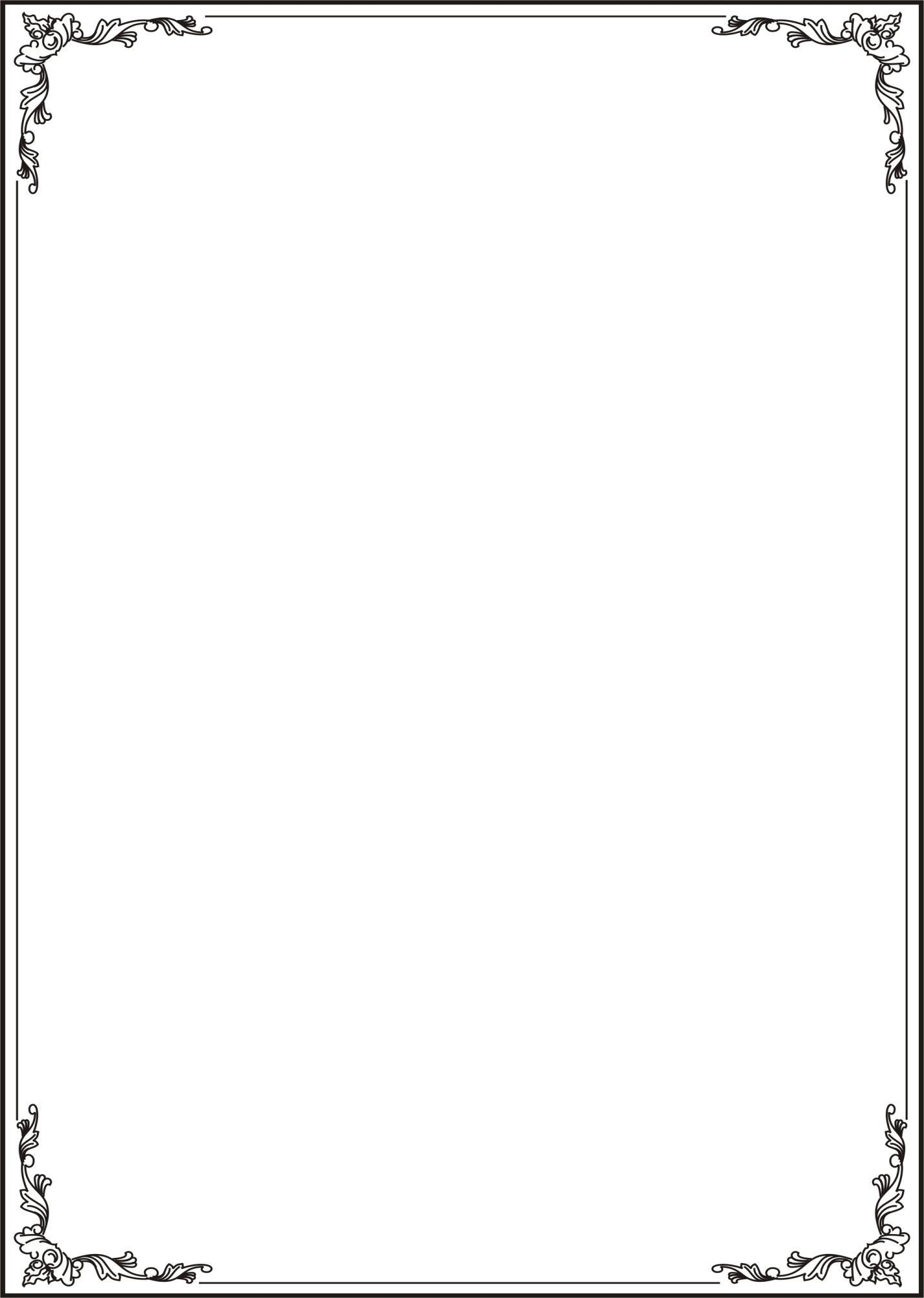
**.**

**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH**

**KHOA ĐÀO TẠO CHẤT LƯỢNG CAO**

**BỘ MÔN ĐIỆN TOÁN ĐÁM MÂY**

-----🙞🙜🕮🙞🙜-----

****

**TÌM HIỂU APACHE HIVE VÀ XÂY DỰNG**

**DATA WAREHOUSE**

Khóa 2019-2022

**GVHD: TS. HUỲNH XUÂN PHỤNG**

**SVTH:**

**LƯU ĐẶC VŨ 20110044**

**LÊ KHÁNH AN 20119316**

**LÊ CÔNG TRÌNH 20110408**

Khóa 2019 -2023

**TP.HCM 11/2022**

**MỤC LỤC**

[LỜI CẢM ƠN 3](#_Toc121529992)

