**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

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

**KHOA CÔNG NGHỆ THÔNG TIN**



**MÔN HỌC: ĐIỆN TOÁN ĐÁM MÂY**

**ĐỀ TÀI:**

**TÌM HIỂU APACHE HIVE VÀ VIẾT ỨNG DỤNG DEMO**

**Giảng viên hướng dẫn:** ***TS. Huỳnh Xuân Phụng***

**Nhóm sinh viên thực hiện:**

|  |  |
| --- | --- |
| 1. Phan Văn Đức Anh | 20110609 |
| 1. Bùi Đặng Quốc Chí | 20110614 |
| 1. Võ Hồng Khang | 20110281 |

***TP. Hồ Chí Minh-11/2022***

**MỤC LỤC**

[MỤC LỤC 2](#_Toc120521446)

[CHƯƠNG 1: APACHE HIVE 1](#_Toc120521447)

[1. Tổng quan về Apache Hive 1](#_Toc120521448)

[**1.1.** **Vị trí, vai trò Hive trong hệ sinh thái Hadoop** 2](#_Toc120521449)

[**1.2.** **Đặc trưng của Hive** 2](#_Toc120521450)

[2. Hive ra đời để làm gì 3](#_Toc120521451)

[3. Kiến trúc Hive 3](#_Toc120521452)

[4. Hoạt động của Hive 5](#_Toc120521453)

[5. Các chế độ hoạt động trong Hive 6](#_Toc120521454)

[6. Tổ chức dữ liệu và kiểu dữ liệu trong Hive 7](#_Toc120521455)

[**6.1.** **Tổ chức dữ liệu** 7](#_Toc120521456)

[**6.2.** **Kiểu dữ liệu** 7](#_Toc120521457)

[7. Tìm hiểu về HiveSQL 9](#_Toc120521458)

[CHƯƠNG 2: Cài Hive trên AWS sử dụng EMR 11](#_Toc120521459)

[CHƯƠNG 3: PHẦN KẾT LUẬN 32](#_Toc120521460)

[**1.** **Kết quả đạt được** 32](#_Toc120521461)

[**1.1. Kiến thức tìm hiểu được** 32](#_Toc120521462)

[**1.2. Chương trình đã làm được** 32](#_Toc120521463)

[**2.** **Ưu điểm** 32](#_Toc120521464)

[**3.** **Nhược điểm** 32](#_Toc120521465)

[**4.** **Hướng phát triển** 33](#_Toc120521466)

[TÀI LIỆU THAM KHẢO 34](#_Toc120521467)

**CHƯƠNG 1: APACHE HIVE**

1. **Tổng quan về Apache 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 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.

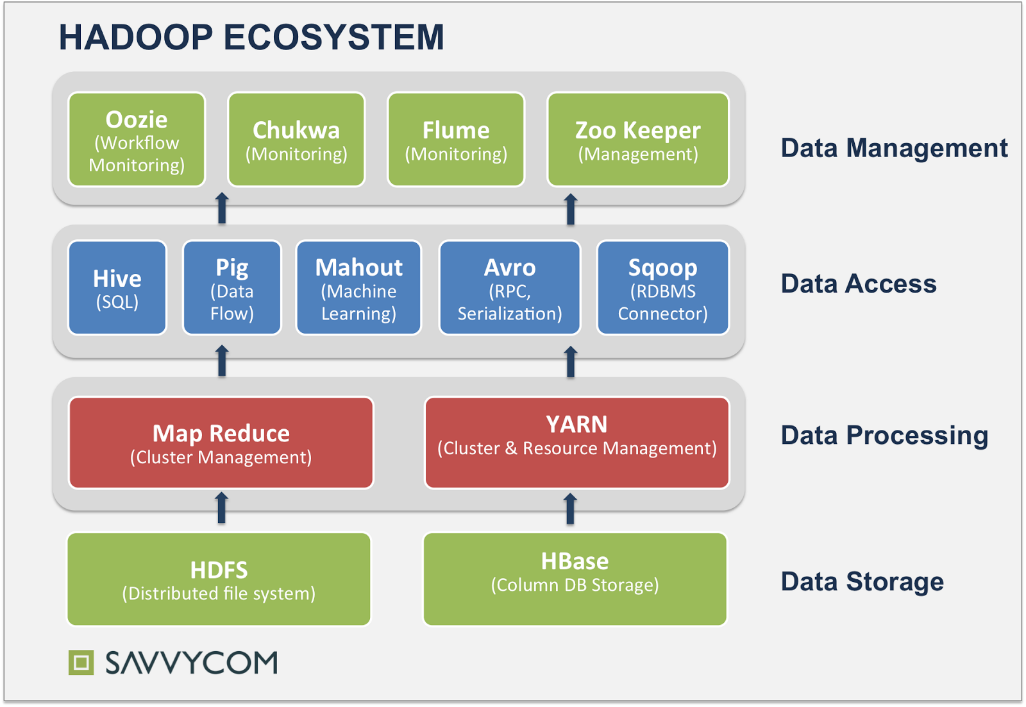
Hive không được thiết kế để cho các giao dịch online (OLTP workloads) và không nên dùng cho các real-time queries và các cập nhật trên từng dòng trong 1 table (row-level). Hive hoạt động tốt nhất cho các batch jobs trên các tập dữ liệu lớn, mà ở đó dữ liệu được thêm vào liên tục (append-only data) ví 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.

