

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



BÁO CÁO ĐỒ ÁN  
CƠ SỞ DỮ LIỆU PHÂN TÁN

ĐỀ TÀI  
KEYDB

**Nhóm sinh viên thực hiện:** Nhóm 14

- |                         |                |
|-------------------------|----------------|
| 1. Cao Hoài Sang        | MSSV: 21522541 |
| 2. Thi Thành Công       | MSSV: 21521897 |
| 3. Nguyễn Trần Gia Kiệt | MSSV: 21522258 |
| 4. Trần Thành Tín       | MSSV: 21522684 |

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



BÁO CÁO ĐỒ ÁN  
CƠ SỞ DỮ LIỆU PHÂN TÁN

ĐỀ TÀI  
KEYDB

**Giảng viên hướng dẫn:** Thầy Nguyễn Hồ Duy Tri

**Nhóm sinh viên thực hiện:** Nhóm TNTA\_CTCA

- |                         |                |
|-------------------------|----------------|
| 1. Cao Hoài Sang        | MSSV: 21522541 |
| 2. Thi Thành Công       | MSSV: 21521897 |
| 3. Nguyễn Trần Gia Kiệt | MSSV: 21522258 |
| 4. Trần Thành Tín       | MSSV: 21522684 |

## LỜI CẢM ƠN

Trong cuộc sống của chúng ta, có lẽ ai cũng đã từng thất bại hoặc thành công, dù như thế nào thì đó cũng là kết quả nỗ lực của mỗi cá nhân cũng như tập thể. Và đằng sau đó chính là sự hỗ trợ giúp đỡ từ mọi người. Xét về mặt thành công, trong thực tế không có sự thành công nào mà không có sự giúp đỡ, nhất là trong học tập. Dân gian ta có câu “Không thầy đố mày làm nên” quả thật là đúng, học sinh không thể thành công nếu không có sự giúp đỡ, truyền đạt cũng như chỉ bảo tận tình của người Thầy. Hôm nay, để có thể hoàn thành được đề án môn học này, nhóm chúng em rất biết ơn những thầy cô đã hỗ trợ tận tình, đã cung cấp cho chúng em nhiều kiến thức cũng như kinh nghiệm của người đi trước.

Lời đầu tiên chúng em xin cảm ơn tập thể cán bộ, giảng viên trường Đại học Công Nghệ Thông Tin – ĐHQG Tp. Hồ Chí Minh, nơi mà chúng em được tự do nghiên cứu cũng như có thêm nhiều kiến thức. Với lòng biết ơn sâu sắc nhất, nhóm chúng em xin gửi đến quý Thầy Cô ở Khoa Hệ thống thông tin – Trường Đại Học Công Nghệ Thông Tin đã cùng với tri thức và tâm huyết của mình để truyền đạt vốn kiến thức quý báu cho chúng em trong suốt thời gian học tập tại trường. Trong đó có *thầy Nguyễn Hồ Duy Tri*, thầy đã cung cấp cho chúng em các kiến thức bổ ích về quá trình làm phần mềm và lời góp ý cho đề án này. Ngoài ra, chúng em cũng gửi lời cảm ơn đến tập thể lớp *IS221.O12* vì thời gian qua đã đồng hành cùng nhau. Cùng nhau học tập, tranh luận một cách sôi nổi để xây dựng bài học một cách tốt nhất. Cảm ơn các bạn đã cùng thảo luận, đánh giá và đóng góp ý kiến, cùng học hỏi nghiên cứu để thực hiện đề án một cách tốt nhất có thể.

Cuối cùng, để đề án được hoàn thành thì không thể nào cảm ơn những người đã làm ra đó, cảm ơn các bạn các thành viên trong nhóm đã chăm chỉ và chịu khó hoàn thành nhiệm vụ đúng tiến độ.

Với những gì đã được giúp đỡ, cuối cùng nhóm đã hoàn thành đề án có tên: “KeyDB”. Trải qua thời gian một học kỳ thực hiện đề tài. Với sự hướng dẫn tận tình cùng những đóng góp quý báu của Thầy và các bạn giúp nhóm chúng em hoàn thành

tốt hơn báo cáo môn học của mình. Bên cạnh việc vận dụng những kiến thức được học trên lớp đồng thời kết hợp với việc học hỏi và tìm hiểu những kiến thức mới. Từ đó, nhóm đã vận dụng tối đa những gì đã tiếp thu được để hoàn thành một báo cáo đồ án tốt nhất. Tuy nhiên, trong quá trình thực hiện, không tránh khỏi những sai sót. Do đó, rất mong nhận được những sự góp ý từ phía Thầy nhằm giúp nhóm hoàn thiện những kiến thức đã học tập và cũng là hành trang để qua đó chúng em tích lũy và học hỏi kinh nghiệm để làm hành trang cho tương lai. Chúng em xin hết và xin một lần nữa gửi lời cảm ơn chân thành đến với thầy!

**Nhóm em xin chân thành cảm ơn!**

## MỤC LỤC

<b>CHƯƠNG I: GIỚI THIỆU .....</b>	<b>9</b>
1. Hệ quản trị CSDL KeyDB.....	9
2. Lý thuyết về NoSQL trong KeyDB.....	10
3. Cơ chế phân tán.....	14
<b>CHƯƠNG II: HƯỚNG DẪN CÀI ĐẶT .....</b>	<b>16</b>
1. Yêu cầu.....	16
2. Cài đặt máy ảo.....	16
3. Cài đặt trên một máy .....	26
4. Cài đặt trên cụm máy phân tán .....	28
<b>CHƯƠNG III: THỰC NGHIỆM MÔ PHỎNG PHÂN TÁN .....</b>	<b>35</b>
1. Mô tả bài toán đặt ra với dữ liệu .....	35
2. Mô tả cấu trúc dữ liệu sử dụng.....	36
3. Các bước thực nghiệm chi tiết .....	41
3.1 Tạo dữ liệu:.....	41
3.2 Cập nhật dữ liệu: .....	51
3.3 Xoá dữ liệu: .....	53
3.4 Một số thao tác khác trên các dữ liệu trong keydb:.....	55
<b>NGUỒN THAM KHẢO .....</b>	<b>60</b>

## DANH SÁCH BẢNG

<i>Table 1 bảng store.....</i>	<i>38</i>
<i>Table 2 bảng category.....</i>	<i>38</i>
<i>Table 3 bảng supplier.....</i>	<i>39</i>

Table 4 bảng products.....	39
Table 5 bảng user.....	40
Table 6 bảng order.....	41
Table 7 bảng detail_order.....	41

## DANH SÁCH HÌNH

Figure 1 Tạo và xóa key.....	12
Figure 2 Truy vấn key .....	12
Figure 3 kiểm tra key có tồn tại không .....	12
Figure 4 Lấy giá trị của một key.....	13
Figure 5 thêm giá trị vào cuối.....	13
Figure 6 thêm bớt giá trị.....	14
Figure 7 vòng tròn 2 chiều gồm 3 nút trong keydb.....	16
Figure 8 tạo máy ảo mới.....	17
Figure 9 chọn kiểu configuration.....	17
Figure 10 CHỌN INSTALL TỪ ĐẦU .....	18
Figure 11 chọn hệ điều hành muốn cài đặt .....	19
Figure 12 đặt tên cho máy ảo .....	20
Figure 13 chỉ định dung lượng ổ đĩa .....	21
Figure 14 thông tin xem lại trước khi nhấn finish .....	22
Figure 15 chọn setting của máy ảnh vừa cài .....	23
Figure 16 chọn nơi chứa file ubuntu.....	24
Figure 17 chọn try or install ubuntu.....	25
Figure 18 chọn vào install ubuntu để cài đặt ubuntu .....	25
Figure 19 bước 1 cài đặt trên 1 máy.....	26
Figure 20 bước 2 cài đặt trên 1 máy.....	26
Figure 21 bước 3 cài đặt trên 1 máy.....	27
Figure 22 bước 2 cài đặt trên 1 máy (1).....	27

<i>Figure 23 bước 4 cài đặt trên 1 máy (2).....</i>	<i>28</i>
<i>Figure 24 thử chạy các lệnh sau khi cài đặt keydb.....</i>	<i>28</i>
<i>Figure 25 tạo thư mục keydb .....</i>	<i>29</i>
<i>Figure 26 tạo thư mục con 6380.....</i>	<i>30</i>
<i>Figure 27 chạy terminal tạo file text.....</i>	<i>31</i>
<i>Figure 28 vào file keydb.config với terminal và copy các lệnh vào .....</i>	<i>32</i>
<i>Figure 29 nhập lệnh .....</i>	<i>33</i>
<i>Figure 30 connect vào port 6380.....</i>	<i>34</i>
<i>Figure 31 thử thêm và lấy dữ liệu từ 2 bên.....</i>	<i>34</i>
<i>Figure 32 thêm dữ liệu store ở máy a.....</i>	<i>42</i>
<i>Figure 33 lấy dữ liệu ở máy b.....</i>	<i>43</i>
<i>Figure 34 thêm dữ liệu category ở máy a.....</i>	<i>43</i>
<i>Figure 35 thêm dữ liệu supplier ở máy a.....</i>	<i>44</i>
<i>Figure 36 thêm dữ liệu products ở máy a.....</i>	<i>45</i>
<i>Figure 37 thêm dữ liệu user ở máy a.....</i>	<i>46</i>
<i>Figure 38 thêm dữ liệu order ở máy a.....</i>	<i>47</i>
<i>Figure 39 thêm từng order vào list tương ứng.....</i>	<i>48</i>
<i>Figure 40 thêm dữ liệu category ở máy a.....</i>	<i>48</i>
<i>Figure 41 thêm dữ liệu detail_order.....</i>	<i>49</i>
<i>Figure 42 lấy dữ liệu products từ cả 2 máy.....</i>	<i>50</i>
<i>Figure 43 lấy dữ liệu category từ cả 2 máy.....</i>	<i>50</i>
<i>Figure 44 lấy tất cả các dữ liệu còn lại (1).....</i>	<i>51</i>
<i>Figure 45 lấy tất cả các dữ liệu còn lại (2).....</i>	<i>51</i>
<i>Figure 46 cập nhập trường quantity của product:1 .....</i>	<i>52</i>
<i>Figure 47 dữ liệu sau chỉnh sửa .....</i>	<i>52</i>
<i>Figure 48 xóa 1 dòng dữ liệu.....</i>	<i>53</i>
<i>Figure 49 xóa 1 hset.....</i>	<i>54</i>
<i>Figure 50 kết quả sau khi xóa.....</i>	<i>55</i>
<i>Figure 51 đếm số trường của hset product:1.....</i>	<i>55</i>

<i>Figure 52</i> tăng giá trị của trường <i>quantity</i> của <i>hset product:1</i> .....	56
<i>Figure 53</i> giảm giá trị của trường <i>quantity</i> của <i>hset product:1</i> .....	57
<i>Figure 54</i> dùng hàm <i>hgetall</i> để lấy thông tin 1 <i>hset</i> .....	58
<i>Figure 55</i> sử dụng <i>lpush</i> và <i>lrange</i> .....	59
<i>Figure 56</i> sử dụng <i>scan</i> để lấy thông tin.....	60



## CHƯƠNG I: GIỚI THIỆU

### 1. Hệ quản trị CSDL KeyDB

KeyDB là một hệ quản trị cơ sở dữ liệu mã nguồn mở lưu trữ dạng key-value, được phát triển dựa trên Redis. KeyDB được tạo ra bởi John Sully vào năm 2019, với mục tiêu cải thiện hiệu năng, độ tin cậy và tính năng của Redis. KeyDB có thể chạy trên nhiều luồng, hỗ trợ đồng bộ hóa nhiều master, cung cấp khả năng backup và khôi phục nhanh, và có thể tương thích với các thư viện và công cụ của Redis. KeyDB gia nhập Snap vào cuối năm 2021.

KeyDB là một phiên bản phân nhánh của Redis với hiệu suất cao, tập trung vào đa luồng, hiệu quả bộ nhớ và khả năng xử lý lớn. Ngoài cải thiện hiệu suất, KeyDB còn cung cấp các tính năng như Sao chép hoạt động, lưu trữ FLASH và hết hạn Subkey. KeyDB có kiến trúc MVCC cho phép bạn thực hiện các truy vấn như KEYS và SCAN mà không chặn cơ sở dữ liệu và làm giảm hiệu suất. KeyDB duy trì khả năng tương thích hoàn toàn với giao thức Redis, các mô-đun và các kịch bản. Điều này bao gồm cả các đảm bảo nguyên tử cho các kịch bản và giao dịch. Vì KeyDB luôn đồng bộ với sự phát triển của Redis, KeyDB là một tập hợp con của các chức năng Redis, khiến KeyDB trở thành một giải pháp thay thế cho các triển khai Redis hiện có.

Trên cùng một phần cứng, KeyDB có thể đạt được khả năng xử lý lớn hơn nhiều so với Redis. Sao chép hoạt động đơn giản hóa việc chuyển đổi dự phòng nóng, cho phép bạn phân phối các ghi trên các bản sao và sử dụng cân bằng tải / chuyển đổi dự phòng dựa trên TCP đơn giản.

Ưu điểm:

- Hiệu suất cao: KeyDB được thiết kế để xử lý các tải công việc nặng với một nút đánh giá benchmark hơn 1 triệu ops/giây<sup>1</sup>. KeyDB là một cơ sở dữ liệu đa luồng và sẽ vượt trội hơn Redis trên cơ sở nút.

- Độ trễ thấp: Bằng cách giữ dữ liệu trong bộ nhớ, KeyDB có thể phục vụ dữ liệu với độ trễ dưới mili giây.
- Nhiều kiểu dữ liệu: KeyDB hỗ trợ nhiều cấu trúc dữ liệu như strings, hashes, lists, sets, sorted sets, bitmaps, hyperloglogs, và streams.
- Tùy chọn lưu trữ linh hoạt: KeyDB cung cấp nhiều tùy chọn lưu trữ như đồ dữ liệu theo chu kỳ vào đĩa hoặc bằng cách thêm từng lệnh vào nhật ký dựa trên đĩa.
- Tính mở rộng: Trong khi một nút KeyDB mở rộng theo chiều dọc, bạn có thể mở rộng theo chiều ngang thông qua sao chép hoạt động, hoặc chế độ cụm (tập dữ liệu được chia sẻ) để đáp ứng các tải công việc lớn hơn nhiều.
- Tính sẵn có cao: KeyDB giới thiệu sao chép hoạt động, cho phép dữ liệu được đồng bộ hóa giữa nhiều phiên bản KeyDB.

Nhược điểm:

- Yêu cầu bộ nhớ cao: Do KeyDB giữ dữ liệu trong bộ nhớ, nó có thể yêu cầu một lượng bộ nhớ đáng kể, đặc biệt là với các tập dữ liệu lớn.
- Khả năng tương thích: Mặc dù KeyDB tương thích với giao thức Redis, nhưng có thể có một số khác biệt nhỏ có thể gây ra vấn đề khi chuyển từ Redis sang KeyDB.
- Hỗ trợ cộng đồng: KeyDB là một sản phẩm mới hơn so với Redis, do đó cộng đồng hỗ trợ có thể không mạnh mẽ như Redis.

## **2. Lý thuyết về NoSQL trong KeyDB**

### **a. Cấu trúc Key-Value Store**

Cấu trúc key-value store (hay còn gọi là key-value database) là một dạng cơ sở dữ liệu đơn giản sử dụng mảng kết hợp, trong đó mỗi key sẽ tồn tại duy nhất một values trong collection.

Phone directory		MAC table	
Key	Value	Key	Value
Paul	(091) 9786453778	10.94.214.172	3c:22:fb:86:c1:b1
Greg	(091) 9686154559	10.94.214.173	00:0a:95:9d:68:16
Marco	(091) 9868564334	10.94.214.174	3c:1b:fb:45:c4:b1

### b. Các kiểu dữ liệu trong KeyDB

KeyDB không chỉ đơn thuần là key-value database, mà là data structures server, nghĩa là nó hỗ trợ nhiều loại values khác nhau như:

**String:** Kiểu dữ liệu cơ bản nhất, cho phép lưu trữ các chuỗi ký tự.

**List:** Một danh sách các chuỗi, được sắp xếp theo thứ tự chèn.

**Set:** Tập hợp các chuỗi không có thứ tự.

**Sorted Set:** Tương tự như Set, nhưng mỗi thành viên đều có một giá trị điểm số liên quan, cho phép bạn sắp xếp các thành viên dựa trên điểm số.

**Hash:** Cấu trúc dữ liệu key-value, cho phép bạn lưu trữ nhiều cặp key-value trong một key duy nhất.

**Bitmaps:** Cho phép bạn thiết lập và lấy giá trị bit trong chuỗi.

**HyperLogLogs:** Cung cấp khả năng ước lượng các phần tử duy nhất trong tập dữ liệu.

**Streams:** Cung cấp khả năng lưu trữ một luồng các thông điệp.

### c. Các thao tác dữ liệu cơ bản trong KeyDB

#### 1. Tạo và xóa keys

**SET:** được sử dụng để thiết lập giá trị cho một key. Cú pháp: SET key value

**DEL:** được sử dụng để xóa một key và giá trị của nó. Cú pháp: DEL key

```
127.0.0.1:6379> set test3 "Tran Van B"
OK
127.0.0.1:6379> del test3
(integer) 1
```

**FIGURE 1 TẠO VÀ XÓA KEY**

**KEYS:** được sử dụng để tìm kiếm các key phù hợp với một mẫu nhất định. Cú pháp:  
**KEYS** pattern

```
127.0.0.1:6379> keys *
1) "sinhvien"
2) "test2"
3) "test"
127.0.0.1:6379> keys test*
1) "test2"
2) "test"
```

**FIGURE 2 TRUY VẤN KEY**

**EXISTS:** được sử dụng để kiểm tra xem 1 key có tồn tại hay không, nếu tồn tại, lệnh sẽ trả về 1, nếu không sẽ trả về 0. Cú pháp: **EXISTS** key

```
127.0.0.1:6379> exists test
(integer) 1
127.0.0.1:6379> exists nontest
(integer) 0
```

**FIGURE 3 KIỂM TRA KEY CÓ TỒN TẠI KHÔNG**

**GET:** được sử dụng để lấy giá trị lưu trữ bởi một key. Cú pháp: **GET** key

```
127.0.0.1:6379> get test  
"Nguyen Van A"
```

FIGURE 4 LẤY GIÁ TRỊ CỦA MỘT KEY

## 2. Thêm giá trị vào cuối một key

**APPEND:** được sử dụng để thêm dữ liệu vào cuối giá trị hiện tại của một key. Cú pháp: APPEND key value

```
127.0.0.1:6379> get test  
"Nguyen Van A"  
127.0.0.1:6379> append test " B"  
(integer) 14  
127.0.0.1:6379> get test  
"Nguyen Van A B"
```

FIGURE 5 THÊM GIÁ TRỊ VÀO CUỐI

## 3. Thêm, bớt giá trị số nguyên tố của một key

**INCR:** được sử dụng để tăng giá trị lưu trữ của một key (số nguyên) lên 1 đơn vị

**INCRBY:** được sử dụng để tăng giá trị lưu trữ của một key (số nguyên) lên n đơn vị

**DECR:** được sử dụng để giảm giá trị lưu trữ của một key (số nguyên) đi 1 đơn vị

**DECRBY:** được sử dụng để giảm giá trị lưu trữ của một key (số nguyên) đi n đơn vị

```
127.0.0.1:6379> set rating 10
OK
127.0.0.1:6379> incr rating
(integer) 11
127.0.0.1:6379> decr rating
(integer) 10
127.0.0.1:6379> incrby rating 5
(integer) 15
127.0.0.1:6379> decrby rating 5
(integer) 10
```

**FIGURE 6 THÊM BỚT GIÁ TRỊ**

### **3. Cơ chế phân tán**

#### **3.1 Active Replication (hoạt động nhân bản)**

KeyDB hỗ trợ chế độ "Active Replicas" (còn được biết đến là "Active Active"). Điều này đơn giản hóa đáng kể các kịch bản chuyển đổi khi replicas không cần phải được thăng cấp thành active masters. Ngoài ra, hỗ trợ "Active Replicas" còn có thể được sử dụng để phân tán tải trong các tình huống có lượng ghi cao.

