**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP.HCM**

**Ảnh có chứa văn bản, mẫu họa

Mô tả được tạo tự độngKHOA CÔNG NGHỆ THÔNG TIN**

-----🙞🙜🕮🙞🙜-----

**ĐỒ ÁN MÔN HỌC**

**ĐIỆN TOÁN ĐÁM MÂY**

**Đề tài: Tìm hiểu về Apache Hive và viết ứng dụng demo**

**GVHD: Huỳnh Xuân Phụng**

**Sinh viên thực hiện:**

* Phạm Trung Kiên 19110384
* Bùi Anh Đức 19110348
* Trần Việt Anh 19110325

**MỤC LỤC**

[CHƯƠNG 1: APACHE HIVE 1](#_Toc60843509)

[1. Tổng quan về Apache Hive 1](#_Toc60843510)

[**1.1.Vị trí, vai trò Hive trong hệ sinh thái Hadoop** 2](#_Toc60843511)

[**1.2.Đặc trưng của Hive** 2](#_Toc60843512)

[2.Hoạt động của Hive 3](#_Toc60843513)

[3.Kiến trúc Hive 4](#_Toc60843514)

[4.Tổ chức dữ liệu trong Hive 4](#_Toc60843515)

[**4.1.Tổ chức dữ liệu** 4](#_Toc60843516)

[**4.2.Kiểu dữ liệu** 5](#_Toc60843517)

[5.Ngôn ngữ truy vấn HiveQL 7](#_Toc60843518)

[CHƯƠNG 2: THIẾT KẾ VÀ XÂY DỰNG DATA WAREHOUSE 9](#_Toc60843519)

[1. Thiết kế Data warehouse 9](#_Toc60843520)

[**1.1.Thiết kế Business process và xây dựng Bus matrix** 9](#_Toc60843521)

[**1.2.Xây dựng Dimension và các bảng Fact** 9](#_Toc60843522)

[**1.2.1.Dimension Customers** 9](#_Toc60843523)

[**1.2.2.DimEmployees** 10](#_Toc60843524)

[**1.2.3.DimProducts** 10](#_Toc60843525)

[**1.2.4.Dimension Date** 11](#_Toc60843526)

[**1.2.5.Bảng Fact Oders** 11](#_Toc60843527)

[**1.3**.**Schema** 12](#_Toc60843528)

[2. Thực hiện xây dựng Data warehouse 12](#_Toc60843529)

[**2.1.Stagging dữ liệu** 12](#_Toc60843530)

[**2.2.Load dữ liệu vào DW** 15](#_Toc60843531)

[**2.2.1.Load FactOrder** 15](#_Toc60843532)

[**2.2.2.Load Dimentions** 16](#_Toc60843533)

[3.Truy vấn và báo cáo 18](#_Toc60843534)

[3.1.Số sản phẩm và số tiền bán được của từng nhân viên 18](#_Toc60843535)

[3.2.Số lượng bán được của từng sản phẩm 19](#_Toc60843536)

[3.3.Khách hàng mua nhiều sản phẩm nhất 19](#_Toc60843537)

[4.Kết luận 20](#_Toc60843538)

# Chương 1: APACHA HIVE

## Tổng quan về Hive

Apache Hive là 1 kho dữ liệu (data warehouse) hỗ trợ người sử dụng có thể dễ dàng hơn trong việc quản lý và truy vấn đối với các tập dữ liệu lớn được lưu trữ trên các hệ thống lưu trữ phân tán (distributed storage). Hive được xây dựng dựa trên cơ sở của Apache Hadoop, nó cung cấp các tính năng chính sau:

* Công cụ cho phép dễ dàng thực hiện tác vụ như trích xuất, vận chuyển và lưu trữ dữ liệu.
* Cơ chế để xử lý cho nhiều định dạng dữ liệu khác nhau.
* Truy cập tới dữ liệu dạng files được lưu trữ trực tiếp ở trong Apache HDFS hoặc đối với nhiều hệ thống lưu trữ dữ liệu khác như Apache HBase.
* Thực hiện query thông qua MapReduce.

Hive định nghĩa ra một ngôn ngữ truy vấn đơn giản có cú pháp gần giống với SQL (SQL-like query language) được gọi là HiveQL, nó cho phép người sử dụng đã quen thuộc với các truy vấn SQL thực hiện việc truy vấn dữ liệu. Ngoài ra ngôn ngữ này còn cho phép các lập trình viên người đã quen thuộc với MapReduce framework có thể nhúng các mappers và reducers cho chính họ viết ra để thực thi nhiều hơn nữa các phân tích phức tập mà không được hỗ trợ bởi các hàm đã có sẵn trong ngôn ngữ HiveQL. HiveQL cung có thể được mở rộng với các custom scalar functions (UDF’s), aggregations (UDAF’s) và các table funtions (UDTF’s)

Hive không yêu cầu dữ liệu phải được đọc và ghi dưới một định dạng của riêng Hive (Hive format). Hive hoạt động tốt trên Thrift và các định dạng dữ liệu riêng của người sử dụng.

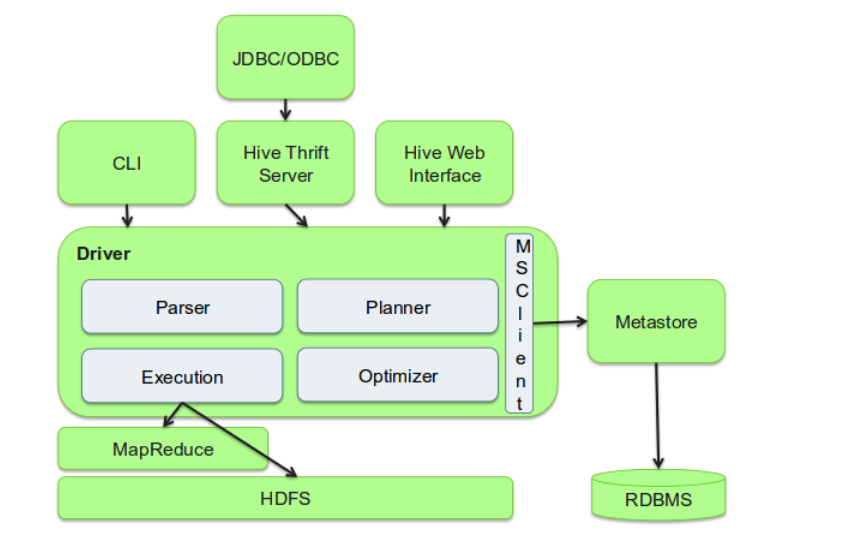
Hive không được thiết kế để cho các giao dịch online (OLTP workloads) và không nên dùng cho các real-time queries và các cập nhật trên từng dòng trong 1 table (row-level). Hive hoạt động tốt nhất cho các batch jobs trên các tập dữ liệu lớn, mà ở đó dữ liệu được thêm vào liên tục (append-only data) ví dụ như web logs. Hive có khả năng mở rộng theo chiều ngang tốt (thực thi tốt trên 1 hadoop cluster có số tượng máy biến đổi), có khả năng tích hợp với MapReduce framework và UDF, UDAF, UDTF; có khả năng chống chịu lỗi và mềm dẻo đối với các dữ liệu đầu vào của chính nó.