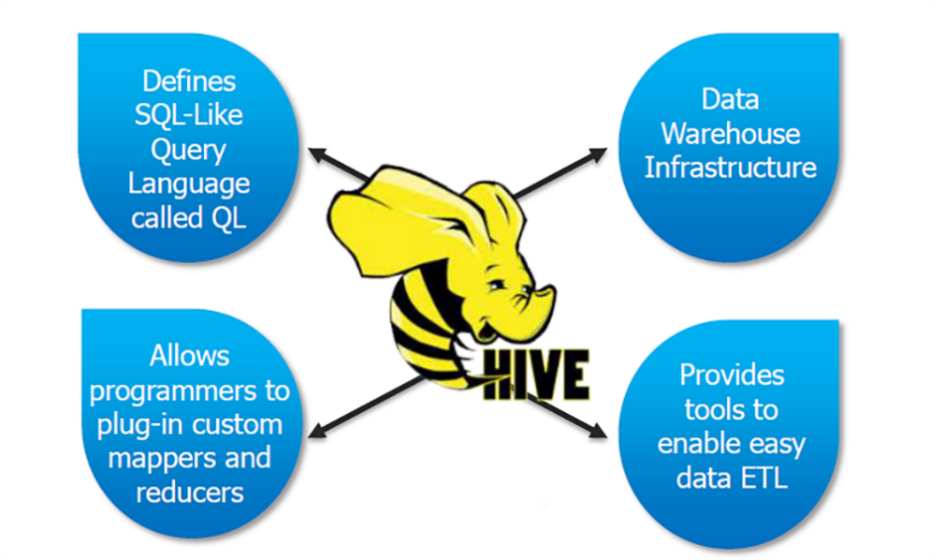


* 1. **Vị trí, vai trò Hive trong hệ sinh thái Hadoop**

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 và giúp truy vấn và phân tích dễ dàng.



* 1. **Đặc trưng của Hive**



Lưu trữ lược đồ trong cơ sở dữ liệu và xử lý dữ liệu vào HDFS.

Được thiết kế cho OLAP.

Cung cấp ngôn ngữ kiểu SQL để truy vấn được gọi là HiveQL hoặc HQL.

Quen thuộc, nhanh chóng, có khả năng mở rộng.

1. **Hive ra đời để làm gì**

Hive là một dự án phần mềm cung cấp các truy vấn và phân tích dữ liệu. Nó tạo điều kiện thuận lợi cho việc đọc, viết và xử lý các tập dữ liệu rộng được lưu trữ trong bộ lưu trữ phân tán và được truy vấn bằng các cú pháp ngôn ngữ truy vấn cấu trúc (SQL).

Hive không được xây dựng cho khối lượng công việc xử lý giao dịch trực tuyến (OLTP). Nó thường được sử dụng cho các tác vụ lưu trữ dữ liệu như đóng gói dữ liệu, truy vấn đặc biệt và phân tích các tập dữ liệu khổng lồ.

Nó được thiết kế để nâng cao khả năng mở rộng, khả năng mở rộng, hiệu suất, khả năng chịu lỗi và kết hợp lỏng lẻo với các định dạng đầu vào của nó.

1. **Kiến trúc 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.