Trong chế độ này, mỗi lệnh được thực thi trên tất cả các bản sao. Điều này đảm bảo rằng tất cả các bản sao luôn giữ dữ liệu nhất quán. Điều này có nghĩa là mỗi khi một lệnh được thực thi trên một bản sao, nó cũng sẽ được thực thi trên tất cả các bản sao khác. Điều này giúp đảm bảo rằng tất cả các bản sao luôn có cùng một trạng thái và không có sự không nhất quán nào xảy ra. Điều này rất quan trọng trong các hệ thống phân tán, nơi mà việc duy trì nhất quán giữa các bản sao là một thách thức lớn.

**Cách hoạt động:** Mặc định, KeyDB hoạt động giống như Redis và chỉ cho phép giao tiếp một chiều từ master đến replica. Một tùy chọn cấu hình mới có tên "active-replica" đã được thêm vào, và khi đặt thành true cũng áp dụng "replica-read-only no". Dưới chế độ này, KeyDB sẽ chấp nhận replicas ngay cả khi kết nối với master bị ngắt. Nó cũng cho phép các kết nối vòng tròn chéo, trong đó hai nút là master của nhau.

**Multi-master:** Ngoài ra, Keydb có hỗ trợ multiple masters khi nhân bản, chế độ mà Multi-Master và Active Replica được kích hoạt đồng thời là lựa chọn đầy triển vọng nhất, bởi vì nó cho phép bạn triển khai những kịch bản sao chép dữ liệu phức tạp giữa các trường hợp KeyDB. Điều quan trọng là chế độ này còn giúp bạn thực hiện các thao tác đa dạng trên bất kỳ trường hợp cụ thể nào.

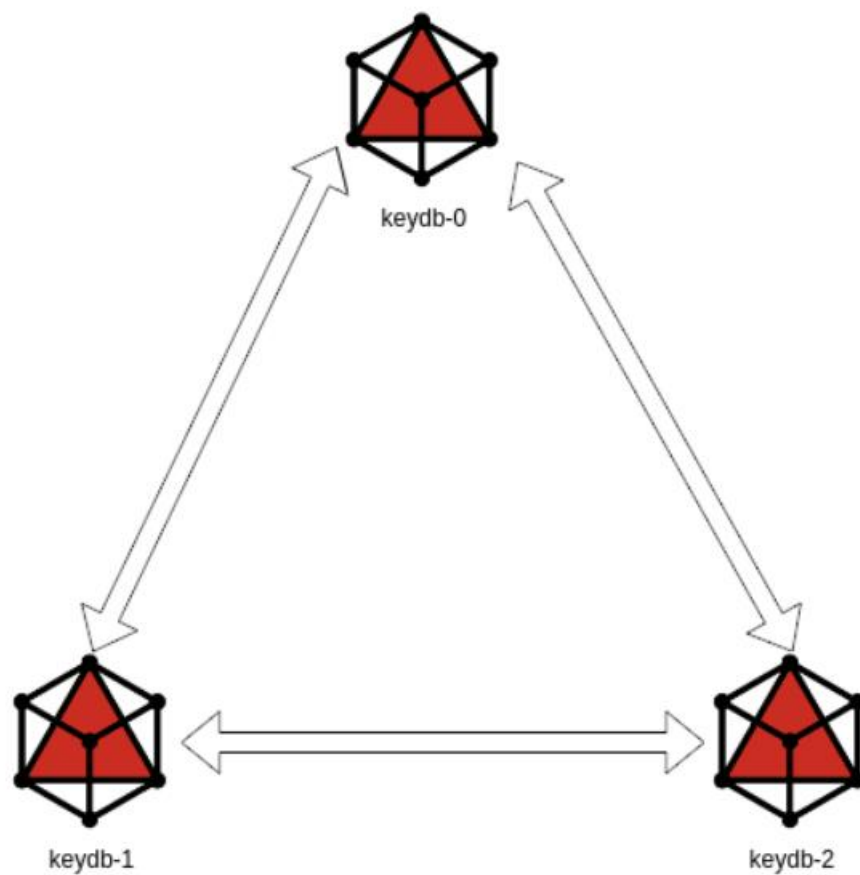


FIGURE 7 VÒNG TRÒN 2 CHIỀU GỒM 3 NÚT TRONG KEYDB

## CHƯƠNG II: HƯỚNG DẪN CÀI ĐẶT

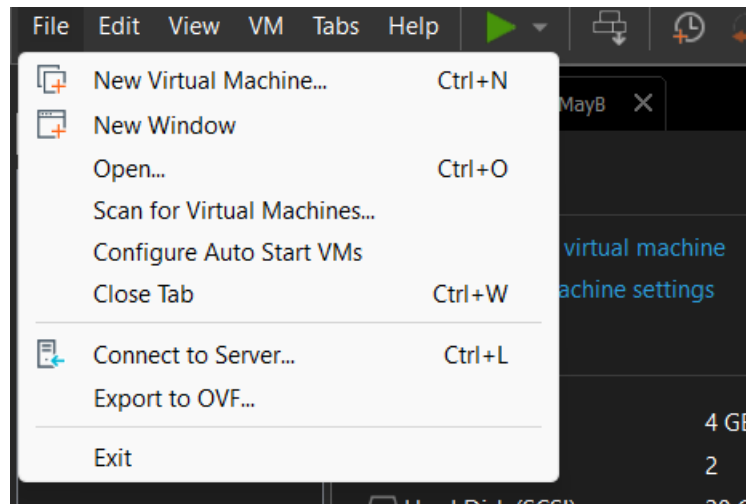
### 1. Yêu cầu

- Vmware Workstation Pro (hoặc Virtual Box)
- Ubuntu 22.04.3 LTS

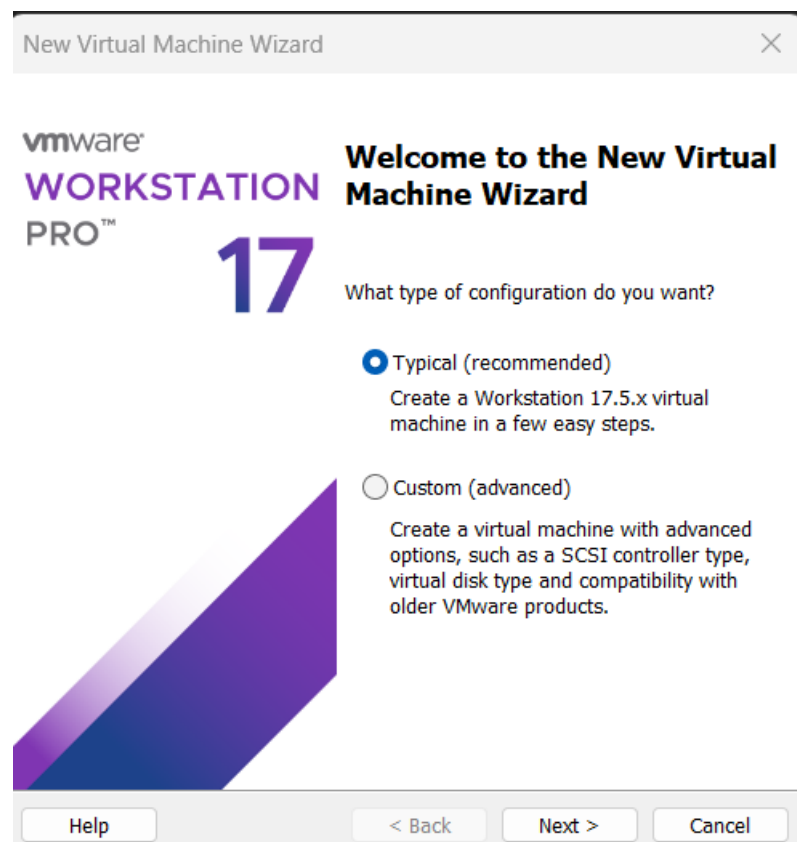
### 2. Cài đặt máy ảo

- Bước 1: Trên thanh taskbar, ta chọn File à New Virtual Machine

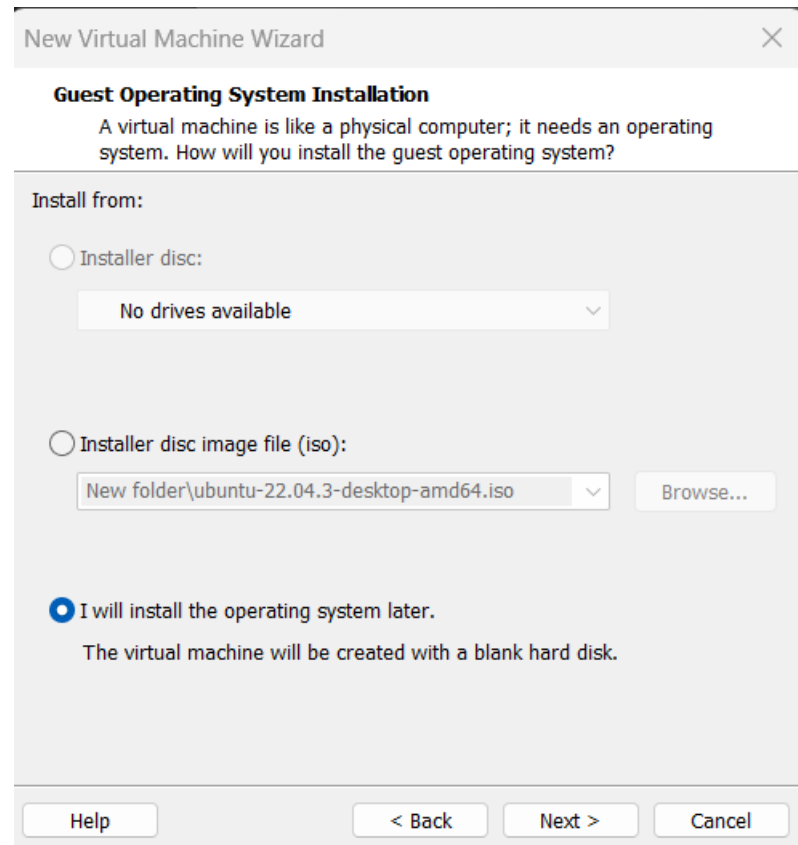


**FIGURE 8 TẠO MÁY ẢO MỚI**

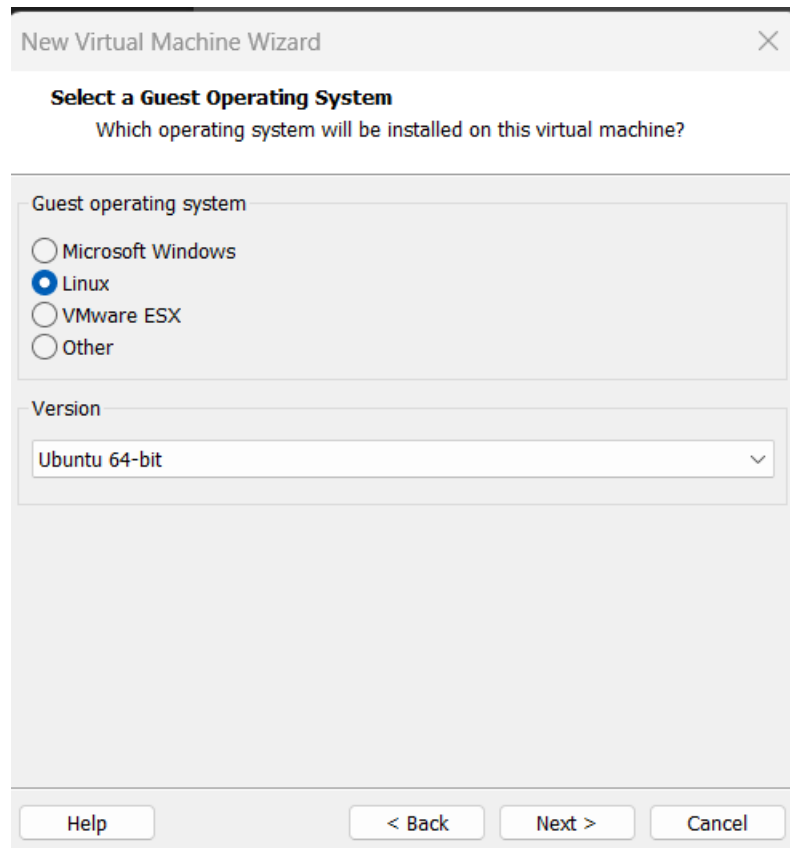
- Bước 2: Chọn kiểu configuration là Typical

**FIGURE 9 CHỌN KIỂU CONFIGURATION**

- Bước 3:

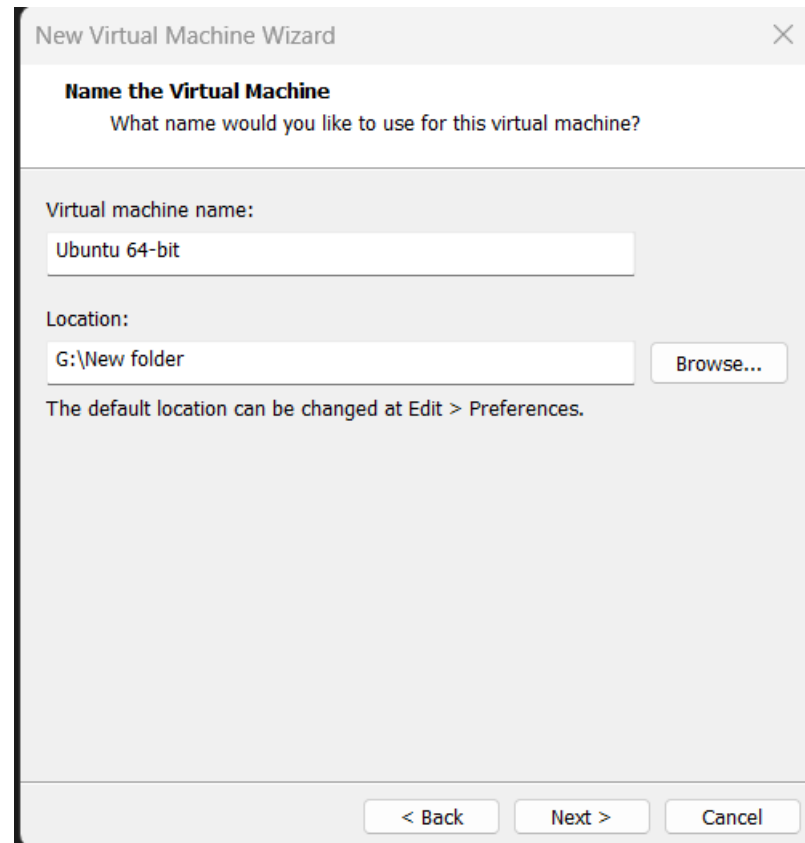
**FIGURE 10 CHỌN INSTALL TỪ ĐÂU**

- Bước 4: Chọn hệ điều hành Linux



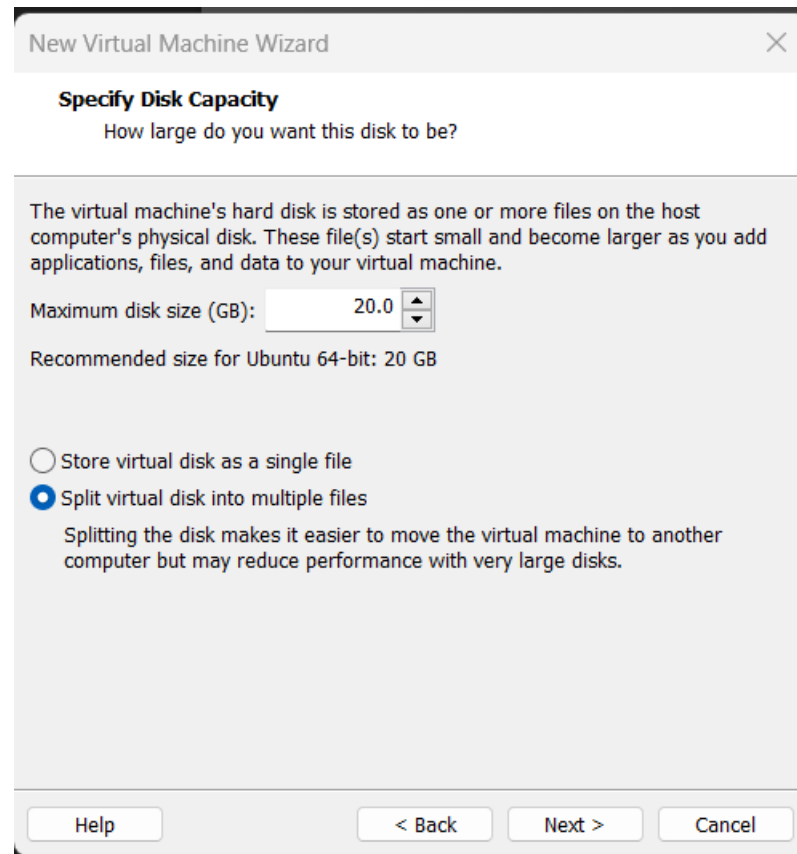
**FIGURE 11 CHỌN HỆ ĐIỀU HÀNH MUỐN CÀI ĐẶT**