[MỞ ĐẦU 4](#_Toc121529993)

[**1.** **Tính cấp thiết của đề tài** 4](#_Toc121529994)

[**2.** **Đối tượng nghiên cứu** 4](#_Toc121529995)

[**3.** **Phạm vi nghiên cứu** 4](#_Toc121529996)

[**4.** **Kết quả dự kiến** 4](#_Toc121529997)

[NỘI DUNG 5](#_Toc121529998)

[Chương 1: Cơ sở lý thuyết 5](#_Toc121529999)

[**1.** **Tổng quan về Apache Hive:** 5](#_Toc121530000)

[**2.** **Kiến trúc của Apache Hive:** 6](#_Toc121530001)

[**3.** **Quá trình hoạt động của Apache Hive:** 7](#_Toc121530002)

[**4.** **Tổ chức dữ liệu trong Apache Hive:** 8](#_Toc121530003)

[**5.** **Các kiểu dữ liệu trong Apache Hive:** 9](#_Toc121530004)

[Chương 2: Cài đặt, thiết kế và xây dựng Data warehouse 9](#_Toc121530005)

[**1.** **Cài đặt Apache Hadoop:** 9](#_Toc121530006)

[**2.** **Cài đặt Apache Hive:** 14](#_Toc121530007)

[**3.** **Thiết kế cơ sở dữ liệu:** 18](#_Toc121530008)

[**4.** **Cài đặt Data warehouse với EMR trên AWS** 18](#_Toc121530009)

[KẾT LUẬN 19](#_Toc121530010)

[TÀI LIỆU THAM KHẢO 20](#_Toc121530011)

# **LỜI CẢM ƠN**

Lời đầu tiên nhóm xin phép được gửi lời cảm ơn chân thành đến Khoa Đào Tạo Chất Lượng Cao – Trường Đại Học Sư Phạm Kỹ Thuật Thành Phố Hồ Chí Minh đã tạo điều kiện cho nhóm chúng em được học tập, phát triển nền tảng kiến thức sâu sắc để có thể thực hiện đề tài này.

Bên cạnh đó chúng em xin gửi lời cảm ơn đến thầy Huỳnh Xuân Phụng đã chỉ dạy những kiến thức bổ ích của bộ môn Điện Toán Đám Mây và hướng dẫn chúng em thực hiện đề tài TÌM HIỂU APACHE HIVE VÀ XÂY DỰNG DATA WAREHOUSE.

Chúng em xin cảm ơn thầy vì thời gian qua đã hướng dẫn và chỉ dạy chúng em nhiệt tình giúp chúng em nắm được những kiến thức bổ ích của môn học này.

Tuy nhiên, trong quá trình tìm hiểu đề tài, chúng em có thể còn nhiều thiếu sót nên mong thầy có thể thông cảm và góp ý để chúng em có thể hoàn thiện đề tài hơn.

Chúng em xin chân thành cảm ơn!

# **MỞ ĐẦU**

1. **Tính cấp thiết của đề tài**

Hiện nay, thuật ngữ Big Data được sử dụng cho các bộ tập dữ liệu khổng lồ bao gồm khối lượng lớn, tốc độ cao và nhiều loại dữ liệu đang tăng lên từng ngày. Sử dụng các hệ thống quản lý dữ liệu truyền thống, rất khó để xử lý Big Data. Do đó, Quỹ phần mềm Apache (Apache Software Foundation) đã giới thiệu một framework có tên là Hadoop. Trong đó, Hive là một công cụ để xử lý dữ liệu có cấu trúc trong Hadoop được dùng để tóm tắt Big Data và giúp cho việc truy vấn và phân tích trở nên dễ dàng hơn.