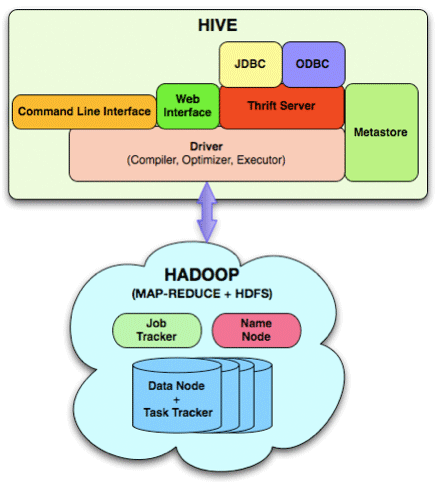
- Hive Driver: thành phần nhận các truy vấn và chuyển các truy vấn này thành các MapReduce Jobs để tiến hành xử lý yêu cầu của người sử dụng.

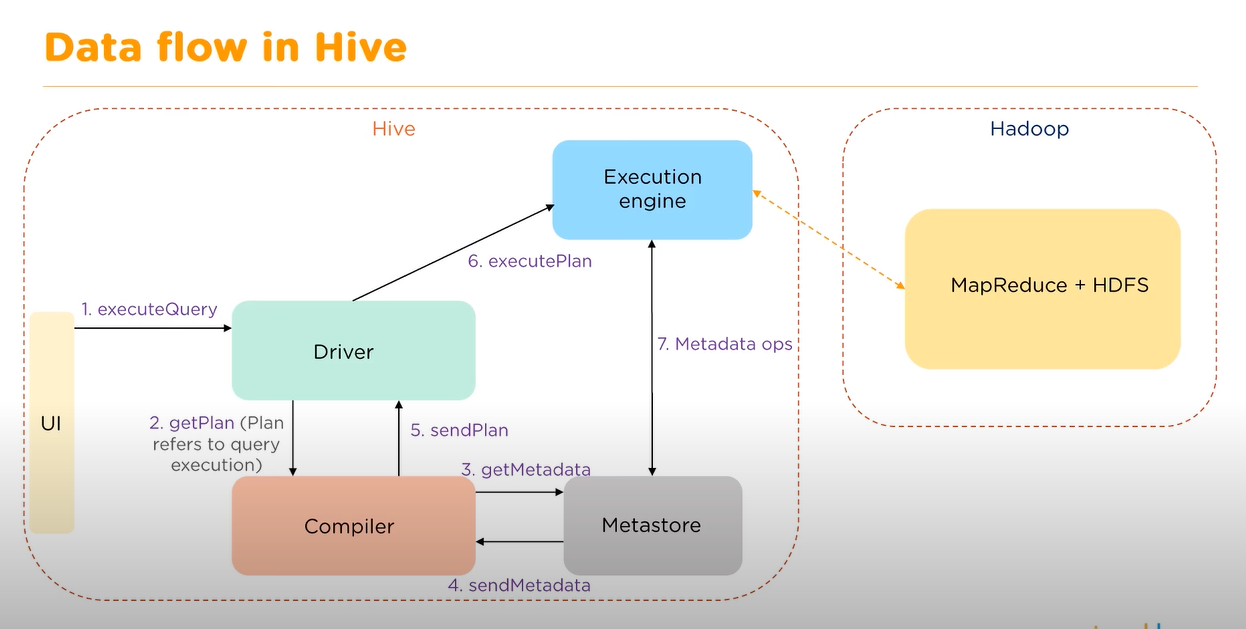
+ Driver: nhận các truy vấn, thành phần này thực hiện việc quản lý các sessions và cung cấp các API để thực thi và lấy dữ liệu trên JDBC/ODBC interfaces.

+ 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.

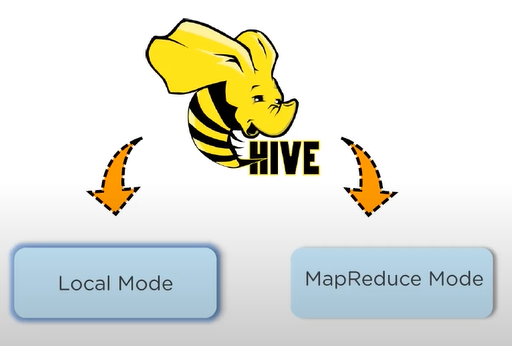
- Hive Metastore: thành phần lưu trữ các metadata của Hive: table, partion, buckets bao gồm cả thông tin về các column trong mỗi table, các serializers và desrializers cần thiết để thực hiện việc đọc và ghi dữ liệu. Metastore sử dụng một cơ sở dữ liệu quan hệ để lưu trữ dữ liệu của chính mình.