- Bước 5: Đặt tên và thư mục nơi đặt máy ảo



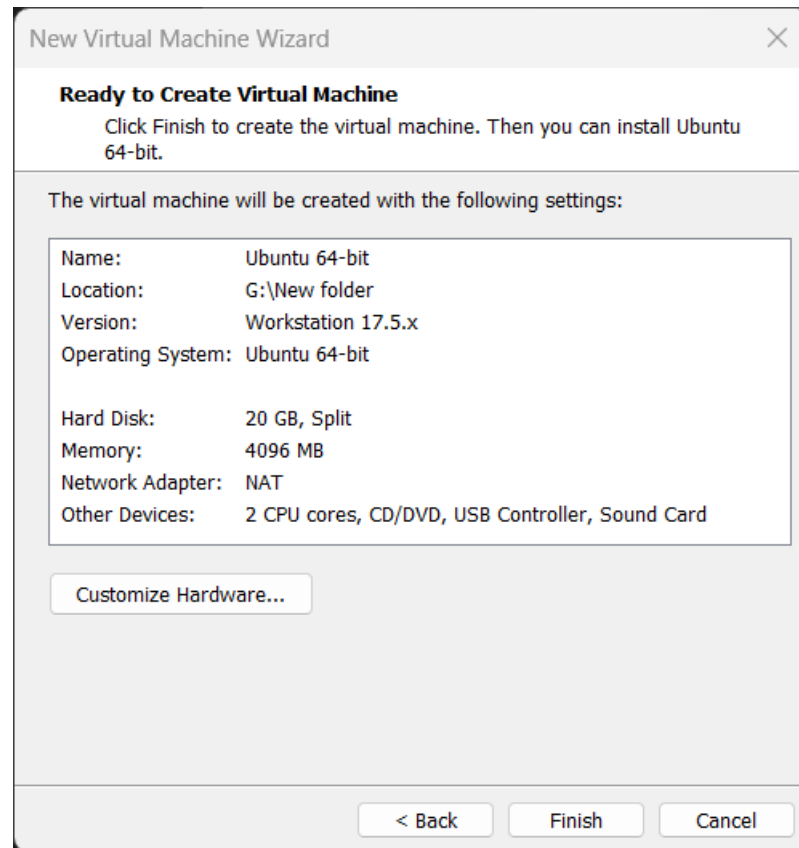
**FIGURE 12 ĐẶT TÊN CHO MÁY ẢO**

- Bước 6: Chỉ định dung lượng ổ đĩa và chọn chia ổ đĩa ảo thành nhiều file



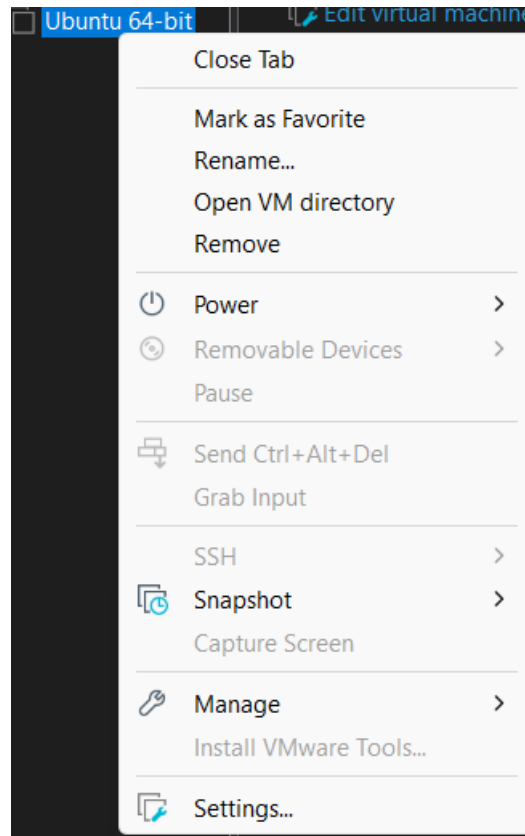
**FIGURE 13** CHỈ ĐỊNH DUNG LƯỢNG Ổ ĐĨA

- Bước 7: Thông tin sau khi cấu hình, sau khi coi xong ta chọn finish



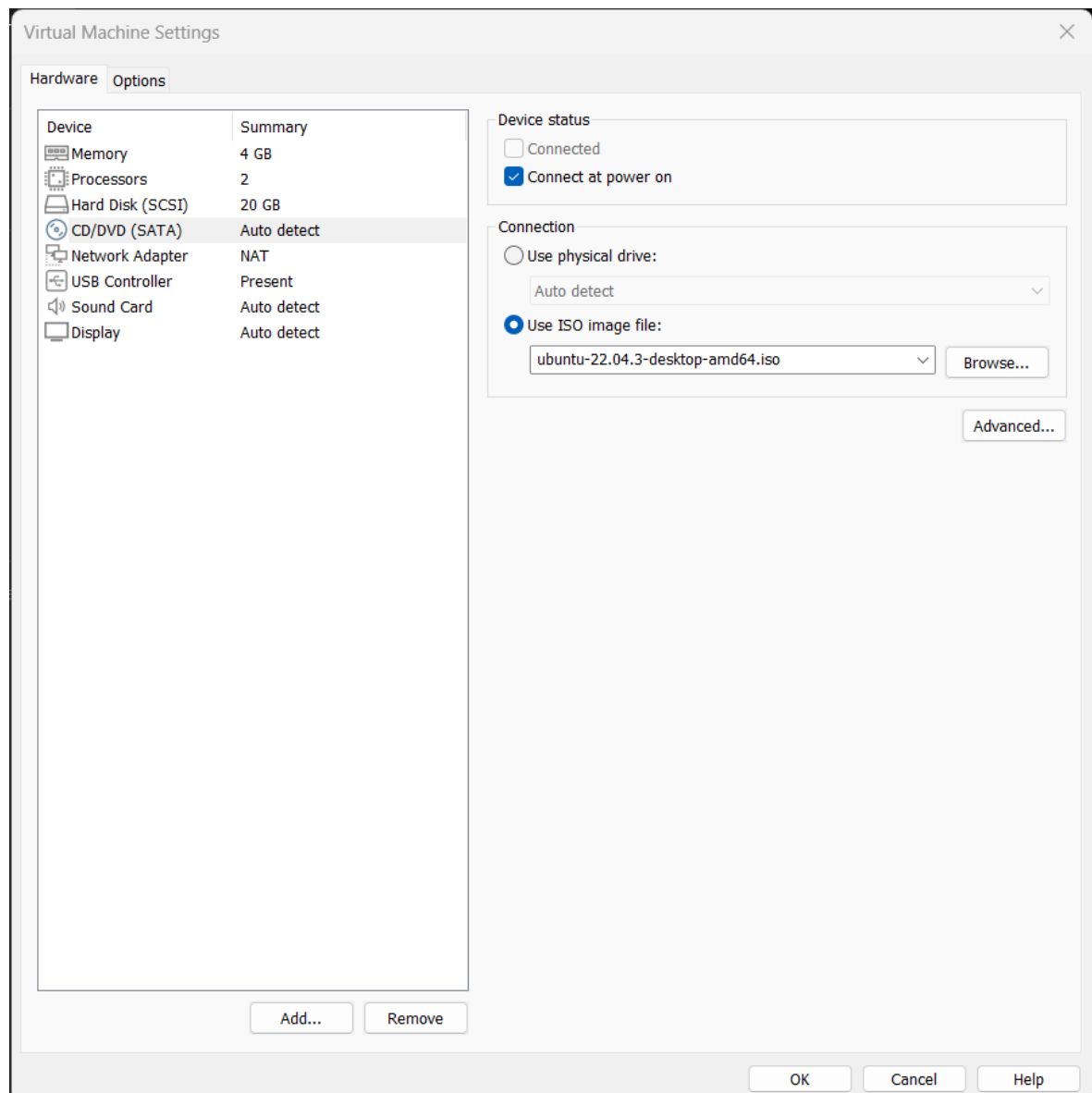
**FIGURE 14 THÔNG TIN XEM LẠI TRƯỚC KHI NHẤN FINISH**

- Bước 8: Sau đó ta vào phần Setting của máy ảo vừa tạo



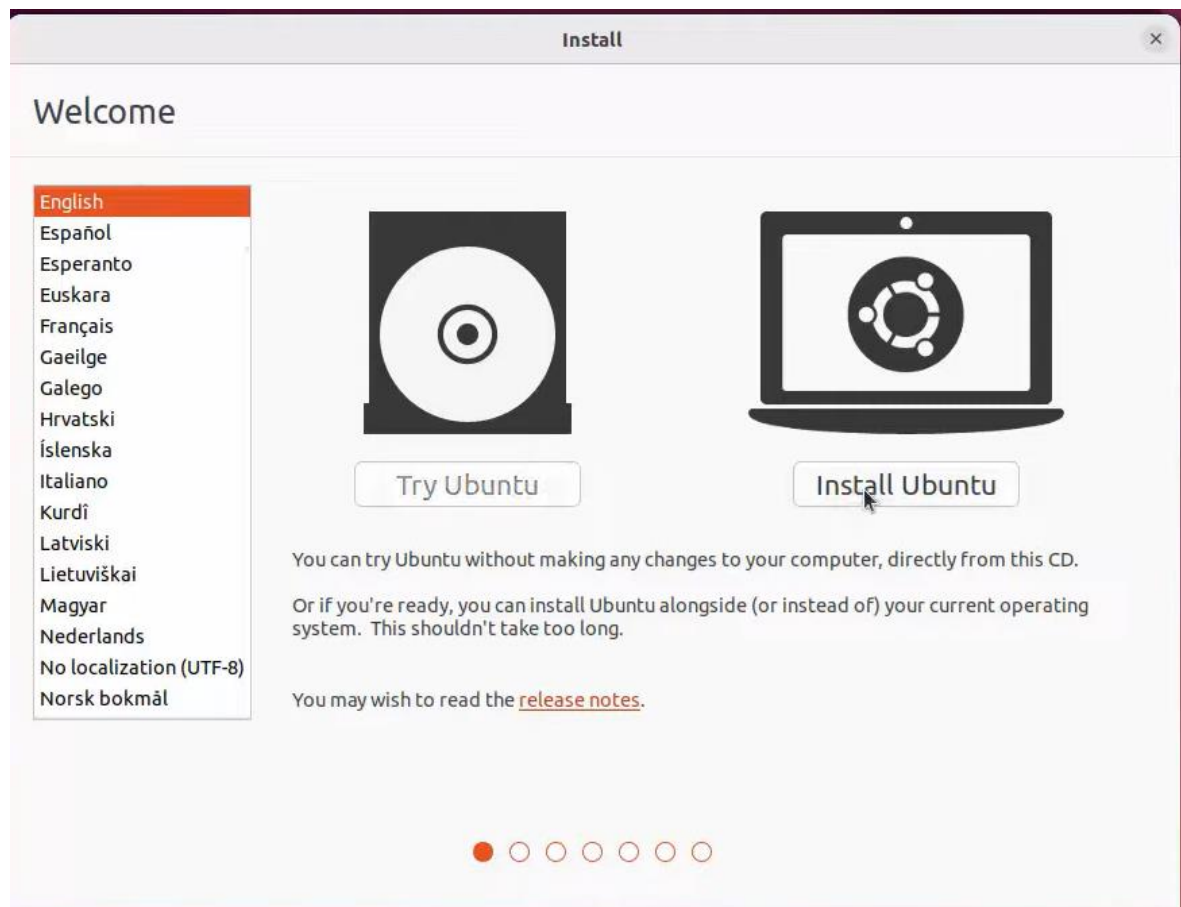
**FIGURE 15 CHỌN SETTING CỦA MÁY ẢNH VỪA CÀI**

- Bước 9: Chọn folder chứa ổ đĩa Ubuntu, sau đó nhấn OK

**FIGURE 16 CHỌN NƠI CHỨA FILE UBUNTU**

- Bước 10: Sau đó ta khởi động máy ảo và install Ubuntu để cài đặt hệ điều hành



**FIGURE 17 CHỌN TRY OR INSTALL UBUNTU****FIGURE 18 CHỌN VÀO INSTALL UBUNTU ĐỂ CÀI ĐẶT UBUNTU**

### 3. Cài đặt trên một máy

Nhóm em sử dụng KeyDB's PPA Repository để cài đặt KeyDB. PPA cho phép người dùng dễ dàng cài đặt, cập nhật, gỡ bỏ với các lệnh như apt, dpkg. Để có thể cài được keydb-ppa, chỉ cần làm theo các bước dưới đây

- Bước 1: chạy lệnh \$echo "deb https://download.keydb.dev/open-source-dist \$(lsb\_release -sc) main" | sudo tee /etc/apt/sources.list.d/keydb.list

```
kietgia05@kietgia05-virtual-machine:~/KeyDB$ echo "deb https://download.keydb.dev/open-source-dist $(lsb_release -sc) main" | sudo tee /etc/apt/sources.list.d/keydb.list
deb https://download.keydb.dev/open-source-dist jammy main
```

FIGURE 19 BƯỚC 1 CÀI ĐẶT TRÊN 1 MÁY

- Bước 2: chạy lệnh \$sudo wget -O /etc/apt/trusted.gpg.d/keydb.gpg <https://download.keydb.dev/open-source-dist/keyring.gpg>

```
kietgia05@kietgia05-virtual-machine:~/KeyDB$ sudo wget -O /etc/apt/trusted.gpg.d/keydb.gpg https://download.keydb.dev/open-source-dist/keyring.gpg
--2023-12-19 10:09:39-- https://download.keydb.dev/open-source-dist/keyring.gpg
Đang phân giải download.keydb.dev (download.keydb.dev)... 18.221.162.228
Đang kết nối tới download.keydb.dev (download.keydb.dev)|18.221.162.228|:443...
đã kết nối.
Đã gửi yêu cầu HTTP, đang đợi câu trả lời... 200 OK
Kích thước: 2299 (2.2K) [application/octet-stream]
Đang ghi vào: '/etc/apt/trusted.gpg.d/keydb.gpg'

/etc/apt/trusted.gpg 100%[=====] 2.25K --.-KB/s trong 0s

2023-12-19 10:09:40 (614 MB/s) - đã lưu '/etc/apt/trusted.gpg.d/keydb.gpg' [2299/2299]
```

FIGURE 20 BƯỚC 2 CÀI ĐẶT TRÊN 1 MÁY

- Bước 3: chạy lệnh \$sudo apt update

```
kietgia05@kietgia05-virtual-machine:~/KeyDB$ sudo apt update
Lấy:1 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Lấy:2 https://download.keydb.dev/open-source-dist jammy InRelease [3,583 B]
Lấy:3 https://download.keydb.dev/open-source-dist jammy/main amd64 Packages [3,147 B]
Tìm thấy:4 http://kh.archive.ubuntu.com/ubuntu jammy InRelease
Lấy:5 http://kh.archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Tìm thấy:6 http://kh.archive.ubuntu.com/ubuntu jammy-backports InRelease
Lấy:7 http://kh.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [1,263 kB]
Lấy:8 http://kh.archive.ubuntu.com/ubuntu jammy-updates/main i386 Packages [547 kB]
Đã lấy về 2,046 kB mất 7 giây (291 kB/g).
Đang đọc các danh sách gói... Xong
Đang xây dựng cây quan hệ phụ thuộc... Xong
Đang đọc thông tin về tình trạng... Xong
175 gói có thể được cập nhật. Chạy "apt list --upgradable" để xem chúng.
N: Skipping acquire of configured file 'main/binary-i386/Packages' as repository 'https://download.keydb.dev/open-source-dist jammy InRelease' doesn't support a architecture 'i386'
```

FIGURE 21 BƯỚC 3 CÀI ĐẶT TRÊN 1 MÁY

- Bước 4: chạy lệnh \$sudo apt install keydb

```
kietgia05@kietgia05-virtual-machine:~/KeyDB$ sudo apt install keydb
Đang đọc các danh sách gói... Xong
Đang xây dựng cây quan hệ phụ thuộc... Xong
Đang đọc thông tin về tình trạng... Xong
The following additional packages will be installed:
  keydb-server keydb-tools libsnpappy1v5
Những gói MỚI sau sẽ được CÀI ĐẶT:
  keydb keydb-server keydb-tools libsnpappy1v5
0 nâng cấp, 4 được cài đặt mới, 0 cần gỡ bỏ, và 175 chưa được nâng cấp.
Cần phải lấy 12.7 MB từ kho chứa.
Sau thao tác này, 39.5 MB dung lượng đĩa sẽ bị chiếm dụng.
Bạn có muốn tiếp tục không? [C/k] c
Lấy:1 https://download.keydb.dev/open-source-dist jammy/main amd64 keydb-tools a
md64 6:6.3.4-1~jammy1 [12.6 MB]
8% [Đang kết nối đến kh.archive.ubuntu.com] [1 keydb-tools 1,245 kB/12.
```

FIGURE 22 BƯỚC 2 CÀI ĐẶT TRÊN 1 MÁY (1)



```
kietgia05@kietgia05-virtual-machine: ~/KeyDB
Đang chuẩn bị mở gói .../libsnappy1v5_1.1.8-1build3_amd64.deb ...
Đang mở gói libsnappy1v5:amd64 (1.1.8-1build3) ...
Chọn gói chưa được chọn trước đây keydb-tools.
Đang chuẩn bị mở gói .../keydb-tools_6%3a6.3.4-1~jammy1_amd64.deb ...
Đang mở gói keydb-tools (6:6.3.4-1~jammy1) ...
Chọn gói chưa được chọn trước đây keydb-server.
Đang chuẩn bị mở gói .../keydb-server_6%3a6.3.4-1~jammy1_amd64.deb ...
Đang mở gói keydb-server (6:6.3.4-1~jammy1) ...
Chọn gói chưa được chọn trước đây keydb.
Đang chuẩn bị mở gói .../keydb_6%3a6.3.4-1~jammy1_all.deb ...
Đang mở gói keydb (6:6.3.4-1~jammy1) ...
Đang cài đặt libsnappy1v5:amd64 (1.1.8-1build3) ...
Đang cài đặt keydb-tools (6:6.3.4-1~jammy1) ...
Đang cài đặt keydb-server (6:6.3.4-1~jammy1) ...
Created symlink /etc/systemd/system/keydb.service → /lib/systemd/system/keydb-server.service.
Created symlink /etc/systemd/system/multi-user.target.wants/keydb-server.service → /lib/systemd/system/keydb-server.service.
Đang cài đặt keydb (6:6.3.4-1~jammy1) ...
Đang xử lý các bẫy dành cho man-db (2.10.2-1) ...
Đang xử lý các bẫy dành cho libc-bin (2.35-0ubuntu3.1) ...
kietgia05@kietgia05-virtual-machine: ~/KeyDB$
```

FIGURE 23 BƯỚC 4 CÀI ĐẶT TRÊN 1 MÁY (2)