1. **Đối tượng nghiên cứu**

Đối với đề tài này, đối tượng nghiên cứu là Big Data. Đồng thời kèm theo đó là các công nghệ áp dụng để xây dựng warehouse đơn giản, cụ thể như:

* JDK và JRE: Bộ công cụ phát triển Java.
* Hadoop: Apache framework mã nguồn mở cho phép phát triển các ứng dụng phân tán (distributed processing) để lưu trữ và quản lý các tập dữ liệu lớn.
* Apache Hive: Công cụ cơ sở hạ tầng kho dữ liệu để xử lý dữ liệu có cấu trúc trong Hadoop.
* Apache Derby: External database để cấu hình Metastore.
* Amazon và EMR (Elastic MapReduce)

1. **Phạm vi nghiên cứu**

Đề tài này chủ yếu tập trung vào việc xử lý và phân tích các câu truy vấn đối với dữ liệu lớn khi sử dụng Hive.

1. **Kết quả dự kiến**

* Cài đặt được Apache Hadoop và Apache Hive
* Xây dựng được một data warehouse đơn giản bằng các câu truy vấn.
* Phân tích, thống kê các dữ liệu từ các câu truy vấn tùy vào mục đích của người dùng.
* Sử dụng EMR (Elastic MapReduce) trên AWS để cài đặt xử lý data warehouse

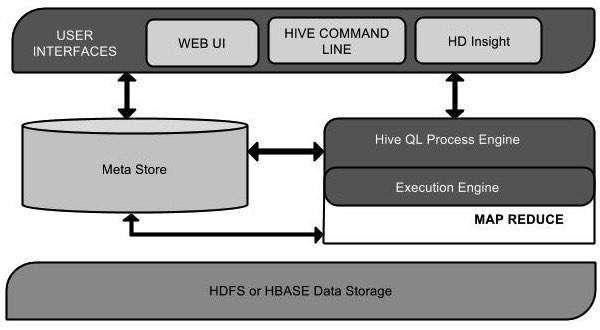
# **NỘI DUNG**

# **Chương 1: Cơ sở lý thuyết**

1. **Tổng quan về Apache Hive:**

* Hive là một công cụ cơ sở hạ tầng kho dữ liệu để xử lý dữ liệu có cấu trúc trong Hadoop. Nó nằm trên đỉnh Hadoop để tóm tắt Dữ liệu lớn nên giúp cho việc truy vấn và phân tích trở nên dễ dàng mà không cần hiểu biết nhiều về MapReduce.
* Cụ thể hơn, 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 và 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 không phải là một cơ sở dữ liệu quan hệ nhưng 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.

1. **Kiến trúc của Apache Hive:**

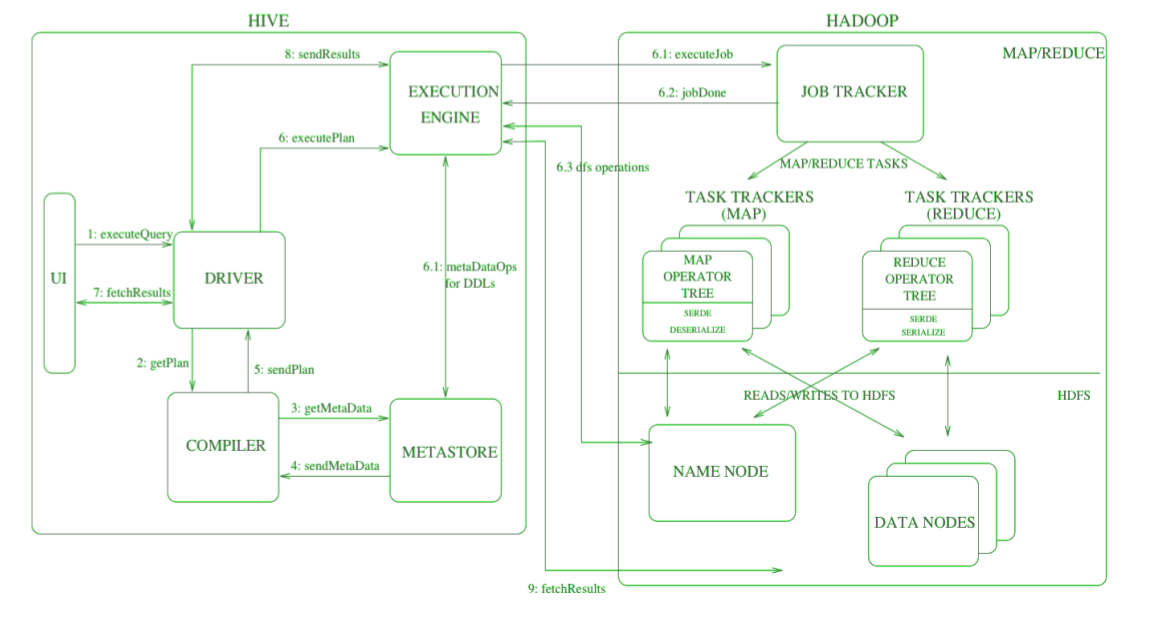


***Hình 1: Kiến trúc của Hive***

Kiến trúc của Hive có rất nhiều thành phần khác nhau, tuy nhiên, có 5 thành phần chính được sử dụng nhiều nhất:

* User Interface: chính là giao diện người dùng mà Hive hỗ trợ, bao gồm: Hive Web UI, Hive command line và Hive HD Insight, nó giúp tạo ra sự tương tác giữa người dùng với HDFS.
* Meta Store: đây chính là nơi mà Hive chọn các máy chủ cơ sở dữ liệu để lưu trữ như: các loại lược đồ, các metadata, các cột, các bảng, các loại dữ liệu trong một bảng, một cột và dữ liệu ánh xạ của HDFS.
* HiveQL Process Engine: HiveQL làm việc tương tự nhe SQL để truy vấn các thông tin về lược đồ trên hệ thống. Ngoài ra, đây còn là một phương pháp nhằm thay thế cho chương trình MapReduce. Vì thế, các lập trình viên thay vì phải viết chương trình MapReduce bằng Java tương đối phức tạp và mất khá nhiều thời gian, thì họ có thể viết những câu truy vấn bằng HiveQL để xử lý công việc được dễ dàng hơn.
* Execution Engine: đây là phần kết hợp giữa 2 công cụ xử lý: HiveQL + MapReduce, và nó chính là công cụ thực thi Hive Execution Engine. Công cụ này giúp thực thi và xử lý các câu truy vấn dữ liệu.
* HDFS hoặc HBASE: đây chính là hệ thống các tệp phân tán của Hadoop. Và HBASE chính là các kỹ thuật dùng để lưu trữ dữ liệu vào hệ thống các tệp phân tán đó.

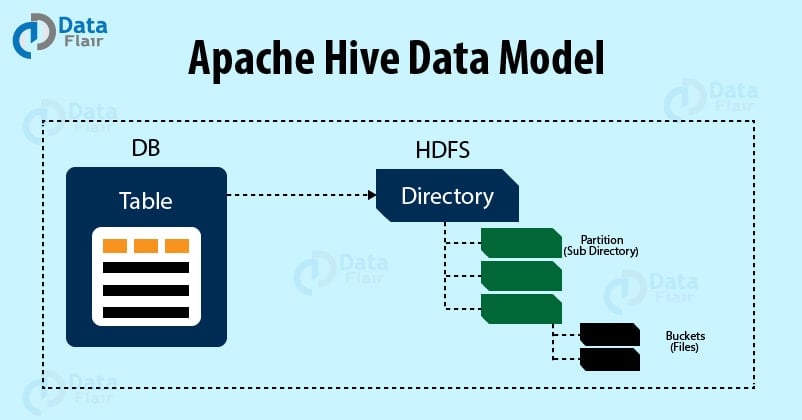
1. **Quá trình hoạt động của Apache Hive:**



***Hình 2: Quá trình hoạt 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)
2. Driver tạo ra mới 1 session cho truy vấn này và gửi query tới compiler để nhận lấy Execution Plan (Bước 2)
3. Compilter nhận các metadata cần thiết từ Metastore (Bước 3, 4). 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 5)
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)
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)
7. **Tổ chức dữ liệu trong Apache Hive:**



**Hình 3: Tổ chức dữ liệu trong Hive**

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.

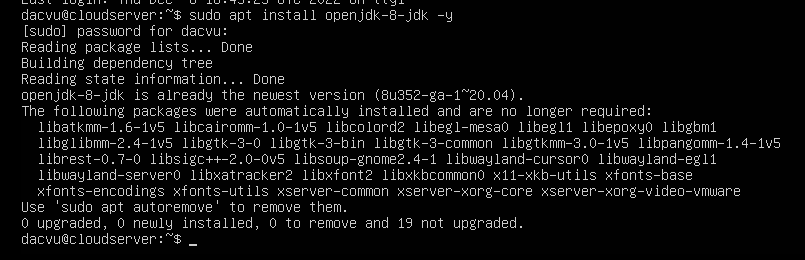
1. **Các kiểu dữ liệu trong Apache Hive:**

* 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.