Các thành phần cấu hình Hive bao gồm HCatalog và WebHCat. HCatalog là một thành phần của Hive. Đây là lớp quản lý lưu trữ cho Hadoop (table and management layer), nó cho phép người dùng với các công cụ xử lý dữ liệu khác nhau bao gồm cả Pig và MapReduce thực thi hoạt động đọc, ghi một cách dễ dàng hơn. WebHCat cung cấp một dịch vụ cho phép bạn có thể thực thi Hadoop MapReduce (hoặc YARN), Pig, Hive.

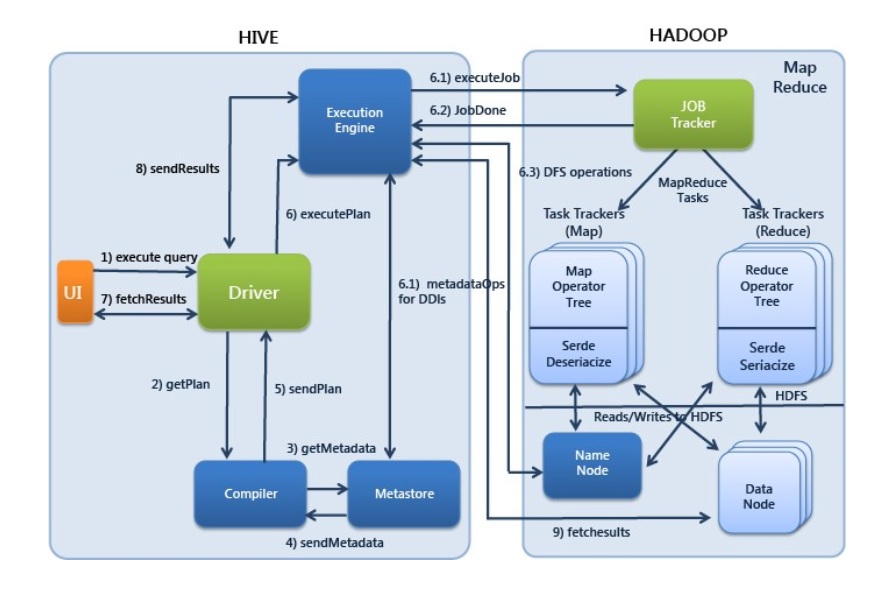
## Kiến trúc của Hive

Hive có các thành phần chính là:

* Hive UI: cung cấp giao diện cho phép người sử dụng tương tác với hệ thống Hive. Hive cung cấp nhiều phương thức khác nhau cho phép người sử dụng tương tác với Hive:
* CLI: giao diện dạng shell cho phép người sử dụng tương tác trực tiếp qua command line.
* Hive Web Interface: giao diện Web cho phép người sử dụng thực hiện các truy vấn thông qua giao diện Web.
* Hive Thrift Server: cho phép các client từ nhiều ngôn ngữ lập trình khác nhau có thể thực hiện tương tác với Hive.
* Hive Driver: thành phần nhận các truy vấn và chuyển các truy vấn này thành các MapReduce Jobs để tiến hành xử lý yêu cầu của người sử dụng.
* Driver: nhận các truy vấn, thành phần này thực hiện việc quản lý các sessions và cung cấp các API để thực thi và lấy dữ liệu trên JDBC/ODBC interfaces.
* Compiler: thành phần hiện việc phân tích ngữ nghĩa đối với các query, lấy các thông tin metadata cần thiết về table và partion từ metastore để sinh ra các excution plan.
* Execute engine: thành phần thực thi các execution plan được tạo bởi compiler (submit các job tới MapReduce). Ngoài ra thành phần execution enginen này thực hiện việc quản lý các dependencies của các bước trong mỗi execution plan, thực thi từng bước này.
* Hive Metastore: thành phần lưu trữ các metadata của Hive: table, partion, buckets bao gồm cả thông tin về các column trong mỗi table, các serializers và desrializers cần thiết để thực hiện việc đọc và ghi dữ liệu. Metastore sử dụng một cơ sở dữ liệu quan hệ để lưu trữ dữ liệu của chính mình.



## Động của Hive

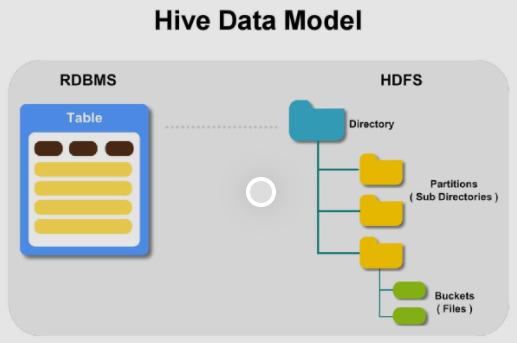


Quy trình hoạt động của Hive có thể được mô tả theo các bước sau:

1. Các truy vấn tới từ User Interface (CLI, Hive Web Interface, Thirft Server) được gửi tới thành phần Driver (Bước 1 hình 3.1)
2. Driver tạo ra mới 1 session cho truy vấn này và gửi query tới compiler để nhận lấy Execution Plan (Bước 2 hình 3.1)
3. Compilter nhận các metadata cần thiết từ Metastore (Bước 3, 4 hình 3.1). Các metadata này sẽ được sử dụng để kiểm tra các biểu thức bên trong query mà Compiler nhận được.
4. Plan được sinh ra bởi Compiler (thông tin về các job (map-reduce) cần thiết để thực thi query sẽ được gửi lại tới thành phần thực thi (Bước 5hình 3.1)
5. Execution engine nhận yêu cầu thực thi và lấy các metadata cần thiết và yêu cầu mapreduce thực thi công việc (Bước 6.1, 6.2, 6.3 hình 3.1)
6. Khi output được sinh ra, nó sẽ được ghi dưới dạng 1 temporary file, temorary file này sẽ cung cấp các thông tin cần thiết cho các stages khác của plan. Nội dung của các temporary file này được execution đọc trực tiếp từ HDFS như là 1 phần của các lời gọi từ Driver (bước 7, 8, 9 hình 3.1)

