

**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**---------oOo---------**

**BÁO CÁO TIỂU LUẬN**



**CƠ SỞ DỮ LIỆU PHÂN TÁN**

**ĐỀ TÀI: TÌM HIỂU CƠ CHẾ PHÂN TÁN TRONG HQT CSDL NOSQL - HADOOP/HBASE**

***Giảng viên hướng dẫn:***

Ths. Nguyễn Hồ Duy Tri

**Nhóm:** 3

***Sinh viên thực hiện***

Đoàn Ngọc Tuấn - 21521623

Doãn Công Trí - 21520492

Trần Quốc Hưng - 21522127

Trần Lê Tứ - 21522746

***Thành phố Hồ Chí Minh, ngày 25 tháng 12 năm 2023***

**LỜI CẢM ƠN**

Lời đầu tiên nhóm em xin cảm ơn tập thể cán bộ, giảng viên trường Đại học Công Nghệ Thông Tin – ĐHQG Tp. Hồ Chí Minh đã tạo một môi trường tích cực cho sinh viên chúng em được thỏa sức sáng tạo, học tập và làm việc, cũng như trau dồi kiến thức cho chính bản thân mình.

Và đặc biệt, nhóm xin gửi lời cảm ơn chân thành và sự tri ân sâu sắc đối với thầy Nguyễn Hồ Duy Tri - giảng viên lý thuyết và thực hành môn Cơ sở dữ liệu phân tán. Cảm ơn vì sự nhiệt tình, chu đáo, tận tâm của thầy trong công tác giảng dạy cũng như hỗ trợ nhóm em trong quá trình thực hiện đồ án môn học này.

Ngoài ra không thể không cảm ơn tập thể lớp IS211.O12 nói chung và những thành viên trong nhóm nói riêng đã có những đóng góp, ý kiến để nhóm có thể cải thiện chất lượng dự án. Cảm ơn vì các bạn đã đồng hành cùng chúng mình trong suốt quá trình thực hiện đồ án môn học.

Trong quá trình thực tập, cũng như là trong quá trình làm đồ án môn học, khó tránh khỏi sai sót, rất mong thầy bỏ qua. Đồng thời do trình độ lý luận cũng như kinh nghiệm thực tiễn còn hạn chế nên bài báo cáo không thể tránh khỏi những thiếu sót, nhóm em rất mong nhận được ý kiến đóng góp từ thầy để nhóm em học thêm được nhiều kinh nghiệm và sẽ hoàn thành tốt hơn những đồ án khác trong tương lai.  
Em xin chân thành cảm ơn!

Nhóm sinh viên thực hiện

Nhóm 3

**NHẬN XÉT CỦA GIẢNG VIÊN**

...............................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................

# **MỤC LỤC**

[**MỤC LỤC 4**](#_l84n0alw4g7c)

[**TÓM TẮT NỘI DUNG ĐỀ TÀI 5**](#_29xr5eh4vlni)

[**Chương 1: GIỚI THIỆU 6**](#_jj3klscf8uu0)

[1.1. TỔNG QUAN VỀ HỆ QUẢN TRỊ CSDL NOSQL 6](#_6ucpkr1z0ajr)

[1.2. HỆ QUẢN TRỊ CSDL HBASE 8](#_48dtqpwuuddp)

[1.2.1. Tổng quan về HBase 8](#_bevpt02wau6k)

[1.2.1.1 Hệ quản trị cơ sở dữ liệu Apache HBase 8](#_626kndo42h9)

[1.2.1.2 Lịch sử hình thành và tổ chức quản lý 9](#_bb28snynefkx)

[1.2.1.3 Kiến trúc HBase 10](#_5lo4be15ih41)

[1.2.1.4 Data Flow trong HBase 12](#_652paevkyhiw)

[1.2.3 Mô hình lưu trữ 13](#_dsaobkcdac4x)

[1.2.5. Ngôn ngữ thao tác với dữ liệu 14](#_30hkgvc5e9r0)

[1.2.6. Cơ chế phân tán 15](#_84k7t4g7hp7i)

[**Chương 2: HƯỚNG DẪN CÀI ĐẶT 16**](#_m6bxru9v0a8r)

[2.1. YÊU CẦU CÀI ĐẶT 16](#_w85khbueiel)

[2.2. CÁC BƯỚC CÀI ĐẶT 16](#_bkuwcz9b0z0h)

[2.2.1 Cấu hình máy ảo & cài đặt Hadoop 16](#_fb661itftx8g)

[2.2.2 Cài đặt Hadoop 21](#_egvi39q498p0)

[2.2.3 Cài đặt Zookeeper 40](#_szxgq12g3wak)

[2.2.4 Cài đặt HBase 46](#_rsmygnrqa871)

[**Chương 3: THỰC NGHIỆM MÔ PHỎNG PHÂN TÁN 53**](#_ww7lxh7tpryg)

[3.1. MÔ TẢ BÀI TOÁN ĐẶT RA VỚI DỮ LIỆU 53](#_vyqbmdfhdd4w)

[3.2. MÔ TẢ CẤU TRÚC DỮ LIỆU SỬ DỤNG 55](#_dqo4db6lebp)

[3.3. CÁC BƯỚC THỰC NGHIỆM 59](#_az8tj2n67mv4)

[3.3.1 Tạo Table và các column family 59](#_45jfvxd)

[3.3.2 Thêm dữ liệu 61](#_2koq656)

[3.3.3 Cập nhật dữ liệu 63](#_zu0gcz)

[3.3.4 Xóa dữ liệu 64](#_3jtnz0s)

[3.3.5 Các thao tác với dữ liệu khác 64](#_1yyy98l)

[**TÀI LIỆU THAM KHẢO 68**](#_o51pl2k6vmxf)

# **TÓM TẮT NỘI DUNG ĐỀ TÀI**

NoSQL (Non-relational Data Management System or Not Only SQL) is becoming increasingly popular and widely used in information technology applications. NoSQL systems provide flexible data storage and retrieval capabilities, suitable for applications with high scalability and performance requirements. One of these systems is HBase, a widely used database management system, especially in the field of Big Data.

HBase is a database management system based on Hadoop, an open-source project under Apache, developed and expanded from Google's Big Data storage project. HBase is written in Java and can store extremely large amounts of data, from terabytes to petabytes. The HBase prototype was created as a Hadoop contribution in February 2007. From October 2007 to September 2009, versions 0.81.1, 0.19.0, and 0.20.0 were released successively.

With outstanding features such as fast data filtering, storage of Big Data, the ability to store billions of rows and columns, real-time data querying, REST protocol support, and consistent data read and write mechanisms based on Hadoop, HBase is supported for various languages such as Java, PHP, and Python, etc.

The architecture of HBase consists of four basic components: HMaster, which is the central component in the HBase architecture, responsible for monitoring all RegionServers; HRegionServer, which directly manages the HRegions; HRegions, the fundamental architectural component of the HBase cluster, consisting of Memstore and Hfile; and Zookeeper, the monitoring center and configuration information storage.

HBase is a column-oriented database, and its tables are row-oriented. The table schema specifies column families, which are key-value pairs. A table can have multiple column families, and each column family can have any number of columns. The next column values are stored continuously on disk. Each cell of the table has its own metadata, such as a timestamp and other information.

The distributed mechanism of Apache's HBase follows a master-slave architecture. With this mechanism, ideally, the master receives all requests, and the actual work is performed by the slaves. However, in reality, for reading and writing data, HBase clients will directly communicate with specific Region Servers (slaves) responsible for handling row keys for all data operations. The master is only used by clients for table creation, modification, and deletion (HBaseAdmin).

With these advantages, HBase is used by technology companies worldwide on a large scale.

# **Chương 1: GIỚI THIỆU**

## **1.1. TỔNG QUAN VỀ HỆ QUẢN TRỊ CSDL NOSQL**

Cơ sở dữ liệu NoSQL (Non-relational Data Management System – Not Only SQL) là cơ sở dữ liệu không phải dạng bảng và lưu trữ dữ liệu khác với các bảng quan hệ. Cơ sở dữ liệu NoSQL có nhiều loại dựa trên mô hình dữ liệu của chúng. Các loại chính là document, key-value, column và graph. Chúng cung cấp các lược đồ linh hoạt và mở rộng quy mô một cách dễ dàng với lượng lớn dữ liệu và lượng người dùng tải cao.

* Cơ sở dữ liệu NoSQL xuất hiện vào cuối những năm 2000 khi chi phí lưu trữ giảm đáng kể. Đã qua rồi cái thời cần tạo ra một mô hình dữ liệu phức tạp, khó quản lý để tránh trùng lặp dữ liệu. Các nhà phát triển (thay vì lưu trữ) đang trở thành chi phí chính của việc phát triển phần mềm, do đó, cơ sở dữ liệu NoSQL được tối ưu hóa cho năng suất của nhà phát triển.
* Khi chi phí lưu trữ giảm nhanh chóng, lượng dữ liệu mà các ứng dụng cần để lưu trữ và truy vấn tăng lên. Dữ liệu này có đủ hình dạng và kích thước – có cấu trúc, bán cấu trúc và đa hình – và việc xác định trước lược đồ trở nên gần như không thể. Cơ sở dữ liệu NoSQL cho phép các nhà phát triển lưu trữ một lượng lớn dữ liệu phi cấu trúc, mang lại cho chúng rất nhiều tính linh hoạt.
* NoSQL đặc biệt nhấn mạnh đến mô hình lưu trữ cặp key - value và hệ thống lưu trữ phân tán:
* Phi quan hệ (Non-relational): relational là thuật ngữ sử dụng đến các mối quan hệ giữa các bảng trong cơ sở dữ liệu quan hệ (Relational Database Management System) sử dụng mô hình gồm 2 loại khóa: khóa chính (primary key) và khóa phụ (foreign key) để ràng buộc dữ liệu nhằm thể hiện tính nhất quán dữ liệu từ các bảng khác nhau. Non-relational là khái niệm không sử dụng các ràng buộc dữ liệu cho tính nhất quán dữ liệu.
* Lưu trữ dữ liệu phân tán.
* Triển khai đơn giản, dễ nâng cấp và mở rộng.
* Mô hình dữ liệu và truy vấn linh hoạt.
* Một số đặc điểm nhận dạng cho thế hệ CSDL mới này bao gồm: schema-free, hỗ trợ mở rộng dễ dàng, API đơn giản, nhất quán cuối (eventual consistency), không giới hạn không gian dữ liệu,...
* Có nhiều cách phân loại các cơ sở dữ liệu NoSQL khác nhau, mỗi loại với các loại và loại con khác nhau, một số trong số đó có thể chồng chéo lên nhau. Một phân loại cơ bản dựa trên mô hình dữ liệu, với các ví dụ:
* Column: HBase, Accumulo, Cassandra, Druid, Vertica
* Document: Apache CouchDB, Clusterpoint, Couchbase, DocumentDB, HyperDex, Lotus Notes, MarkLogic, MongoDB, OrientDB, Qizx, RethinkDB
* Key-value: Aerospike, CouchDB, Dynamo, FairCom c-treeACE, FoundationDB, HyperDex, MemcacheDB, MUMPS, Oracle NoSQL Database, OrientDB, Redis, Riak, Berkeley DB.
* Graph: AllegroGraph, InfiniteGraph, MarkLogic, Neo4J, OrientDB, Virtuoso, Stardog.
* Multi-model: Alchemy Database, ArangoDB, CortexDB, FoundationDB, MarkLogic, OrientDB.
* Cơ sở dữ liệu NoSQL được sử dụng trong hầu hết mọi ngành. Các trường hợp sử dụng bao gồm từ mức độ quan trọng cao (ví dụ: lưu trữ dữ liệu tài chính và hồ sơ chăm sóc sức khỏe) đến thú vị hơn và phù phiếm hơn (ví dụ: lưu trữ các kết quả đọc IoT từ hộp vệ sinh mèo thông minh).Khi quyết định sử dụng cơ sở dữ liệu nào, những người ra quyết định thường tìm thấy một hoặc nhiều yếu tố sau đây dẫn họ đến việc chọn cơ sở dữ liệu NoSQL:
* Phát triển Agile tốc độ nhanh
* Lưu trữ dữ liệu có cấu trúc và bán cấu trúc
* Khối lượng dữ liệu khổng lồ
* Yêu cầu đối với kiến ​​trúc quy mô

## **1.2. HỆ QUẢN TRỊ CSDL HBASE**

### **1.2.1. Tổng quan về HBase**

#### **1.2.1.1 Hệ quản trị cơ sở dữ liệu Apache HBase**

Hbase là hệ quản trị cơ sở dữ liệu dựa trên Hadoop, đây là mã nguồn mở nằm trong dự án của Apache, phát triển và mở rộng từ dự án lưu trữ Big Data của google. (được xây dựng dựa trên Google Bigtable). Hbase được viết bằng ngôn ngữ Java có thể lưu trữ dữ liệu cực lớn từ terabytes đến petabytes.

HBase thực chất là một NoSQL điển hình nên vì thế các table của HBase không có một schemas cố định nào và cũng không có mối quan hệ giữa các bảng. Hiện nay, có rất nhiều công ty và tập đoàn công nghệ lớn trên thế giới sử dụng HBase, có thể kể đến: Facebook, Twitter, Yahoo, Adobe….

**Các tính năng của Hbase**

* Thời gian lọc dữ liệu nhanh
* Lưu trữ dữ liệu Big-Data, có thể lưu trữ hàng tỷ rows và columns
* Có độ ổn định và giảm thiểu rủi ro (failover) khi lưu một lượng lớn dữ liệu.
* Truy vấn dữ liệu theo thời gian thực
* Cung cấp giao thức REST, giúp trả về dữ liệu theo các định dạng khác nhau như plain text, json, xml. Nhờ đó chúng ta có thể khai thác dữ liệu không cần qua API từ phần mềm thứ 3.
* Nhất quán cơ chế đọc và ghi dữ liệu dựa trên Hadoop
* Nhiều extension hỗ trợ Hbase cho nhiều ngôn ngữ như Java, PHP, Python…
* Lưu trữ dữ liệu đáng tin cậy, được các hãng công nghệ trên thế giới sử dụng trên quy mô lớn.

#### **1.2.1.2 Lịch sử hình thành và tổ chức quản lý**

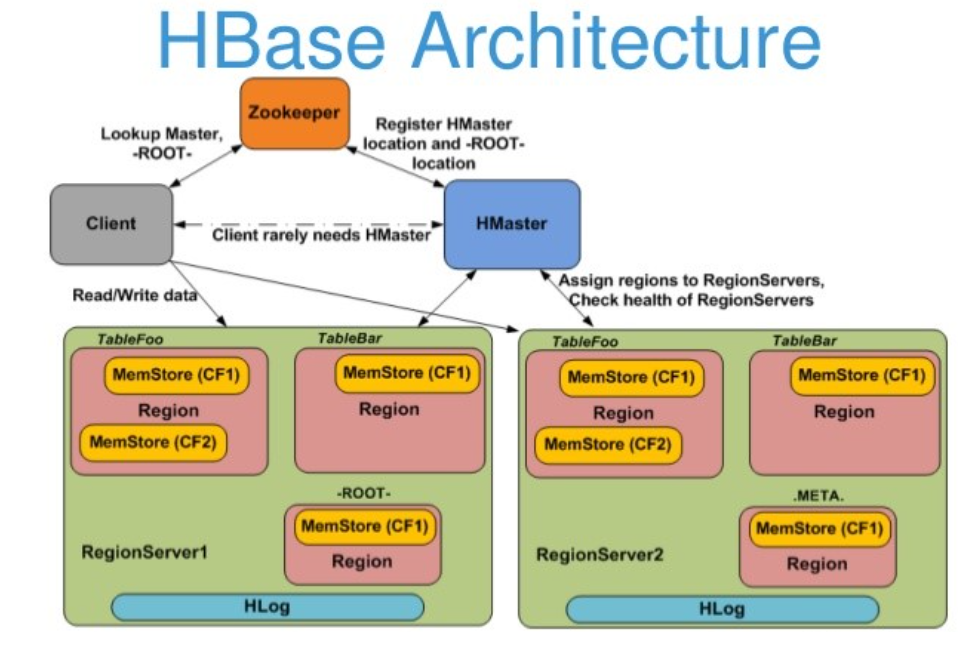
* **Năm phát hành:** 2007

Apache HBase được phát hành lần đầu tiên vào tháng 2 năm 2007. Sau đó vào tháng 1 năm 2008, HBase trở thành một dự án con của Apache Hadoop. Năm 2010, HBase trở thành dự án cấp cao nhất của Apache.

| **Năm** | **Sự kiện** |
| --- | --- |
| Tháng 11 năm 2006 | Google đã phát hành bài báo trên BigTable. |
| Tháng 2 năm 2007 | Nguyên mẫu HBase ban đầu được tạo ra như một đóng góp của Hadoop. |
| Tháng 10 năm 2007 | HBase có thể sử dụng đầu tiên cùng với Hadoop 0.15.0 đã được phát hành. |
| Tháng 1 năm 2008 | HBase trở thành dự án phụ của Hadoop. |
| Tháng 10 năm 2008 | HBase 0.18.1 đã được phát hành. |
| Tháng 1 năm 2009 | HBase 0.19.0 đã được phát hành. |
| Tháng 9 năm 2009 | HBase 0.20.0 đã được phát hành. |
| Tháng 5 năm 2010 | Hbase trở thành dự án cấp cao nhất của Apache |

* **Tổ chức quản lý:** HBase ban đầu là một dự án của công ty Powerset, một công ty về tìm kiếm và ngôn ngữ tự nhiên có trụ sở tại San Francisco. Microsoft mua lại Powerset vào năm 2008.
* **Nguyên nhân ra đời**: Apache HBase bắt đầu như một dự án của công ty Powerset do nhu cầu xử lý lượng dữ liệu khổng lồ cho mục đích tìm kiếm ngôn ngữ tự nhiên.