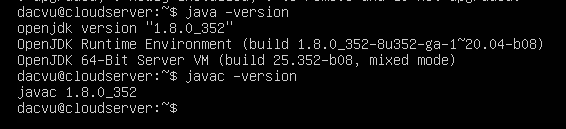
# **Chương 2: Cài đặt, thiết kế và xây dựng Data warehouse**

1. **Cài đặt Apache Hadoop:**

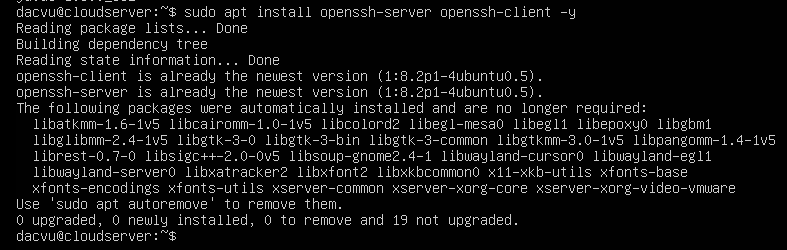
* Hadoop là framework được viết bằng Java nên nó đòi hỏi có môi trường khởi chạy là Java Runtime Environment (JRE) và công cụ phát triển Java Development Kit (JDK). Ở đây chúng ta cài đặt phiên bản OpenJDK 8 để tương thích với Apache Hadoop 3.x



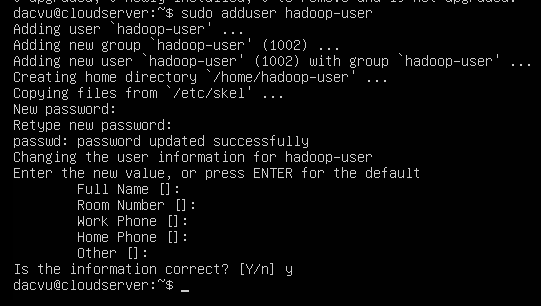
* Sau khi cài đặt thành công, ta xác nhận lại phiên bản của java



* Tiến hành cài đặt OpenSSH server và client



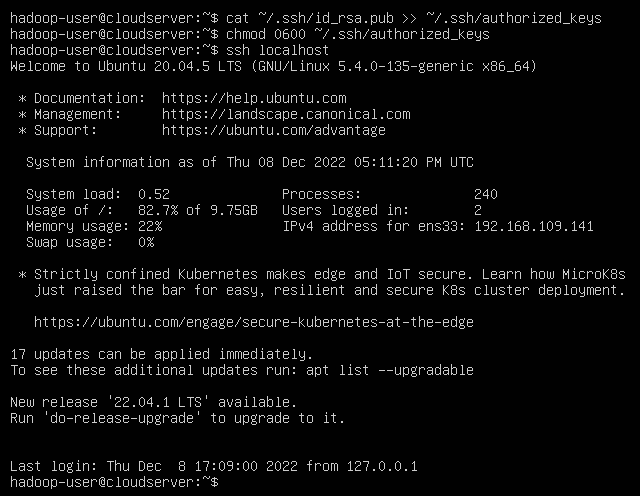
* Tạo Hadoop user, ở đây ta lấy tên là hadoop-user



* Khởi tạo cặp khóa SSH và xác định nơi lưu trữ



* Lưu khóa public vào thư mục ssh và cài đặt quyền cho user



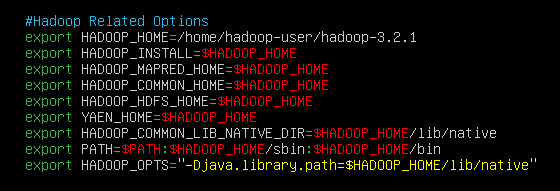
* Cài đặt hadoop phiên bản 3.2.1 thông qua lên wget và link <https://archive.apache.org/dist/hadoop/common/hadoop-3.2.1/hadoop-3.2.1.tar.gz>



* Giải nén file vừa tải về



* Cấu hình file bashrc, thêm những dòng dưới đây vào cuối file



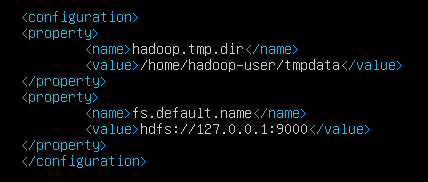
* Sau khi cấu hình xong thì dùng lệnh source để áp dụng ngay những thay đổi vào môi trường đang chạy