## Mô hình dữ liệu trong Hive

### 1.4.1 Tổ chức dữ liệu



Dữ liệu trong Hive được tổ chức thành các kiểu sau:

* Databases: là namespace cho các tables, dùng để nhóm và quản lý các nhóm tables khác nhau.
* Tables: tương tự như table trong các hệ cơ sở dữ liệu quan hệ. Trong Hive table có thể thực hiện các phép toán filter, join và union… Mặc định thì dữ liệu của Hive sẽ được lưu bên trong thư mục warehouse trên HDFS. Tuy nhiên Hive cũng cung cấp kiểu external table cho phép ta tạo ra và quản lý các table mà dữ liệu của nó đã tồn tại từ trước khi ta tạo ra table này hoặc nó được lưu trữ ở 1 thư mục khác bên trong hệ thống HDFS. Tổ chức row và column bên trong Hive có nhiều điểm tương đồng với tổ chức Row và Column trong các hệ cơ sở dữ liệu quan hệ. Hive có 2 kiểu table đó là: Managed Table và External tables.
* Partions: Mỗi table có thể có 1 hoặc nhiều các khóa mà từ đó xác định dữ liệu sẽ được lưu trữ ở đâu. Ví dụ table web\_log có thể phân chia dữ liệu của mình theo từng ngày là lưu dữ liệu của mỗi ngày trong 1 thư mục khác nhau bên dưới đường dẫn warehouse.

Ví dụ: /warehouse/web\_log/date=”01-01-2014″

* Buckets: Dữ liệu trong mỗi partion có thể được phân chia thành nhiều buckets khác nhau dựa trên 1 hash của 1 colume bên trong table. Mỗi bucket lưu trữ dữ liệu của nó bên dưới 1 thư mục riêng. Việc phân chia các partion thành các bucket giúp việc thực thi các query dễ dàng hơn.

### 1.4.2 Kiểu dữ liệu

Kiểu dữ liệu nguyên thủy:

Mỗi columns có 1 kiểu dữ liệu cố định. Các kiểu dữ liệu nguyên thủy sau sẽ được hỗ trợ đối với Hive:

* Integers:
* TINYINT – 1 byte integer
* SMALLINT – 2 byte integer
* INT – 4 byte integer
* BIGINT – 8 byte integer
* Boolean type
* BOOLEAN – TRUE/FALSE
* Floating point numbers
* FLOAT – single precision
* DOUBLE – Double precision
* String type
* STRING – sequence of characters in a specified character set

Các kiểu dữ liệu khác:

* Structs: là kiểu dữ liệu mà mỗi phần tử bên trong đó có thể được truy cập thông qua việc sử dụng ký hiệu (.)

Ví dụ, với kiểu dữ liệu STRUCT {a INT; b INT} ví dụ trường a của nó có thể truy cập thông qua c.a

* Maps (key-value tuples): là kiểu dữ liệu mà các phần tử sẽ được truy cập thông qua ký hiệu [‘element name’]. Đối với map M thực hiện việc map dữ liệu đối với khóa ‘group’ -> thì dữ liệu sẽ được sử dụng bởi trường M[‘group’]
* Arrays (indexable lists): Kiểu mảng.

## HiveSQL (HQL)

Ngôn ngữ truy vấn Hive cung cấp các toán tử cơ bản giống SQL. Đây là một số tác vụ mà HQL có thể làm dễ dàng.

* Tạo và quản lý tables và partitions.
* Hỗ trợ các toán tử Relational, Arithmetic và Logical khác nhau.
* Evaluate functions
* Tải về nội dung 1 table từ thư mục cục bộ hoặc kết quả của câu truy vấn đến thư mục HDFS.

Đây là ví dụ truy vấn HQL:

SELECT upper(name), salesprice

FROM sales;

SELECT category, count(1)

FROM products

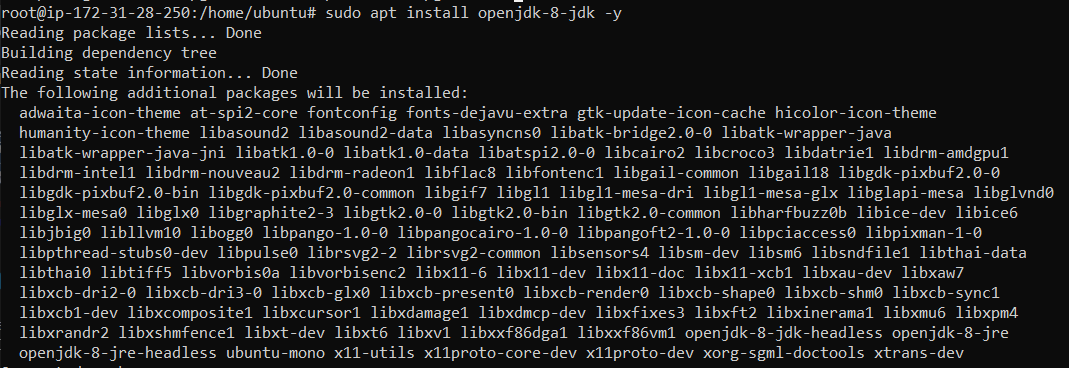
GROUP BY category;