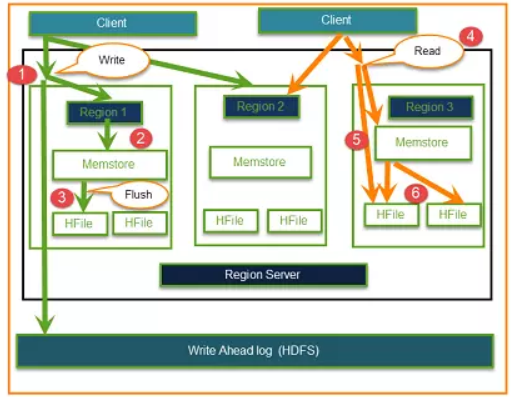
#### **1.2.1.3 Kiến trúc HBase**

****

Kiến trúc HBase bao gồm 4 thành phần cơ bản:

* **HMaster**: chính là thành phần trung tâm trong kiến trúc của Hbase, nó giám sát tất cả các RegionServer trong cụm. Trong môi trường bao gồm nhiều cụm như Hadoop thì HMaster sẽ nằm ở NameNode, mọi thay đổi liên quan đến metadata đều phải thông qua HMaster, cụ thể:
* Cung cấp quyền admin, tính toán đến các Region Servers
* Gán các Regions cho Region Servers
* HMaster cũng đảm nhận nhiệm vụ cân bằng tải hoặc xử lí lỗi ở các node con trong 1 cụm
* Những thao tác liên quan đến metadata hoặc DDL tới cơ sở dữ liệu HBase
* **HBase RegionServer / HRegionServer**: Nhận trực tiếp yêu cầu DML (read, write) từ Client mà không cần thông qua HMaster. Khi HRegionServer nhận yêu cầu từ người dùng, nó thực hiện gán yêu cầu này cho các Regions tương ứng, HRegionServer còn chứa HLog dùng để chứa mọi log files. Trong môi trường bao gồm nhiều cụm như Hadoop thì HRegionServers sẽ nằm trên các DataNode, HMaster sẽ liên lạc với HRegionServers khi có các thao tác sau:
* Quản lý các Regions
* Phân phối các Regions tự động
* Nhận các lệnh DML
* Liên lạc trực tiếp với client
* **HRegions**: là thành phần kiến trúc cơ sở của Hbase cluster, nó bao gồm 2 thành phần chính là Memstore và Hfile. Memstore giống như một bộ nhớ cache, data đi vào đầu tiên sẽ nằm ở Memstore, được sắp xếp lại và cuối cùng là được đưa vào HFile, nếu ta sử dụng Apache HBase trên một hệ thống Hadoop cluster, các HFile này sẽ được lưu trữ vào trong Hadoop Distributed File System (HDFS).
* **Zookeeper**: ZooKeeper là trung tâm điều khiển của HBase, nó mang nhiệm vụ quan trọng là duy trì những thông tin cấu hình, cung cấp cơ chế đồng bộ phân tán cho toàn cơ sở dữ liệu. Cơ chế đồng bộ phân tán là việc truy cập phân tán đến các cụm đang chạy với nhiệm vụ điều phối nhiệm vụ giữa các node một cách chính xác, tránh xảy ra lỗi. Nếu Client muốn giao tiếp với Regions thì phải thông qua ZooKeeper trước, cụ thể của ZooKeeper như sau:
* Duy trì thông tin thiết lập.
* Cung cấp cơ chế đồng bộ phân tán.
* Thiết lập kết nối giữa Client với HRegionServers.
* Kiểm tra liên tục các lỗi xảy ra với các cụm.

#### **1.2.1.4 Data Flow trong HBase**

****

**Write operations**

**Step 1:** Client muốn write data, tạo kết nối lần đầu tiên với Regions server và sau đó là regions.

**Step 2:** Regions liên lạc với memstore, lưu lại liên kết với column family.

**Step 3:** Dữ liệu trước tiên sẽ được lưu trữ tại Memstore. Tại đây dữ liệu sẽ được sắp xếp (sorted) trước khi chuyển tới HFile. Có 2 lý do chính cho việc sử dụng Memstore là :

* Hệ thống lưu trữ dữ liệu phân tán dựa trên row Key nên cần sắp xếp trước khi lưu trữ.
* Tối ưu hóa luồng ghi dữ liệu khi sử dụng kiến trúc The Log-Structured Merge Tree

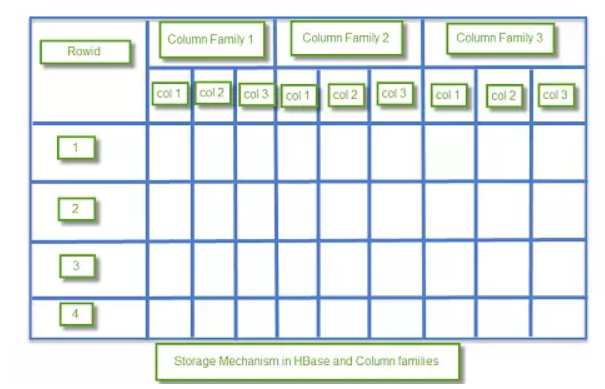
**Read operations**

**Step 4:** Client muốn đọc data từ Regions

**Step 5:** Client có thể trực tiếp truy tập tới Mem store và yêu cầu dữ liệu.

**Step 6:** Client get data từ HFile.

### **1.2.2 Mô hình lưu trữ**

HBase là một cơ sở dữ liệu theo hướng cột và dữ liệu được lưu trữ trong bảng. Các bảng được sắp xếp bởi RowId.

Các Column Family có trong lược đồ là các cặp key-value. Column Family có nhiều cột (col). Giá trị col được lưu trữ trong bộ nhớ đĩa. Mỗi ô của bảng có dữ liệu Meta riêng như timestamp và các thông tin khác.

Trong một HBase

* **Table** là một tập hợp các row
* **Rows** là tập hợp các Column family
* **Column** **family** là tập hợp các column
* **Column** là tập hợp các key - value

### **1.2.3. Ngôn ngữ thao tác với dữ liệu**

Một số lệnh thao tác cơ bản trong HBase:

* **create**: tạo bảng mới

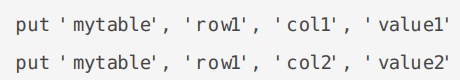
Để tạo một bảng mới với tên là **mytable** và hai cột là **col1** và **col2**, ta sử dụng cú pháp sau:



* **describe**: mô tả thuộc tính bảng
* **put**: thêm dữ liệu vào bảng

**put <'tablename'>,<'rowname'>,<'columnvalue'>,<'value'>**

Thêm dữ liệu vào bảng **mytable:**

****

Lệnh này sẽ thêm hai giá trị **value1** và **value2** vào hàng **row1** của bảng **mytable**, trong hai cột **col1** và **col2**.

* **get**: truy vấn dữ liệu từ bảng

**get <'tablename'>, <'row key'>, <'filters'>**

* **scan**: duyệt dữ liệu trong bảng

Thao tác scan được sử dụng để đọc nhiều hàng của một bảng. Nó khác với Get ở chỗ chúng ta cần chỉ định một tập hợp các hàng để đọc. Sử dụng Quét, chúng ta có thể lặp qua một dải hàng hoặc tất cả các hàng trong một bảng.



* **delete**: xóa dữ liệu khỏi bảng

Sử dụng để xóa một hàng hoặc một tập hợp các hàng khỏi bảng HBase.

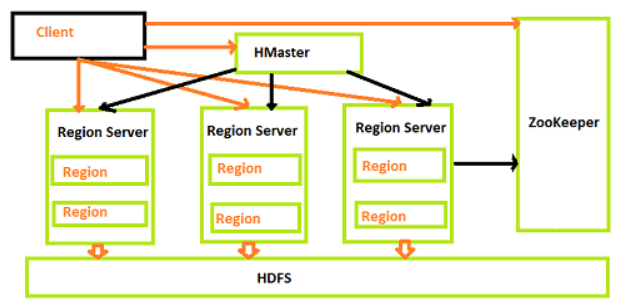


* **disable**: tạm dừng bảng
* **enable**: kích hoạt lại bảng
* **drop**: xóa bảng
* **Hbase Data Types**

Trong Apache HBase, không có khái niệm kiểu dữ liệu như vậy. Nó là một loại cơ sở dữ liệu byte-in và byte-out, trong đó, khi một giá trị được chèn vào, nó sẽ được chuyển đổi thành một mảng byte bằng giao diện **Put** và **Result**. Apache HBase sử dụng khung tuần tự hóa để chuyển đổi dữ liệu người dùng thành mảng byte.

### **1.2.4. Cơ chế phân tán**

Cơ chế phân tán của HBase của Apache tuân theo cơ chế **master – slave.** Với cơ chế này, lẽ ra master nhận tất cả các yêu cầu và công việc thực sự được thực hiện bởi các slave, nhưng trên thực tế, để đọc và ghi dữ liệu, máy khách HBase sẽ chuyển trực tiếp đến Region Server (là slave) cụ thể chịu trách nhiệm xử lý các khóa hàng cho tất cả các hoạt động dữ liệu. Master chỉ được khách hàng sử dụng cho các hoạt động tạo, sửa đổi và xóa bảng (HBaseAdmin ).



# **Chương 2: HƯỚNG DẪN CÀI ĐẶT**

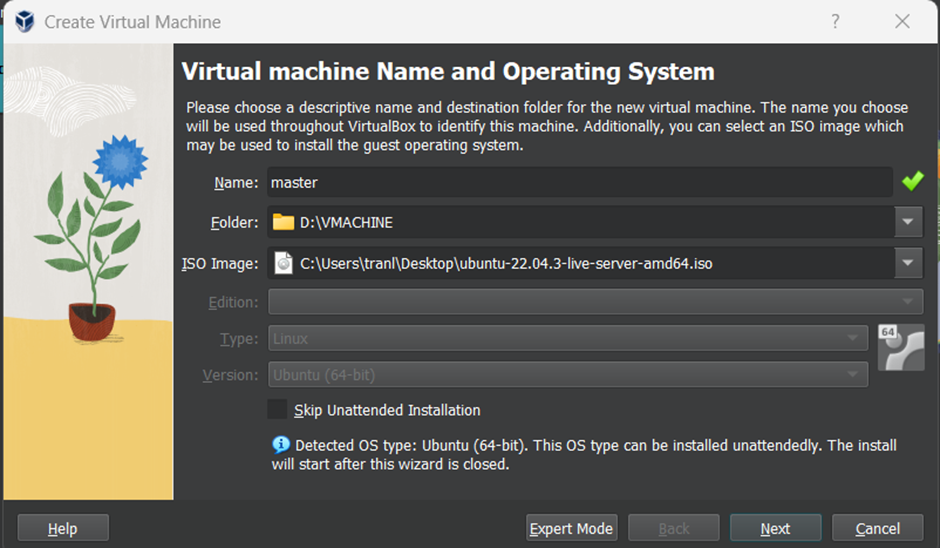
## **2.1. YÊU CẦU CÀI ĐẶT**

* Oracle VM VirtualBox
* Ubuntu Server 22.04
* Openjdk-8-jdk
* Hadoop 3.3.2
* Apache Zookeeper 3.6.3
* Apache Storm 2.1.1
* Apache HBase 2.4.0