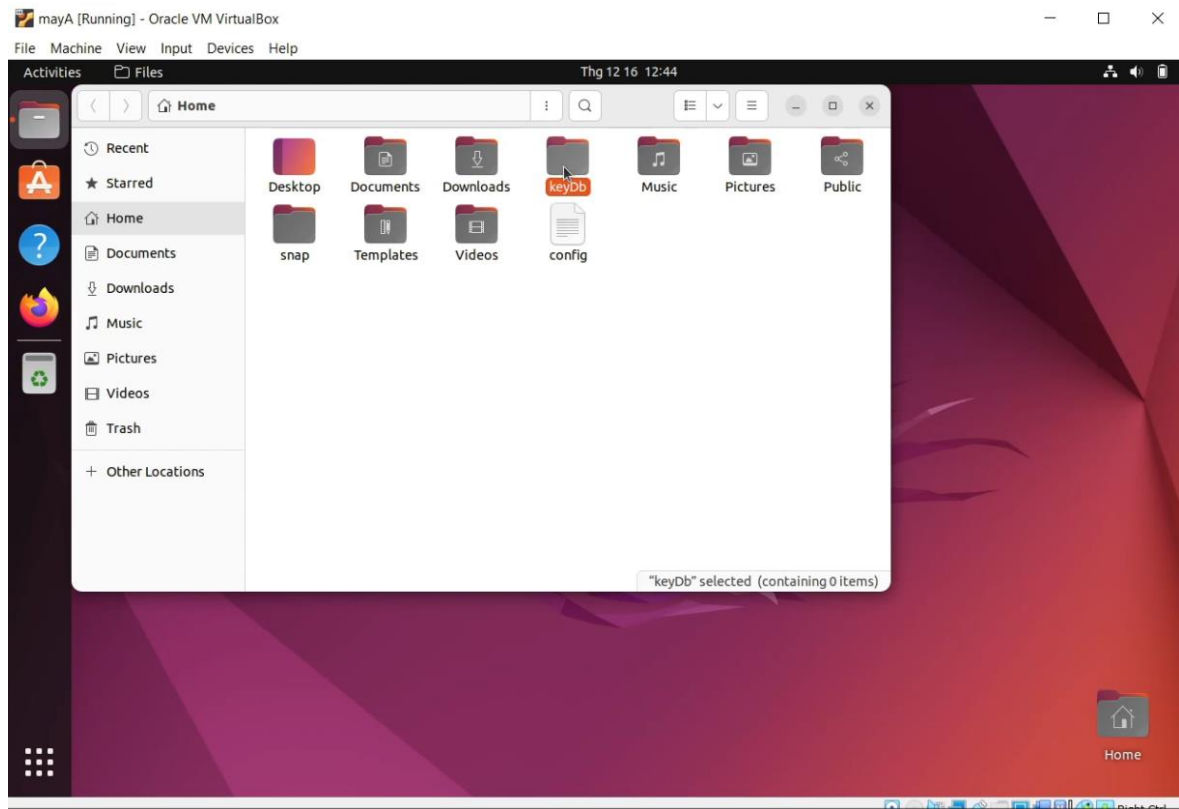
```
kietgia05@kietgia05-virtual-machine: ~/KeyDB$ keydb-cli
Message of the day:
  KeyDB has now joined Snap! See the announcement at: https://docs.keydb.dev/news

127.0.0.1:6379> keys *
(empty array)
127.0.0.1:6379>
```

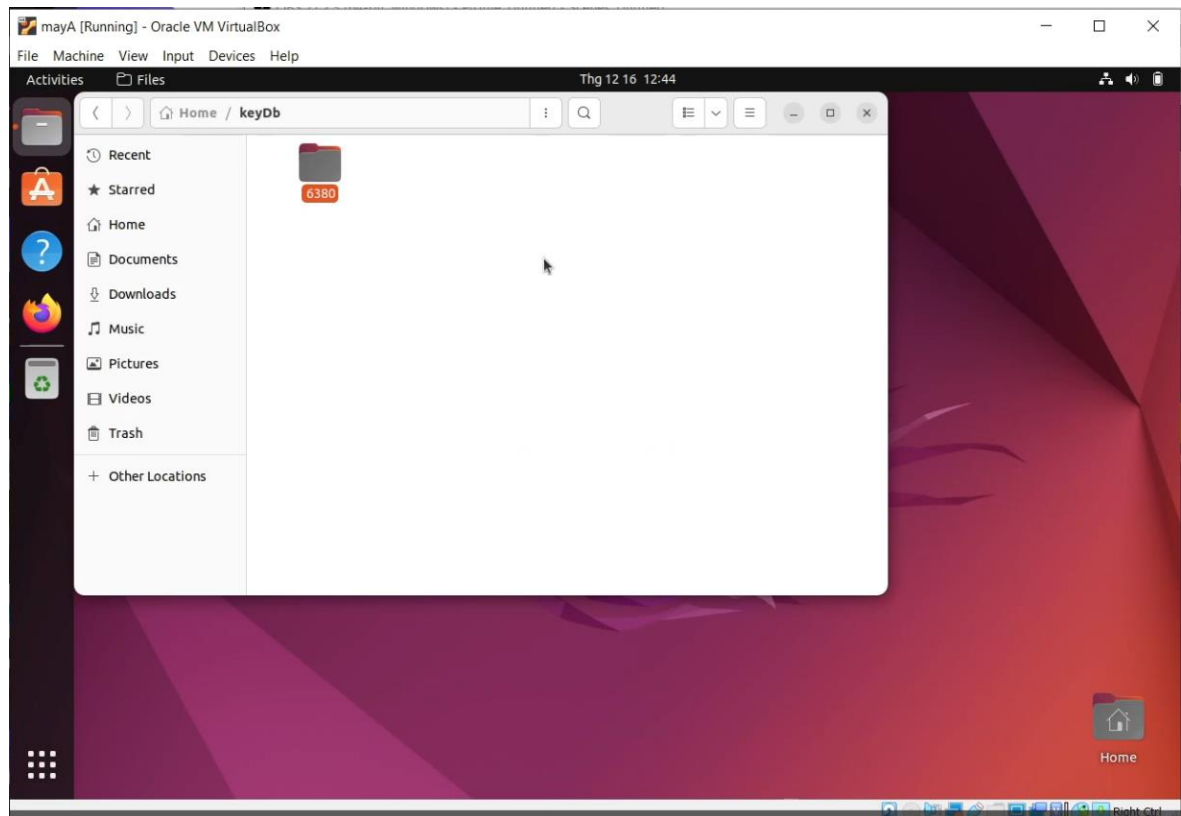
FIGURE 24 THỬ CHẠY CÁC LỆNH SAU KHI CÀI ĐẶT KEYDB

#### 4. Cài đặt trên cụm máy phân tán

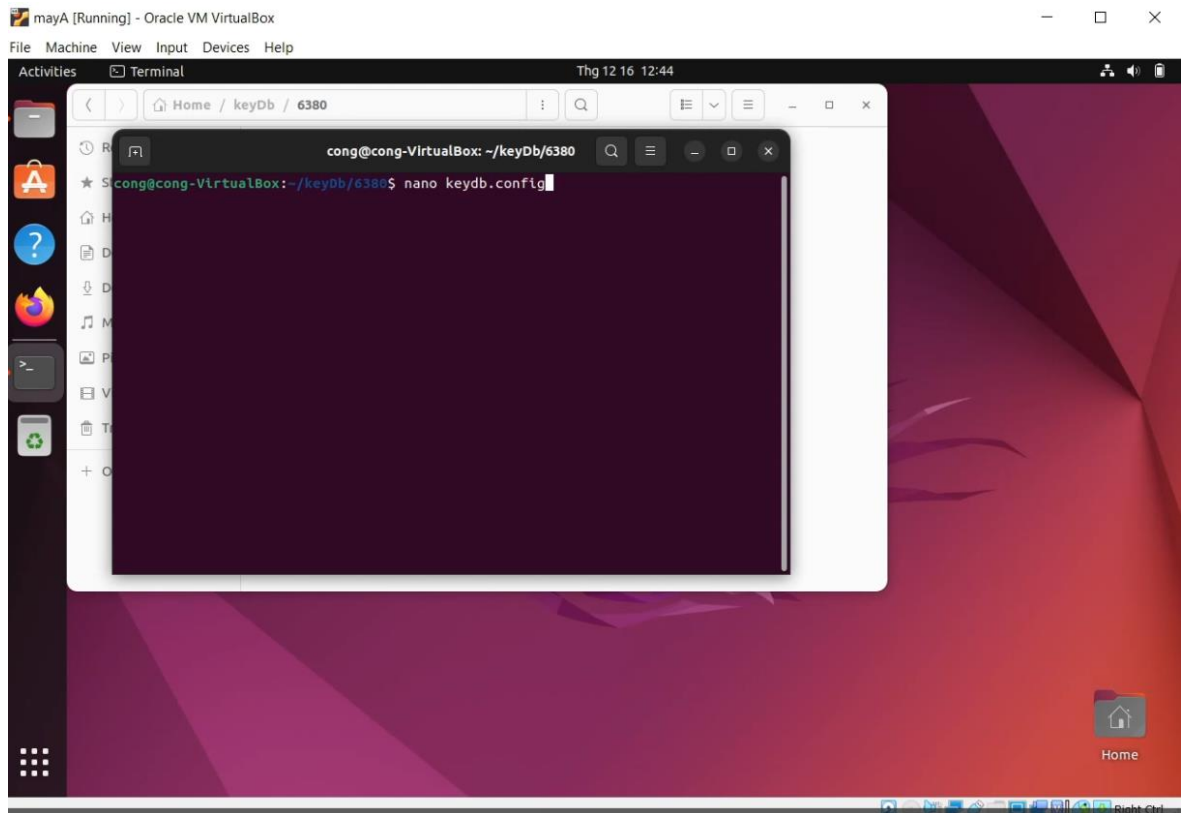
- Bước 1: Tại mỗi máy tạo 1 thư mục KeyDb:

**FIGURE 25 TẠO THƯ MỤC KEYDB**

- Bước 2: trong thư mục keyDb tạo thêm 1 thư mục 6380(port để kết nối):

**FIGURE 26 TẠO THƯ MỤC CON 6380**

- Bước 3: trong thư mục 6380 vừa tạo, mở terminal tại đó và tạo 1 file text keydb.config như sau:



**FIGURE 27 CHẠY TERMINAL TẠO FILE TEXT**

- Bước 4: vào file keydb.config và copy các dòng sau đây vào:  
LƯU Ý: thay địa chỉ ip nguồn và đích theo ip của 2 máy đang cần kết nối  
port 6380

appendonly yes

bind 127.0.0.1 (địa chỉ ip nguồn)

active-replica yes

multi-master yes

replicaof (địa chỉ ip đích) 6380

dir ./

loglevel notice

logfile 6380.log

save 900 1

save 300 10

save 60 10000



```
GNU nano 6.2
port 6380
cluster-enabled no
cluster-config-file nodes.conf
cluster-node-timeout 5000
appendonly yes
bind 127.0.0.1 192.168.1.155
active-replica yes
multi-master yes
#slaveof 192.168.1.174 6380
replicaof 192.168.1.174 6380
dir ./
loglevel notice
logfile 6380.log

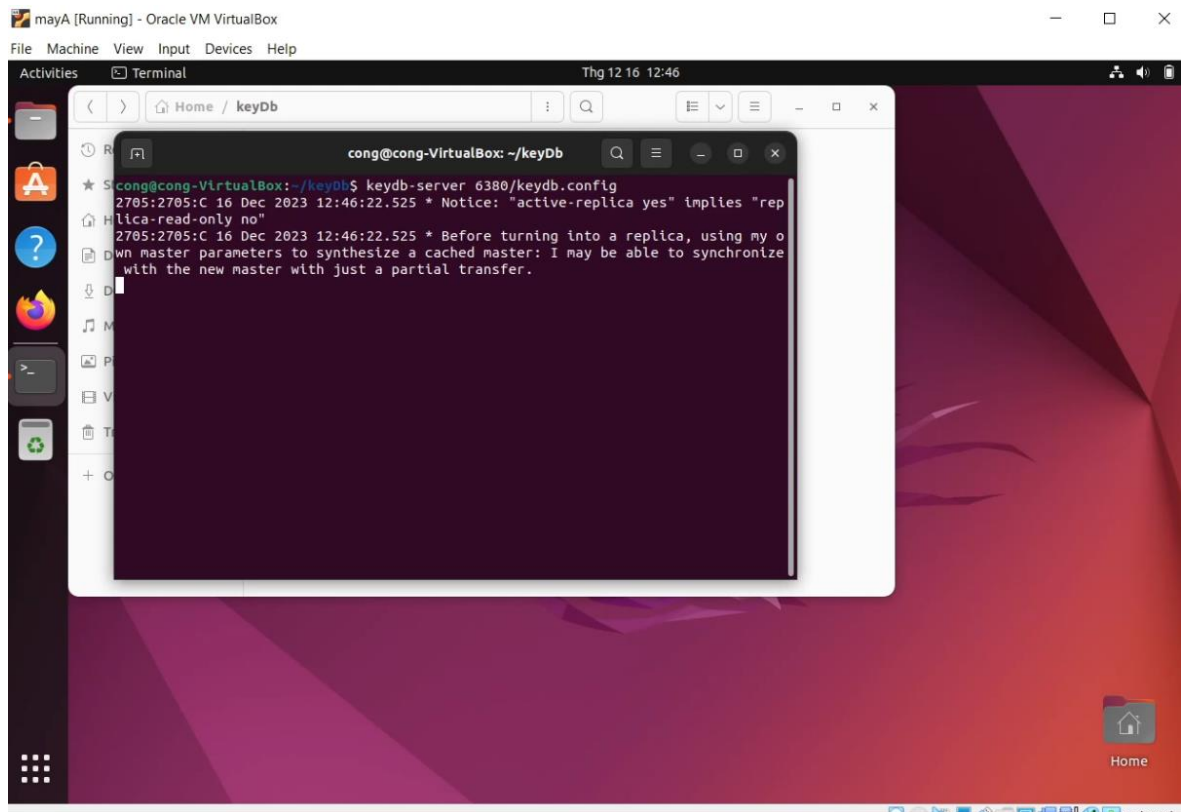
save 900 1
save 300 10
save 60 10000
```

**FIGURE 28** VÀO FILE KEYDB.CONFIG VỚI TERMINAL VÀ COPY CÁC LỆNH VÀO

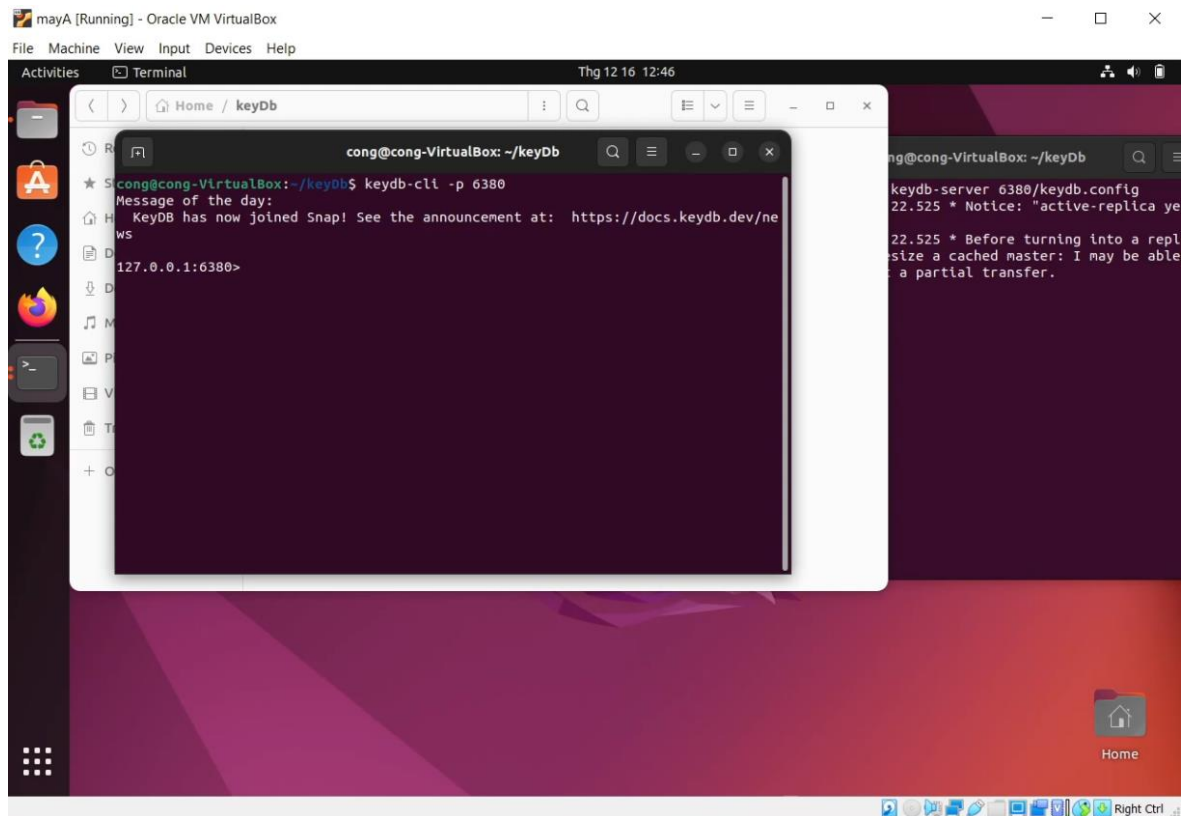
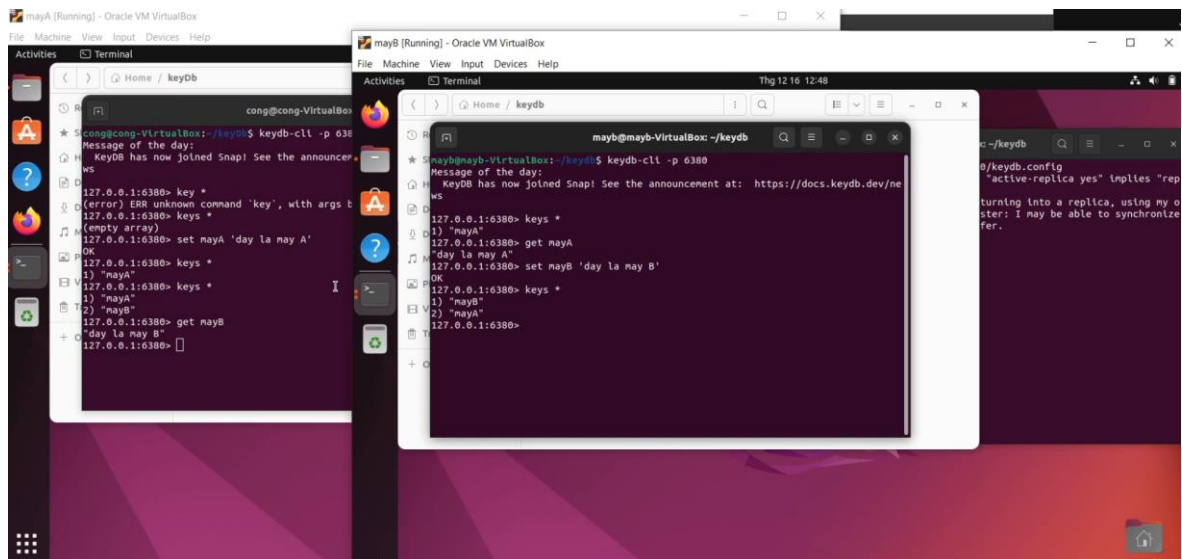
- Bước 5: Mở terminal tại folder keydb và nhập lệnh:

keydb-server 6380/keydb.config



**FIGURE 29 NHẬP LỆNH**

- Bước 6: kiểm thử bằng cách connect vào port 6380 vừa tạo ở 2 máy và tạo, lấy dữ liệu của nhau:

**FIGURE 30 CONNECT VÀO PORT 6380****FIGURE 31 THỬ THÊM VÀ LẤY DỮ LIỆU TỪ 2 BÊN**

5. Một số vấn đề khi cài đặt keydb:

a. Không thể connect keydb-cli:

Đã Thử: `sudo apt-get install redis-server` và không còn lỗi

b. Gặp lỗi illegal instructor thì làm như sau:

Bước 1: chạy cmd của window với quyền admin

Bước 2: nhập `bcdedit /set hypervisorlaunchtype off` và enter

Bước 3: nhập `DISM /Online /Disable-Feature:Microsoft-Hyper-V` và enter

Bước 4: Tắt máy với lệnh `shutdown -s -t 2` và đợi vài giây sau đó mở máy lại

c. Nên chuyển network sang Bridge Adapter

### CHƯƠNG III: THỰC NGHIỆM MÔ PHỎNG PHÂN TÁN

1. Mô tả bài toán đặt ra với dữ liệu

- **CSDL Hệ thống cửa hàng kinh doanh thiết bị công nghệ - Ttech**

Hệ thống cửa hàng kinh doanh thiết bị công nghệ có 3 cửa hàng, với trụ sở chính nằm ở Quận 1 thuộc Thành phố Hồ Chí Minh, 2 cửa hàng còn lại lần lượt nằm ở Thành phố Thủ Đức thuộc Thành phố Hồ Chí Minh và Quận Bình Thạnh thuộc Thành Phố Hồ Chí Minh

- Các chức năng của hệ thống

- Đăng nhập hệ thống
- Quản lý danh mục sản phẩm
- Quản lý sản phẩm
- Quản lý người dùng
- Quản lý đơn hàng
- Quản lý nhà cung cấp

- Thống kê doanh thu của cửa hàng
- Thống kê doanh thu của các cửa hàng
- Lý do phân tán dữ liệu
  - Hệ thống có nhiều cửa hàng nằm ở nhiều nơi khác nhau ở Thành phố Hồ Chí Minh và có thể mở rộng ra quy mô cả nước, do đó có nhu cầu trao đổi và xử lý thông tin giữa các cửa hàng.
  - Trong thực tế, có cửa hàng được phân tán khắp nơi, trong khi đó, dữ liệu quản lý ngày càng lớn và phục vụ cho đa người dùng phân tán, vì vậy CSDL phân tán là con đường thích hợp nhất.
  - Nhằm thích ứng tốt hơn với việc phân bố rộng rãi của các cửa hàng trong hệ thống. Quan trọng hơn, nhiều ứng dụng hiện tại của công nghệ máy tính cũng được phân tán.
  - Các CSDL hiện tại cần kết nối với nhau: CSDL phân tán là giải pháp tự nhiên khi có các CSDL đang tồn tại và sự cần thiết xây dựng một ứng dụng toàn cục.
  - Sự lớn mạnh của hệ thống: Hệ thống cửa hàng có thể phát triển mở rộng bằng cách thành lập thêm các đơn vị mới, vừa có tính tự trị, vừa có quan hệ với các đơn vị tổ chức khác.
  - Giảm chi phí truyền thông: Nhiều ứng dụng cục bộ làm giảm chi phí truyền thông.
  - Nâng cao hiệu suất: Có cơ chế xử lý song song và phân mảnh dữ liệu theo ứng dụng làm cực đại hóa tính cục bộ của ứng dụng.
  - Tăng độ tin cậy và tính sẵn sàng: Nếu có một thành phần nào đó của hệ thống bị hỏng, hệ thống vẫn có thể duy trì hoạt động.

## 2. Mô tả cấu trúc dữ liệu sử dụng

- Lược đồ cơ sở dữ liệu của mỗi cửa hàng TTech

**STORE** (store\_id, name\_st, address, sdt)

**Tên từ:** Mỗi cửa hàng có mã cửa hàng (store\_id) dùng để phân biệt các cửa hàng với nhau, ngoài ra còn lưu tên cửa hàng (name\_st), số điện thoại của cửa hàng đó (sdt), địa chỉ nơi cửa hàng được xây dựng và hoạt động (address). Trường tên cửa hàng, số điện thoại là duy nhất.

**CATEGORY** (category\_id, category\_name)

**Tên từ:** Mỗi loại sản phẩm có mã loại sản phẩm (category\_id) là duy nhất, ngoài ra còn lưu tên nhà cung cấp (category\_name).

**SUPPLIER** (supplier\_id, supplier\_name)

**Tên từ:** Mỗi nhà cung cấp có mã nhà cung cấp (supplier\_id) là duy nhất, ngoài ra còn lưu tên loại sản phẩm (supplier\_name).

**PRODUCTS** (product\_id, name\_pr, price, detail, supplier\_id, category\_id, quantity)

**Tên từ:** Mỗi sản phẩm có mã sản phẩm (product\_id) riêng để phân biệt với các sản phẩm khác. Ngoài ra còn lưu tên sản phẩm (product\_id), giá tiền (price), chú thích về sản phẩm (detail), mã nhà cung cấp (supplier\_id), mã loại sản phẩm (category\_id), số lượng (quantity).

**USER** (user\_id, name, email, phone, gender, role)

**Tên từ:** Mỗi người dùng có mã người dùng khác nhau (user\_id) để phân biệt với những người dùng khác. Ngoài ra còn lưu tên người dùng (name), email người dùng (email), số điện thoại người dùng (phone), mật khẩu người dùng (password), vai trò người dùng (role), nếu role có giá trị là 1 thì người đó là nhân viên cửa hàng, role có giá trị là 0 thì người đó là khách hàng.

**ORDER** (order\_id, user\_id, create\_order\_at, store\_id, total)

**Tên từ:** Mỗi đơn hàng có mã đơn hàng (order\_id) là duy nhất, ngoài ra còn lưu thông tin mã người dùng (user\_id), ngày lập hoá đơn (create\_order\_at), mã cửa hàng nơi bán sản phẩm (store\_id), tổng tiền (total).

**DETAIL\_ORDER** (order\_id, product\_id, price, quantity\_pr)

**Tên từ:** Chi tiết đơn hàng lưu trữ thông tin mã sản phẩm (product\_id) và mã đơn hàng (order\_id) mà chi tiết đơn hàng thuộc về, ngoài ra còn có giá tiền sản phẩm đó và số lượng sản phẩm đó được mua.

- Dữ liệu mẫu ở các bảng
  - Bảng STORE

**TABLE 1 BẢNG STORE**

key	value		
	Name_st	address	sdt
store:1	Ttech Quận 1	25A Mai Thị Lưu, phường Đa Kao, Quận 1, Thành phố Hồ Chí Minh	0123789456
store:2	Ttech Thủ Đức	34A Đường Số 12, phường Trường Thọ, Thành phố Thủ Đức, Thành phố Hồ Chí Minh	0147963852
store:3	Ttech Bình Thạnh	243A Nguyễn Thượng Hiền, Quận Bình Thạnh, Thành phố Hồ Chí Minh	0369784512

- Bảng CATEGORY

**TABLE 2 BẢNG CATEGORY**

key	value
	category_name
cate:1	Điện thoại
cate:2	Tablet
cate:3	Laptop

- Bảng SUPPLIER

**TABLE 3 BẢNG SUPPLIER**

key	value
	supplier_name
supp:1	Thế giới di động
supp:2	FPT Shop
supp:3	Phong Vũ

## ○ Bảng PRODUCTS

**TABLE 4 BẢNG PRODUCTS**

key	value					
	name_pr	price	detail	supplier_id	category_id	quantity
product:1	Iphone15	31,999,000	none	supp:1	cate:1	50
product:2	Samsung Zflip 5	16,090,000	none	supp:2	cate:1	50
product:3	MSI GF63 Thin 11UC- 444VN	13,490,000	none	supp:3	cate:3	50
product:4	Samsung Galaxy Tab S9	9,990,000	none	supp:1	cate:2	50
product:5	Samsung Galaxy Tab S9+	21,990,000	none	supp:1	cate:2	50

## ○ Bảng USER

**TABLE 5 BẢNG USER**

key	value				
	name	email	phone	gender	role
user:1	Nguyễn Trần Gia Kiệt	ntgkiet@gmail.com	0387518892	nam	admin
user:2	Thi Thành Công	congthanhthi@gmail.com	0951753486	nam	admin
user:3	Cao Hoài Sang	chsang@gmail.com	0751953684	nam	admin
user:4	Trần Thành Tin	tinthanh@gmail.com	0153789455	nam	admin
user:5	Nguyễn Thị Thùy	thithuy@gmail.com	0931475742	nữ	client
user:6	Nguyễn Thị Kiều Oanh	oanhnguyen@gmail.com	0167114889	nữ	client
user:7	Trần Thị Lê Na	lenatiktok@gmail.com	0245966434	nữ	client
user:8	Lữ Tiến Toàn	celano340@gmail.com	0921442131	nam	client
user:9	Hà Thục Uyển	chanhleockt@gmail.com	0741331776	nữ	client



user:10	Bùi Thị Hồng Uyên	hoangbui@gmail.com	0992459644	nữ	client
---------	-------------------------	--------------------	------------	----	--------

○ Bảng ORDER

**TABLE 6 BẢNG ORDER**

key	value			
	user_id	create_order_at	store_id	total
order:1:user5	user:5	12/12/2023	store:1	16,090,000
order:2:user6	user:6	12/12/2023	store:2	45,489,000
order:3:user7	user:7	12/12/2023	store:1	9,990,000
order:4:user8	user:8	12/12/2023	store:3	21,990,000

○ Bảng DETAIL\_ORDER

**TABLE 7 BẢNG DETAIL\_ORDER**

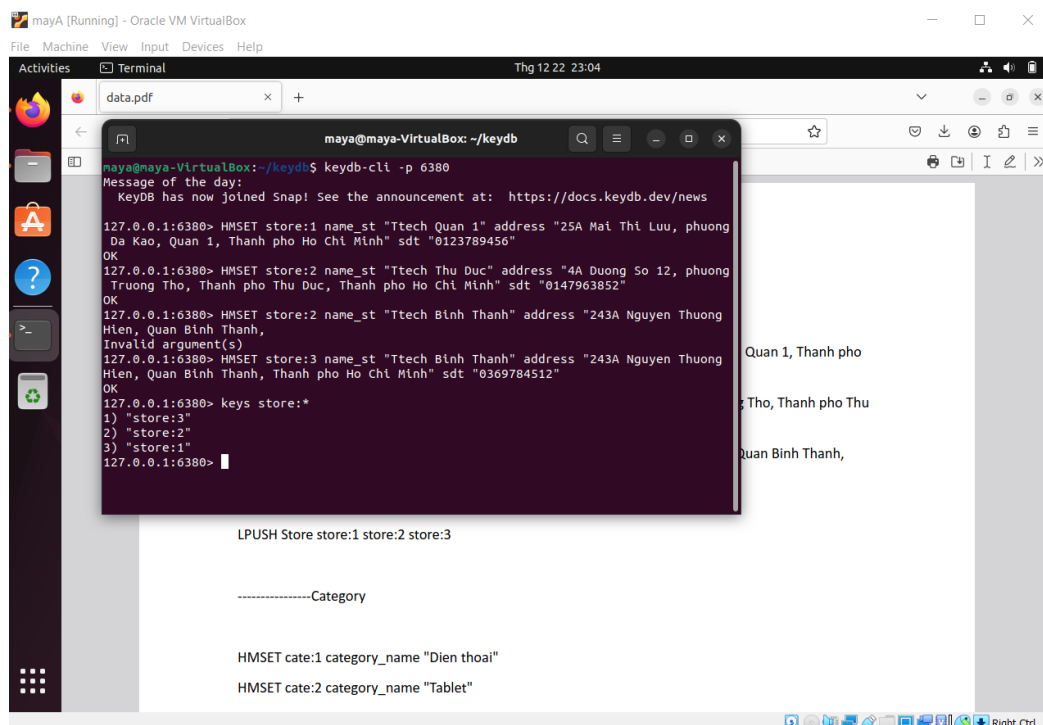
key	value		
	product_id	price	quantity_pr
ordDe:1:order1	product:2	16,090,000	1
ordDe:1:order2	product:1	31,999,000	1
ordDe:2:order2	product:3	13,490,000	1
ordDe:1:order3	product:4	9,990,000	1
ordDe:1:order4	product:5	21,990,000	1

### 3. Các bước thực nghiệm chi tiết

#### 3.1 Tạo dữ liệu:

Trong keydb không có cách tạo bảng thông thường, mà nó lưu dữ liệu dưới dạng key-value, vì thế cách tạo dữ liệu cho từng mục có thể làm bằng hset như sau:

- Dữ liệu STORE: insert 2 dòng dữ liệu từ 1 máy và đều có thể select tất cả dữ liệu đã insert từ 2 máy
  - o Thêm và lấy dữ liệu từ máy A:



**FIGURE 32 THÊM DỮ LIỆU STORE Ở MÁY A**

- o Lấy dữ liệu đó từ máy B:

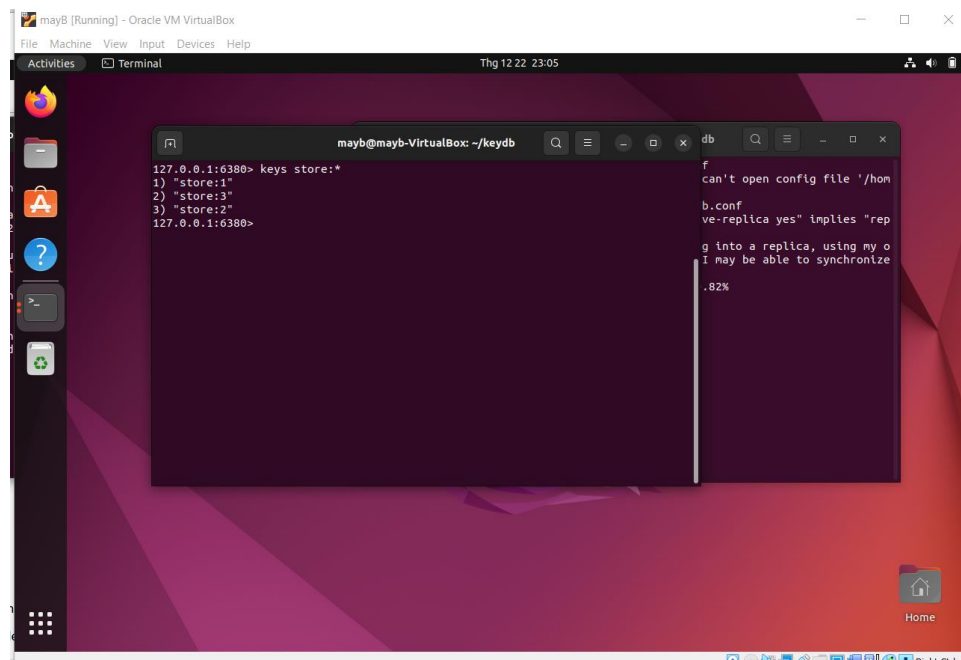


FIGURE 33 LẤY DỮ LIỆU Ở MÁY B

- Dữ liệu CATEGORY tương tự:

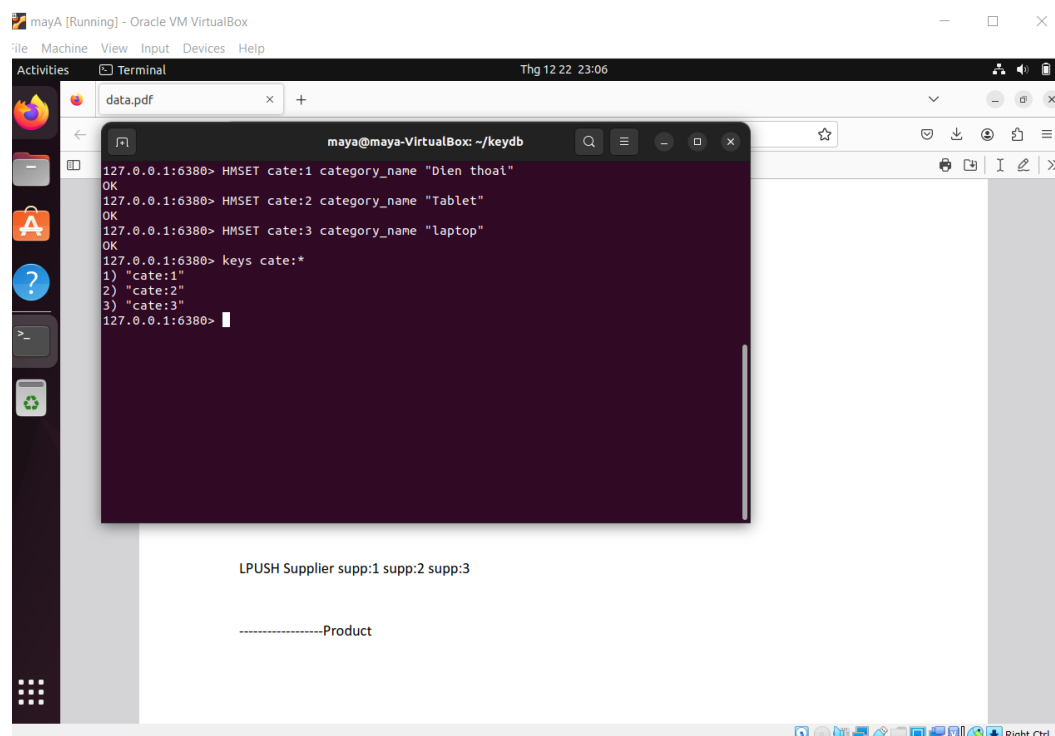
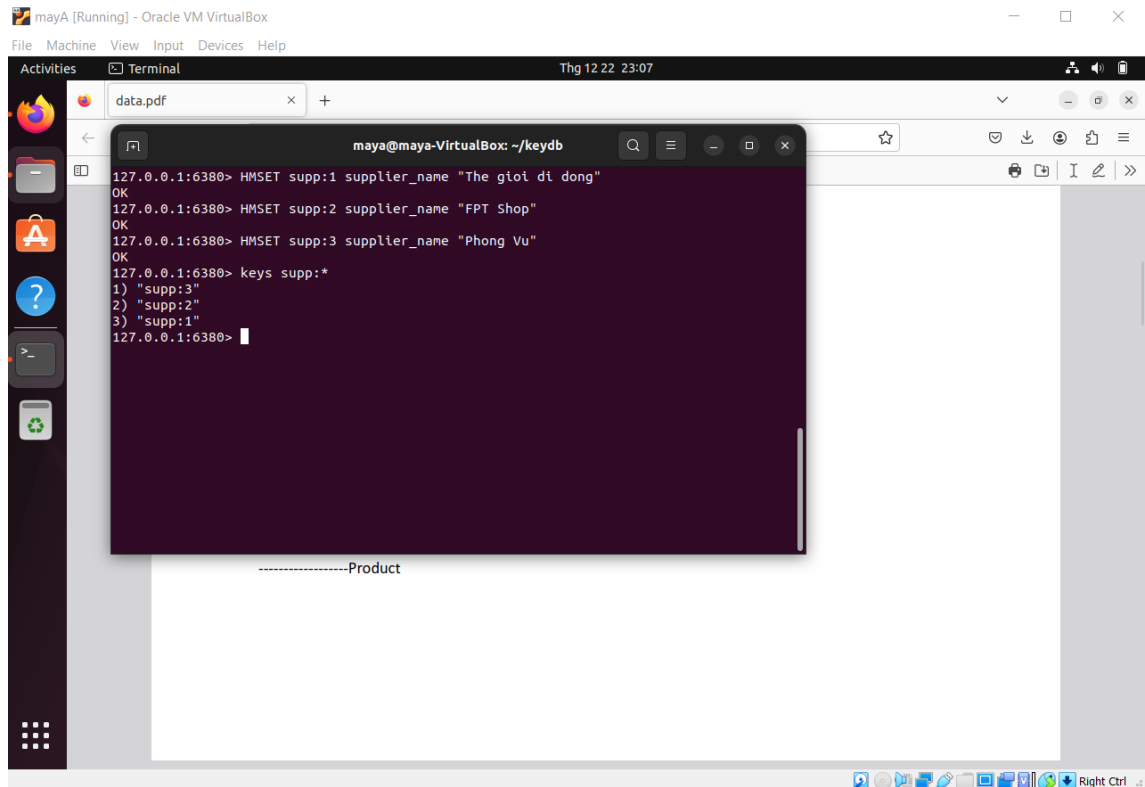