# Chương 2: CÀI ĐẶT, THIẾT KẾ VÀ XÂY DỰNG

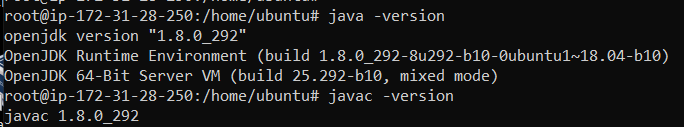
## 2.1 Cài đặt

### 2.1.1 Cài đặt môi trường Hadoop

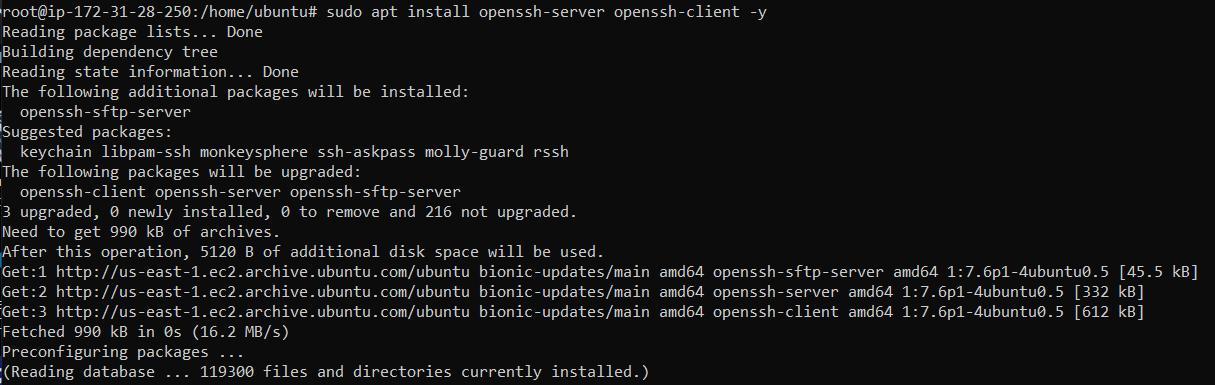
Cài đặt OpenJDK



Kiểm tra phiên bản java

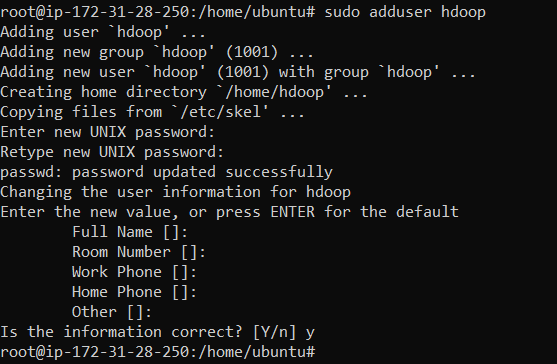


Cài đặt OpenSSH

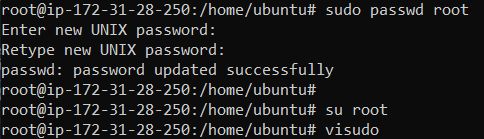


Sau khi cài đặt OpenJDK và OpenSSH, ta cần 1 người dùng để sử dụng để tiếp tục cài đặt Hadoop

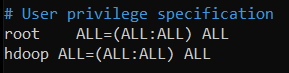
Ở đây, tụi em tạo user tên là hdoop



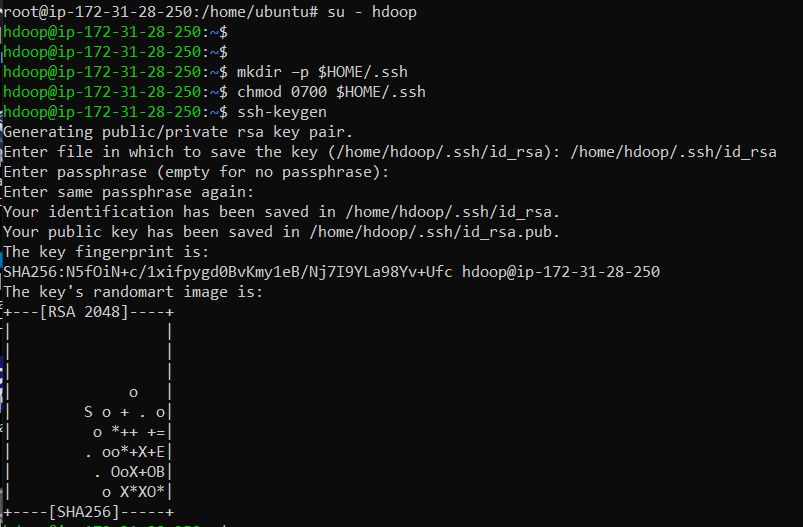
Sau khi tạo xong, ta cần thêm quyền root vào user vừa tạo



Sử dụng lệnh visudo và thêm quyền root vào dưới root ALL=(ALL:ALL) ALL



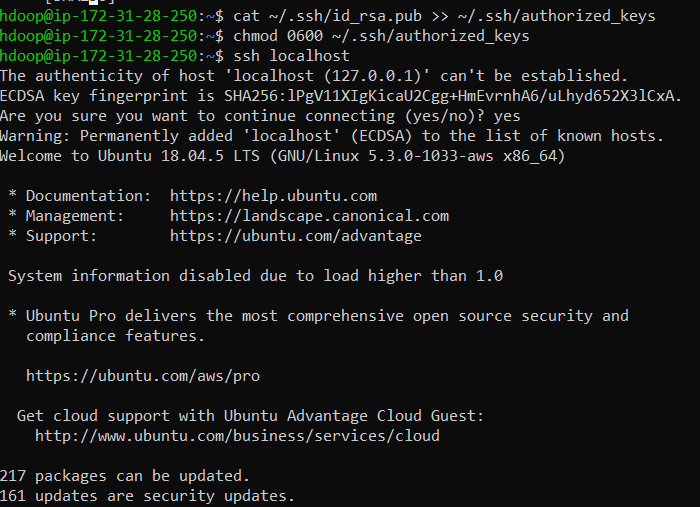
Bật không mật khẩu cho người dùng hdoop để tránh phải nhập lại mật khẩu nhiều lần



Sử dụng lệnh cat để lưu trữ khóa công khai dưới dạng allow\_keys trong thư mục ssh

Sau đó đặt quyền cho user bằng lệnh chmod

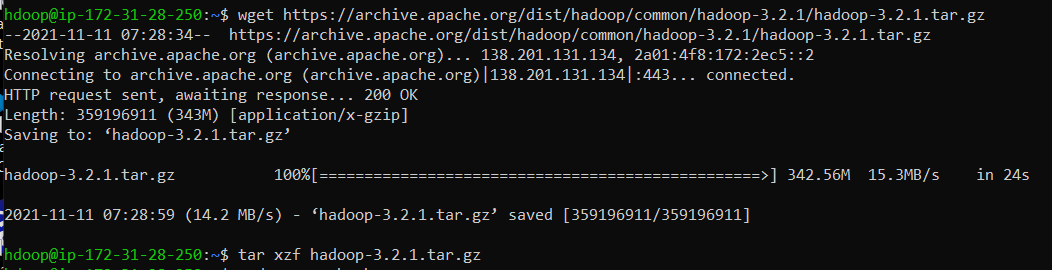
Người dùng sau đó có thể ssh mà không cần dùng mật khẩu mỗi lần. Có thể kiểm tra mọi thứ đã thiết lập chính xác bằng cách sử dụng user hdoop đã tạo ở trên để ssh vào localhost



Tiếp theo là tải gói hadoop về, ở đây chúng em sử dụng bản 3.2.1

Link tham khảo tải: <https://archive.apache.org/dist/hadoop/common/hadoop-3.2.1/hadoop-3.2.1.tar.gz>

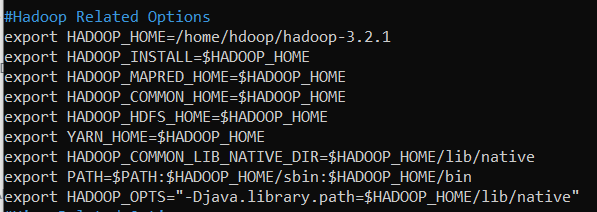
Sau khi tải xong, chúng ta giải nén bằng lệnh tar xzf