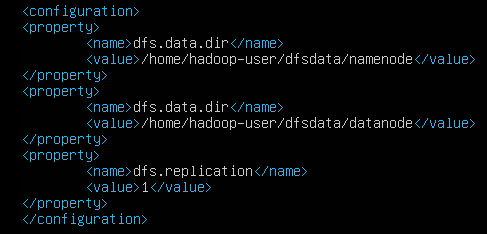
* Cấu hình file hadoop-env.sh, tìm dòng chứa biến $JAVA\_HOME đã được comment trong dấu #, bỏ dấu # và sửa lại thành như sau



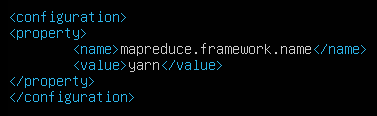
* Cấu hình file core-site.xml, thêm những dòng dưới đây vào cặp thẻ configuration



* Cấu hình file hdfs-site.xml, thêm những dòng dưới đây vào cặp thẻ configuration



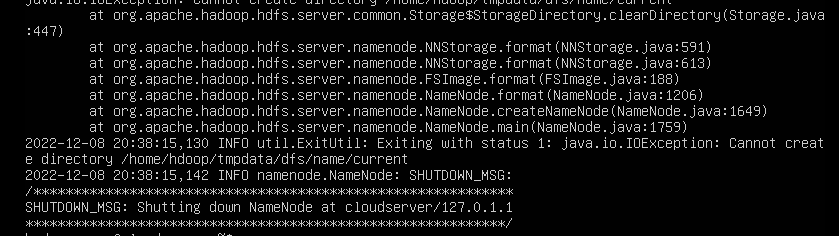
* Cấu hình file mapred-site.xml, thêm những dòng dưới đây vào cặp thẻ configuration



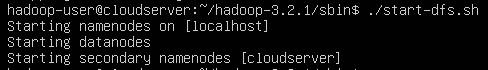
* Cấu hình file yarn-site.xml, thêm những dòng dưới đây vào cặp thẻ configuration



* Định dạng NameNode trước khi khởi chạy dịch vụ Hadoop lần đầu tiên



* Khởi chạy NameNode và DataNode



* Khởi chạy YARN và nodemanagers



* Kiểm tra xem các tiến trình với lệnh jps



1. **Cài đặt Apache Hive:**

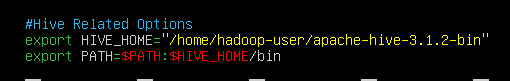
* Sau khi cài đặt xong Apache Hadoop thì ta tiến hành cài đặt Apache Hive phiên bản 3.1.2 tương thích với phiên bản của Hadoop thông qua lệnh wget và link <https://downloads.apache.org/hive/hive-3.1.2/apache-hive-3.1.2-bin.tar.gz>



* Sau khi tải xong tệp thì ta tiến hành giải nén



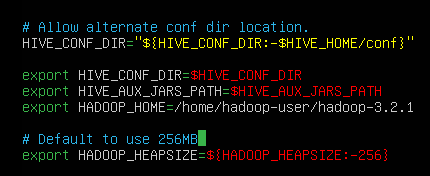
* Cấu hình file .bashrc, thêm những dòng dưới đây vào file



* Sau khi cấu hình xong thì dùng lệnh source để áp dụng ngay những thay đổi vào môi trường đang chạy



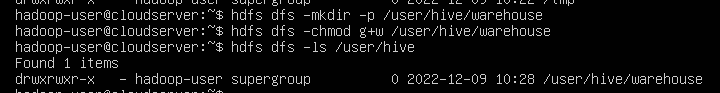
* Cấu hình file hive-config.sh, thêm biến HADOOP\_HOME là đường dẫn của thư mục Hadoop như sau



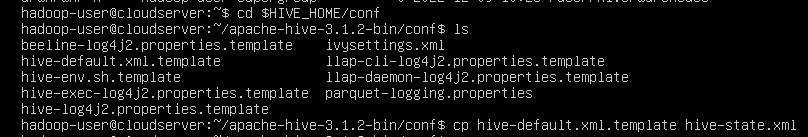
* Khởi tạo thư mục tmp trong lớp lưu trữ HDFS để lưu trữ những dữ liệu trung gian mà Hive gửi tới HDFS, cấp quyền write và execute cho nhóm tmp và kiểm tra lại