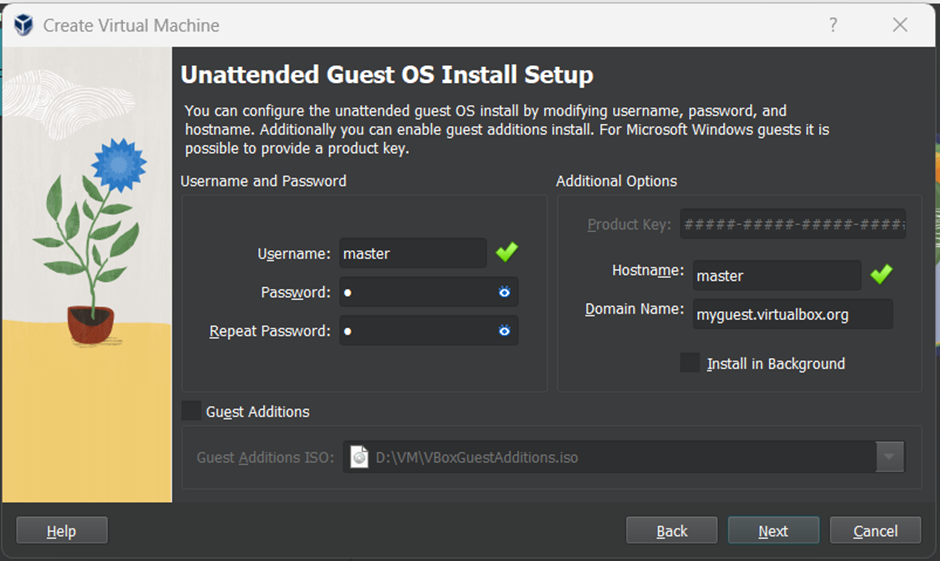
## **2.2. CÁC BƯỚC CÀI ĐẶT**

### **2.2.1 Cấu hình máy ảo & cài đặt Hadoop**

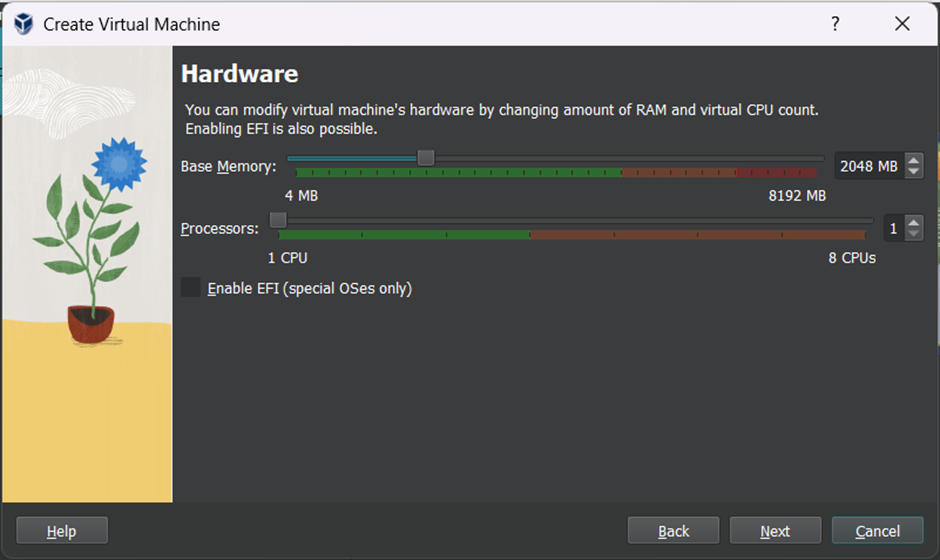
* **Tạo máy ảo**



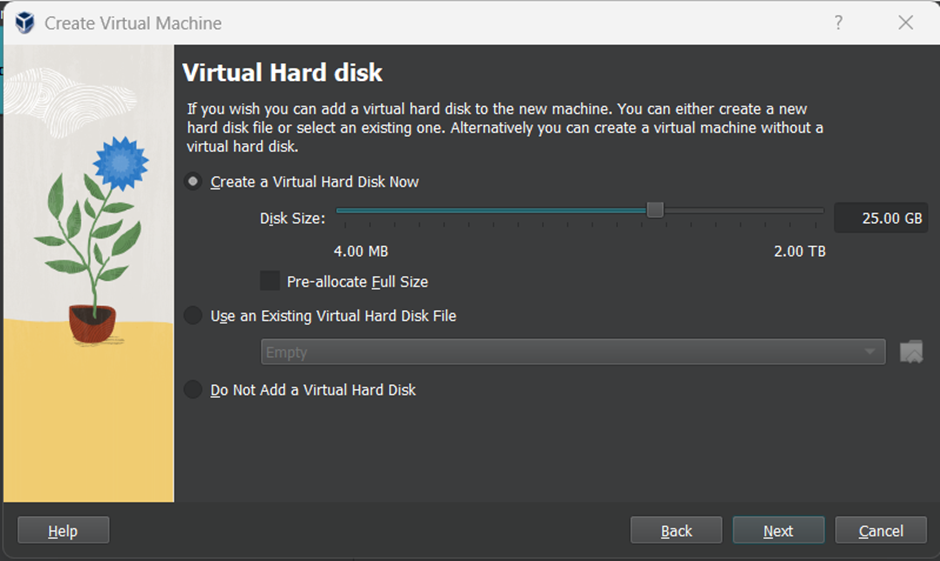
* **Tạo username & pass**



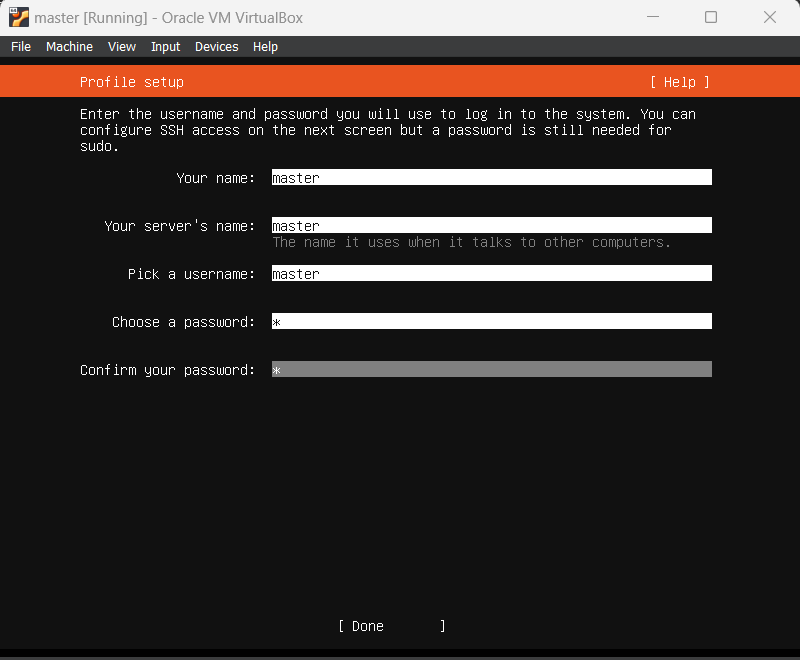
* **Cấu hình bộ nhớ cho máy ảo**



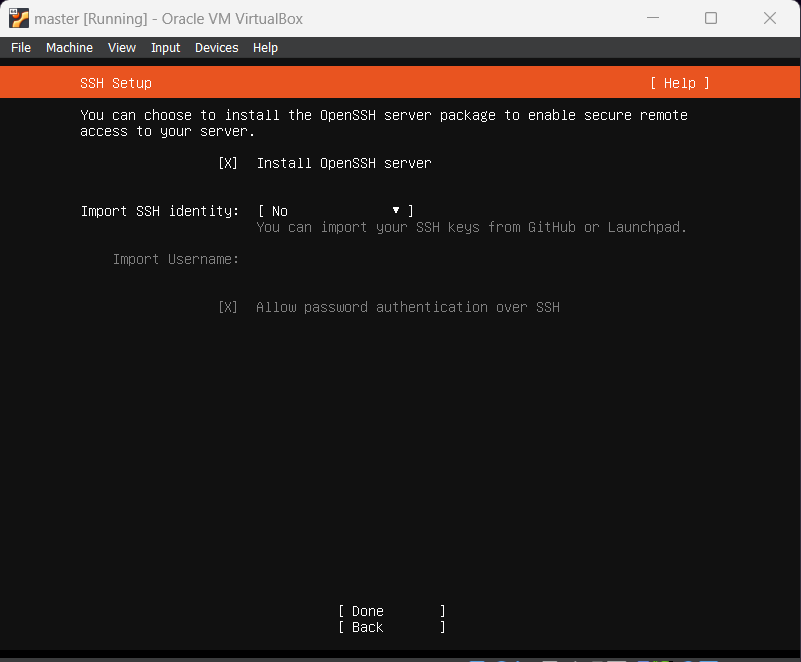
* **Cấu hình ổ cứng**



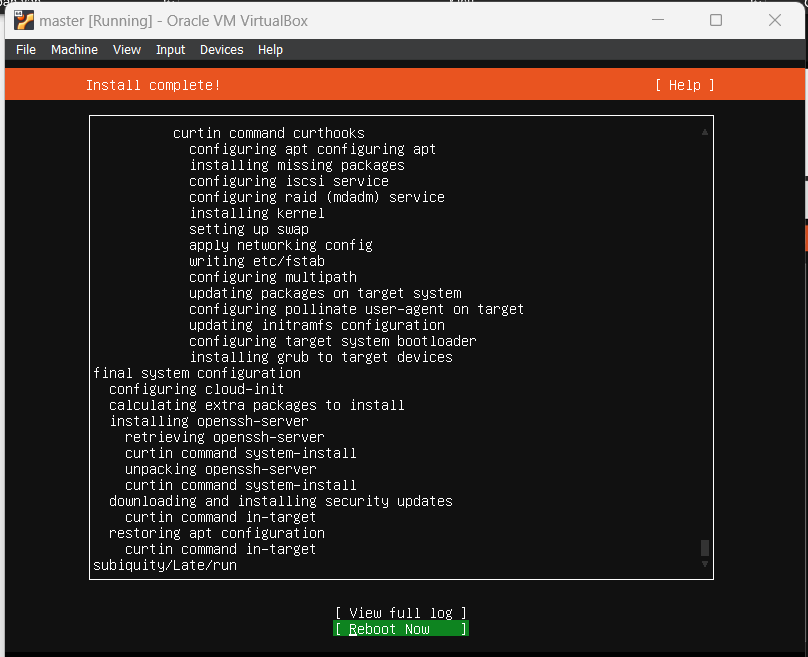
* **Đặt username và password cho máy ảo**



* **Chọn Install OpenSSH Server và ấn Done**



* **Đợi cài đặt hoàn tất, chọn Reboot Now**



### **2.2.2 Cài đặt Hadoop**

* **Tạo user hadoopuser và cấp tất cả quyền:**

*sudo adduser hadoopuser*

*sudo usermod -aG sudo hadoopuser*





* **Thực hiện đăng nhập vào hadoopuser**

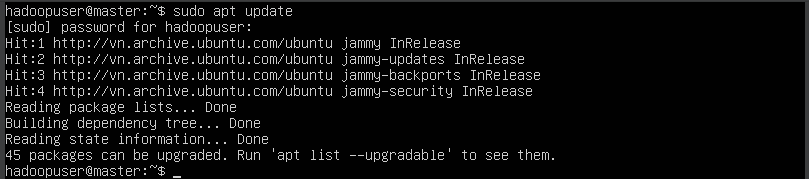


* **Thực hiện update và cài đặt jdk:**

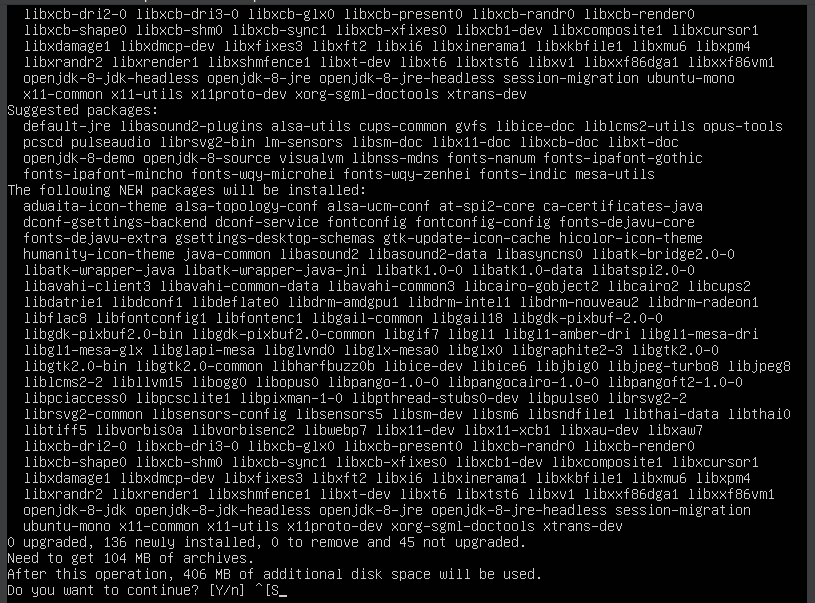
*sudo apt update*

*sudo apt install openjdk-8-jdk*

*Sudo apt update*



*Install openjdk*



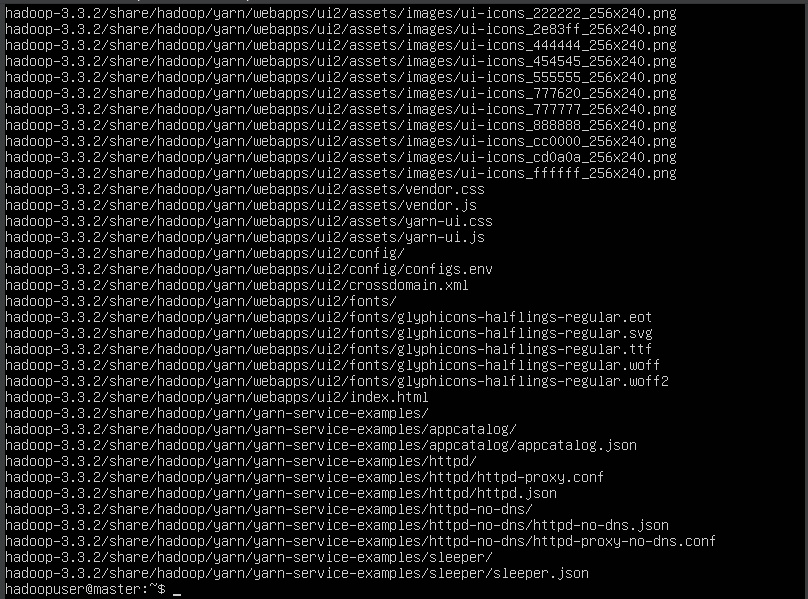
* **Thực hiện download hadoop:**

*wget https://archive.apache.org/dist/hadoop/common/hadoop-3.3.2/hadoop-3.3.2.tar.gz*

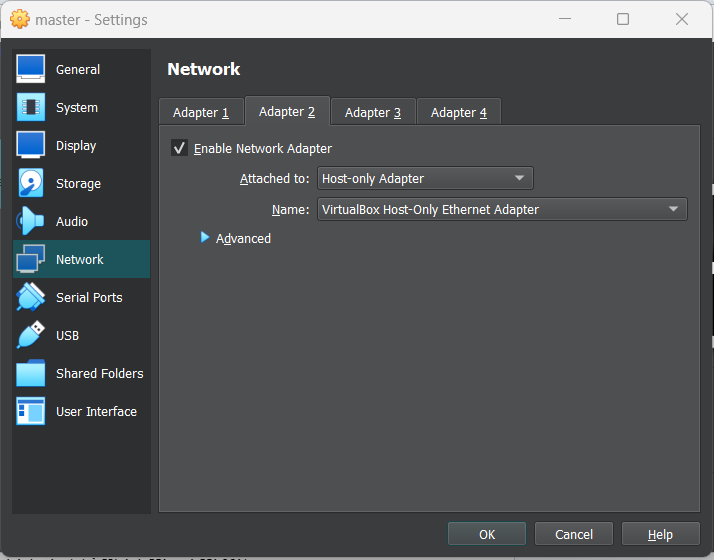


* **Giải nén file hadoop**

*tar -zxvf hadoop-3.3.2.tar.gz*

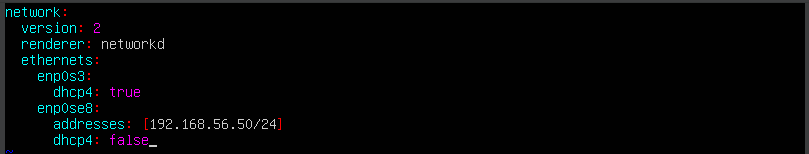


* **Power off the Machine và Thay đổi adapter 2**

****

* **Thực hiện tạo file 01-netcfg.yaml và cấu hình mạng như dưới đây**

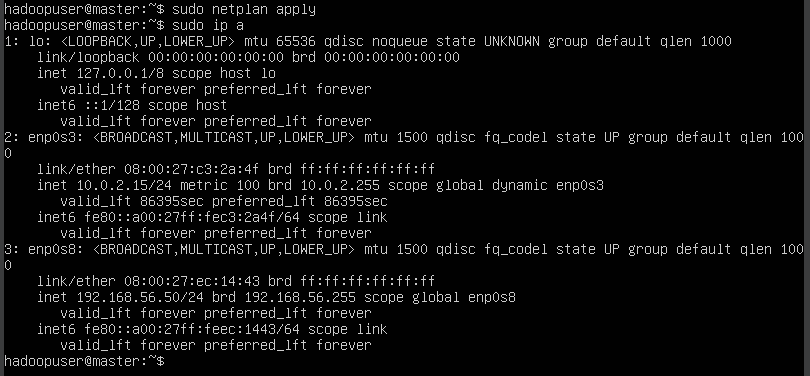
*sudo vim /etc/netplan/01-netcfg.yaml*

****

* **Thực hiện apply netplan và kiểm tra ip:**

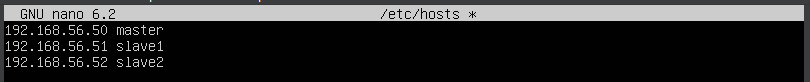
*sudo netplan apply*

*sudo ip a*

**

* **Thực hiện chỉnh sửa file host và thay đổi hostname**

*sudo nano /etc/hosts*

**

*sudo hostname master*

**

* **Thực hiện chỉnh sửa file ~/.bashrc**

*sudo nano ~/.bashrc*

thêm vào những lệnh như sau:

**

* ***Thực hiện các lệnh sau:***

*source ~/.bashrc*

*git clone* [*https://github.com/nilesh-g/hadoop-cluster-install.git*](https://github.com/nilesh-g/hadoop-cluster-install.git)

**

* **Copy từ hadoop-cluster-install/master qua máy master**

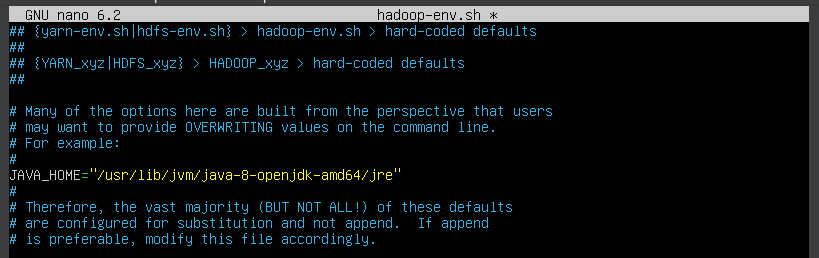
*cp hadoop-cluster-install/master/\* hadoop-3.3.2/etc/hadoop/*