Tiếp theo ta cần cấu hình các biến môi trường Hadoop(bashrc)



Xác định nội dung biến môi trường bằng các lệnh sau:

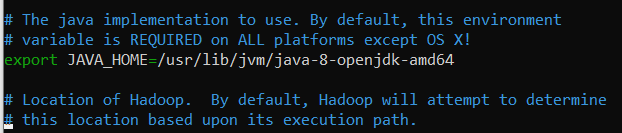


Sau khi xác định xong, ta lưu lại bằng lệnh sau

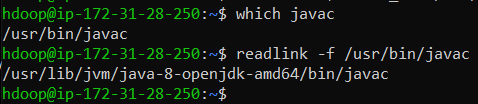
Chỉnh sửa tệp hadoop.env.sh



Bỏ ghi chú biến $JAVA\_HOME(bỏ dấu ‘#’) và thêm đường dẫn đầy đủ đến cài đặt OpenJDK trên hệ thống



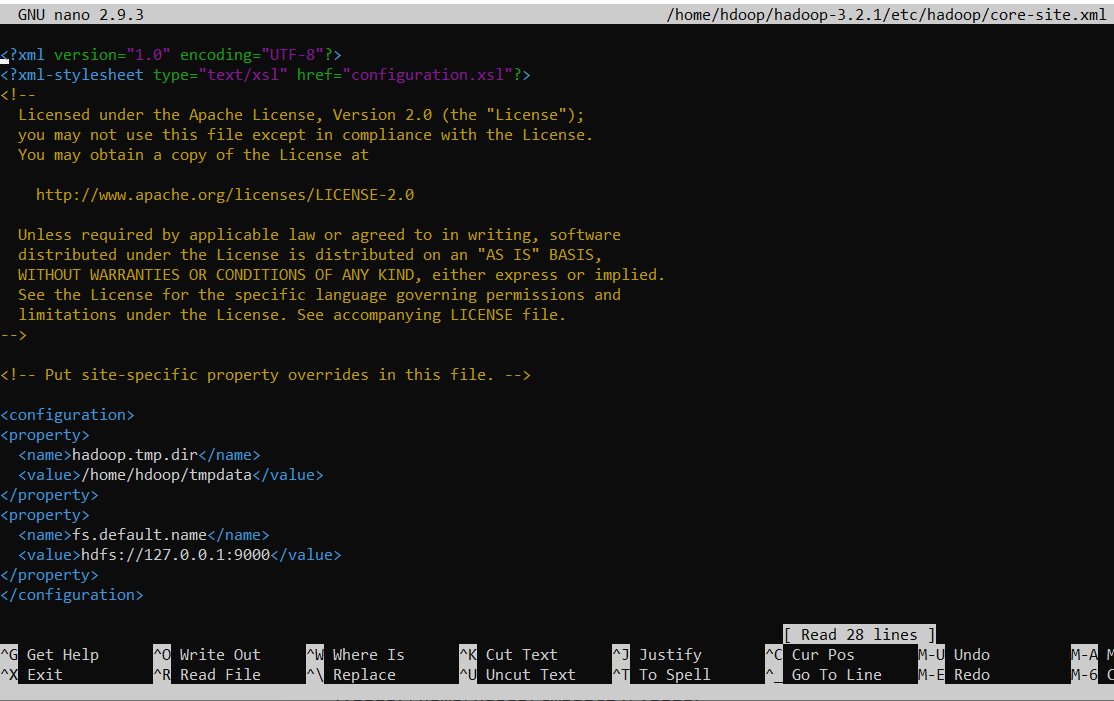
Nếu như không biết đường dẫn Java chính xác, hãy chạy lệnh sau:



Chỉnh sửa tệp core-site.xml



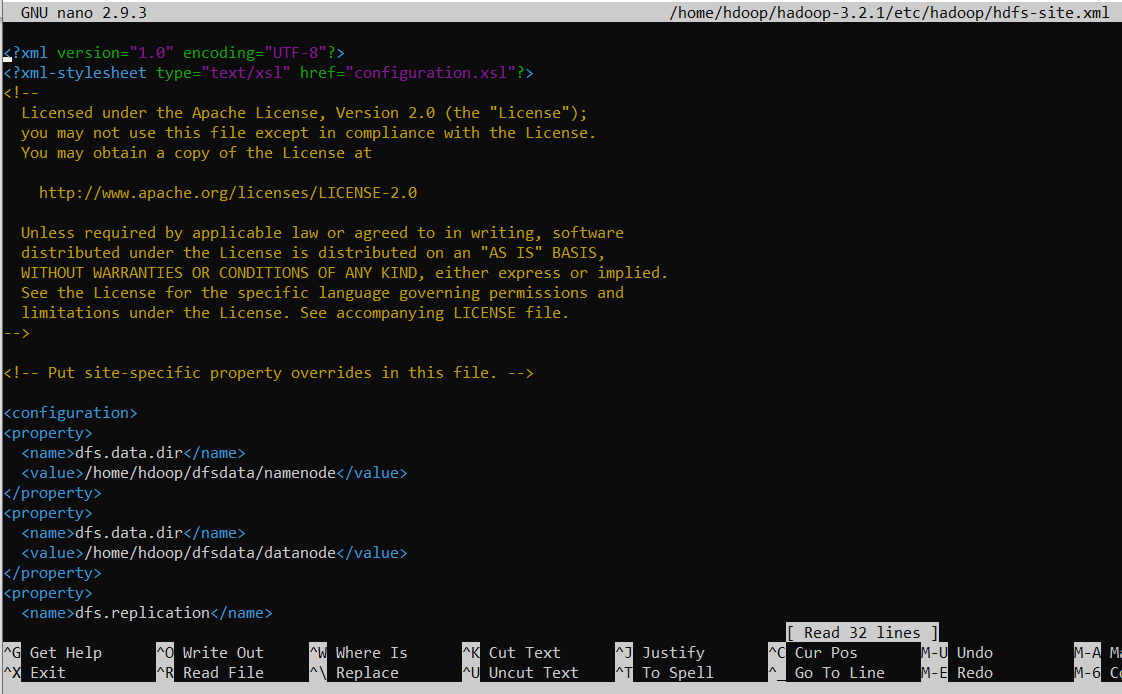
Thêm cấu hình sau để ghi đè các giá trị mặc định cho thư mục tạm thời và thêm URL HDFS để thay thế cài đặt hệ thống tệp cục bộ mặc định:



Chỉnh sửa tệp hdfs-site.xml



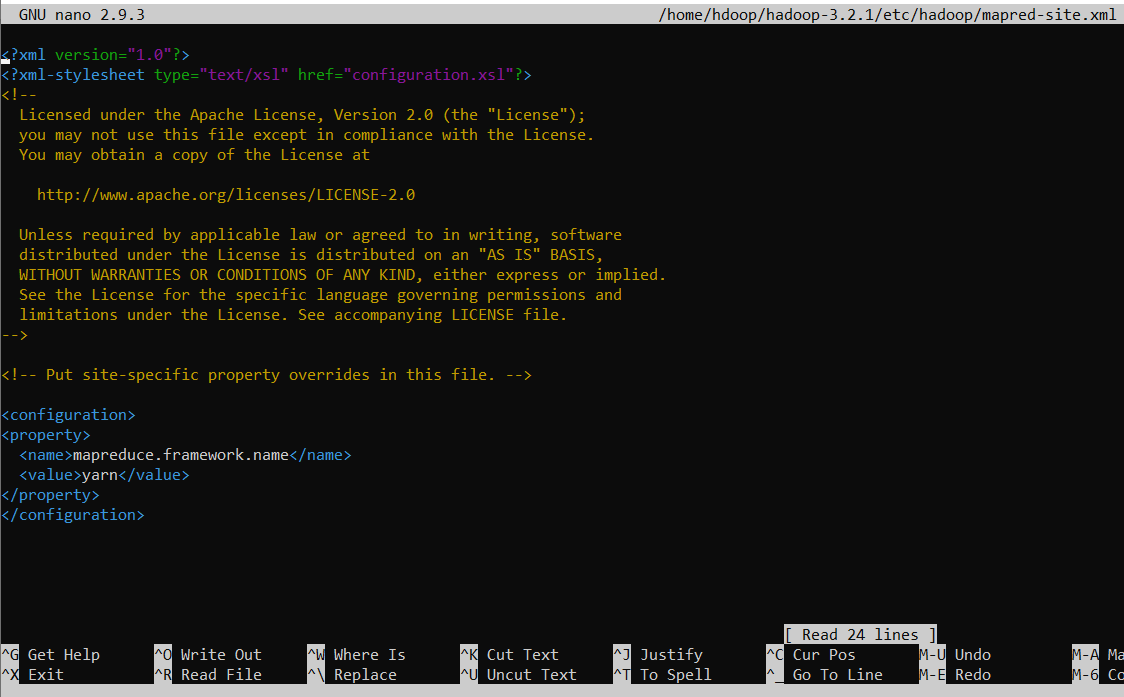
Thêm cấu hình sau vào tệp và nếu cần, hãy điều chỉnh các thư mục NameNode và DataNode cho các vị trí tùy chỉnh:



Chỉnh sửa tệp mapred-site.xml



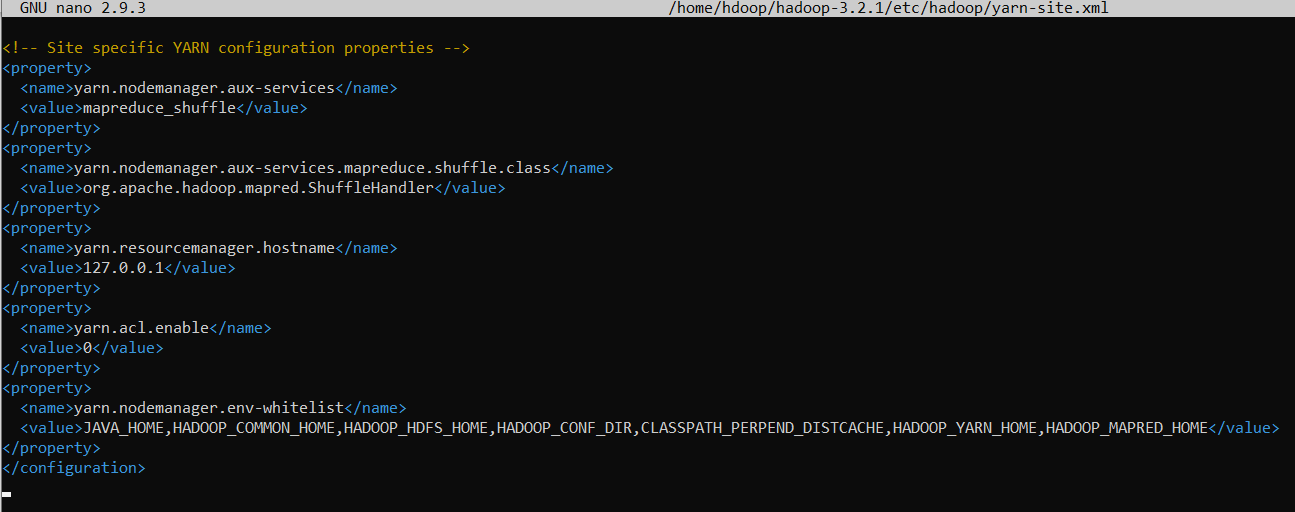
Thêm cấu hình sau để thay đổi giá trị MapReduce framework mặc định thành yarn:



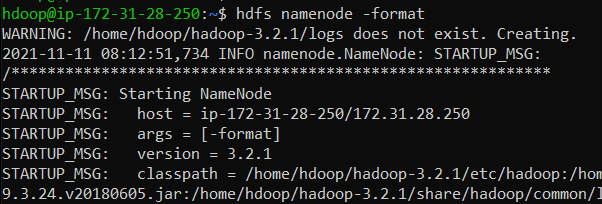
Chỉnh sửa tập tin yarn-site.xml

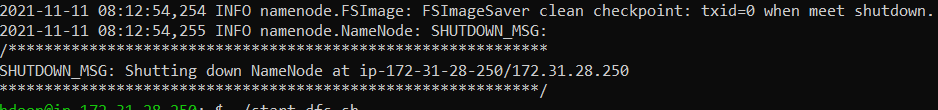


Thêm các cấu hình sau vào file

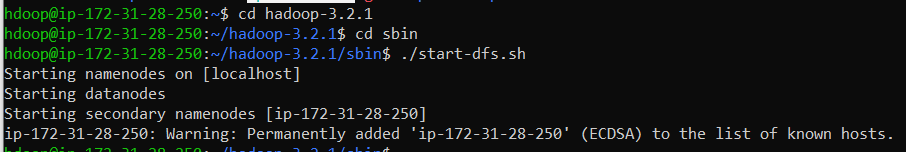


Định dạng HDFS NameNode:



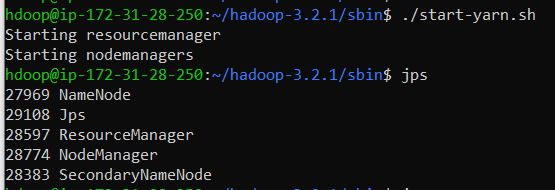


Khởi động Hadoop Cluster



Như với lệnh trước, đầu ra thông báo rằng các quá trình đang bắt đầu

Nhập lệnh đơn giản jps để kiểm tra xem tất cả các daemon có đang hoạt động và đang chạy như các quy trình Java hay không

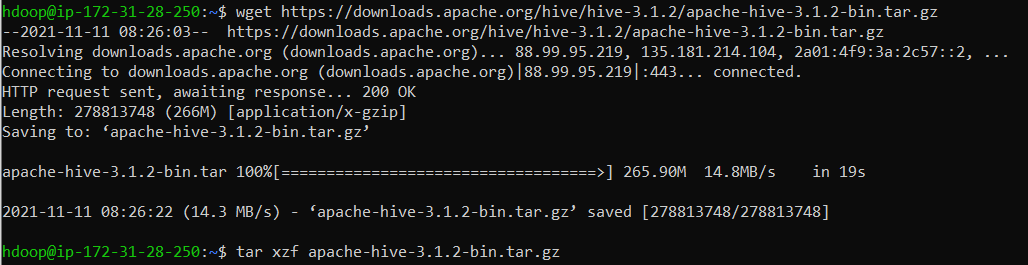


### 2.1.2 Cài đặt Apache Hive

Tải gói apache hive về, chúng em sử dụng bản 3.1.2

Link tham khảo tải: <https://downloads.apache.org/hive/hive-3.1.2/apache-hive-3.1.2-bin.tar.gz>

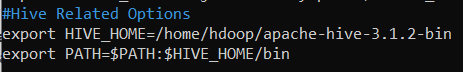
Sau khi tải xong, giải nén bằng lệnh tar xzf



Cấu hình các biến môi trường Hive (bashrc)



Thêm các biến môi trường hive sau vào tệp .bashrc



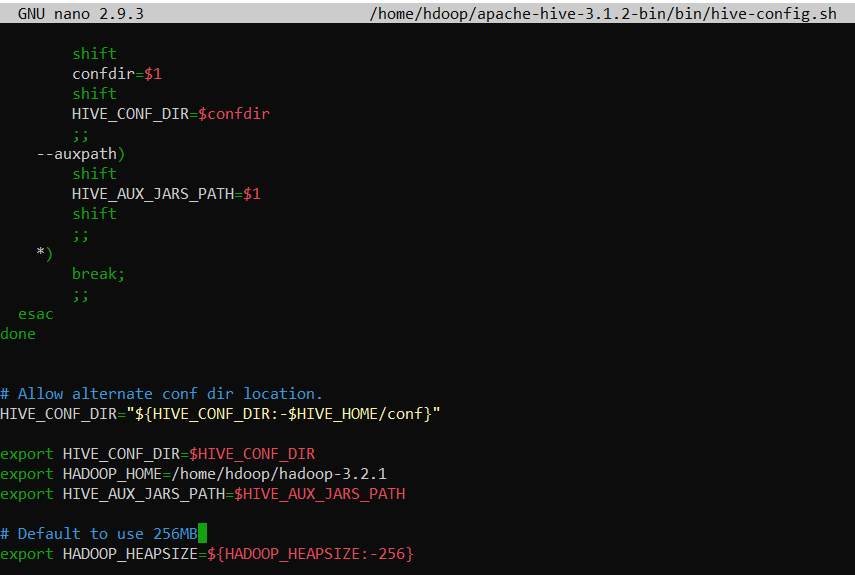
Sau khi cấu hình xong, chạy lệnh sau để lưu tệp



Chỉnh sửa tập hive-config.sh



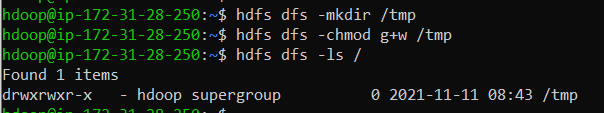
Thêm biến HADOOP\_HOME và đường dẫn đầy đủ đến thư mục Hadoop