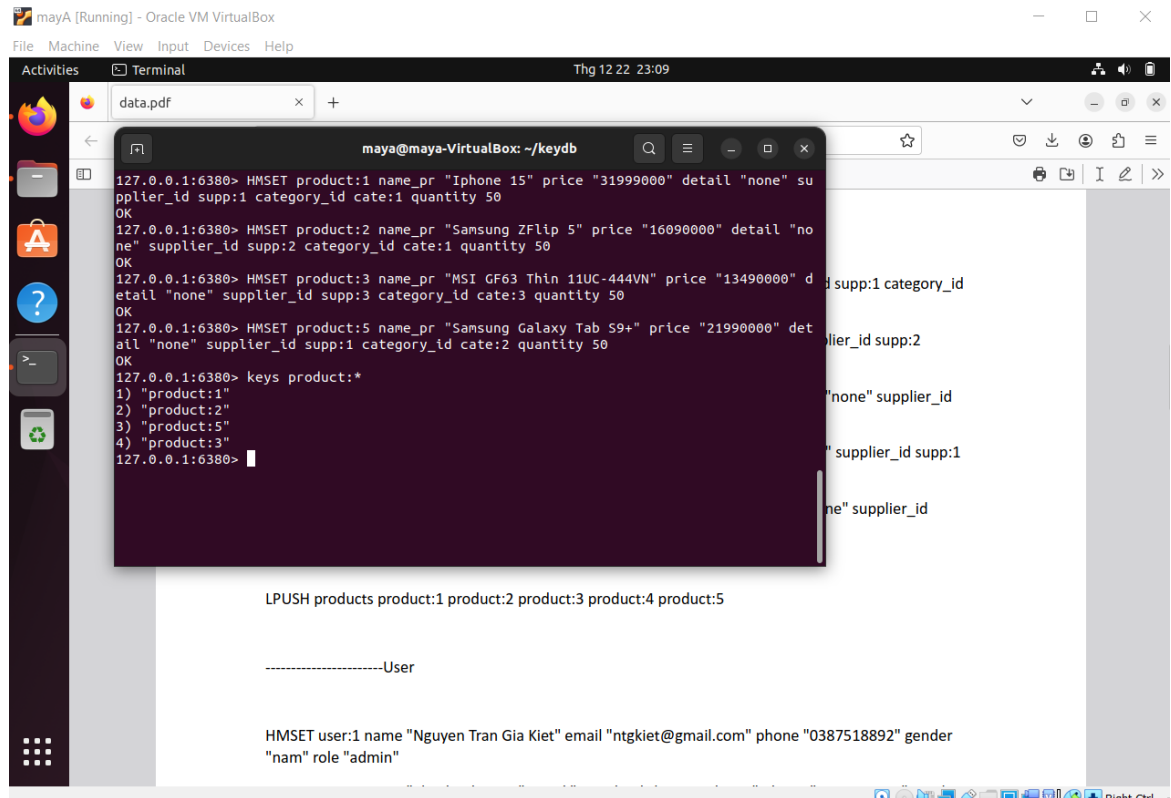
FIGURE 34 THÊM DỮ LIỆU CATEGORY Ở MÁY A

- Dữ liệu SUPPLIER:



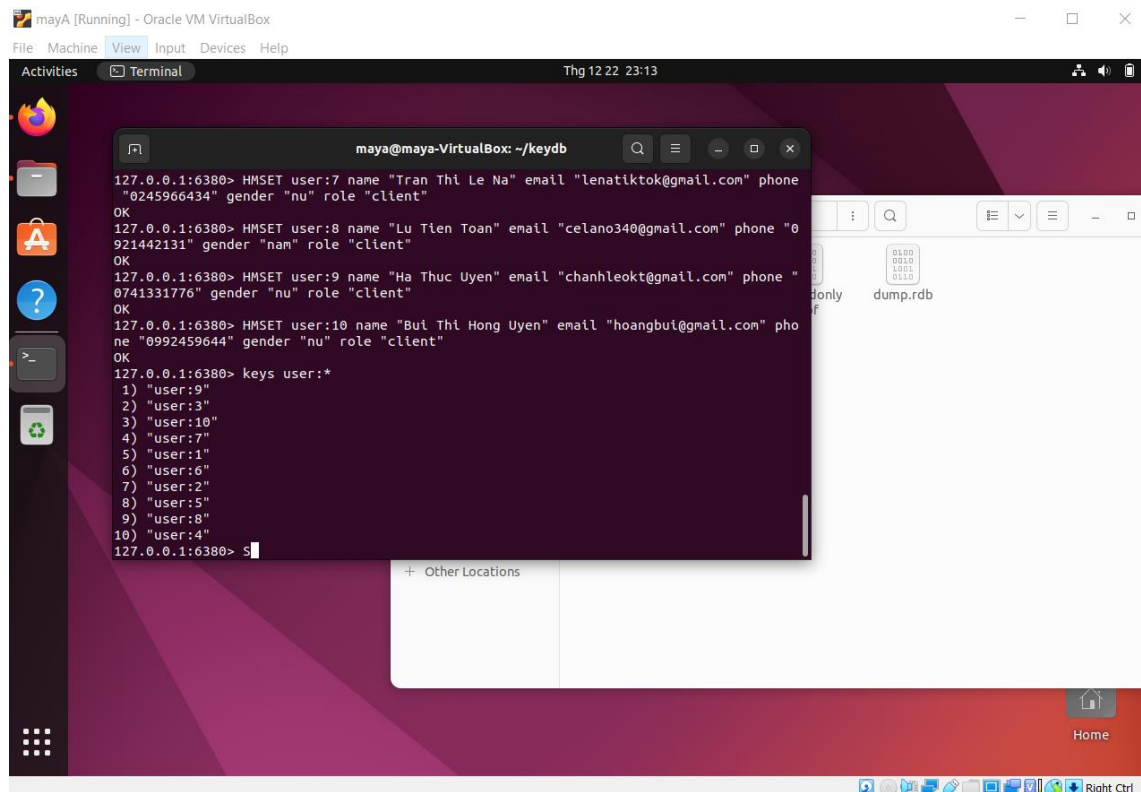
**FIGURE 35 THÊM DỮ LIỆU SUPPLIER Ở MÁY A**

- Dữ liệu PRODUCTS:



**FIGURE 36 THÊM DỮ LIỆU PRODUCTS Ở MÁY A**

- Dữ liệu USER:



**FIGURE 37** THÊM DỮ LIỆU USER Ở MÁY A

- Dữ liệu ORDER: ở đây sẽ lưu các dữ liệu của order và kèm theo user để nhận biết đó là order của user nào:

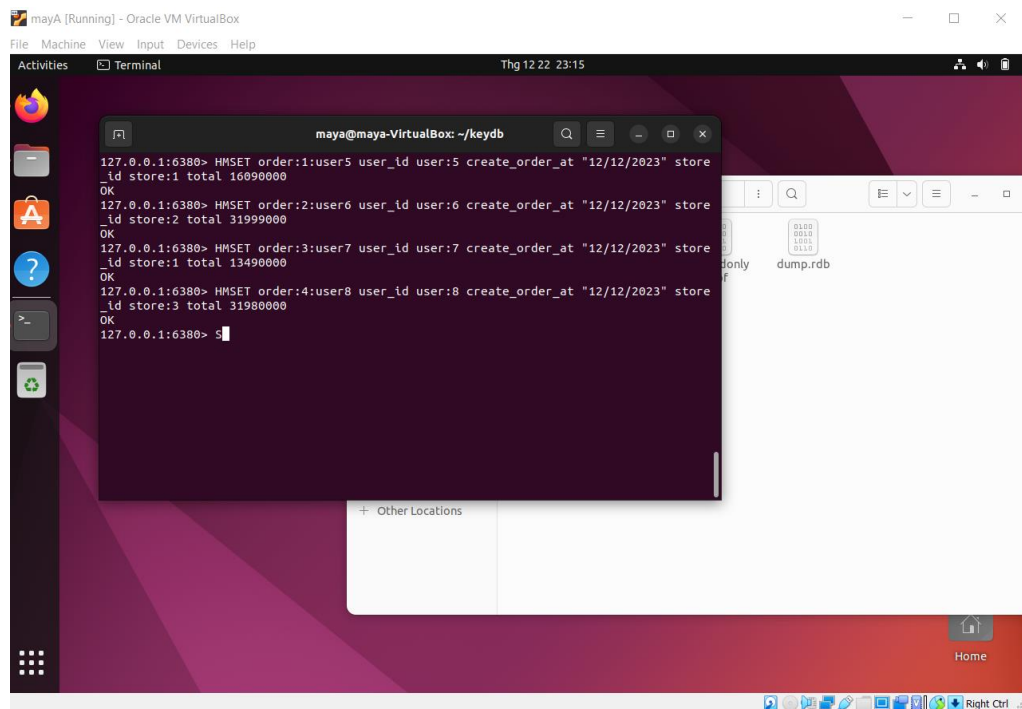
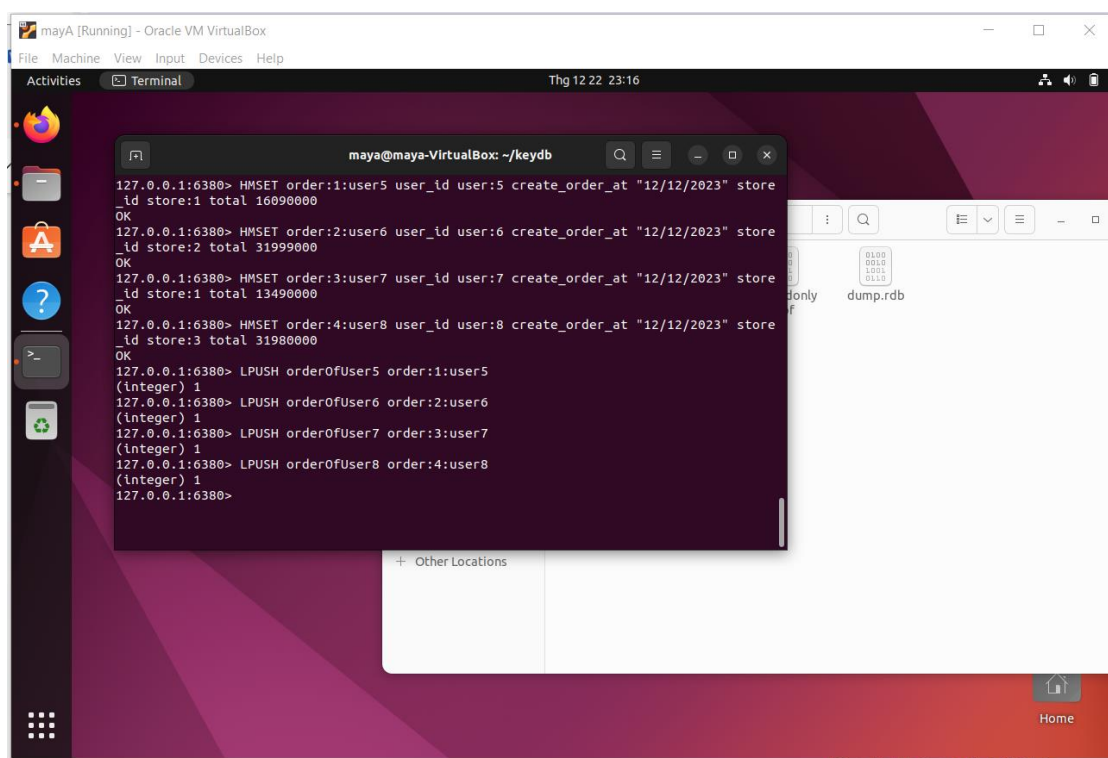


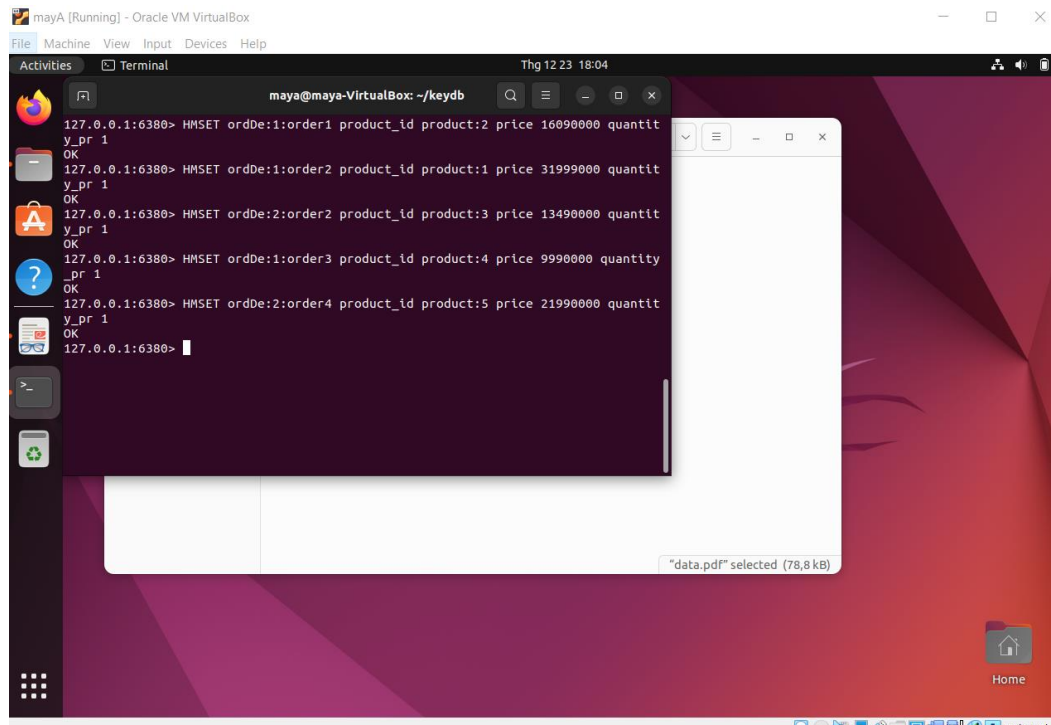
FIGURE 38 THÊM DỮ LIỆU ORDER Ở MÁY A

- Sau đó sẽ đưa từng order của từng user vào list để dễ tìm:



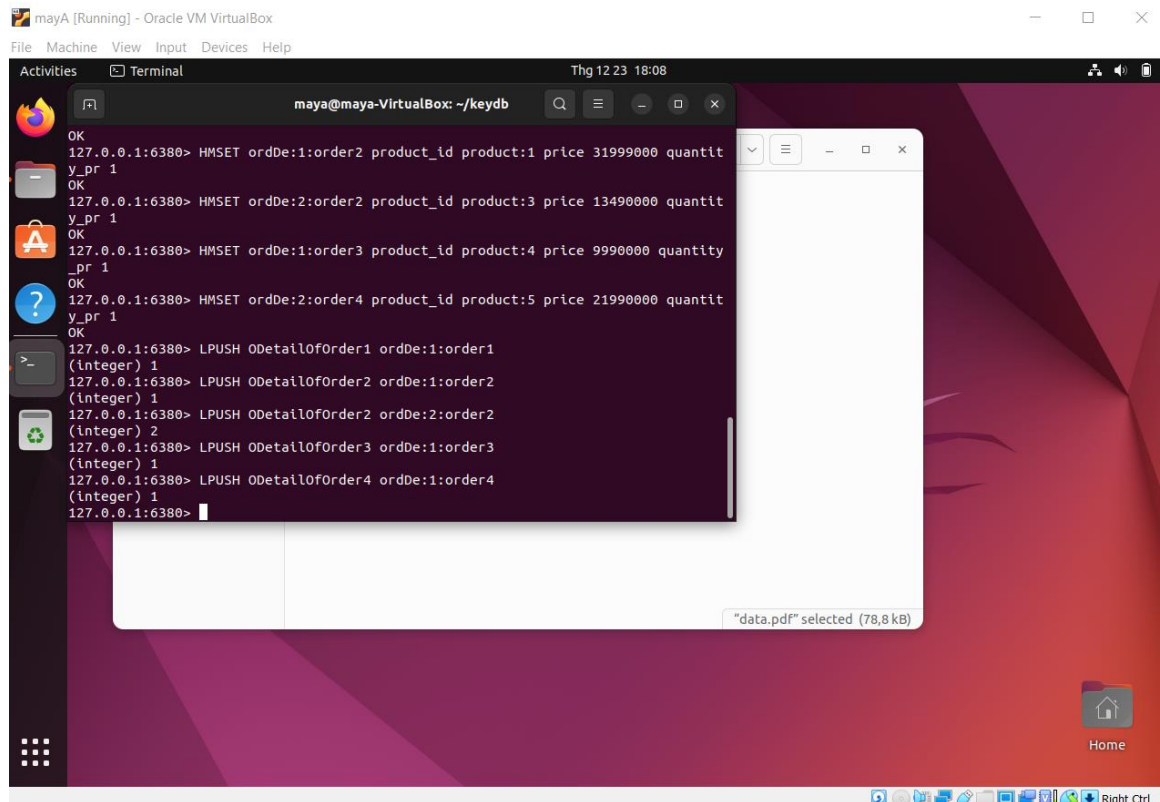
**FIGURE 39 THÊM TỪNG ORDER VÀO LIST TƯƠNG ỨNG**

- Dữ liệu `DETAIL_ORDER`: sẽ lưu các dữ liệu của `detail_order` kèm theo `order` để nhận biết đó là detail của order nào:

**FIGURE 40 THÊM DỮ LIỆU CATEGORY Ở MÁY A**

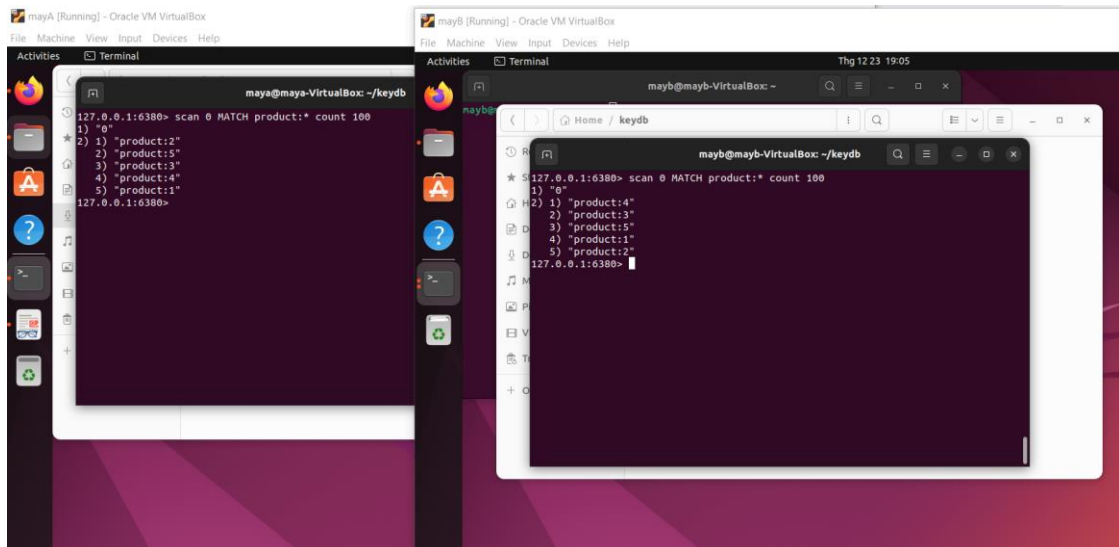
- Sau đó sẽ đưa từng `detail_order` của từng `order` vào các list để dễ tìm kiếm:



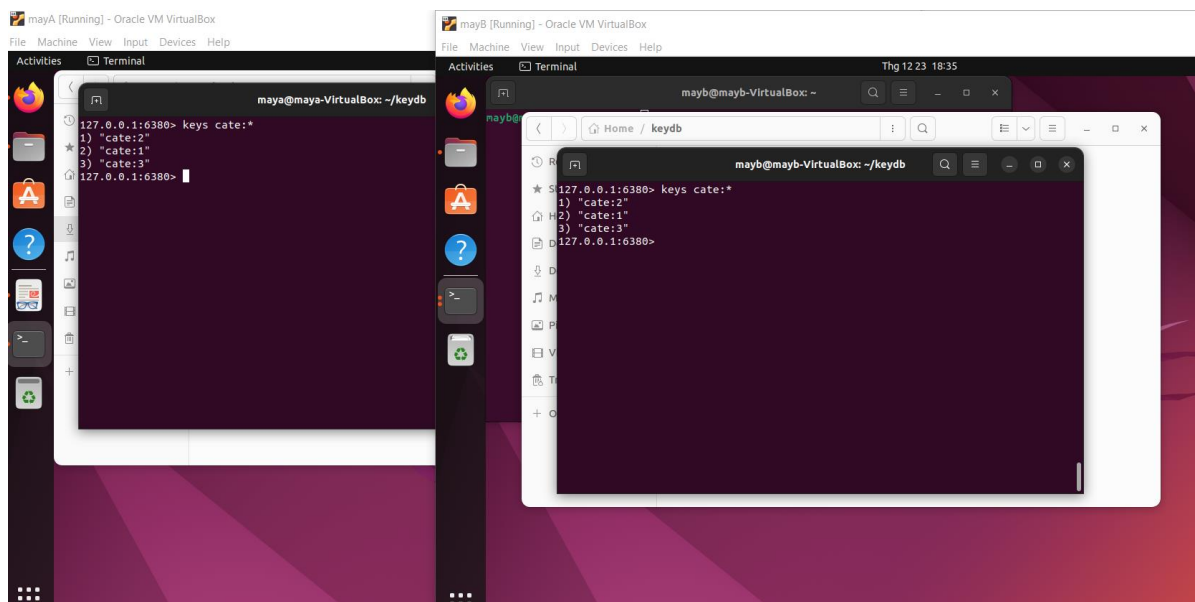
**FIGURE 41 THÊM DỮ LIỆU DETAIL\_ORDER**

Tất cả dữ liệu trên đều có thể được truy xuất từ 2 máy:

- Ví dụ dữ liệu product:

**FIGURE 42 LẤY DỮ LIỆU PRODUCTS TỪ CẢ 2 MÁY**

- Dữ liệu category:

**FIGURE 43 LẤY DỮ LIỆU CATEGORY TỪ CẢ 2 MÁY**

- Tương tự với tất cả dữ liệu còn lại:

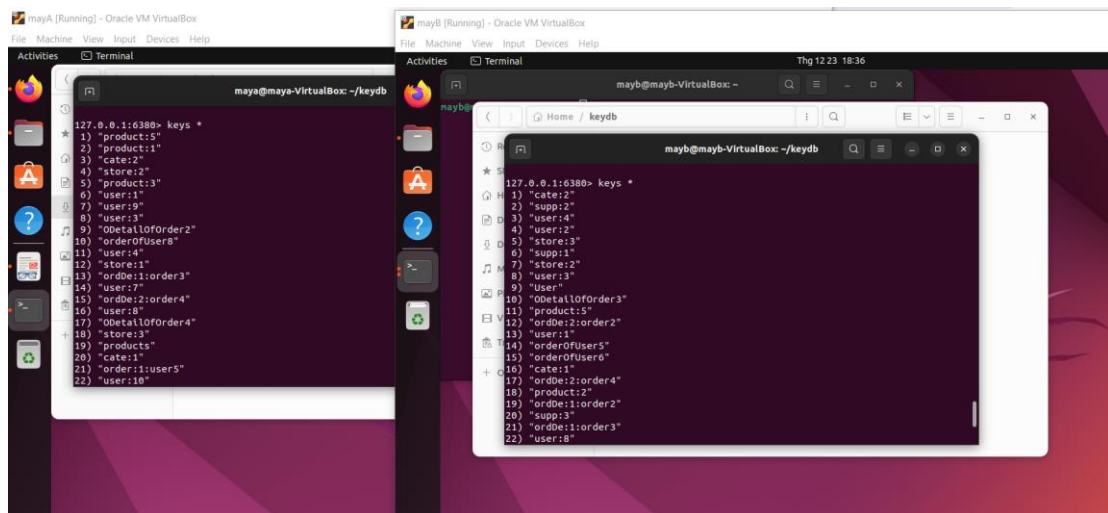


FIGURE 44 LẤY TẤT CẢ CÁC DỮ LIỆU CÒN LẠI (1)

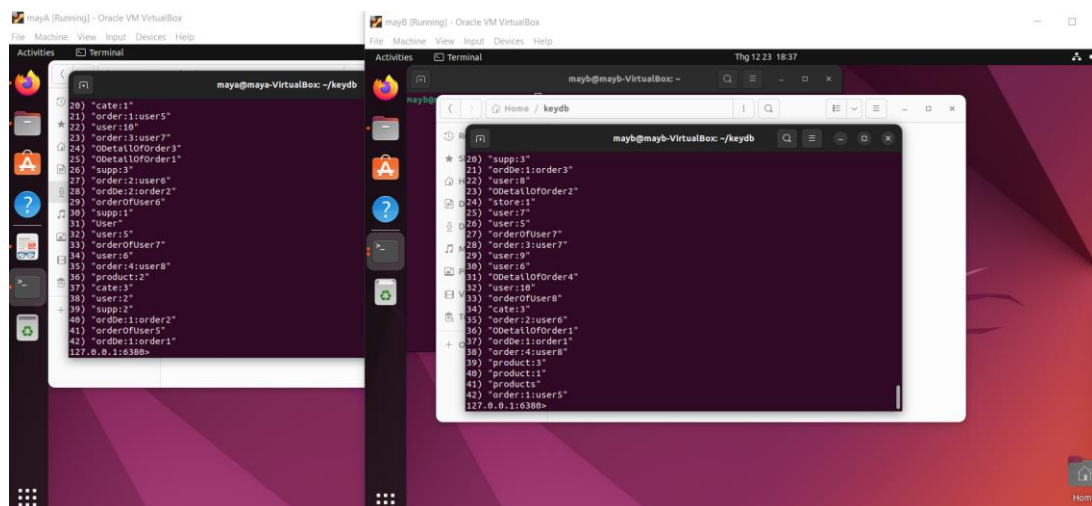
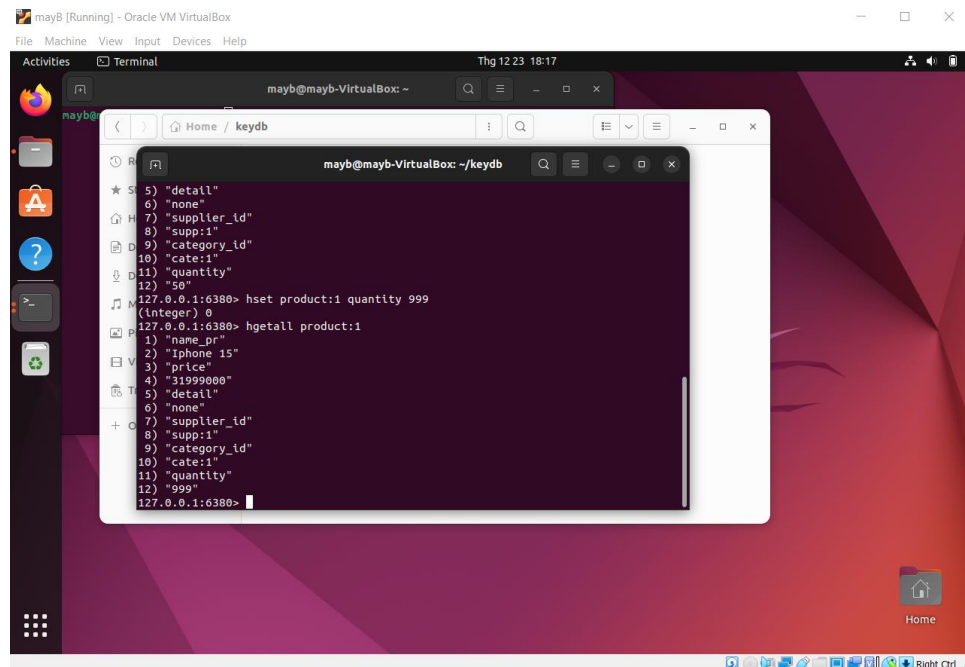


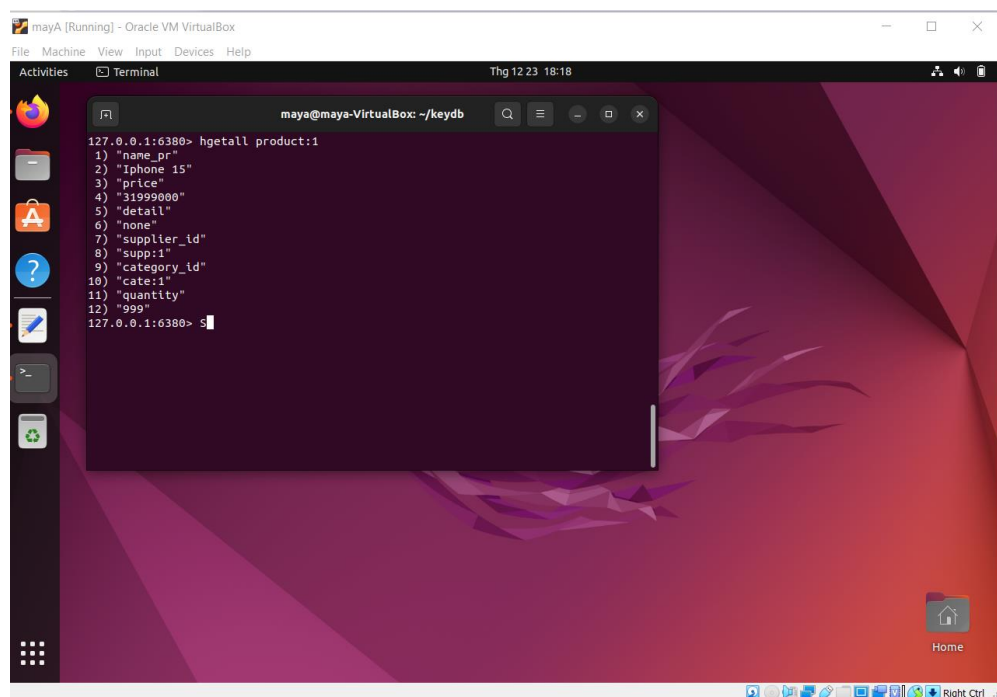
FIGURE 45 LẤY TẤT CẢ CÁC DỮ LIỆU CÒN LẠI (2)

### 3.2 Cập nhật dữ liệu:

Ví dụ cập nhật trường quantity của product:1 thành 999 tại máy B:

**FIGURE 46 CẬP NHẬP TRƯỜNG QUANTITY CỦA PRODUCT:1**

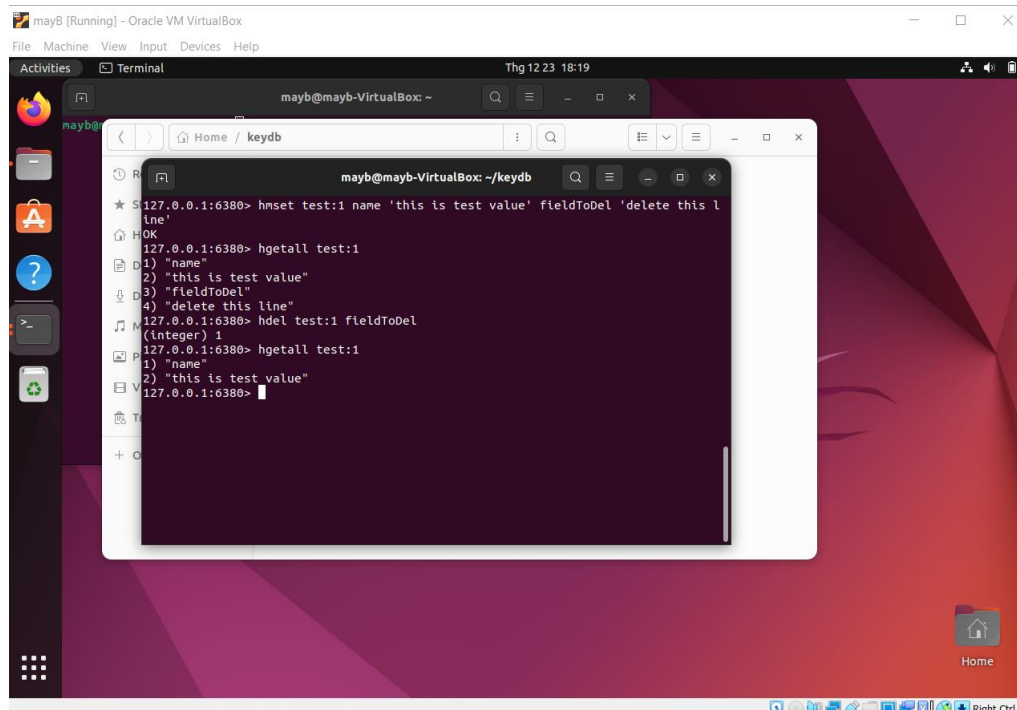
- Tại máy A khi đó:

**FIGURE 47 DỮ LIỆU SAU CHỈNH SỬA**

### 3.3 Xóa dữ liệu:

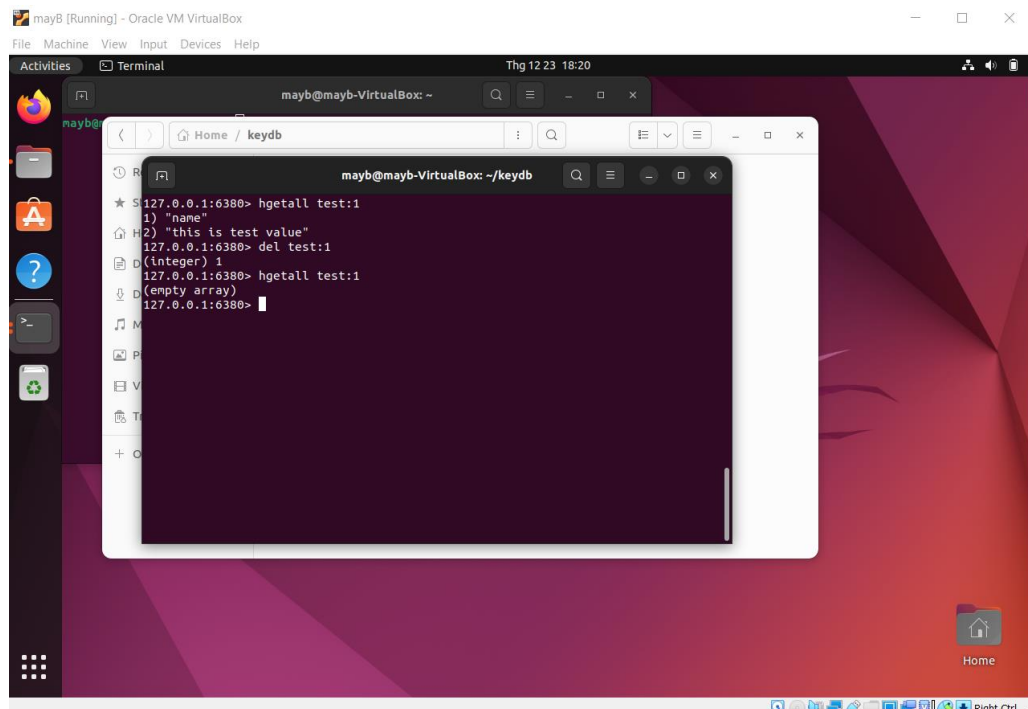
Trong keydb ta có thể xóa 1 dòng dữ liệu hoặc cả 1 hset như sau:

- Xóa 1 dòng dữ liệu 'fieldToDel' trong hset test:1

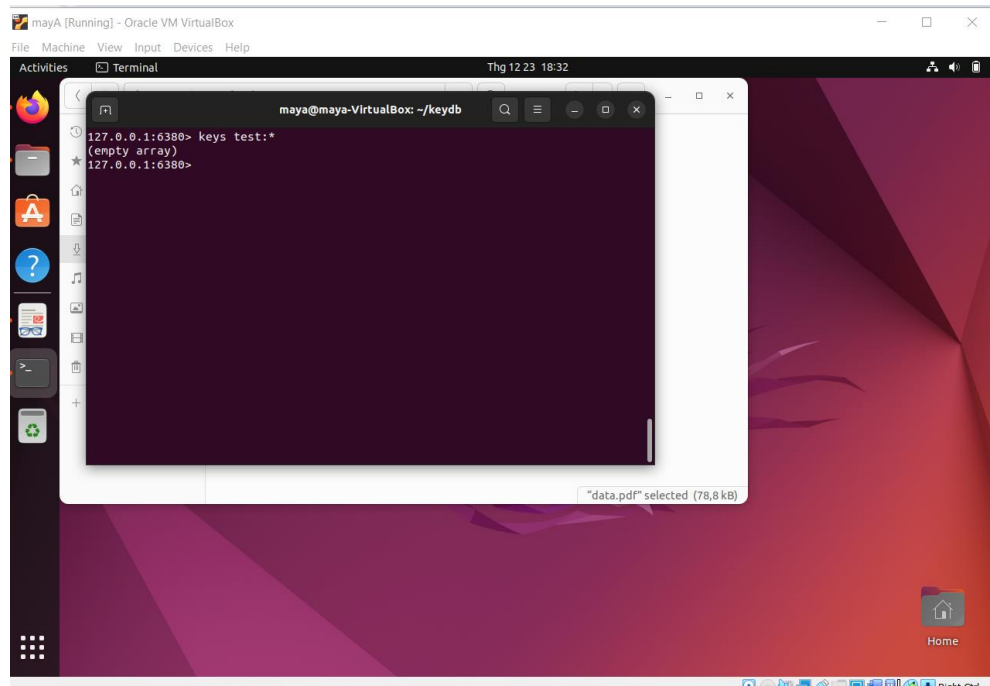


**FIGURE 48 XÓA 1 DÒNG DỮ LIỆU**

- Xóa 1 hset tại máy B:

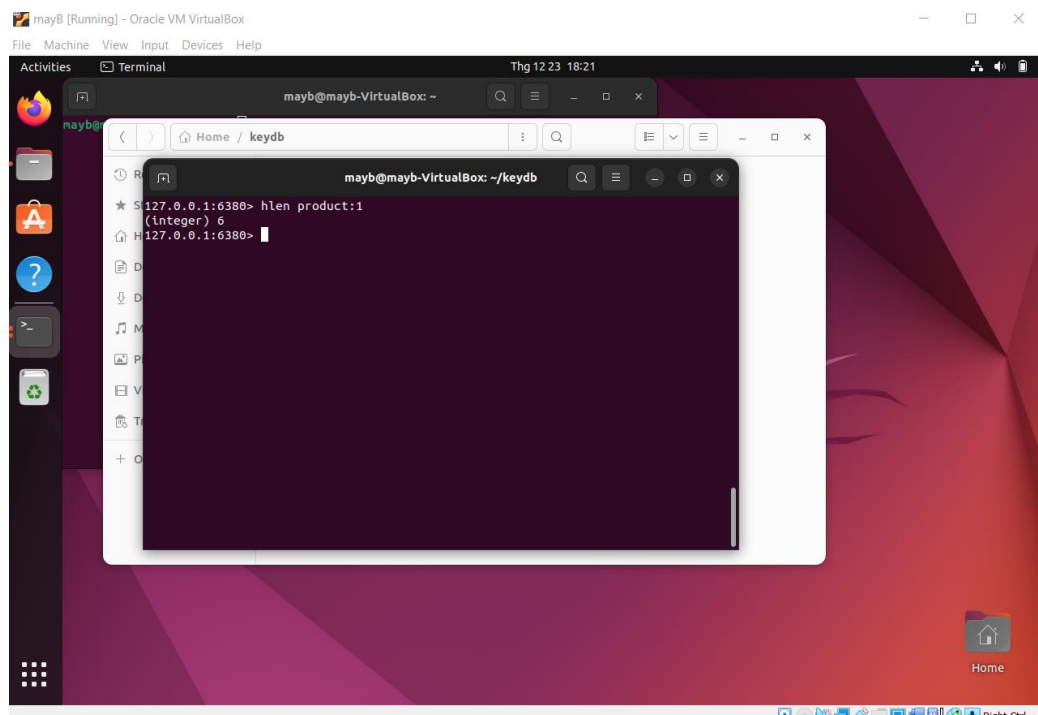
**FIGURE 49 XOÁ 1 HSET**

- Khi đó tại máy A cũng đã không còn hset test:

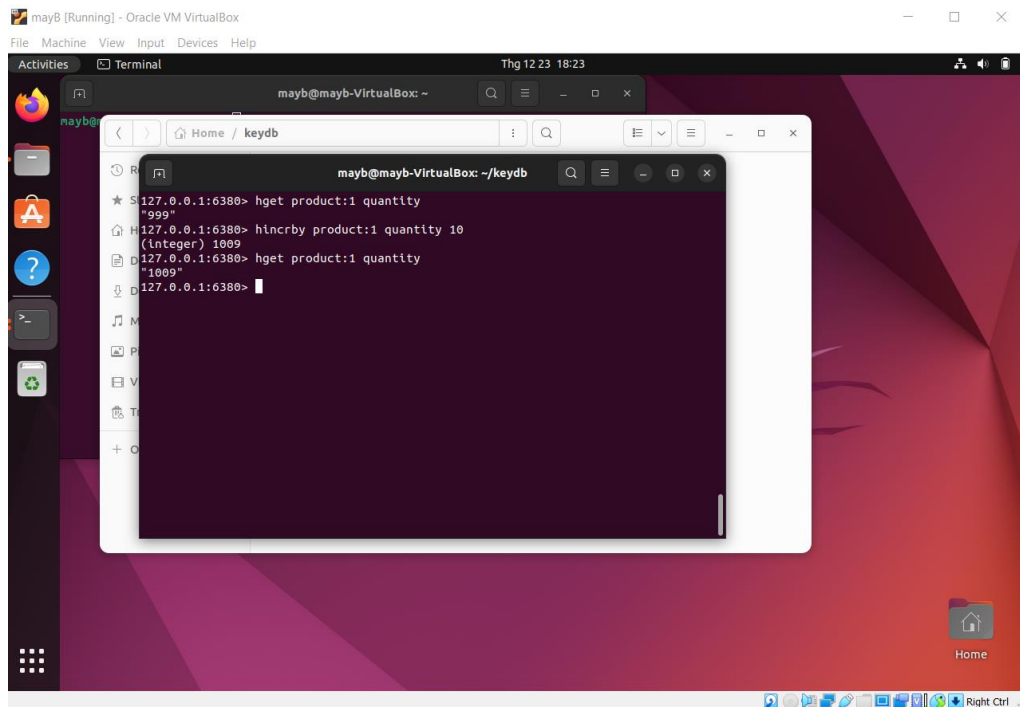
**FIGURE 50 KẾT QUẢ SAU KHI XOÁ**

### 3.4 Một số thao tác khác trên các dữ liệu trong keydb:

- Đếm số trường của 1 hset sử dụng hlen:

**FIGURE 51 ĐẾM SỐ TRƯỜNG CỦA HSET PRODUCT:1**

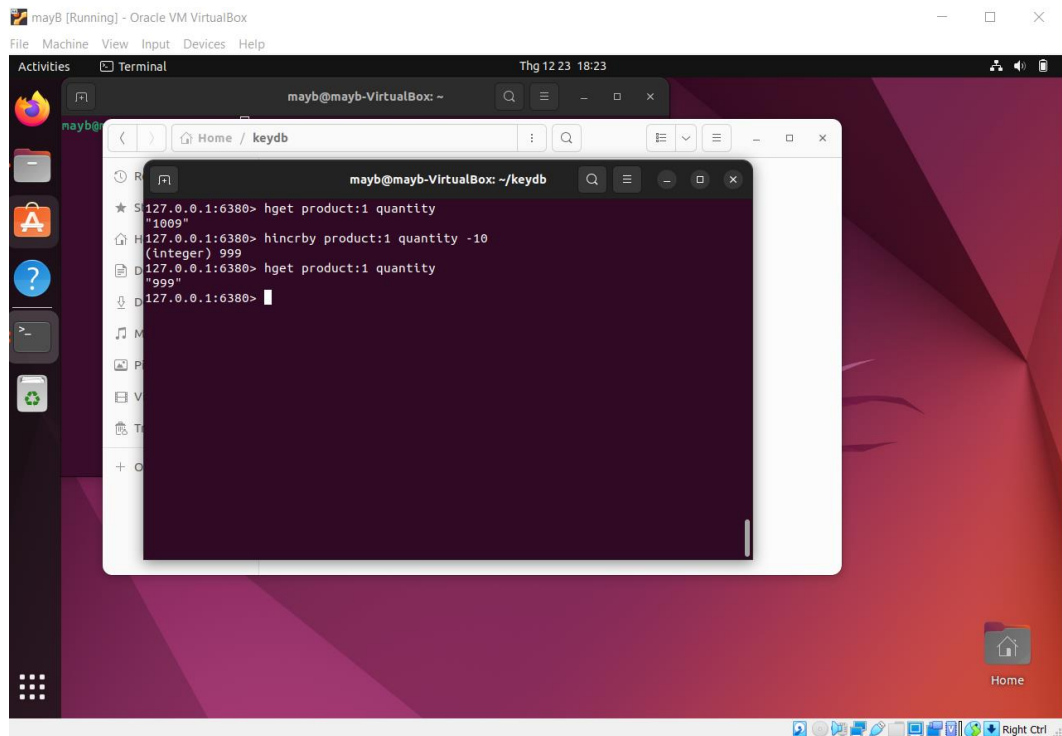
- Nếu dòng dữ liệu đó là kiểu integer thì có thể dùng hàm HINCRBY để tăng hoặc giảm giá trị:
  - Dùng HINCRBY để tăng giá trị:



**FIGURE 52 TĂNG GIÁ TRỊ CỦA TRƯỜNG QUANTITY CỦA HSET PRODUCT:1**

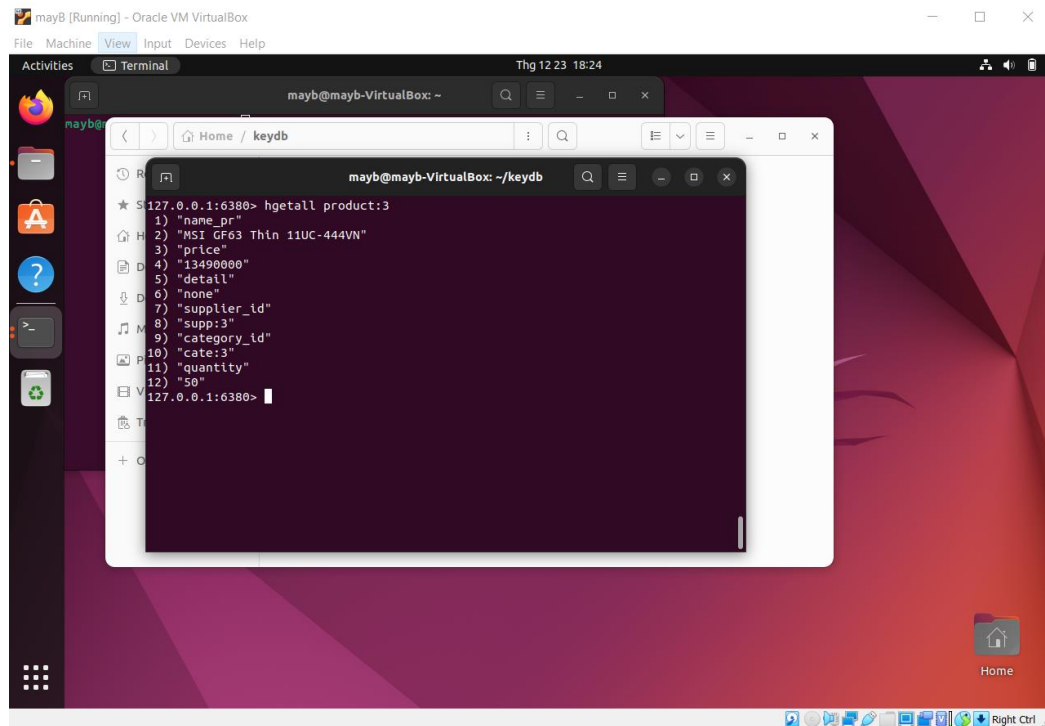
- Để giảm thì chỉ cần ghi số âm:





**FIGURE 53 GIẢM GIÁ TRỊ CỦA TRƯỜNG QUANTITY CỦA HSET PRODUCT:1**

- Sau khi có thể dùng hàm HGETALL để lấy giá trị cụ thể của 1 HSET:

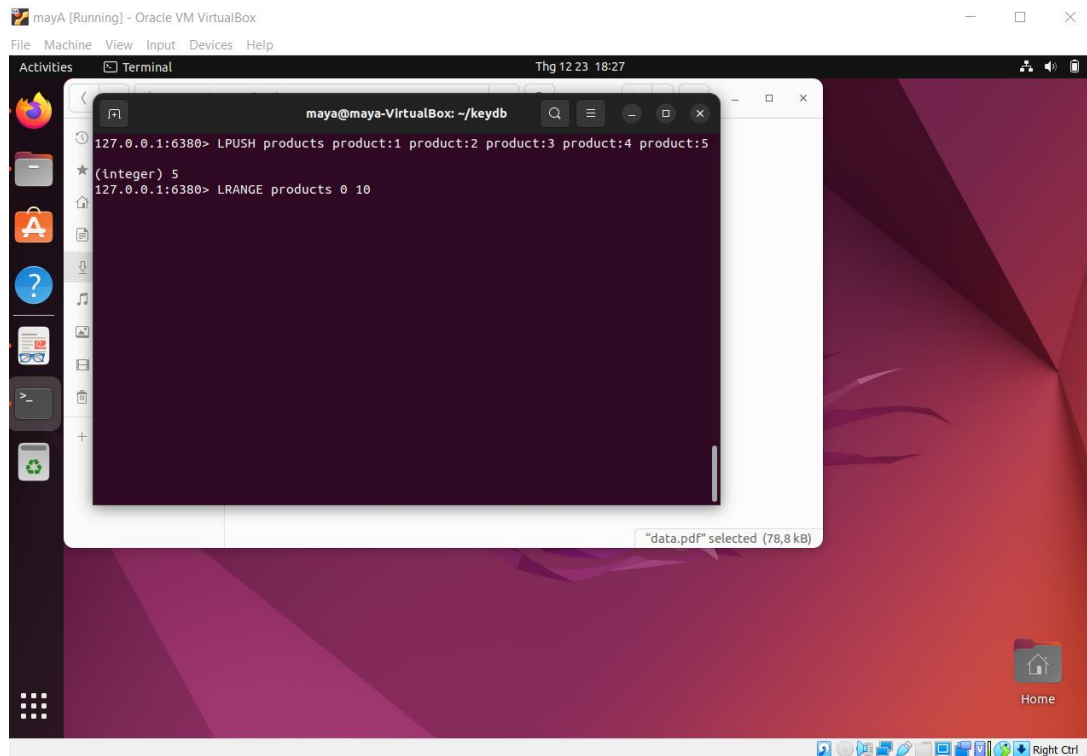


The screenshot shows a terminal window titled 'mayb@mayb-VirtualBox: ~' with a file explorer in the background. The terminal displays the command 'hgetall product:3' and its output, which is a list of 12 items representing a product's details.

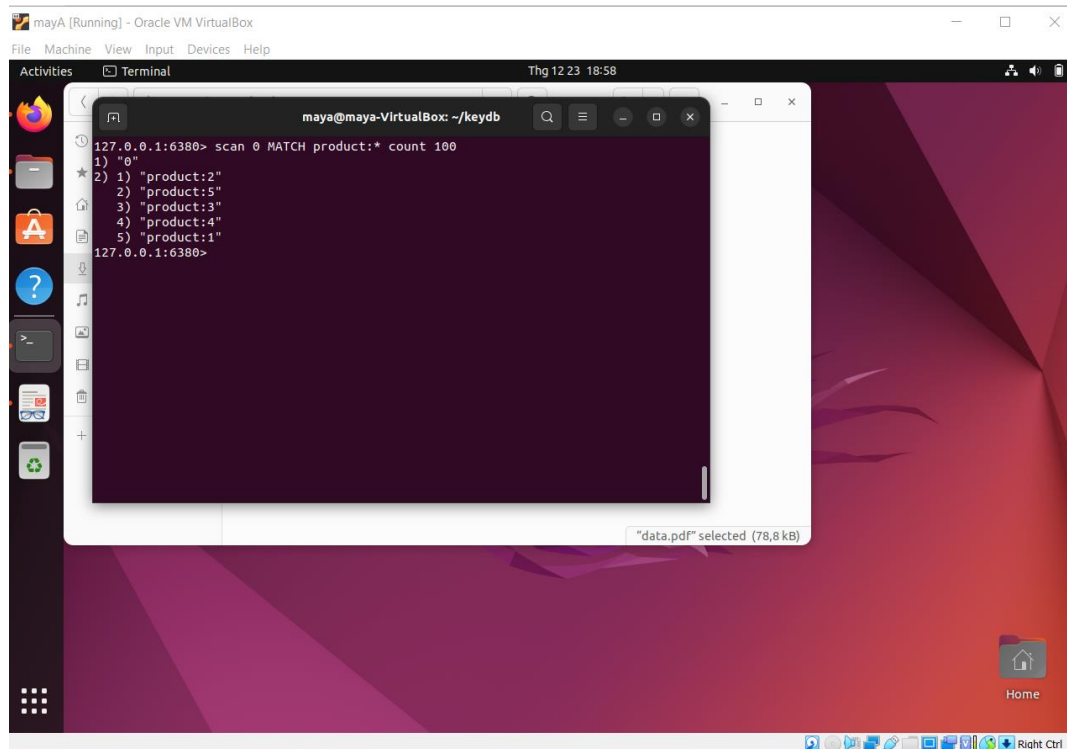
```
127.0.0.1:6380> hgetall product:3
1) "name_pr"
2) "MSI Gf63 Thin 11UC-444VN"
3) "price"
4) "13490000"
5) "detail"
6) "none"
7) "supplier_id"
8) "supp:3"
9) "category_id"
10) "cate:3"
11) "quantity"
12) "50"
127.0.0.1:6380>
```

**FIGURE 54 DÙNG HÀM HGETALL ĐỂ LẤY THÔNG TIN 1 HSET**

- Có thể dùng LPUSH hoặc RPUSH để push các giá trị vào 1 list và dùng LRange để hiển thị dữ liệu trong 1 list:

**FIGURE 55 SỬ DỤNG LPUSH VÀ LRANGE**

- Có thể dùng SCAN để tìm kiếm các key tương tự như lệnh keys. Nhưng tại sao nên dùng SCAN ? Vì SCAN được thiết kế để duyệt qua các key trong một cơ sở dữ liệu mà không ảnh hưởng đến hiệu suất của máy chủ. Còn việc sử dụng KEYS có thể ảnh hưởng đến hiệu suất nếu có nhiều key. Ví dụ dùng SCAN để lấy tất cả key product:



**FIGURE 56 SỬ DỤNG SCAN ĐỂ LẤY THÔNG TIN**

## NGUỒN THAM KHẢO

KeyDB. (n.d.). *Keydb documentation & learning center*. Retrieved from <https://docs.keydb.dev/>