* **Kiểm tra lại các file**

*cd hadoop-3.3.2/etc/hadoop/*

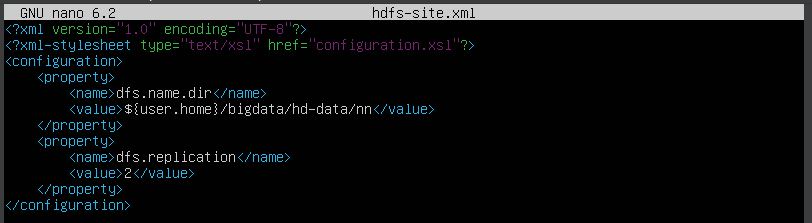
*sudo nano hadoop-env.sh*

****

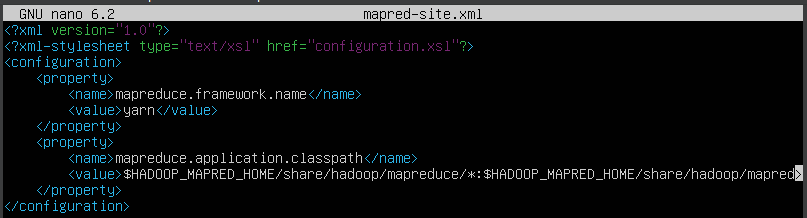
*sudo nano core-site.xml*



*sudo nano hdfs-site.xml*



*sudo nano mapred-site.xml*



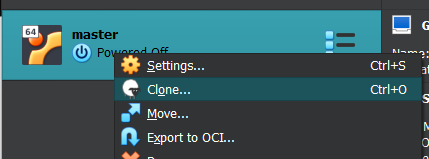
*sudo nano yarn-site.xml*

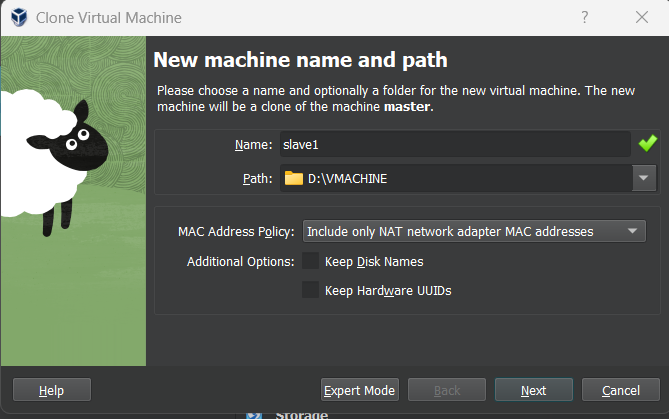
**

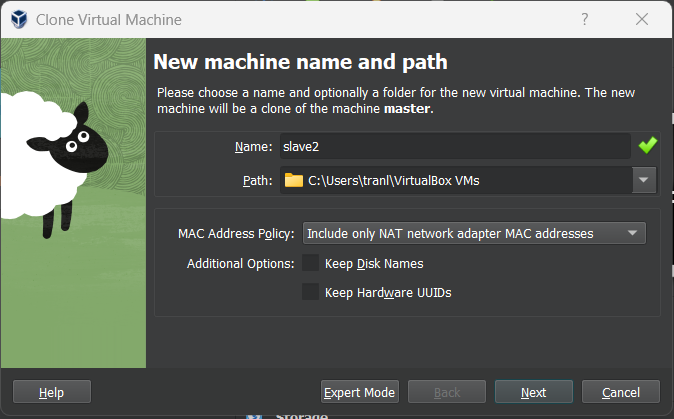
*sudo nano workers*

**

* **Thực hiện clone master thành 2 máy slave(sau khi đã cài hadoop) như sau:**

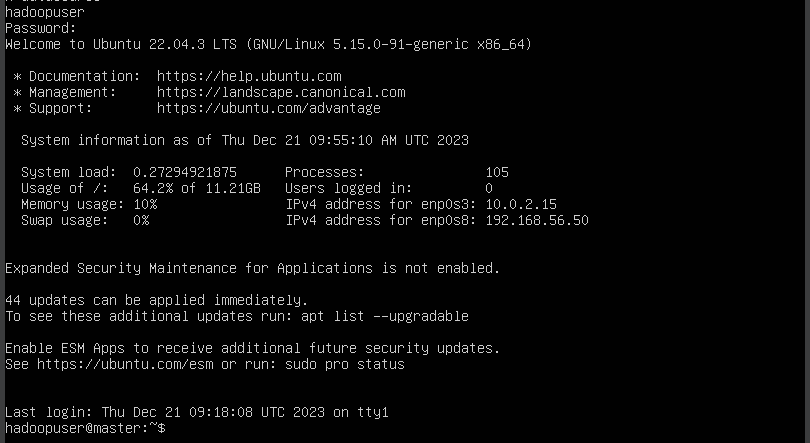






* **Thực hiện ở cả 2 máy slaves những bước sau:**

**Đăng nhập vào hadoopuser:**

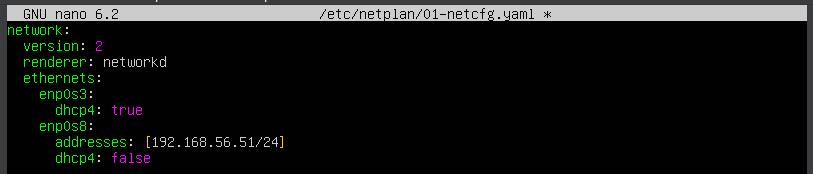
****

**Cấu hình file 01-netcfg.yaml:**

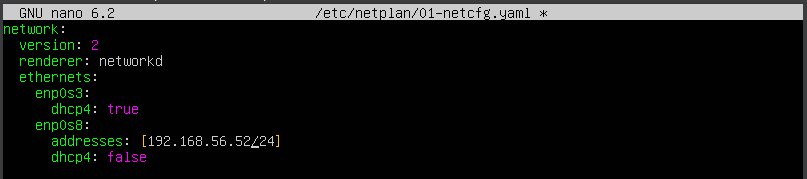
*sudo nano /etc/netplan/01-netcfg.yaml*

*sudo netplan apply*

**Ở slave1:**

****

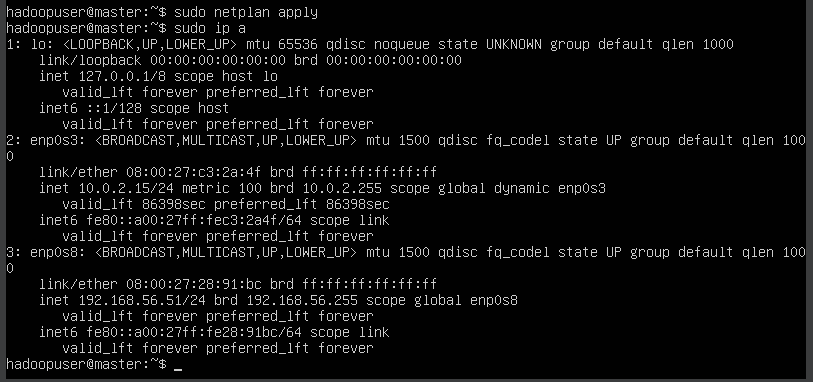
**Ở slave2:**



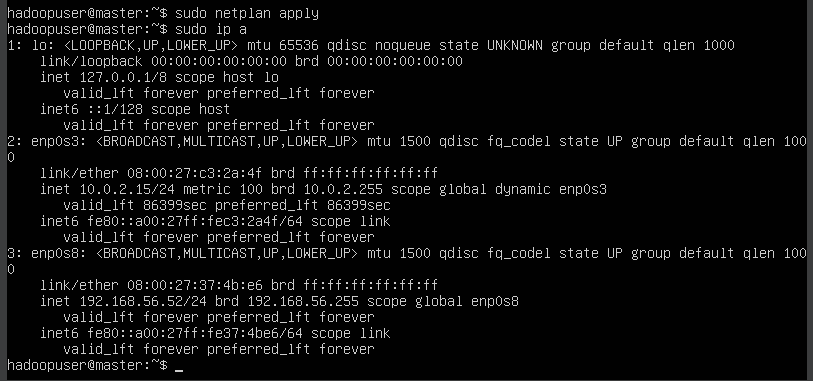
**Kiểm tra lại ip đã được chỉnh sửa:**

*sudo ip a*

**Ở slave1:**



**Ở slave2:**



**Chỉnh sửa file hostname:**

*sudo nano /etc/hostname*

**Ở slave1:**

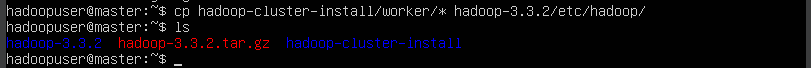


**Ở slave2:**

****

**Copy từ hadoop-cluster-install/workder qua:**

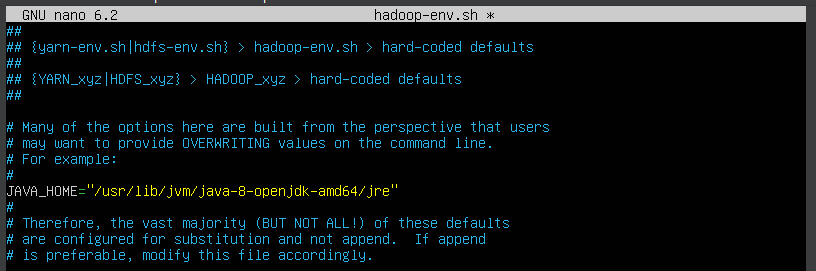
*cp hadoop-cluster-install/worker/\* hadoop-3.3.2/etc/hadoop/*



**Ta thực hiện kiểm tra các file vừa copy:**

*cd hadoop-3.3.2/etc/hadoop/*

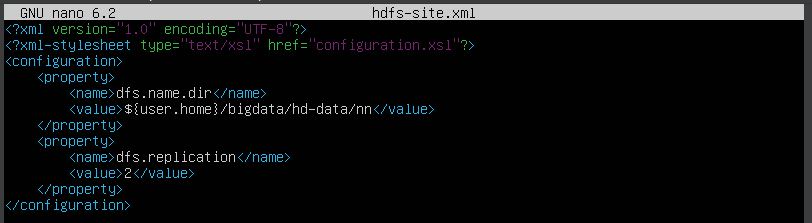
*sudo nano hadoop-env.sh*

**

*sudo nano core-site.xml*



*sudo nano hdfs-site.xml*



*sudo nano yarn-site.xml*

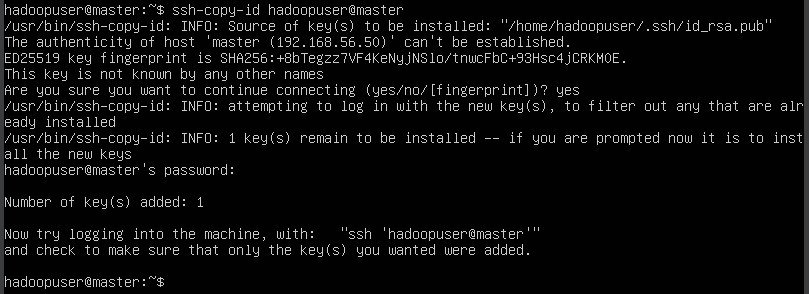
**

* **Mở 3 máy và thực hiện các lệnh sau (trên master):**

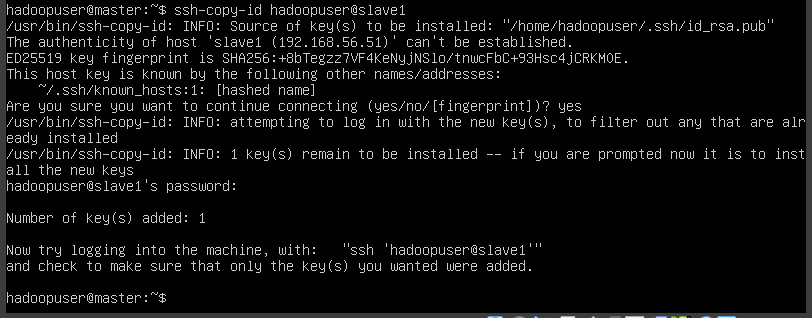
*ssh-keygen -t rsa -P “”*

**

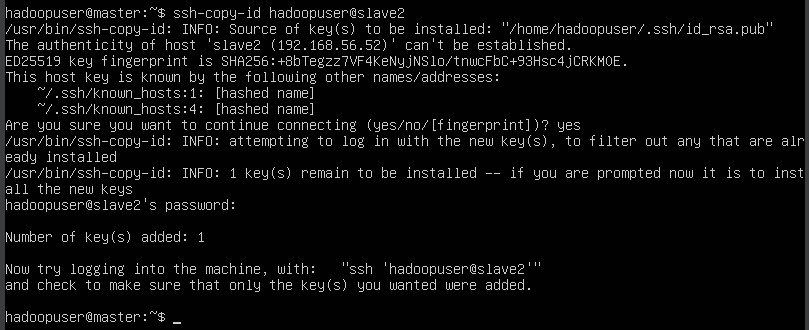
*ssh-copy-id hadoopuser@master*

**

*ssh-copy-id hadoopuser@slave1*

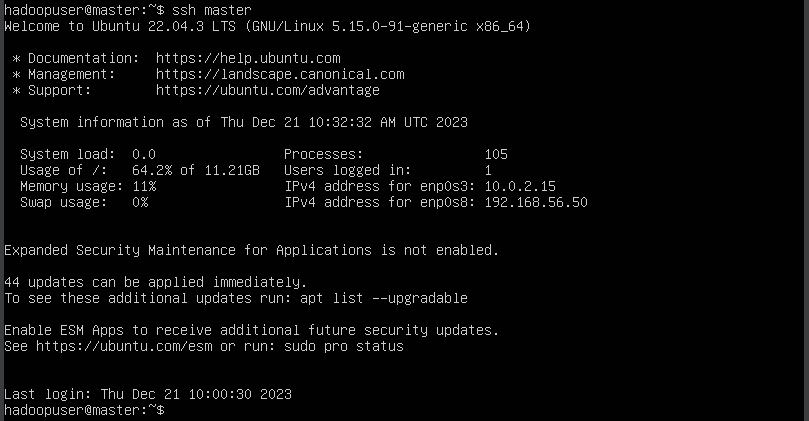
**

*ssh-copy-id hadoopuser@slave2*

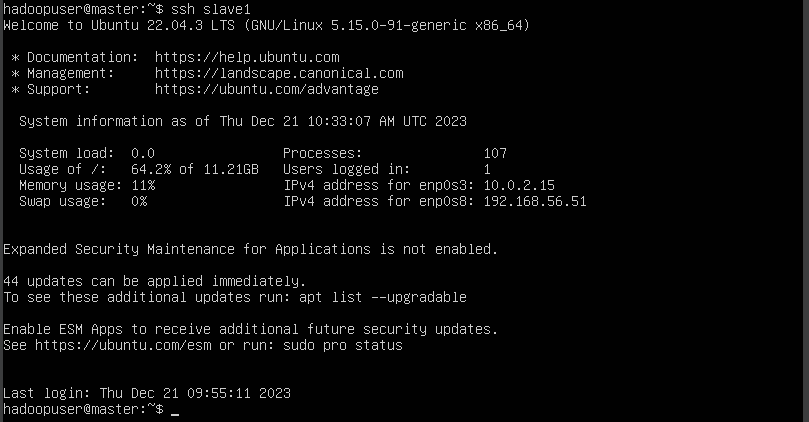
**

* **Kiểm tra kết nối (trên master):**

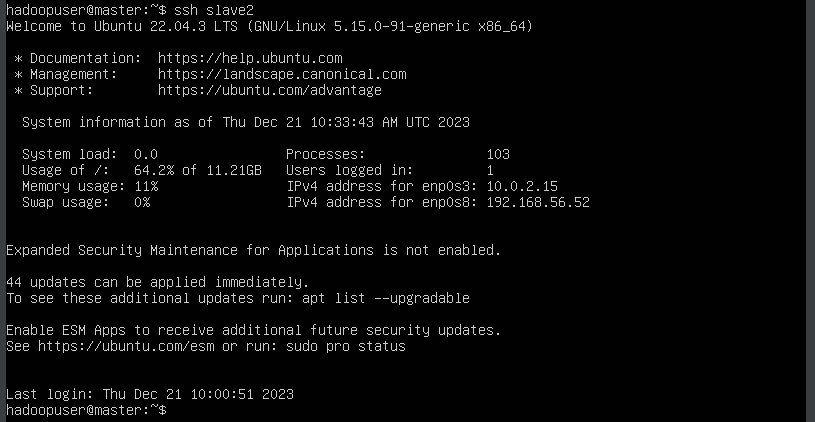
*ssh master*

**

*ssh slave1*

**

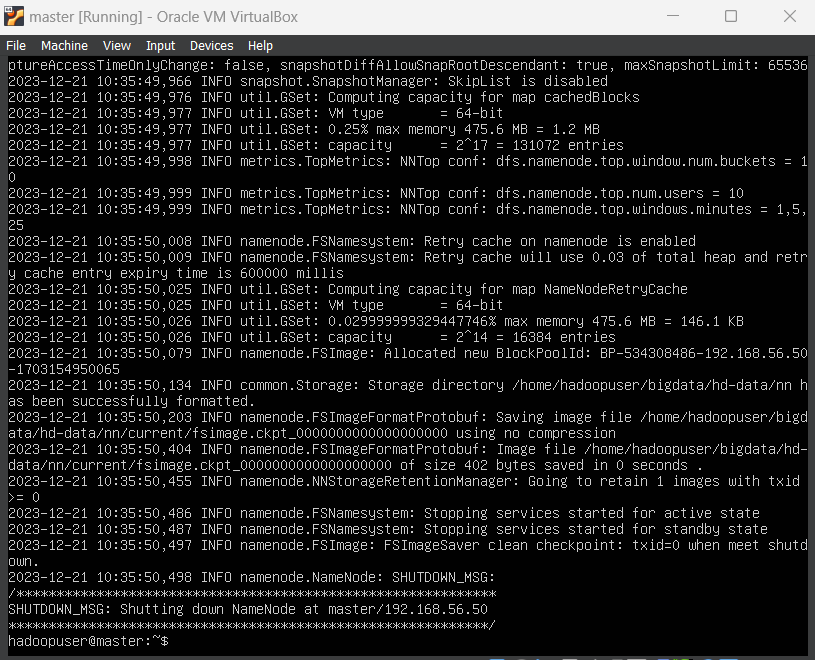
*ssh slave2*

**

* **Thực hiện các lệnh sau (trên master):**

*cd ~*

*hdfs namenode -format*

**

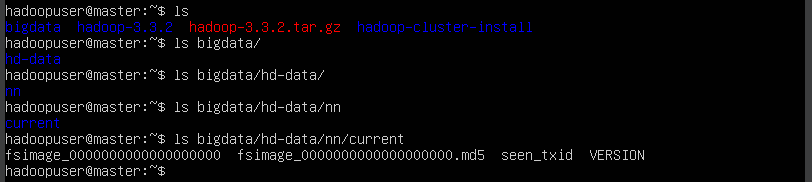
*ls*

*ls bigdata/*

*ls bigdata/hd-data/*

*ls bigdata/hd-data/nn*

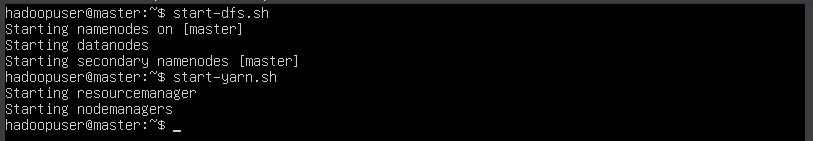
*ls bigdata/hd-data/nn/current*

**

* **Thực hiện khởi chạy hadoop (trên master):**

*start-dfs.sh*

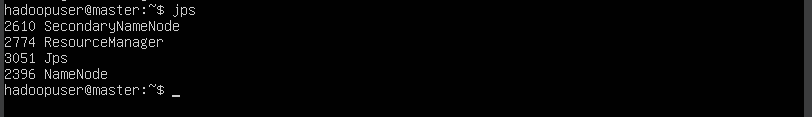
*start-yarn.sh*

**

* **Kiểm tra quá trình hoàn tất (trên cả 3 máy):**

*jps*

**Ở master:**

****

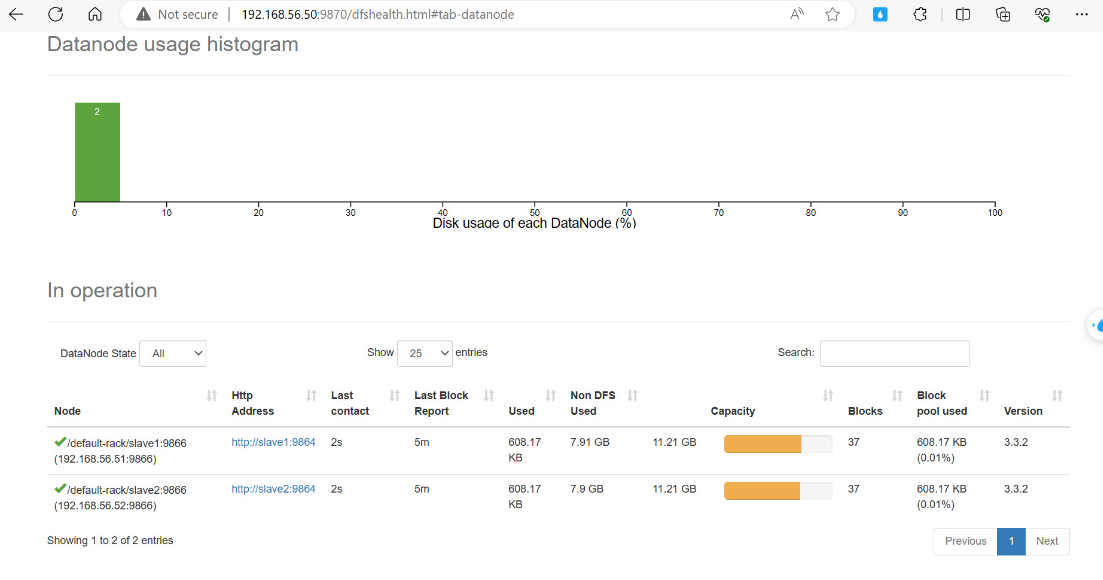
**Ở slave1:**

****

**Ở slave2:**

****

**Kiểm tra trên Web UI của Hadoop**

****

### **2.2.3 Cài đặt Zookeeper**

* **Thực hiện tải zookeeper và apache-storm về máy về máy (master):**

***wget*** [***https://archive.apache.org/dist/zookeeper/zookeeper-3.6.3/apache-zookeeper-3.6.3-bin.tar.gz***](https://archive.apache.org/dist/zookeeper/zookeeper-3.6.3/apache-zookeeper-3.6.3-bin.tar.gz)

**wget**

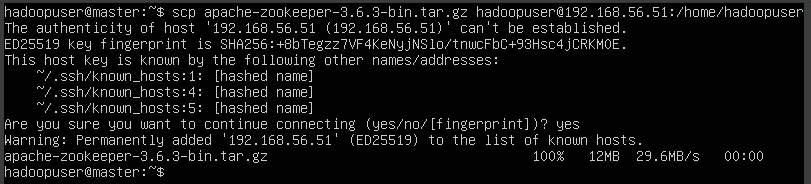
[***https://archive.apache.org/dist/storm/apache-storm-2.2.1/apache-storm-2.2.1.tar.gz***](https://dlcdn.apache.org/storm/apache-storm-2.2.1/apache-storm-2.2.1.tar.gz)

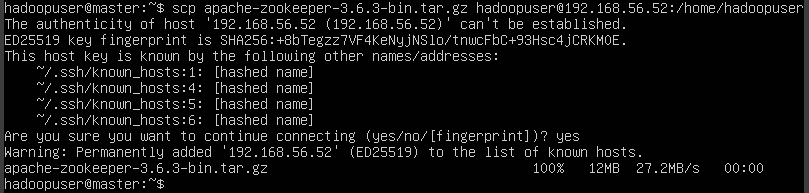
******

****Thực hiện copy file zookeeper qua các máy slave (trên master)**

*scp apache-zookeeper-3.6.3-bin.tar.gz hadoopuser@192.168.56.51:/home/hadoopuser*

*scp apache-zookeeper-3.6.3-bin.tar.gz hadoopuser@192.168.56.52:/home/hadoopuser*

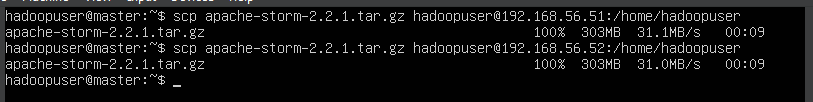
**

**

**Thực hiện copy file storm qua các máy slave (trên master):**

*scp apache-storm-2.2.1.tar.gz hadoopuser@192.168.56.51:/home/hadoopuser*

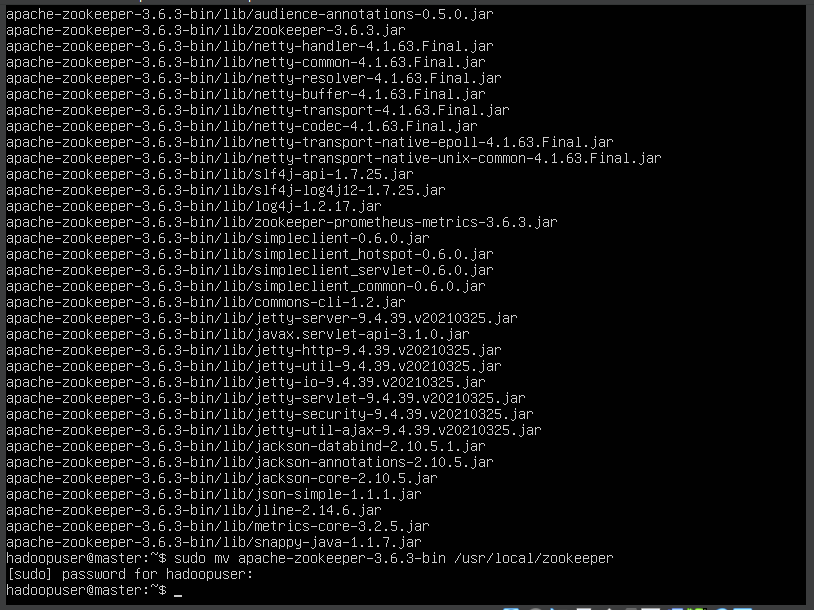
*scp apache-storm-2.2.1.tar.gz hadoopuser@192.168.56.51:/home/hadoopuser*

****

**Giải nén file zookeeper và move vào folder /usr/local/zookeeper (trên cả 3 máy):**

*tar -zxvf apache-zookeeper-3.6.3-bin.tar.gz*

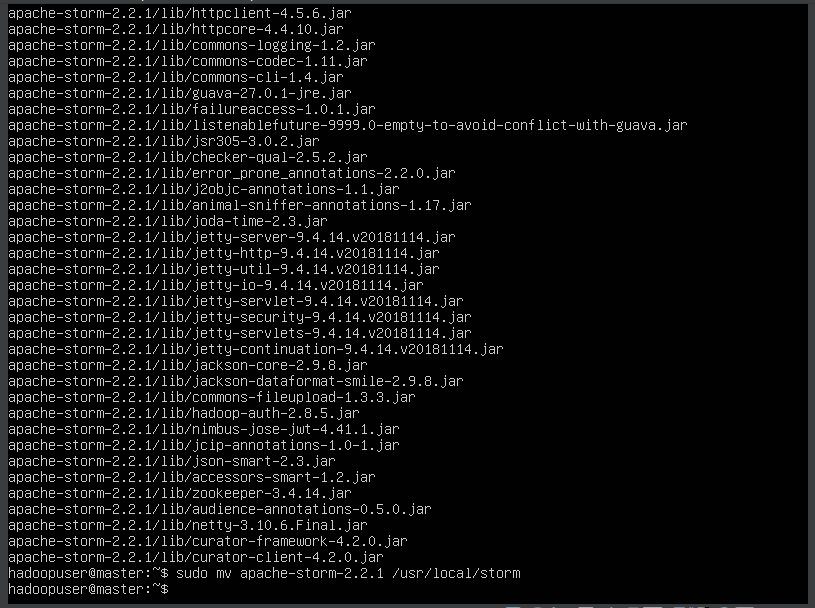
*sudo mv apache-zookeeper-3.6.3-bin /usr/local/zookeeper*

****

**Giải nén file storm và move vào folder /usr/local/storm (trên 3 máy):**

*tar -zxvf apache-storm-2.2.1.tar.gz*

*sudo mv apache-storm-2.2.1 /usr/local/storm*

**

***Chỉnh sửa file ~/.bashrc (trên cả 3 máy):***

*sudo nano ~/.bashrc*

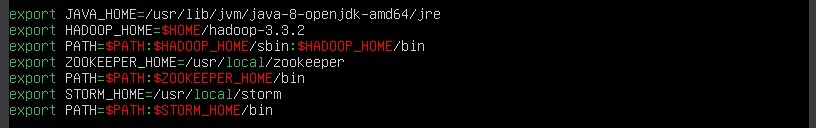
Add những dùng sau:

*export ZOOKEEPER\_HOME=/usr/local/zookeeper*

*export PATH=$PATH:$ZOOKEEPER\_HOME/bin*

*export STORM\_HOME=/usr/local/storm*

*export PATH=$PATH:$STORM\_HOME/bin*

**

**Sau đó chạy ~/.bashrc (trên cả 3 máy):**

*source ~/.bashrc*

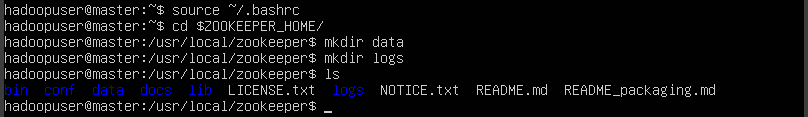
**

**Di chuyển vào folder Zookeeper và tạo folder data và logs (trên cả 3 máy):**

*cd $ZOOKEEPER\_HOME/*

*mkdir data*

*mkdir logs*

****

**Di chuyển vào folder conf và thực hiện copy file zoo\_sample.cfg (trên 3 máy)**

*cd conf/*

*cp zoo\_sample.cfg zoo.cfg*

****

**Thực hiện chỉnh sửa file zoo.cfg (trên cả 3 máy):**

*sudo nano zoo.cfg*

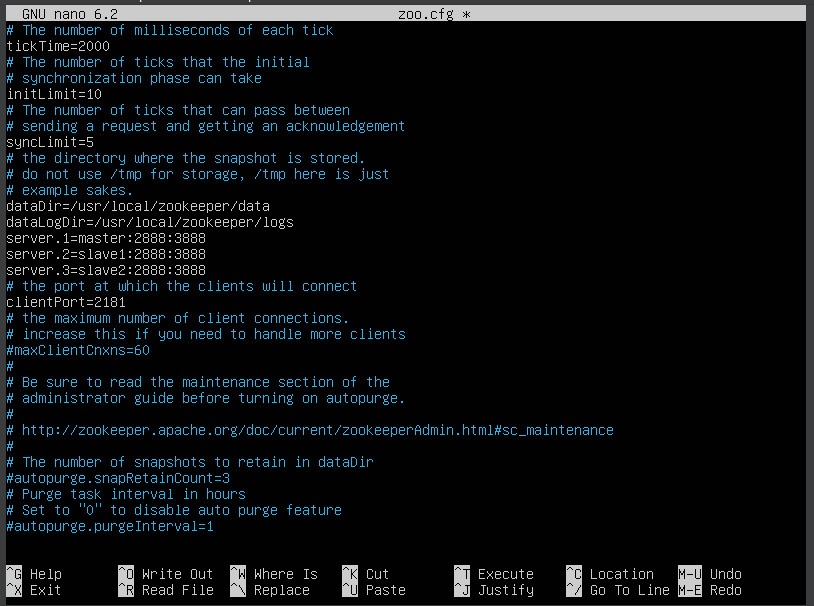
dataDir=/usr/local/zookeeper/data

dataLogDir=/usr/local/zookeeper/logs

server.1=tnmaster:2888:3888

server.2=tnslave1:2888:3888

server.3=tnslave2:2888:3888



**Thực hiện chỉnh sửa myid (trên cả 3 máy):**

*sudo nano /usr/local/zookeeper/data/myid*

**Ở master:**

****

**Ở slave1:**

****

**Ở slave2:**

****

**Thực hiện chạy start zookeeper và kiểm tra tiến trình hoàn tất (trên 3 máy):**

*zkServer.sh start*

*jps*

**Ở master:**

****

**Ở slave1:**

****

**Ở slave2:**

****

### **2.2.4 Cài đặt HBase**

**Thực hiện download Hbase (trên master):**

*wget* [*https://archive.apache.org/dist/hbase/2.4.0/hbase-2.4.0-bin.tar.gz*](https://archive.apache.org/dist/hbase/2.4.0/hbase-2.4.0-bin.tar.gz)