Tạo đường dẫn thư mục Hive trong HDFS

Thêm quyền ghi và thực thi cho file tmp

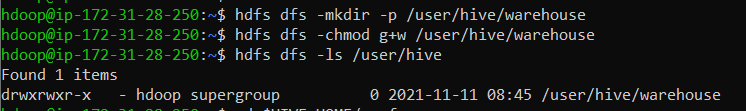
Kiểm tra quyền bằng lệnh –ls/



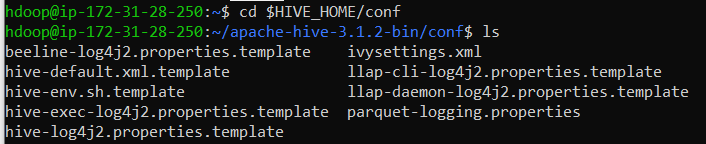
Tạo thư mục warehouse

Thêm quyền ghi và thực thi cho warehouse

Kiểm tra bằng lệnh –ls/



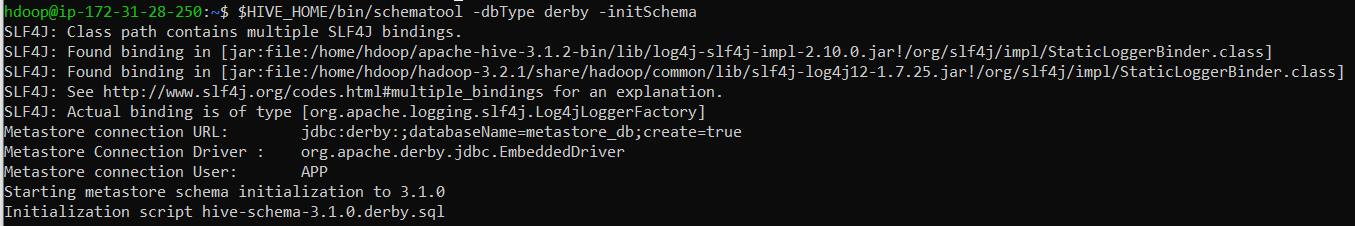
Cấu hình tập hive-site.xml



Sử dụng hive-default.xml.template để tạo tệp hive-site.xml



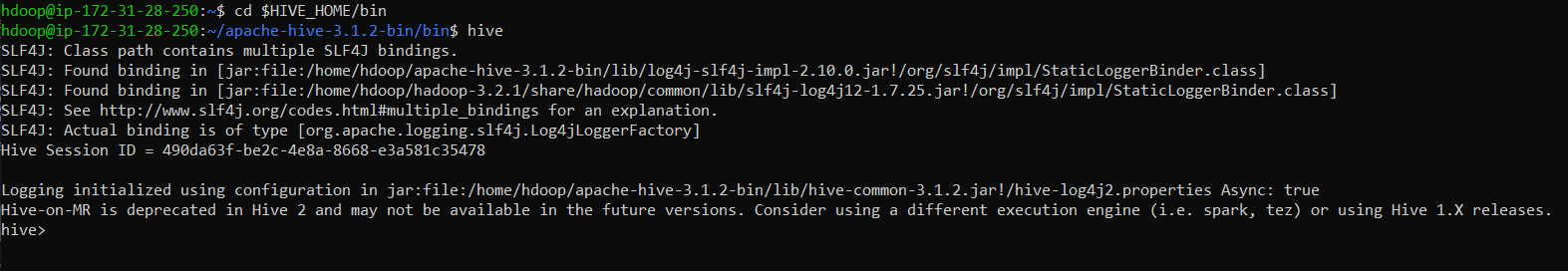
Khởi tạo Derby Database



Sau khi khỏi tạo xong sẽ hiện ra schemaTool conpleted

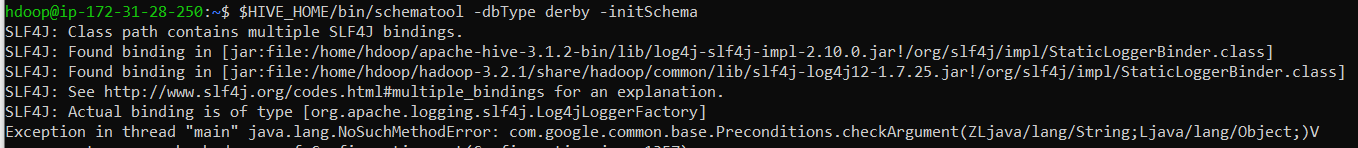


Sau đó ta có thể sử dụng Hive



### 2.1.3 Lỗi phát sinh trong quá trình cài đặt

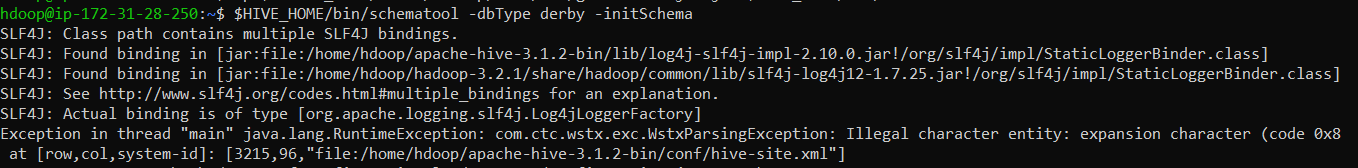
Lỗi 1:



Ta cần xóa guave của hive, rồi copy guave của hadoop qua hive để đồng bộ bằng cách sau:



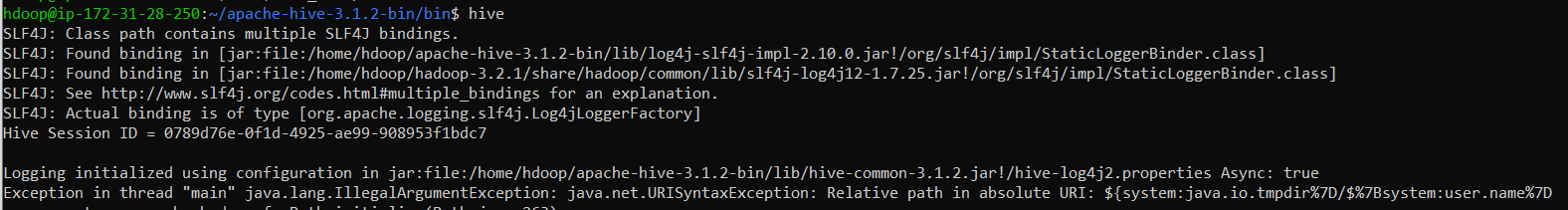
Lỗi 2:



Xóa những kí tự đặc biệt bị dư, vào file xml xóa, dòng 3215 trong file sau:



Lỗi 3:



Thêm property sau vào đầu file hive-site.xml:

<property>

<name>system:java.io.tmpdir</name>

<value>/tmp/hive/java</value>

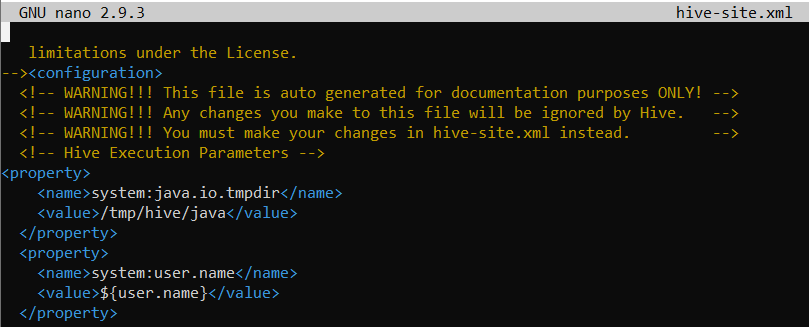
</property>

<property>

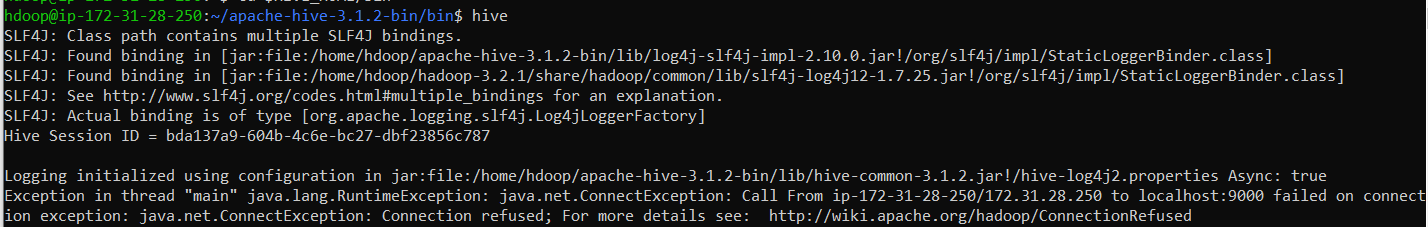
<name>system:user.name</name>

<value>${user.name}</value>

</property>



Lỗi 4:



Exception in thread "main" java.lang.RuntimeException: java.net.ConnectException: Connection refused

Sửa bằng cách thực hiện các bước sau:







