even if no authentication is configured, nor a specific set of interfaces
# are explicitly listed using the "bind" directive.
protected-mode no
#
# Accept connections on the specified port, default is 6379 (IANA #815344).
# If port 0 is specified Redis will not listen on a TCP socket.
port 6379
#
# TCP listen() backlog.
#
# In high requests-per-second environments you need an high backlog in order
# to avoid slow clients connections issues. Note that the Linux kernel
# will silently truncate it to the value of /proc/sys/net/core/somaxconn so
# make sure to raise both the value of somaxconn and tcp_max_syn_backlog
# in order to get the desired effect.
tcp-backlog 511
#
# Unix socket.
#
# Specify the path for the Unix socket that will be used to listen for
#
# Get Help Write Out Where Is Cut Text Justify Cur Pos
# Exit Read File Replace Paste Text To Spell Go To Line

```

(B) Tắt chế độ bảo vệ của Redis server.

HÌNH 2.13: Ràng buộc địa chỉ IP, cho phép truy vấn từ xa.

6. Tại Máy 2, thực hiện câu lệnh sau để kết nối đến máy 1:

```
redis-cli -h <IP công khai của máy 1> -p <port Redis>
```

Trong đó, port Redis mặc định thường là **6379**. Nếu truy vấn thành công sẽ có kết quả như hình 2.14.

```

hoaiobao1407@Admin: ~
hoaiobao1407@Admin:~$ redis-cli -h 26.167.167.11 -p 6379
26.167.167.11:6379> _

```

HÌNH 2.14: Máy 2 truy vấn đến Redis ở máy 1.

7. Cài đặt truy vấn phân tán thành công.

Chương 3

Truy vấn dữ liệu trên môi trường phân tán

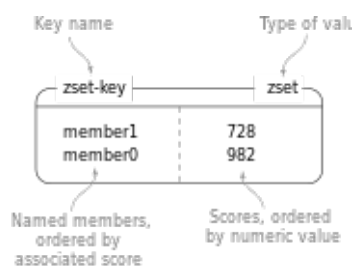
3.1 Mô tả bài toán

Giáo viên chủ nhiệm cần 1 cơ sở dữ liệu đơn giản chỉ lưu trữ tên học sinh và điểm thi của 1 lớp. Một số yêu cầu của giáo viên:

- Dễ dàng truy cập từ xa
- Tốc độ truy xuất nhanh chóng

3.2 Cấu trúc cơ sở dữ liệu sử dụng

Sử dụng sorted set (zset) trong Redis



HÌNH 3.1: Sorted set (zset) trong redis

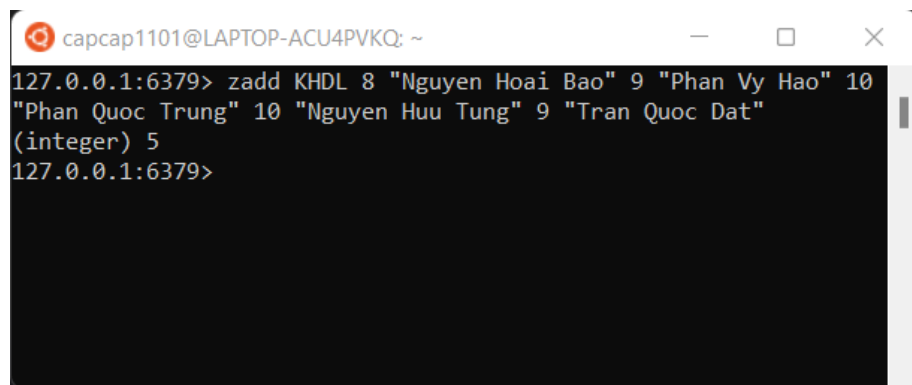
Trong đó:

- zset-key: Tên lớp học. Ví dụ: 10A8, 10A11...
- member0, member1: Tên các học sinh. Ví dụ: Nguyen Hoai Bao, Phan Vy Hao...
- 728, 928: Score của member, là 1 số nguyên mô tả điểm thi của học sinh trên thang điểm 100. Ví dụ: 0, 20, 30...

3.3 Các bước thực hiện

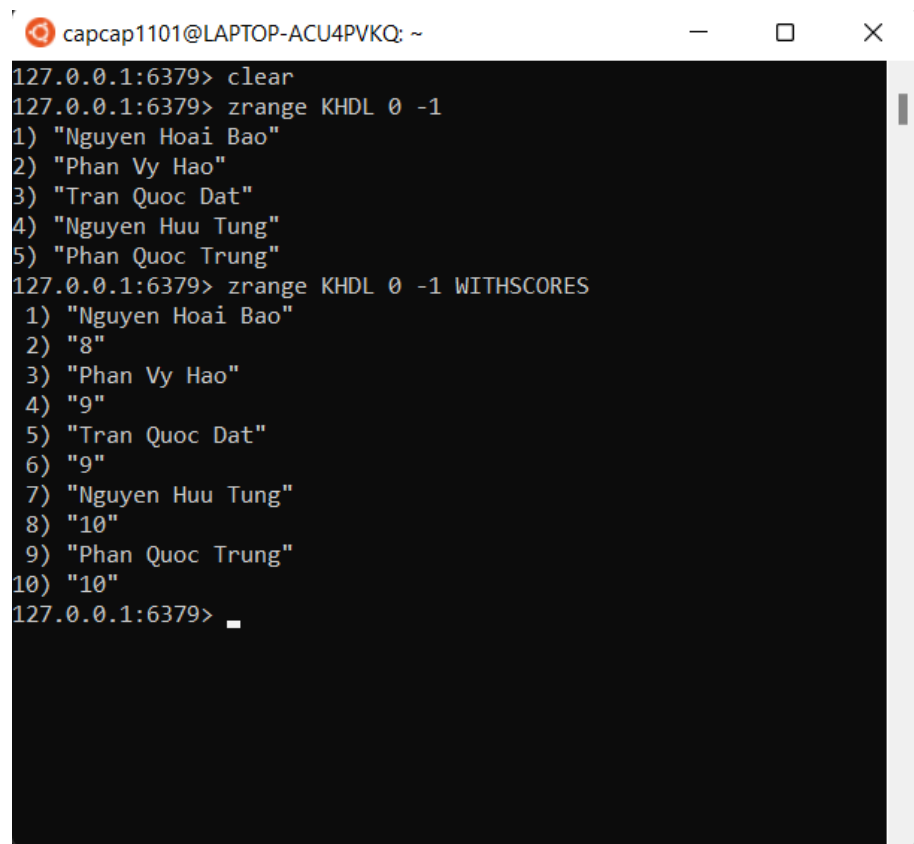
3.3.1 Insert dữ liệu

Máy tính bạn Đăng (capcap1101) đóng vai trò **Máy 1**, lưu dữ liệu **KHDL**.



```
capcap1101@LAPTOP-ACU4PVKQ: ~  
127.0.0.1:6379> zadd KHDL 8 "Nguyen Hoai Bao" 9 "Phan Vy Hao" 10  
"Phan Quoc Trung" 10 "Nguyen Huu Tung" 9 "Tran Quoc Dat"  
(integer) 5  
127.0.0.1:6379>
```

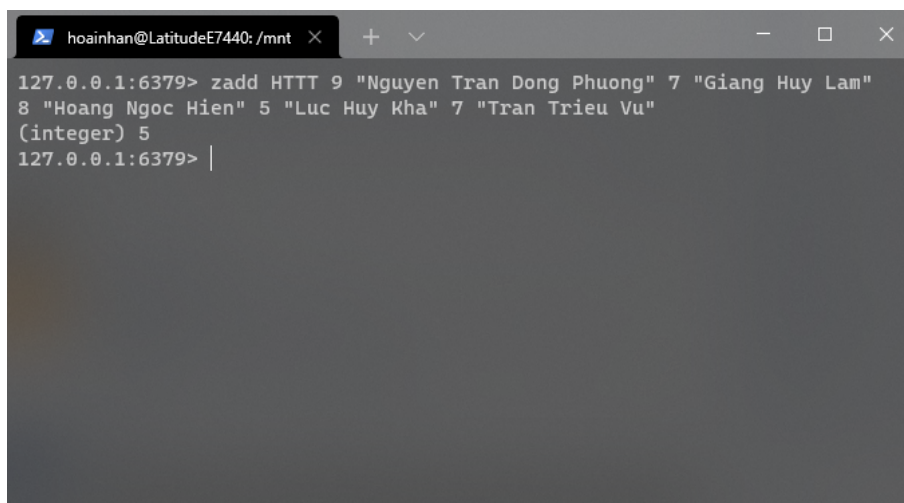
HÌNH 3.2: Máy 1 tạo dữ liệu KHDL.



```
capcap1101@LAPTOP-ACU4PVKQ: ~  
127.0.0.1:6379> clear  
127.0.0.1:6379> zrange KHD 0 -1  
1) "Nguyen Hoai Bao"  
2) "Phan Vy Hao"  
3) "Tran Quoc Dat"  
4) "Nguyen Huu Tung"  
5) "Phan Quoc Trung"  
127.0.0.1:6379> zrange KHD 0 -1 WITHSCORES  
1) "Nguyen Hoai Bao"  
2) "8"  
3) "Phan Vy Hao"  
4) "9"  
5) "Tran Quoc Dat"  
6) "9"  
7) "Nguyen Huu Tung"  
8) "10"  
9) "Phan Quoc Trung"  
10) "10"  
127.0.0.1:6379> _
```

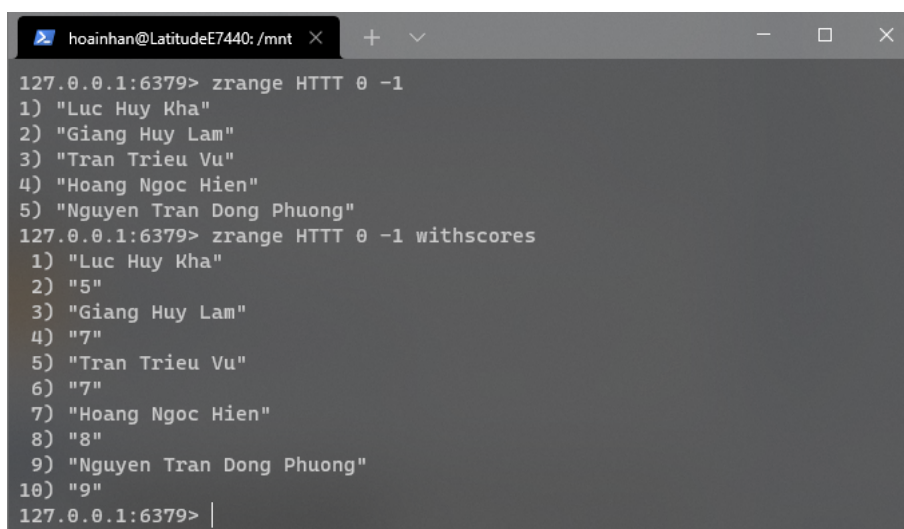
HÌNH 3.3: Máy 1 truy vấn dữ liệu tập trung.

Máy tính bạn Nhân (hoainhan) đóng vai trò **Máy 2**, lưu dữ liệu **HTTT** và sẽ truy vấn sang máy 1.



```
hoainhan@LatitudeE7440: /mnt
127.0.0.1:6379> zadd HTTT 9 "Nguyen Tran Dong Phuong" 7 "Giang Huy Lam"
8 "Hoang Ngoc Hien" 5 "Luc Huy Kha" 7 "Tran Trieu Vu"
(integer) 5
127.0.0.1:6379> |
```

HÌNH 3.4: Máy 2 tạo dữ liệu HTTT.



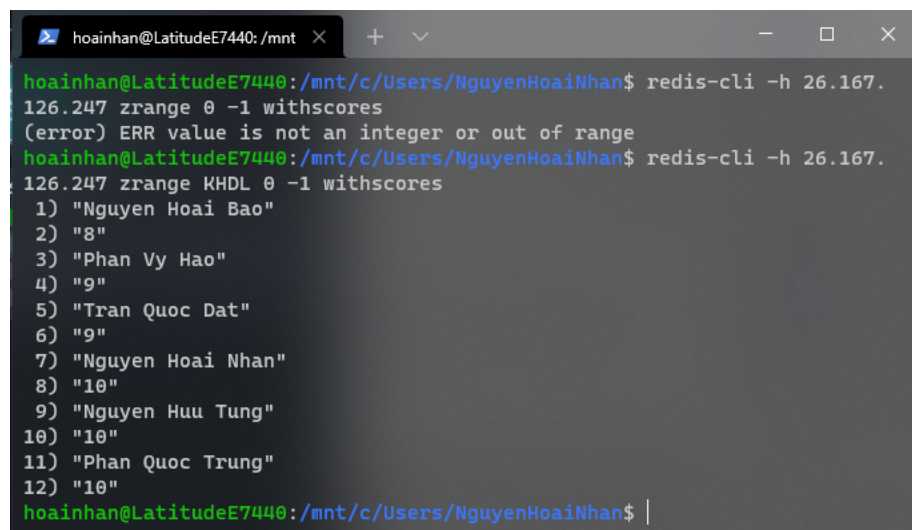
```
hoainhan@LatitudeE7440: /mnt
127.0.0.1:6379> zrange HTTT 0 -1
1) "Luc Huy Kha"
2) "Giang Huy Lam"
3) "Tran Trieu Vu"
4) "Hoang Ngoc Hien"
5) "Nguyen Tran Dong Phuong"
127.0.0.1:6379> zrange HTTT 0 -1 withscores
1) "Luc Huy Kha"
2) "5"
3) "Giang Huy Lam"
4) "7"
5) "Tran Trieu Vu"
6) "7"
7) "Hoang Ngoc Hien"
8) "8"
9) "Nguyen Tran Dong Phuong"
10) "9"
127.0.0.1:6379> |
```

HÌNH 3.5: Máy 2 truy vấn dữ liệu tập trung.

3.3.2 Truy vấn trên môi trường phân tán

3.3.2.1 Tương tác dữ liệu qua lại giữa 2 máy

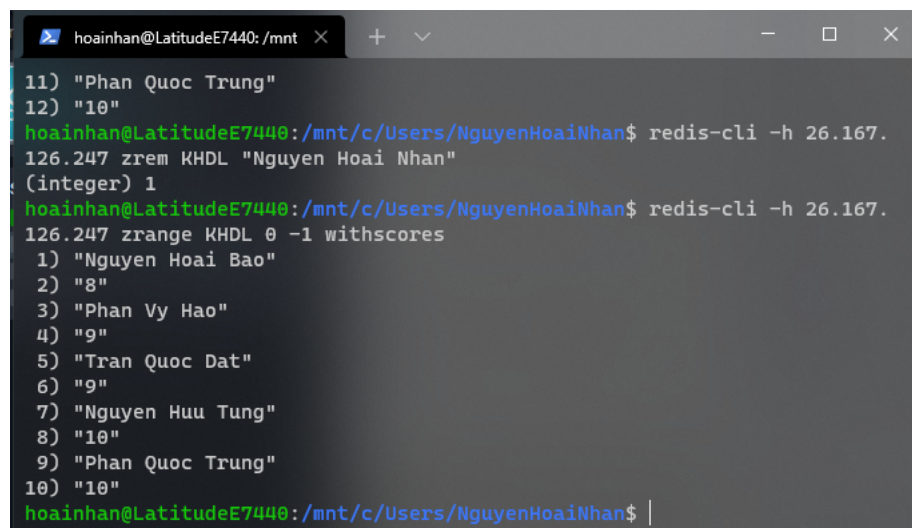
1. Máy 2 truy vấn remote dữ liệu **KHDL** đặt tại máy 1, như hình 3.6.



```
hoainhan@LatitudeE7440: /mnt
hoainhan@LatitudeE7440:/mnt/c/Users/NguyenHoaiNhan$ redis-cli -h 26.167.126.247 zrange 0 -1 withscores
(error) ERR value is not an integer or out of range
hoainhan@LatitudeE7440:/mnt/c/Users/NguyenHoaiNhan$ redis-cli -h 26.167.126.247 zrange KHD 0 -1 withscores
1) "Nguyen Hoai Bao"
2) "8"
3) "Phan Vy Hao"
4) "9"
5) "Tran Quoc Dat"
6) "9"
7) "Nguyen Hoai Nhan"
8) "10"
9) "Nguyen Huu Tung"
10) "10"
11) "Phan Quoc Trung"
12) "10"
hoainhan@LatitudeE7440:/mnt/c/Users/NguyenHoaiNhan$ |
```

HÌNH 3.6: Máy 2 truy vấn dữ liệu phân tán sang máy 1.

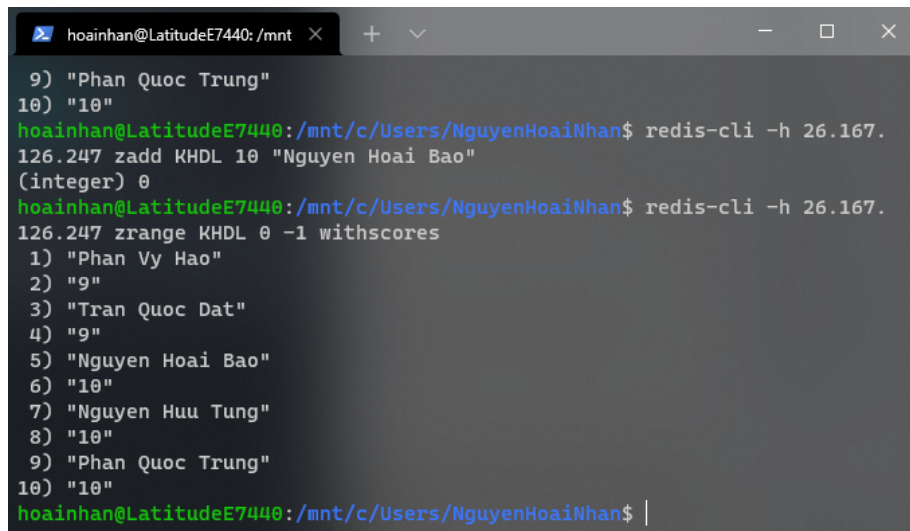
- Máy 2 thực hiện xóa một dữ liệu trong sorted set **KHD** của máy 1, xóa dữ liệu "Nguyen Hoai Nhan" và score 10 bằng lệnh **zrem**, như hình 3.8.



```
hoainhan@LatitudeE7440: /mnt
hoainhan@LatitudeE7440:/mnt/c/Users/NguyenHoaiNhan$ redis-cli -h 26.167.126.247 zrem KHD "Nguyen Hoai Nhan"
(integer) 1
hoainhan@LatitudeE7440:/mnt/c/Users/NguyenHoaiNhan$ redis-cli -h 26.167.126.247 zrange KHD 0 -1 withscores
1) "Nguyen Hoai Bao"
2) "8"
3) "Phan Vy Hao"
4) "9"
5) "Tran Quoc Dat"
6) "9"
7) "Nguyen Huu Tung"
8) "10"
9) "Phan Quoc Trung"
10) "10"
hoainhan@LatitudeE7440:/mnt/c/Users/NguyenHoaiNhan$ |
```

HÌNH 3.7: Máy 2 xóa một dữ liệu phân tán của máy 1.

- Máy 2 thực hiện thêm vào một dữ liệu với member là "Nguyen Hoai Bao" và score 10 bằng lệnh **zadd**, như hình 3.8.

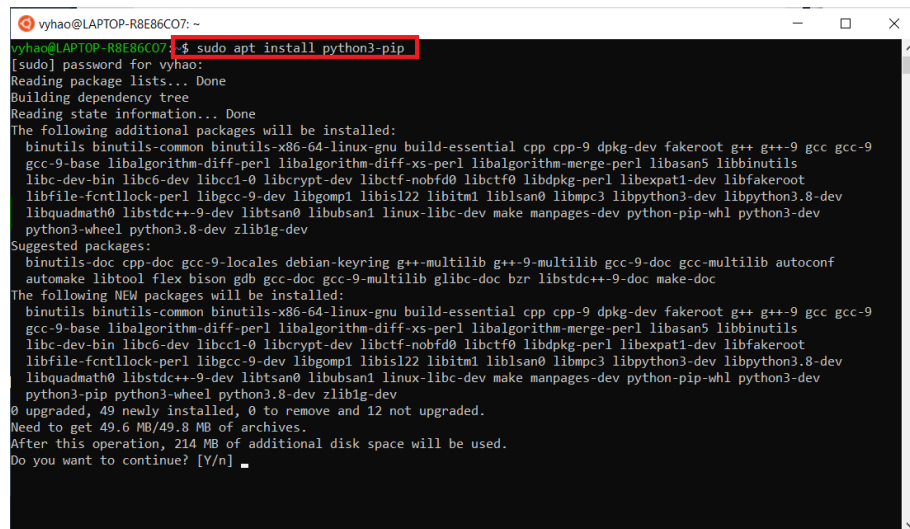


```
hoainhan@LatitudeE7440: /mnt
9) "Phan Quoc Trung"
10) "10"
hoainhan@LatitudeE7440: /mnt/c/Users/NguyenHoaiNhan$ redis-cli -h 26.167.
126.247 zadd KHD 10 "Nguyen Hoai Bao"
(integer) 0
hoainhan@LatitudeE7440: /mnt/c/Users/NguyenHoaiNhan$ redis-cli -h 26.167.
126.247 zrange KHD 0 -1 withscores
1) "Phan Vy Hao"
2) "9"
3) "Tran Quoc Dat"
4) "9"
5) "Nguyen Hoai Bao"
6) "10"
7) "Nguyen Huu Tung"
8) "10"
9) "Phan Quoc Trung"
10) "10"
hoainhan@LatitudeE7440: /mnt/c/Users/NguyenHoaiNhan$ |
```

HÌNH 3.8: Máy 2 thêm một dữ liệu phân tán vào KHD của máy 1.

3.3.2.2 Truy vấn phân tán

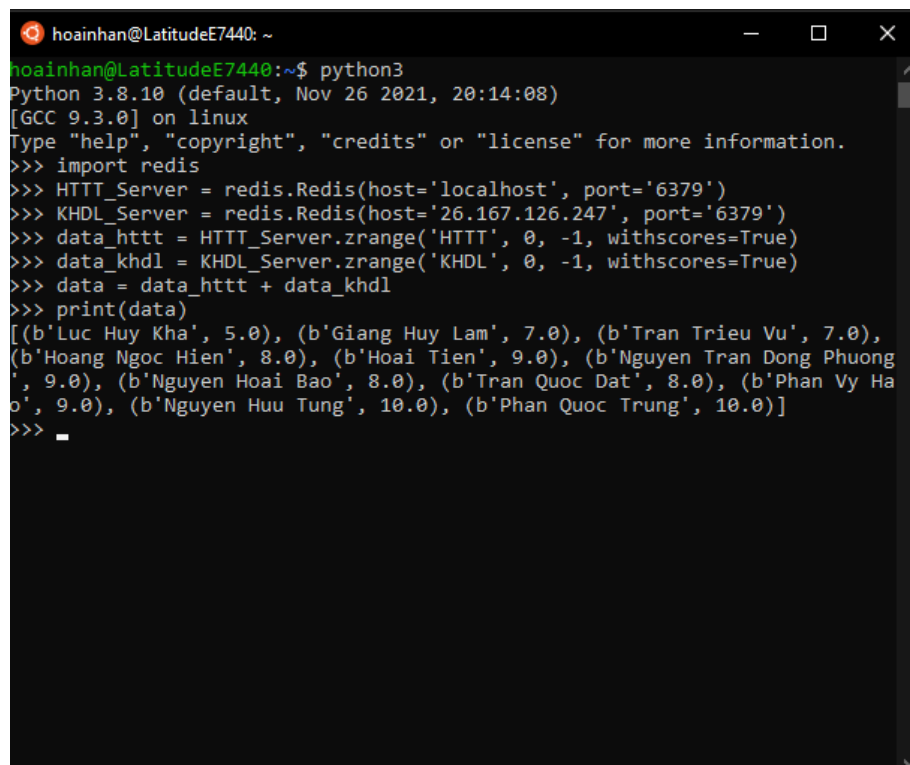
- **Mô tả yêu cầu:** Tại máy 2, thực hiện gộp rồi in ra toàn bộ học sinh và điểm của họ ở cả hai máy.
- Redis là một hệ cơ sở dữ liệu đơn giản, vốn chỉ hỗ trợ các lệnh Get, set và kiểu dữ liệu dạng **key : value**. Vì thế, để truy vấn phân tán đúng nghĩa và hiệu quả, nhóm thực hiện cài đặt thêm một ngôn ngữ lập trình trên *Ubuntu*.
- Vì thế, nhóm sinh viên thực hiện cài thêm package Python cho Ubuntu để thực hiện tác vụ phân tán này, như hình 3.9.



```
vyhao@LAPTOP-R8E86C07: ~
vyhao@LAPTOP-R8E86C07:~$ sudo apt install python3-pip
[sudo] password for vyhao:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  binutils binutils-common binutils-x86-64-linux-gnu build-essential cpp cpp-9 dpkg-dev fakeroot g++ g++-9 gcc gcc-9
  gcc-9-base libalgorithm-diff-perl libalgorithm-diff-xs-perl libalgorithm-merge-perl libasan5 libbinutils
  libc-dev-bin libc6-dev libcc1-0 libcrypt-dev libctf-nobfd0 libctf0 libdpkg-perl libexpat1-dev libfakeroot
  libfile-fcntllock-perl libgcc-9-dev libgomp1 libisl22 libitm1 liblsan0 libmpc3 libpython3-dev libpython3.8-dev
  libquadmath0 libstdc++-9-dev libstdc++9 libubsan1 linux-libc-dev make manpages-dev python-pip-whl python3-dev
  python3-wheel python3.8-dev zlib1g-dev
Suggested packages:
  binutils-doc cpp-doc gcc-9-locales debian-keyring g++-multilib g++-9-multilib gcc-9-doc gcc-multilib autoconf
  automake libtool flex bison gdb gcc-doc gcc-9-multilib glibc-doc bzip libstdc++-9-doc make-doc
The following NEW packages will be installed:
  binutils binutils-common binutils-x86-64-linux-gnu build-essential cpp cpp-9 dpkg-dev fakeroot g++ g++-9 gcc gcc-9
  gcc-9-base libalgorithm-diff-perl libalgorithm-diff-xs-perl libalgorithm-merge-perl libasan5 libbinutils
  libc-dev-bin libc6-dev libcc1-0 libcrypt-dev libctf-nobfd0 libctf0 libdpkg-perl libexpat1-dev libfakeroot
  libfile-fcntllock-perl libgcc-9-dev libgomp1 libisl22 libitm1 liblsan0 libmpc3 libpython3-dev libpython3.8-dev
  libquadmath0 libstdc++-9-dev libstdc++9 libubsan1 linux-libc-dev make manpages-dev python-pip-whl python3-dev
  python3-wheel python3.8-dev zlib1g-dev
0 upgraded, 49 newly installed, 0 to remove and 12 not upgraded.
Need to get 49.6 MB/49.8 MB of archives.
After this operation, 214 MB of additional disk space will be used.
Do you want to continue? [Y/n]
```

HÌNH 3.9: Cài đặt package Python trên Ubuntu.

- Kết quả của tác vụ thể hiện ở hình 3.10.



```
hoainhan@LatitudeE7440: ~
hoainhan@LatitudeE7440:~$ python3
Python 3.8.10 (default, Nov 26 2021, 20:14:08)
[GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import redis
>>> HTTP_Server = redis.Redis(host='localhost', port='6379')
>>> KHDH_Server = redis.Redis(host='26.167.126.247', port='6379')
>>> data_http = HTTP_Server.zrange('HTTP', 0, -1, withscores=True)
>>> data_khdh = KHDH_Server.zrange('KHDH', 0, -1, withscores=True)
>>> data = data_http + data_khdh
>>> print(data)
[(b'Luc Huy Kha', 5.0), (b'Giang Huy Lam', 7.0), (b'Tran Trieu Vu', 7.0),
(b'Hoang Ngoc Hien', 8.0), (b'Hoai Tien', 9.0), (b'Nguyen Tran Dong Phuong',
9.0), (b'Nguyen Hoai Bao', 8.0), (b'Tran Quoc Dat', 8.0), (b'Phan Vy Ha',
9.0), (b'Nguyen Huu Tung', 10.0), (b'Phan Quoc Trung', 10.0)]
>>>
```

HÌNH 3.10: Máy 2 hợp (union) dữ liệu ở cả hai máy.

- Nội dung đoạn code mà nhóm đã sử dụng được trình bày ở phía dưới:

```
1 import redis
2
3 # may 2
4 HTTP_Server = redis.Redis(host='localhost', port='6379')
5 # may 1
6 KHDH_Server = redis.Redis(host='26.167.126.247', port='6379')
7
8 # lay du lieu HTTP
9 data_http = HTTP_Server.zrange('HTTP', 0, -1, withscores=True)
10 data_khdh = KHDH_Server.zrange('KHDH', 0, -1, withscores=True)
11
12 # Union
13 data = data_http + data_khdh
14
15 print(data)
```

LISTING 3.1: Code câu lệnh truy vấn phân tán