* Khởi tạo thư mục warehouse trong thư mục hive để lưu trữ các bảng của Hive, cấp quyền write và execute cho nhóm warehouse và kiểm tra lại



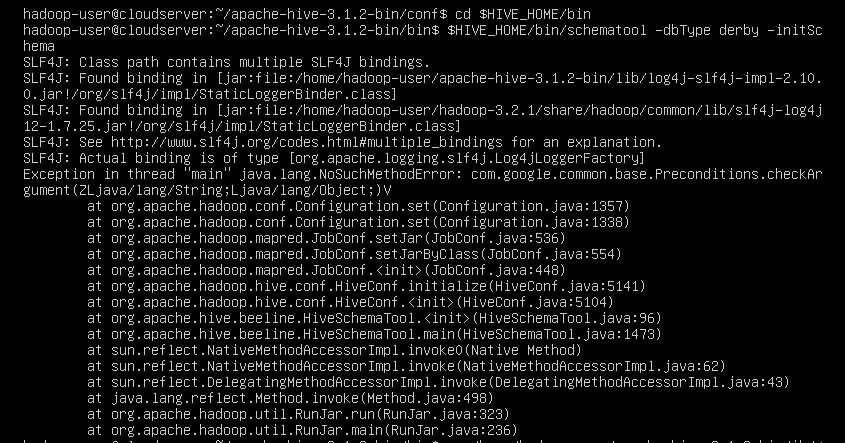
* Liệt kê các file trong thư mục conf với lệnh ls



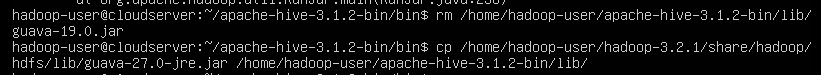
* Dùng hive-default.xml.template để tạo file hive-site.xml



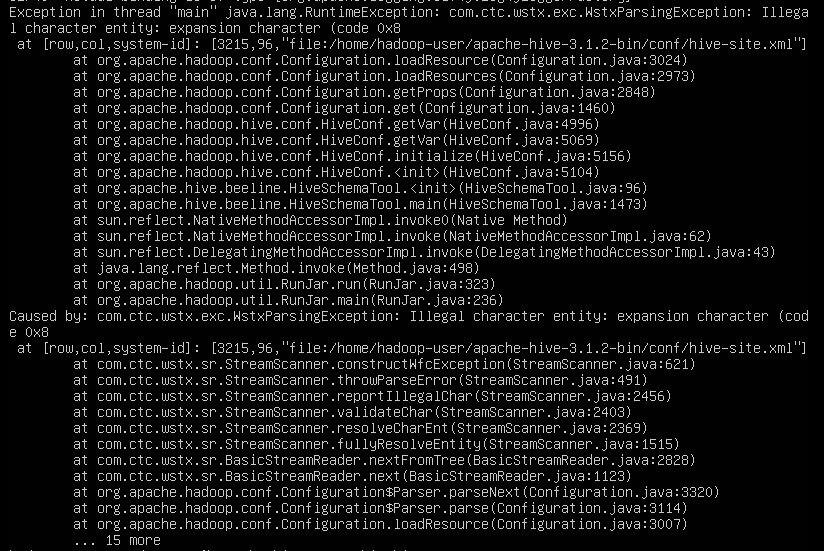
* Khởi tạo Derby database bằng lệnh schematool. Tuy nhiên ở đây ta gặp lỗi không tương thích guava



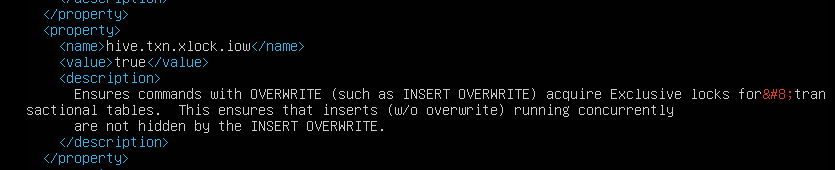
* Ta khắc phục lỗi này bằng cách remove file guava trong thư viện của Hive và copy file guava từ thư viện Hadoop sang



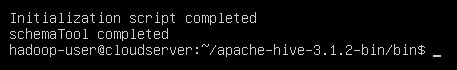
* Sau khi khắc phục được lỗi không tương thích ta sẽ tiếp tục gặp lỗi ký tự nằm trong file hive.xml vừa tạo.\