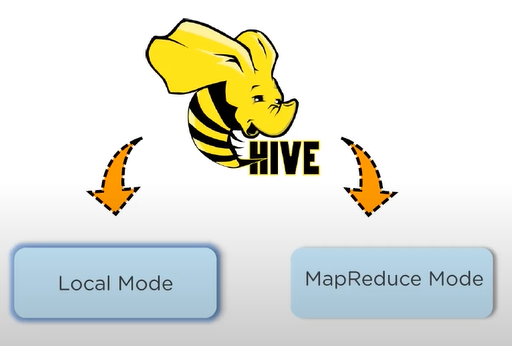
1. **Hoạt động của Hive**

Hình 1. Mô hình 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, ThirftServer) được gửi tới thành phần Driver
2. Driver tạo ra mới 1 session cho truy vấn này và gửi query tới compiler để nhận lấy Execution Plan
3. Compilter nhận các metadata cần thiết từ Metastore. 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.
5. Execution engine nhận yêu cầu thực thi và lấy các metadata cần thiết và yêu cầu mapreduce thực thi công việc.
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.
7. Công cụ thực thi giao tiếp hai chiều với metastore để thực hiện các hoạt động như tạo, xóa bảng. Metastore lưu trữ thông tin về bảng, cột.
8. **Các chế độ hoạt động trong Hive**



**Hình 2. Chế độ hoạt động**

**Local Mode:**

**- Được sử dụng khi hadoop có một nút dữ liệu và dữ liệu nhỏ.**

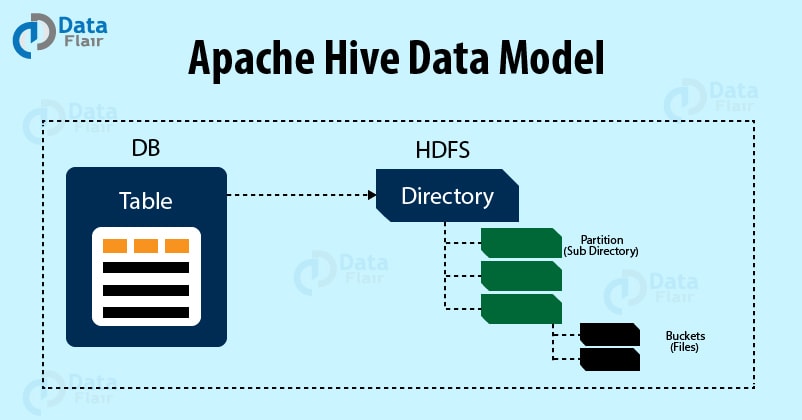
**- Việc xử lý sẽ rất nhanh trên các bộ dữ liệu nhỏ hơn có trong máy cục bộ.**

**MapReduce Mode:**

**- Được sử dụng khi hadoop có nhiều nút dữ liệu và dữ liệu được trải trên các ghi chú dữ liệu khác nhau.**

**- Xử lý bộ dữ liệu lớn có thể hiệu quả hơn khi sử dụng chế độ này.**

1. **Tổ chức dữ liệu và kiểu dữ liệu trong Hive**
   1. **Tổ chức dữ liệu**



Dữ liệu được tổ chức thành 3 định dạng trong HIVE.

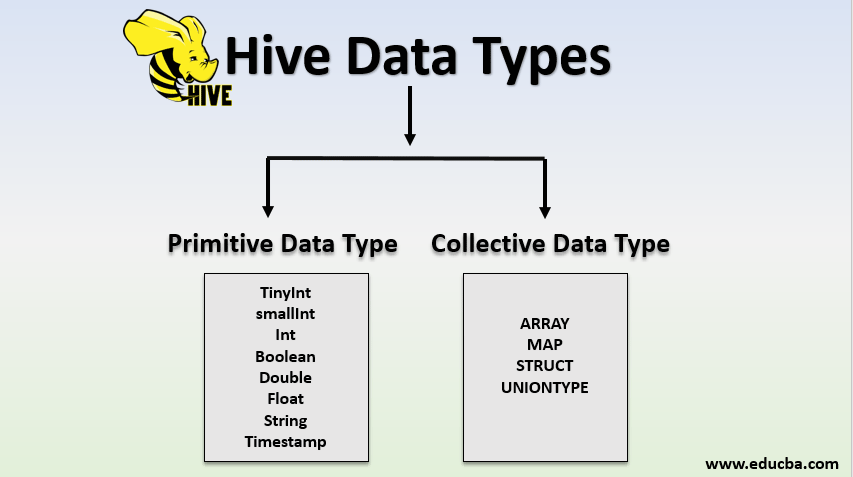
- Tables: Chúng rất tương tự như bảng (tables) trong RDBMS và chứa các dòng (rows). Hive chỉ được xếp lớp trên HDFS, do đó tables được ánh xạ trực tiếp vào các thư mục của hệ thống tập tin. Nó cũng hỗ trợ các tables được lưu trên các hệ thống tập tin khác.