****

**Thực hiện copy qua các máy slave (trên master):**

*scp hbase-2.4.0-bin.tar.gz hadoopuser@192.168.56.51:/home/hadoopuser*

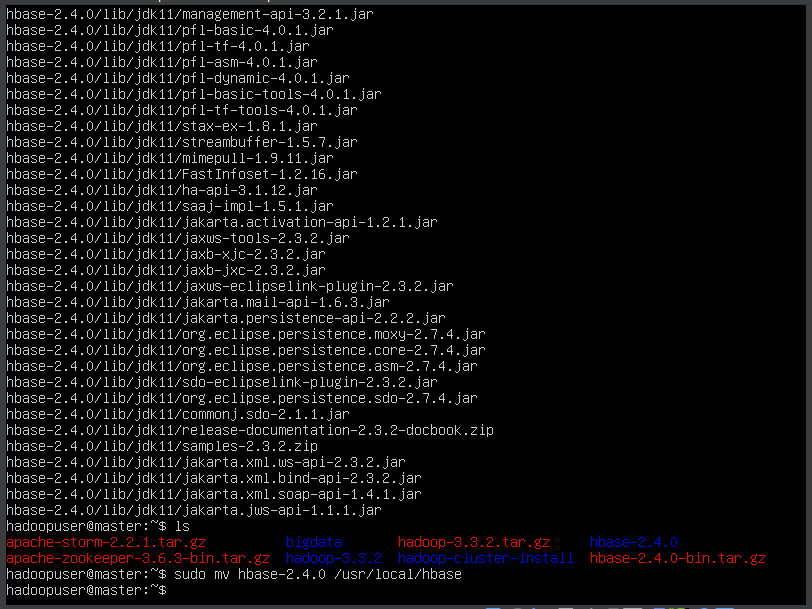
*scp hbase-2.4.0-bin.tar.gz hadoopuser@192.168.56.52:/home/hadoopuser*

****

**Giải nén file hbase và move vào folder /usr/local/hbase (trên cả 3 máy):**

*sudo tar -zxvf hbase-2.4.0-bin.tar.gz*

*sudo mv hbase-2.4.0 /usr/local/hbase*

****

**Thêm các dòng sau vào file .bashrc (trên cả 3 máy):**

*sudo nano ~/.bashrc*

export HBASE\_HOME=/usr/local/hbase

export PATH=$PATH:$HBASE\_HOME/bin

**

Sau đó chạy lệnh source ~/.bashrc

****

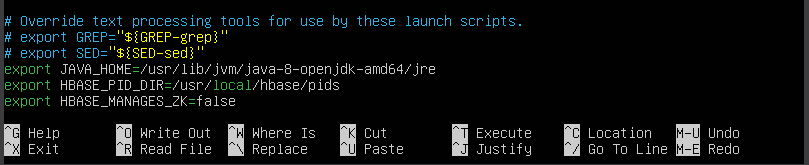
**Thưc hiện chỉnh sửa hbase-env.sh (trên cả 3 máy)**

*sudo nano /usr/local/hbase/conf/hbase-env.sh*

export JAVA\_HOME=/usr/lib/jvm/java-8-openjdk-amd64/jre

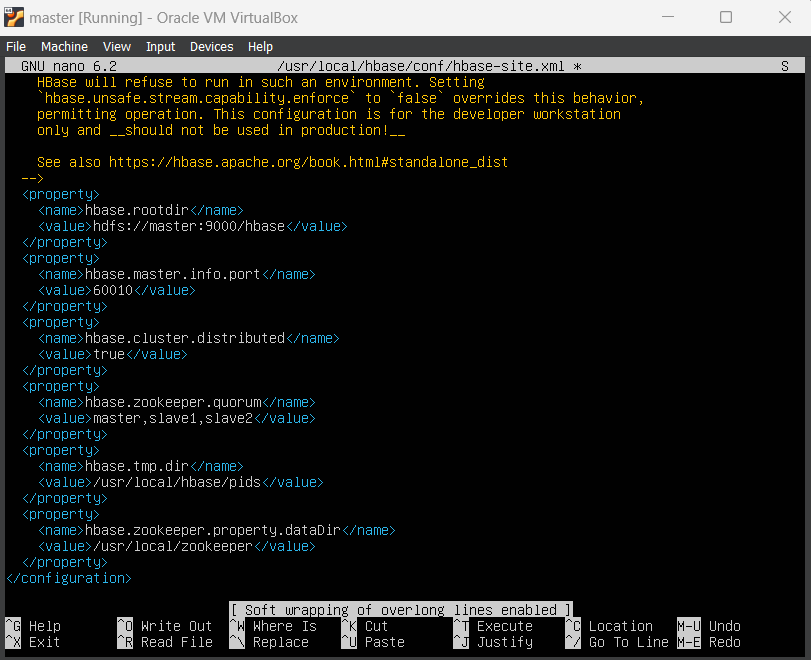
export HBASE\_PID\_DIR=/usr/local/hbase/pids

export HBASE\_MANAGES\_ZK=false



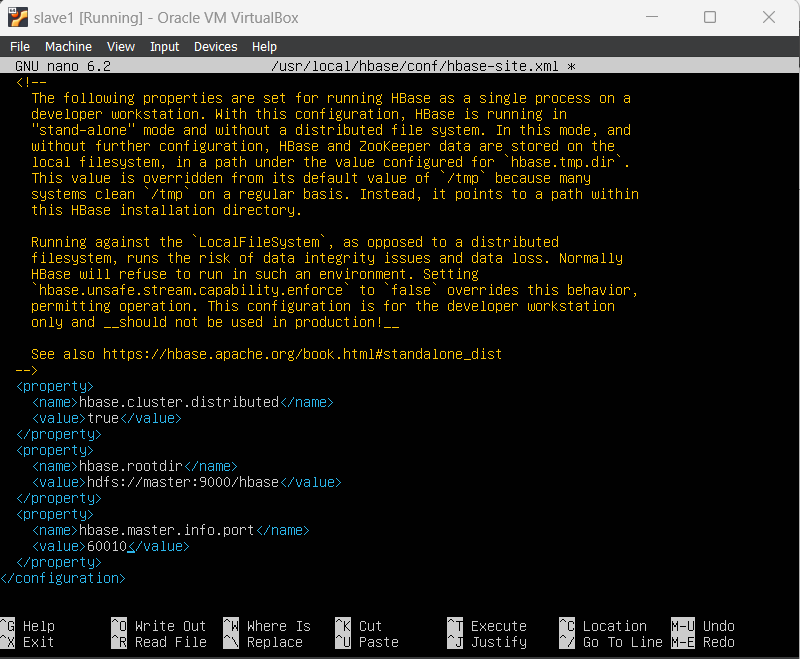
**Chỉnh sửa file hbase-site.xml (trên master):**

*sudo nano /usr/local/hbase/conf/hbase-site.xml*

**

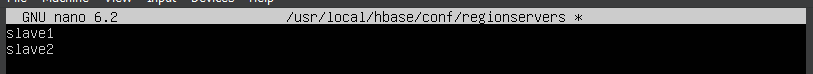
**Chỉnh sửa file hbase-site.xml (trên slave1, slave2):**

*sudo nano /usr/local/hbase/conf/hbase-site.xml*

**

**Thực hiện chỉnh sửa file regionservers (trên master):**

*sudo nano /usr/local/hbase/conf/regionservers*

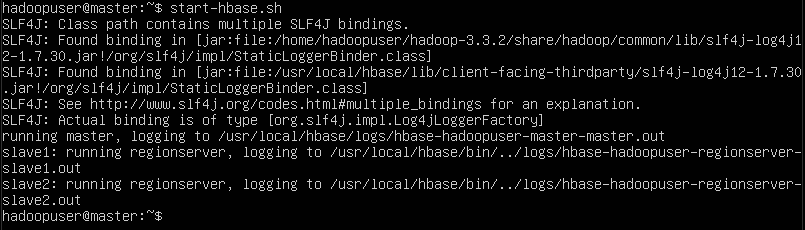
****

**Thực hiện khởi động Hbase (trên master):**

*start-all.sh (khởi động hadoop cluster nếu chưa khởi động)*

*zkServer.sh start (khởi động zookeeper ở cả 3 máy nếu chưa khởi động)*

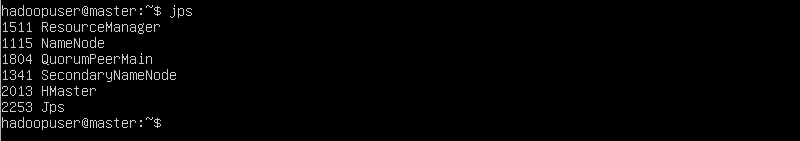
*start-hbase.sh (khởi động hbase)*

**

**Kiểm tra quá trình hoàn tất (trên 3 máy):**

*jps*

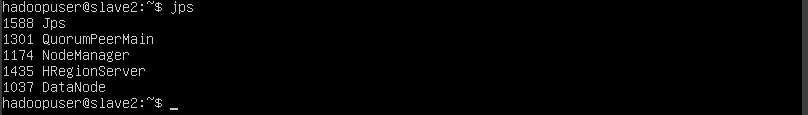
**Ở master:**

****

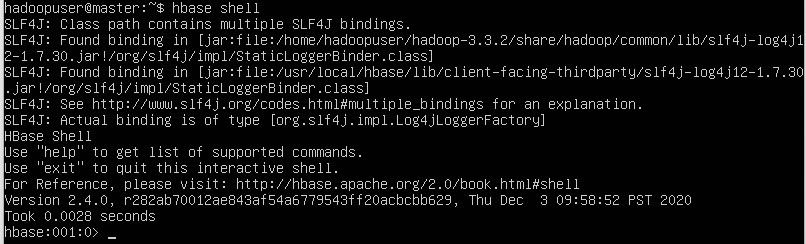
**Ở slave1:**

****

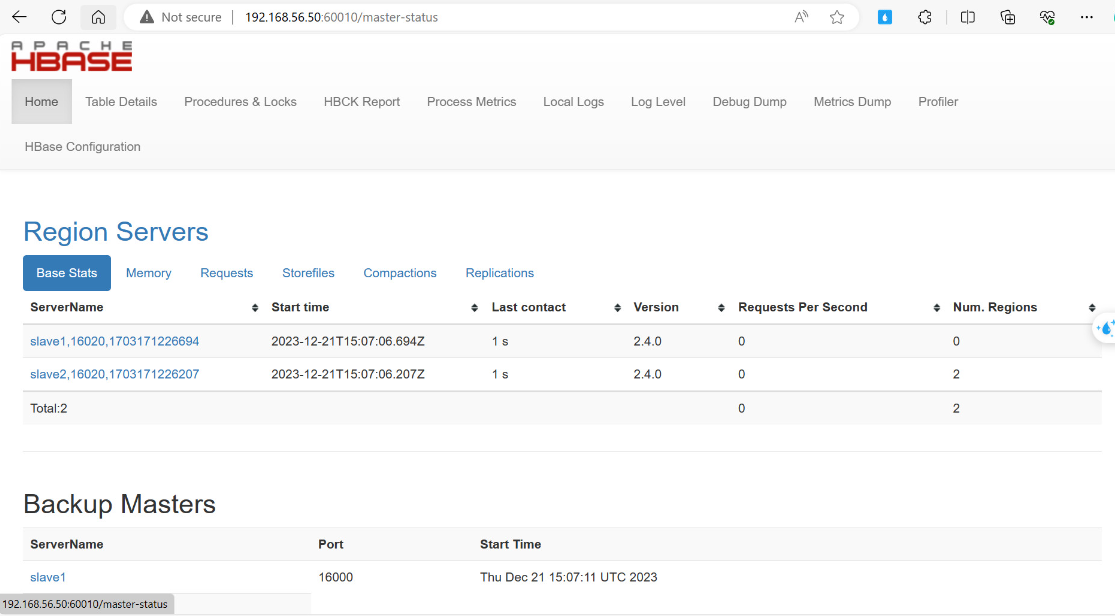
**Ở slave2:**

****

*hbase shell* (ở master)



**Kiểm tra active trên web UI HBase**

****

# **Chương 3: THỰC NGHIỆM MÔ PHỎNG PHÂN TÁN**

## **3.1. MÔ TẢ BÀI TOÁN ĐẶT RA VỚI DỮ LIỆU**

**CSDL Quản lý hệ thống cửa hàng điện tử TechnoShop:**

Hệ thống cửa hàng điện tử TechnoShop có 3 chi nhánh, với trụ sở chính nằm ở TP. Hồ Chí Minh, 2 chi nhánh còn lại ở Khánh Hòa và Hà Nội.

**Các chức năng của hệ thống:**

* Đăng nhập hệ thống
* Quản lý danh mục cửa hàng
* Quản lý danh sách khách hàng
* Quản lý danh sách nhân viên
* Quản lý bán hàng
* Thống kê doanh thu

**Lý do phân tán dữ liệu:**

* Tăng hiệu suất và khả năng mở rộng: Phân tán dữ liệu cho phép dữ liệu được lưu trữ và xử lý trên nhiều nút máy chủ, giúp tăng hiệu suất và khả năng mở rộng của hệ thống. Khi dữ liệu được phân tán, các tác vụ xử lý có thể được chia nhỏ và phân phối trên nhiều nút, giúp tối ưu hóa tài nguyên và giảm thời gian xử lý.
* Đơn vị có nhiều cửa hàng nằm ở các tỉnh thành khác nhau trong cả nước, 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.
* Tối ưu hóa việc truy cập dữ liệu: Phân tán dữ liệu có thể giúp tối ưu hóa việc truy cập dữ liệu bằng cách đặt dữ liệu gần người dùng hoặc ứng dụng sử dụng nó, giúp giảm độ trễ và tăng tốc độ truy cập.
* Tính linh hoạt: Phân tán dữ liệu cho phép hệ thống mở rộng dễ dàng bằng cách thêm nút mới vào cụm, giúp tăng khả năng chịu tải và đáp ứng nhu cầu mở rộng của ứng dụng.
* Cần kết nối các CSDL có sẵn: 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.
* Giảm chi phí truyền thông: Tăng 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: Phân tán dữ liệu giúp tăng độ tin cậy của hệ thống. Khi dữ liệu được sao chép và phân phối trên nhiều nút, nếu một nút gặp sự cố, dữ liệu vẫn có thể được truy cập từ các nút khác, giúp giảm thiểu nguy cơ mất mát dữ liệu.

## **3.2. MÔ TẢ CẤU TRÚC DỮ LIỆU SỬ DỤNG**

**Database TechnoShop**

**CUAHANG** (MaCH, TenCH, SDT, DiaDiem, ChiNhanh)

**Tân từ:** Mỗi cửa hàng có mã cửa hàng (MaCH) dùng để phân biệt các cửa hàng với nhau, ngoài ra còn lưu tên cửa hàng (TenCH), số điện thoại của cửa hàng đó (SDT), địa điểm cửa hàng(DiaDiem), chi nhánh cửa hàng hoạt động (ChiNhanh). Trường mã cửa hàng, tên cửa hàng, số điện thoại là duy nhất.

**KHACHHANG** (MaKH, TenKH, DiaChi, GT, SDT, DiemTichLuy)

**Tân từ:** Mỗi khách hàng có mã khách hàng (MaKH) dùng để phân biệt với các khách hàng khác, ngoài ra còn có lưu tên khách hàng (TenKH), địa chỉ để giao hàng (DiaChi), giới tính của khách hàng (GioiTinh), số điện thoại (SDT) và số điểm tích lũy dựa vào số lần mua hàng (DiemTichLuy).

**NHANVIEN** (MaNV, TenNV, NgSinh, NgayVL, SDT, GioiTinh)

**Tân từ:** Mỗi nhân viên có mã nhân viên (MaNV) để phân biệt với các nhân viên khác. Ngoài ra còn có lưu tên nhân viên (TenNV), ngày sinh (NgaySinh), ngày vào làm (NgayVL), số điện thoại liên lạc (DienThoai) và giới tính của nhân viên (GioiTinh).

**SANPHAM** (MaSP, TenSP, LoaiSP, Gia, ThuongHieu)

**Tân từ:** Mỗi sản phẩm có mã sản phẩm (MaSP) là duy nhất, ngoài ra còn có lưu tên sản phẩm, tên loại sản phẩm (LoaiSP), giá tiền (Gia) và thương hiệu của sản phẩm đó (ThuongHieu).

**HOADON** (MaHD, MaKH, MaCH, MaNV, NgayHD, ThanhTien)

**Tân từ:**  Mỗi hóa đơn sẽ có một mã hóa đơn (MaHD) là duy nhất, ngoài ra còn lưu thông tin ngày lập hóa đơn (NgayHD), mã khách hàng mua sản phẩm (MaKH), mã cửa hàng nơi sản phẩm được bán (MaCH), mã nhân viên lập hóa đơn (MaNV) và tổng thành tiền của hóa đơn đó (ThanhTien).

**CTHD** (SoHD, MaSP, SoLuong)

**Tân từ:** Chi tiết hóa đơn lưu giữ thông tin sản phẩm và số lượng mà hóa đơn sở hữu nó có. Thông tin bao gồm số hóa đơn (SoHD), mã sản phẩm (MaSP), số lượng sản phẩm được mua (SoLuong).

Dữ liệu mẫu ở các bảng

* Bảng CUAHANG

| **RowID** | **ThongTinCH** | | **DiaChi** | |
| --- | --- | --- | --- | --- |
| **MaCH** | **TenCH** | **SDT** | **DiaDiem** | **ChiNhanh** |
| CH01 | TechnoShop HCM | 0389417014 | Khu pho 6, Phuong Linh Trung, Thu Duc, TP HCM | Ho Chi Minh |
| CH02 | TechnoShop HaNoi | 0951761491 | 213 P. Thai Ha, Lang Ha, Dong Da, Ha Noi | Ha Noi |
| CH03 | TechnoShop KhanhHoa | 0780141751 | Ninh Dien, Ninh Tho, Ninh Hoa, Khanh Hoa | Khanh Hoa |

* Bảng KHACHHANG

| **RowID** | **ThongTinKH** | | | | **TichLuy** |
| --- | --- | --- | --- | --- | --- |
| **MaKH** | **TenKH** | **DiaChi** | **GT** | **SDT** | **DiemTichLuy** |
| KH01 | Đoàn Ngọc Tuấn | P. Linh Chiểu, TP.Thủ Đức - TP. Hồ Chí Minh | Nam | 0987164192 | 590 |
| KH02 | Doãn Công Trí | Phường 6, Quận  3, TP, Hồ Chí Minh | Nam | 0983615411 | 268 |
| KH03 | Trần Quốc Hưng | Phường Bến Nghé,  Quận 1, TP. Hồ Chí Minh | Nam | 0356719331 | 1190 |
| KH04 | Trần Lê Tứ | Huyện Hóc Môn,  TP. Hồ Chí Minh | Nam | 0782561511 | 873 |

* Bảng SANPHAM

| **RowID** | **ThongTinSP** | | **ChiTiet** | |
| --- | --- | --- | --- | --- |
| **MaSP** | **TenSP** | **ThuongHieu** | **Gia** | **LoaiSP** |
| SP01 | Apple iPad 10.2-inch (9th Gen) Wi-Fi, 2021 | Apple | 6990000 | Điện thoại |
| SP02 | Điện thoại Samsung Galaxy S23 5G | Samsung | 6600000 | Điện thoại |
| SP03 | Điện Thoại Oppo A57 4GB/128GB - Hàng Chính Hãng | Oppo | 4500000 | Điện thoại |
| SP04 | MacBook Air M1 13 inch 2020 | Apple | 18900000 | Laptop |

* Bảng NHANVIEN

| **RowID** | **TTCaNhan** | | | | **TTLamViec** |
| --- | --- | --- | --- | --- | --- |
| **MaNV** | **TenNV** | **NgaySinh** | **SDT** | **GioiTinh** | **NgayVL** |
| NV01 | Nguyễn Thùy Duyên | 05/09/2000 | 0397172623 | Nu | 12/04/2023 |
| NV02 | Trần Lê Thúy Anh | 04/07/1997 | 0326786536 | Nu | 06/11/2023 |
| NV03 | Huỳnh Khắc Nam | 12/01/2003 | 0982544013 | Nam | 01/01/2023 |

* Bảng HOADON

| **RowID** | **DoiTuongLK** | | | **ChiTiet** | |
| --- | --- | --- | --- | --- | --- |
| **MaHD** | **MaKH** | **MaCH** | **MaNV** | **NgayHD** | **ThanhTien** |
| HD01 | KH01 | CH01 | NV01 | 10/12/2022 | 445000 |
| HD02 | KH02 | CH03 | NV02 | 10/12/2022 | 2250000 |
| HD03 | KH03 | CH02 | NV03 | 10/12/2022 | 250000 |

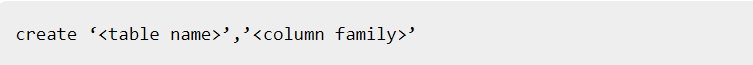
* Bảng CTHD

| **RowID** | **SoHD** | **MaSP** | **SoLuong** |
| --- | --- | --- | --- |
| 1 | HD01 | SP01 | 1 |
| 2 | HD02 | SP02 | 6 |
| 3 | HD01 | SP03 | 3 |
| 4 | HD02 | SP04 | 2 |

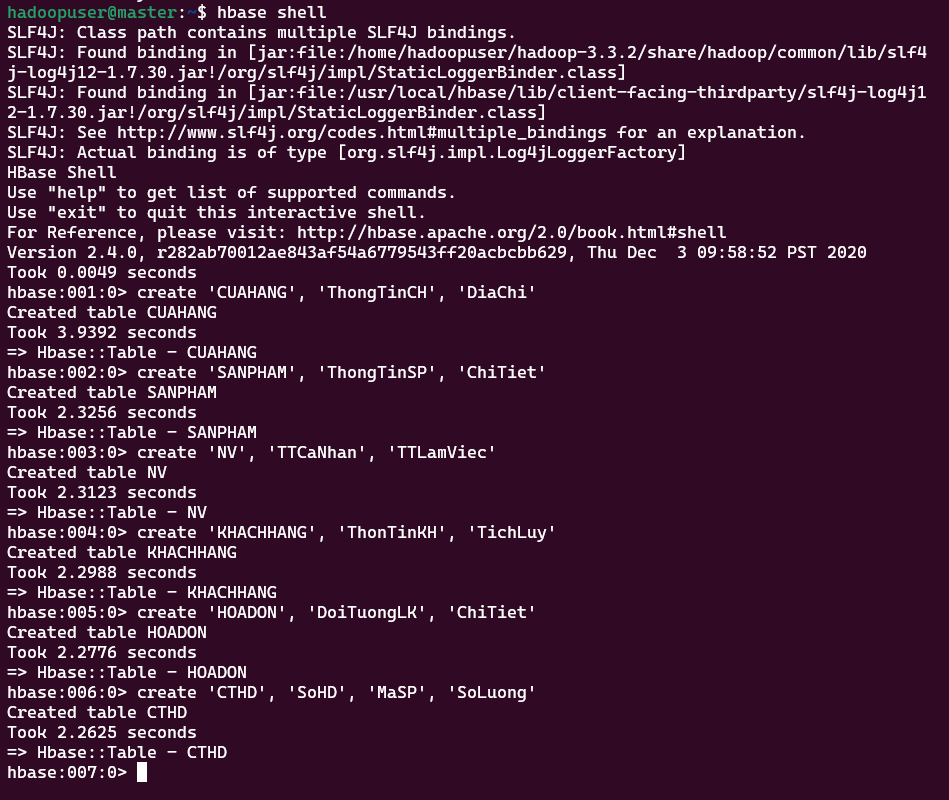
## **3.3. CÁC BƯỚC THỰC NGHIỆM**

### **3.3.1 Tạo Table và các column family**

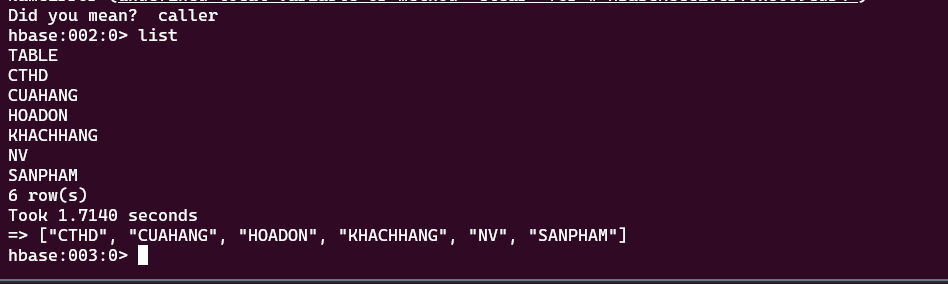
**Câu lệnh:**

****

* Tạo các table ở máy **master**:

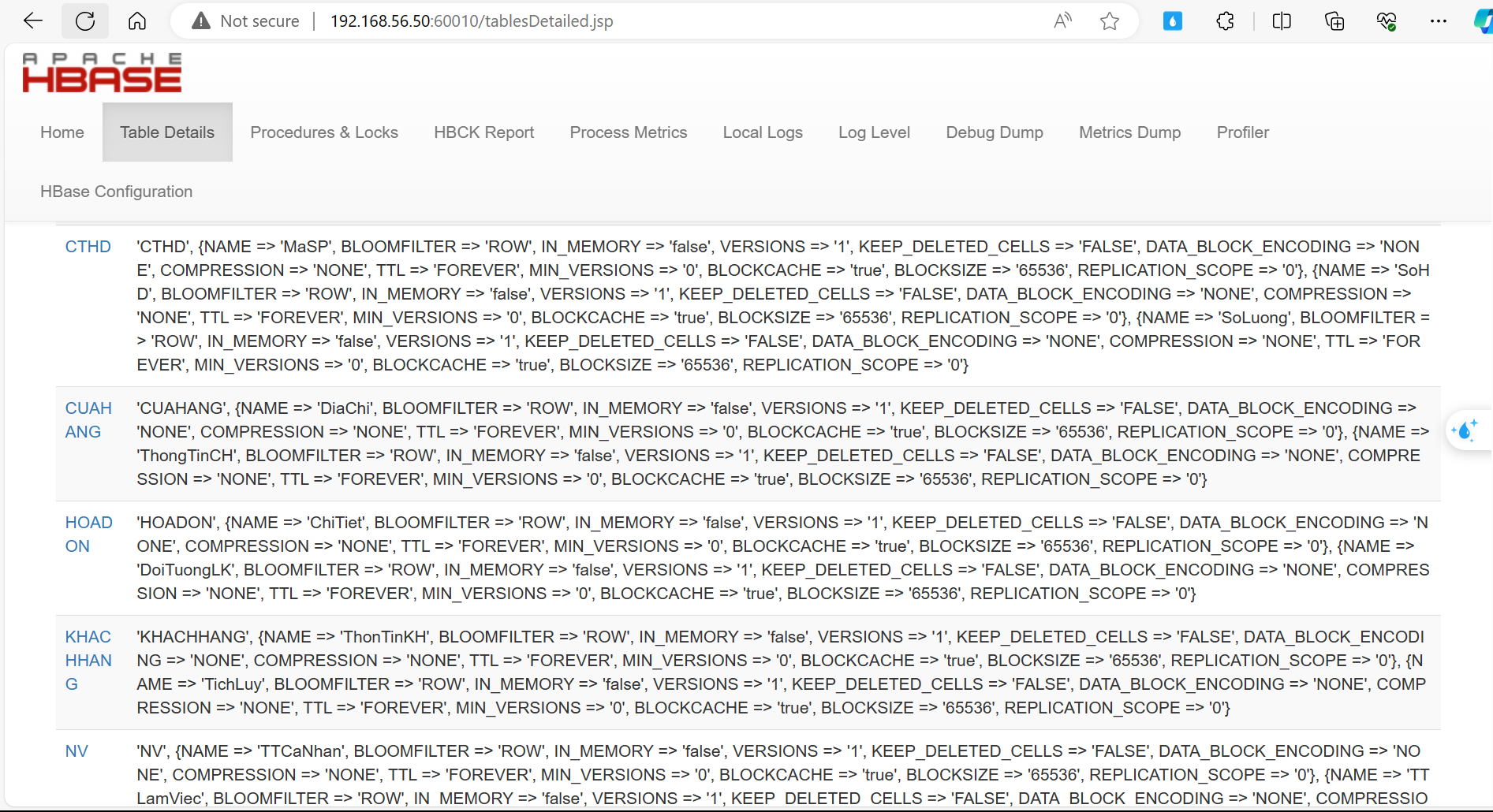


* Qua 2 máy slave kiểm tra ta thấy các table đã được cập nhật qua 2 máy này:



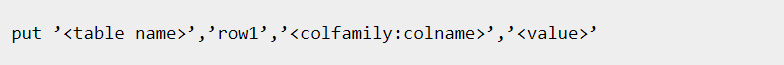


Ta có thể kiểm tra thông tin table trên Web UI:

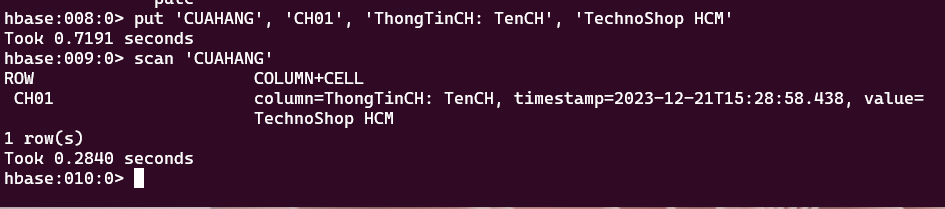


### **3.3.2 Thêm dữ liệu**

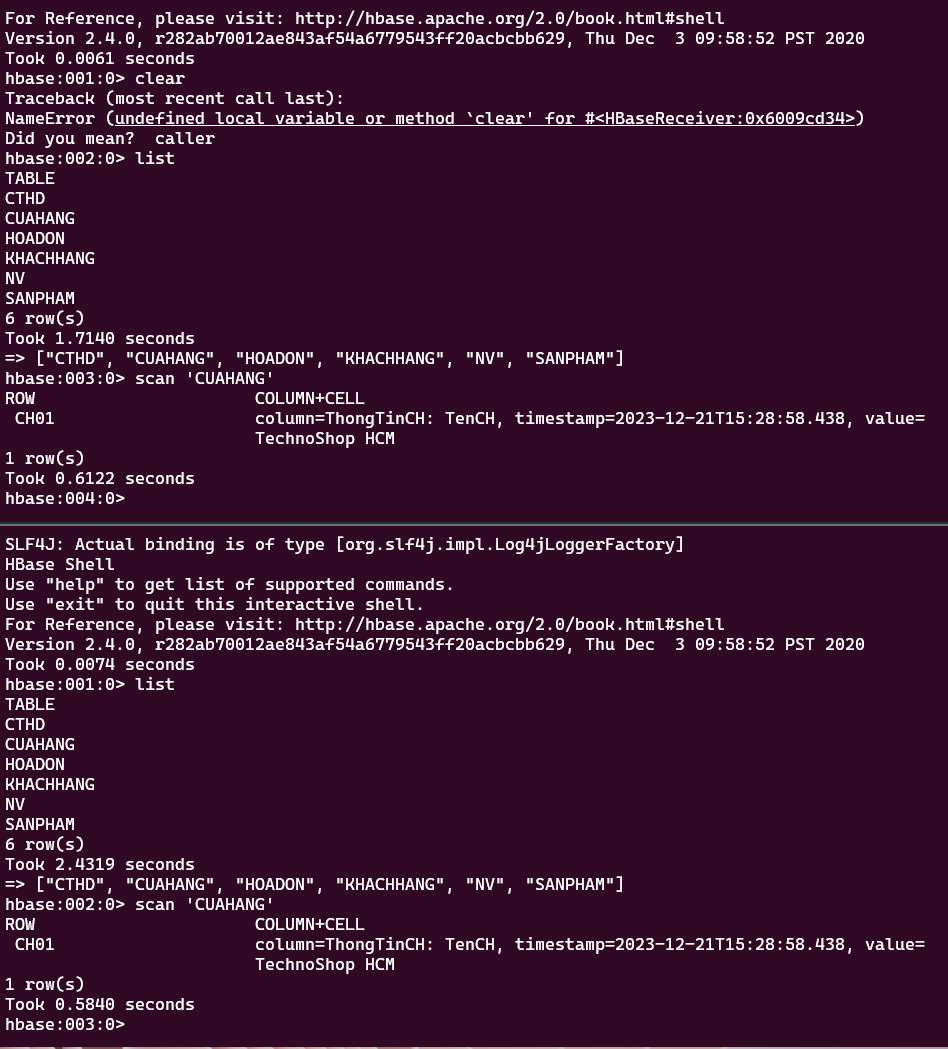
**Câu lệnh:**

****

* Thêm dữ liệu trên máy **master**:



* Kiểm tra trên 2 máy slave:



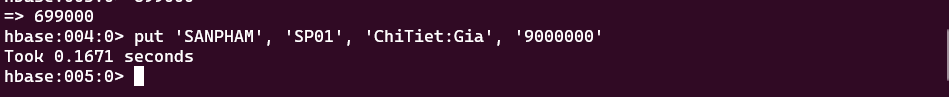
### **3.3.3 Cập nhật dữ liệu**

**Câu lệnh**:

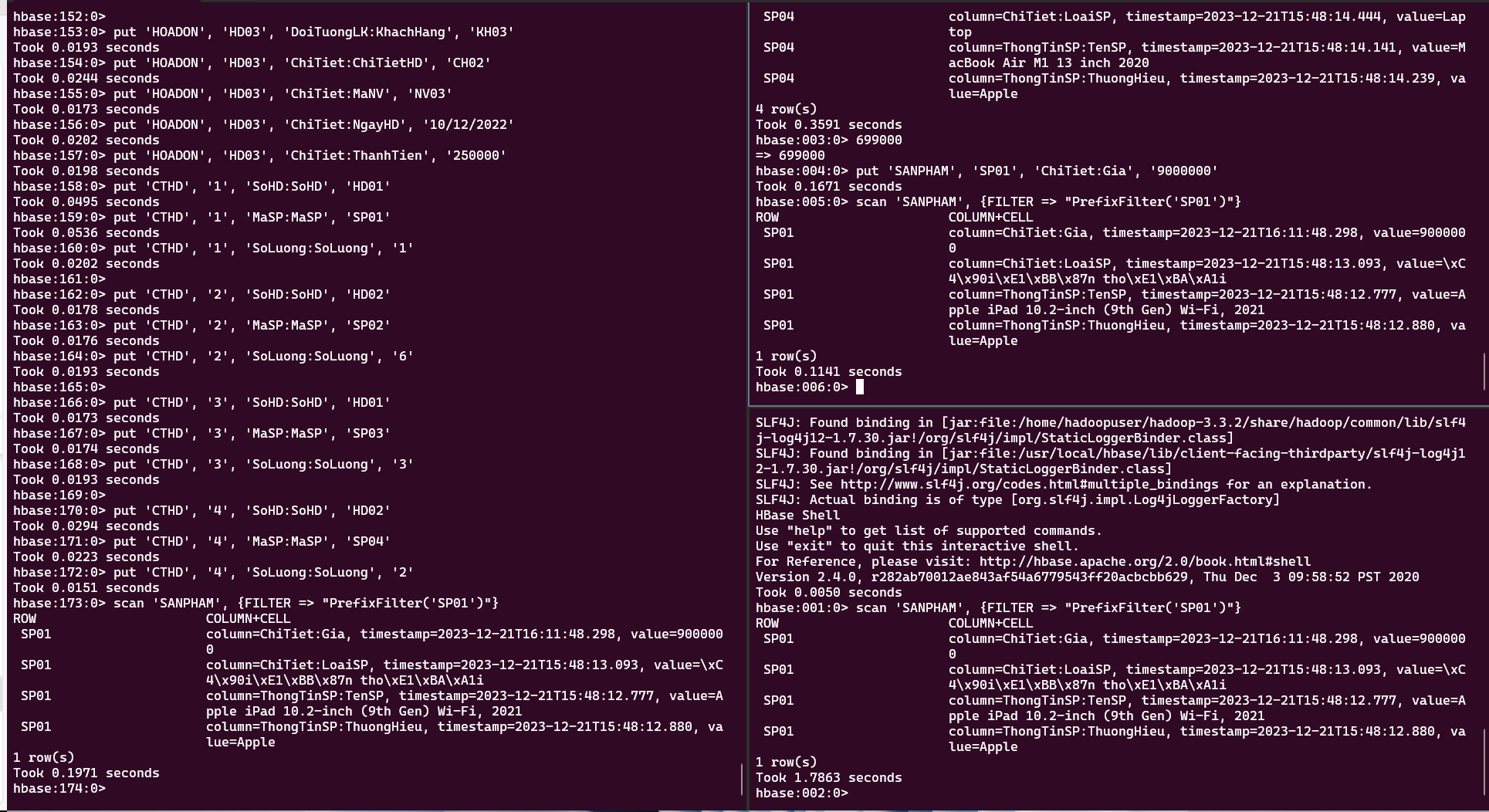


* Sửa dữ liệu trên máy **slave1**:

Ta thấy, ban đầu sản phẩm SP10 có giá là 6990000, thực hiện sửa giá SP10 thành 9000000.



Kết quả sau khi sửa giá (giá được cập nhật lại 1000000 ở cả 3 máy):

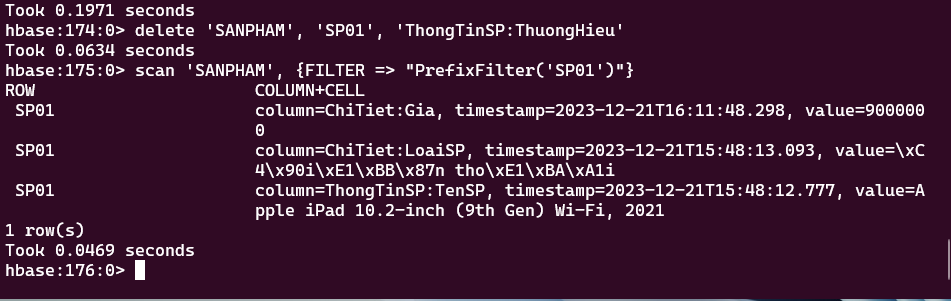


### **3.3.4 Xóa dữ liệu**

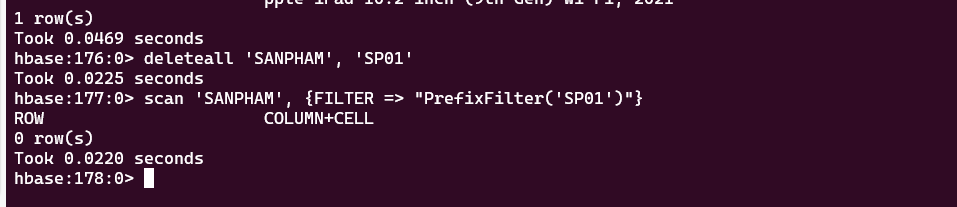
**Câu lệnh:**



* Thực hiện xóa, ta có kết quả:



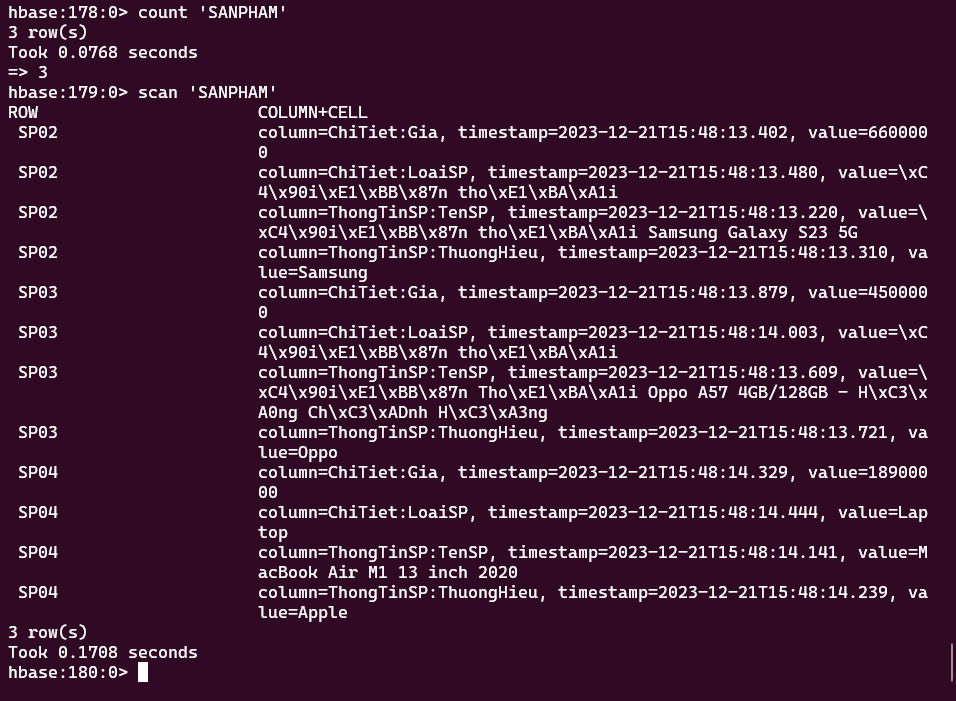
* Câu lệnh xóa tất cả các giá trị của sản phẩm SP01



### **3.3.5 Các thao tác với dữ liệu khác**

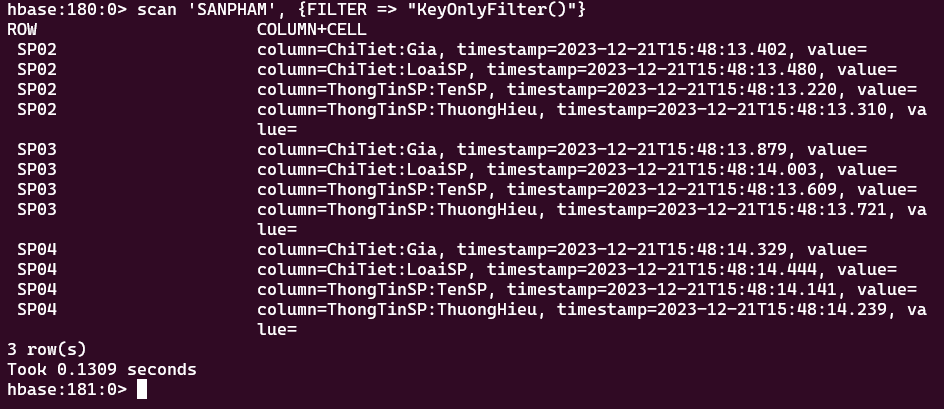
* **Count**

Đếm số hàng của một bảng:



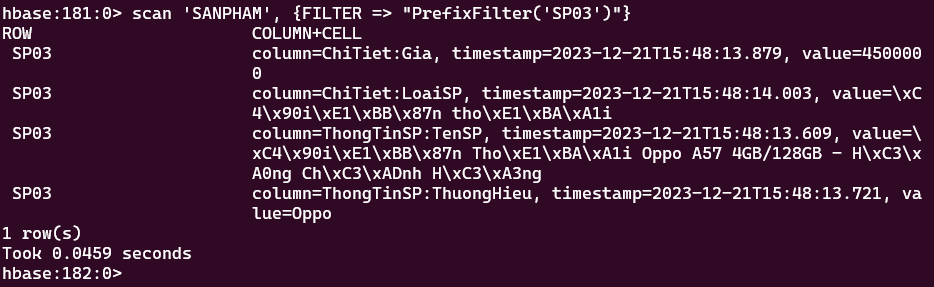
* **KeyOnlyFilter**

Bộ lọc sẽ chỉ trả về các giá trị key trong cặp key-value (value sẽ được viết lại thành trống). Bộ lọc này có thể được sử dụng để lấy tất cả các key mà không cần phải lấy cả các value.



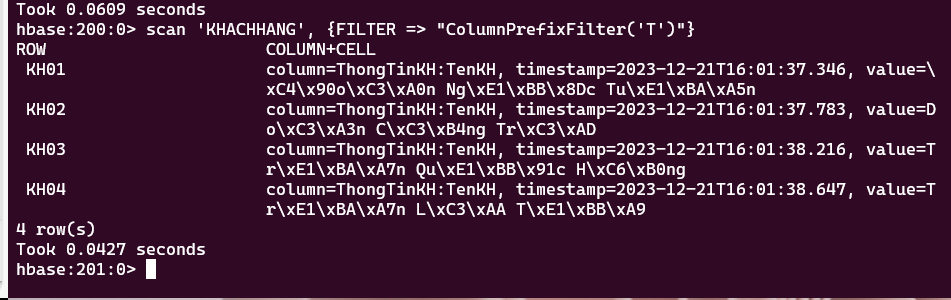
* **PrefixFilter**

Dùng để lọc các record dựa trên RowKey



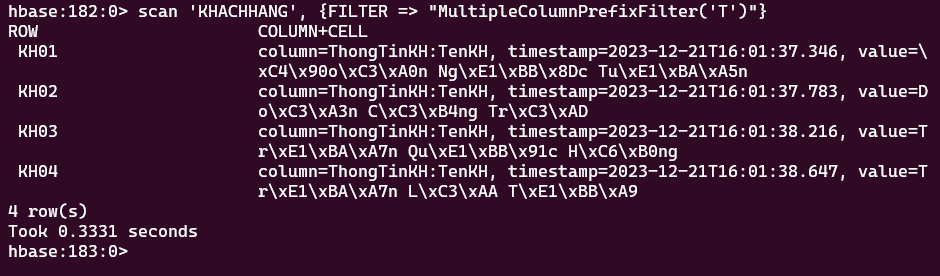
* **ColumnPrefixFilter**

Bộ lọc này được sử dụng để chỉ chọn những key có cột khớp với một tiền tố cụ thể. Dưới đây ta có tiền tố là **‘T’** nên các dòng có tên cột bắt đầu bằng **‘T’** được lọc ra, đó là cột **TenKH.**



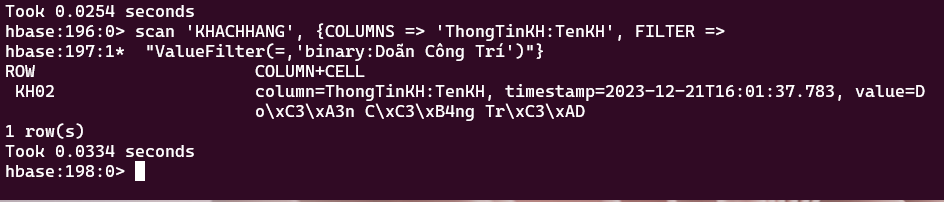
* **MutipleColumnPrefixFilter**

Nó hoạt động giống như bộ lọc ColumnPrefixFilter nhưng cho phép chỉ định nhiều tiền tố.



* **ValueFilter**

Dùng để lọc các records theo value đưa ra



### 

# **TÀI LIỆU THAM KHẢO**

1. <https://itnavi.com.vn/blog/hbase-la-gi#e752>
2. <https://blog.vietnamlab.vn/tong-quan-ve-hbase/>
3. <https://viblo.asia/p/hbase-overview-architecture-va-data-flow-63vKj6J6K2R>
4. <https://www.oreilly.com/library/view/hbase-the-definitive/9781449314682/ch01.html>
5. <https://www.simplilearn.com/tutorials/hadoop-tutorial/hbase#:~:text=Back%20in%20November%202006%2C%20Google,of%20Hadoop%20in%20January%202008>.
6. <https://fr.slideshare.net/tuanbv/hbase-29536077>
7. <https://youtu.be/--lax6MCs6k?si=rbnw2z_6e0BwFtwL>
8. <https://www.youtube.com/watch?v=PEL0e51oeaE>
9. <https://youtu.be/c2Lg5c8v4YQ?si=tQHaz4h2LW_LYhhB>
10. <https://blog.csdn.net/qq_45811072/article/details/121693142>