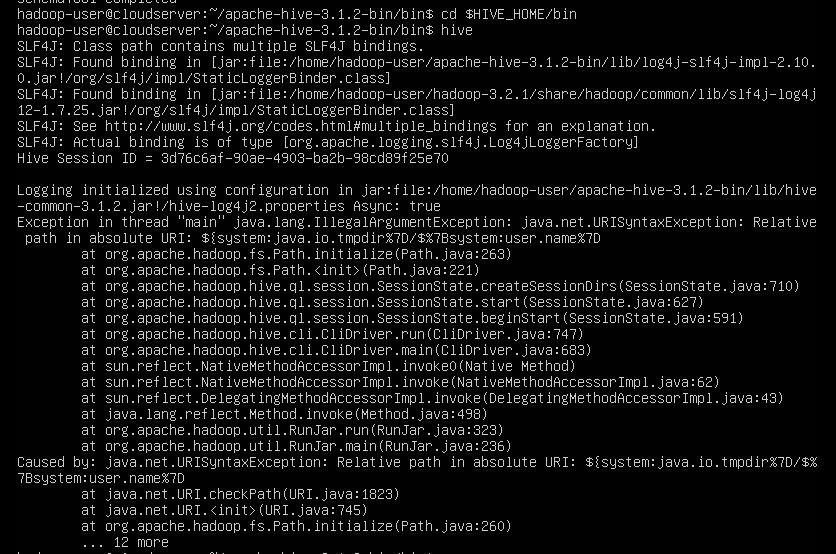
* Ta khắc phục bằng cách tìm đến vị trí bị thừa ký tự và xóa nó đi



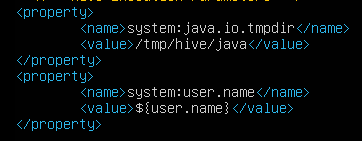
* Sau khi khắc phục xong thì ta đã có thể khởi tạo Derby database



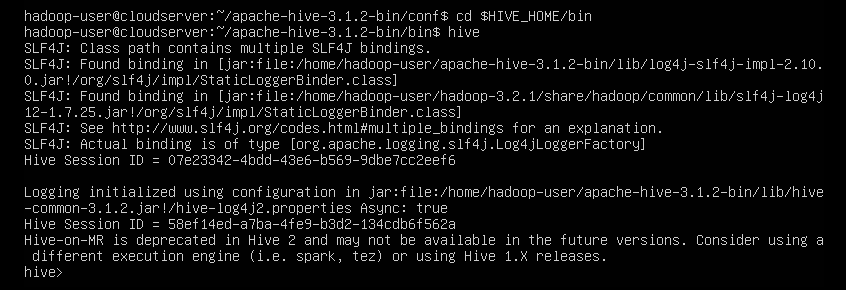
* Ta tiến hành khởi chạy giao diện truy vấn của Apache Hive với lệnh hive. Tuy nhiên ở đây ta lại gặp lỗi đối số

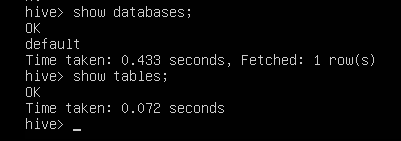


* Để khắc phục lỗi trên thì ta thêm những dòng code dưới đây vào file hive.xml đã tạo ở trên



* Sau khi thêm xong thì giờ đây hive đã có thể chạy bình thường và đã có thể truy vấn





1. **Thiết kế cơ sở dữ liệu:**
2. **Cài đặt Data warehouse với EMR trên AWS**

# **KẾT LUẬN**

# **TÀI LIỆU THAM KHẢO**

1. <https://docs.aws.amazon.com/emr/latest/ManagementGuide/emr-what-is-emr.html>
2. <https://aws.amazon.com/vi/emr/features/hive/?nc1=f_ls>
3. <https://expressmagazine.net/posts/view/3646/gioi-thieu-nhung-kien-thuc-co-ban-ve-apache-hive>
4. <https://itnavi.com.vn/blog/ngon-ngu-lap-trinh-hive-la-gi>
5. <https://www.academia.edu/18578847/Hive>
6. [https://phoenixnap.com/kb/install-hadoop-ubuntu#](https://phoenixnap.com/kb/install-hadoop-ubuntu)
7. <https://phoenixnap.com/kb/install-hive-on-ubuntu>
8. <https://stackoverflow.com/questions/27099898/java-net-urisyntaxexception-when-starting-hive>