- Partitions: Hive tables có thể có nhiều hơn 1 partition. Chúng được ánh xạ với các thư mục con và các hệ thống tập tin.

- Buckets: Trong Hive, dữ liệu có thể được chia thành các buckets. Buckets được lưu trữ như các tập tin trong partition trong hệ thống tập tin.

Hive cũng có metastore để lưu tất cả metadata. Nó là CSDL quan hệ chứa thông tin khác nhau liên quan đến Hive Schema (column types, owners, key-value data, statistics, …). Chúng ta có thể dùng MySQL cho việc này.

* 1. **Kiểu dữ liệu**



Hive là một script language được xây dựng theo cấu trúc gần giống với SQL với những khái niệm căn bản như bảng, cột, dòng và partition (giống như label cho các ngăn chứa). Đồng thời Hive hỗ trợ các kiểu dữ liệu gốc như intergers, floats, doubles và string cũng như các kiểu dữ liệu phức tạp như maps, list và structs.

Các kiểu dữ liệu gốc sau:

- 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

Hive cũng hỗ trợ các kiểu dữ liệu phức tạp như:

- 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.

Các kiểu dữ liệu trên hoàn toàn có thể lồng vào nhau.

Ví dụ: list<map<string, struct<p1:int, p2:int>>

Ví dụ về tạo bảng TEST:

CREATE TABLE TEST (st string, fl float, li list<map<string, struct<p1:int, p2:int>>);

Ta truy xuất đến các trường trong bảng bằng cách sử dụng dấu “.” và dùng dấu “[]” để truy xuất vào các list và array.

**Hạn chế:**

Hive có một vài hạn chế như không cho insert dữ liệu vào bảng. Các câu lệnh quen thuộc trong SQL như INSERT INTO, DELETE hay UPDATE đều được lược bỏ trong Hive.

Hive không hỗ trợ chèn vô bảng hay partition mà ta chỉ có thể overwrite lại toàn bộ dữ liệu INSERT OVERWRITE TABLE TEST

1. **Tìm hiểu về HiveSQL**

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 cũng có thể được mở rộng với các custom scalar functions (UDF’s), aggregations (UDAF’s) và các table funtions (UDTF’s)

• Câu lệnh SELECT được sử dụng để lấy dữ liệu từ một bảng.

• Mệnh đề WHERE hoạt động tương tự như một điều kiện. Nó lọc dữ liệu bằng cách sử dụng điều kiện và cung cấp cho bạn một kết quả hữu hạn. Các toán tử và hàm tích hợp tạo ra một biểu thức, đáp ứng điều kiện.

• Mệnh đề ORDER BY được sử dụng để truy xuất các chi tiết dựa trên một cột và sắp xếp kết quả được đặt theo thứ tự tăng dần hoặc giảm dần.

• Mệnh đề GROUP BY được sử dụng để nhóm tất cả các bản ghi trong tập kết quả bằng cột thu thập cụ thể. Nó được sử dụng để truy vấn một nhóm các hồ sơ.

Cú pháp truy vấn:

SELECT [ALL | DISTINCT] select\_expr, select\_expr, ...

FROM table\_reference

[WHERE where\_condition]

[GROUP BY col\_list]

[HAVING having\_condition]

[ORDER BY col\_list]]

[LIMIT number];

Đây là ví dụ truy vấn HQL:

SELECT product\_name, quantity, price, rate

FROM products;

WHERE product\_id=1;

SELECT author.author\_name,author.author\_id, COUNT(bookid) AS "So luong"

FROM book\_author inner join author

on book\_author.author\_id=author.author\_id

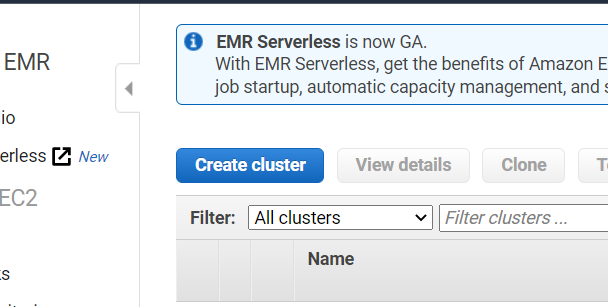
GROUP BY author.author\_name,author.author\_id;

**CHƯƠNG 2: Cài Hive trên AWS sử dụng EMR**

- Tìm kiếm service có tên là EMR



- Sau đó chọn Create Cluster



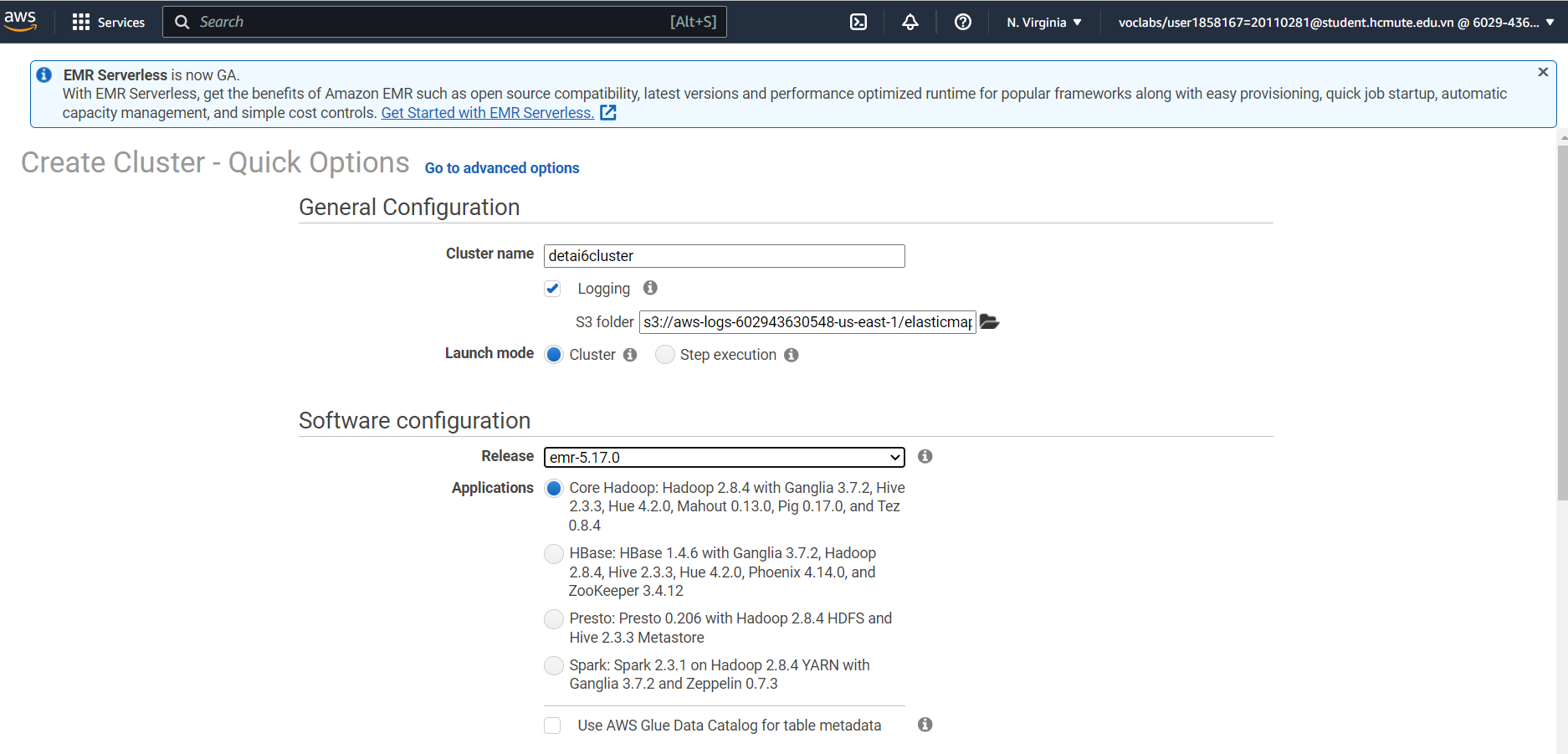
***- General Configuration***

Cluster name là tên của cluster: detai6cluster

S3 folder tức là folder cluster

***- Software configuration***

Release thì chọn phiên bản mà ta muốn sử dụng, ở đây nhóm em chọn emr-5.17.0

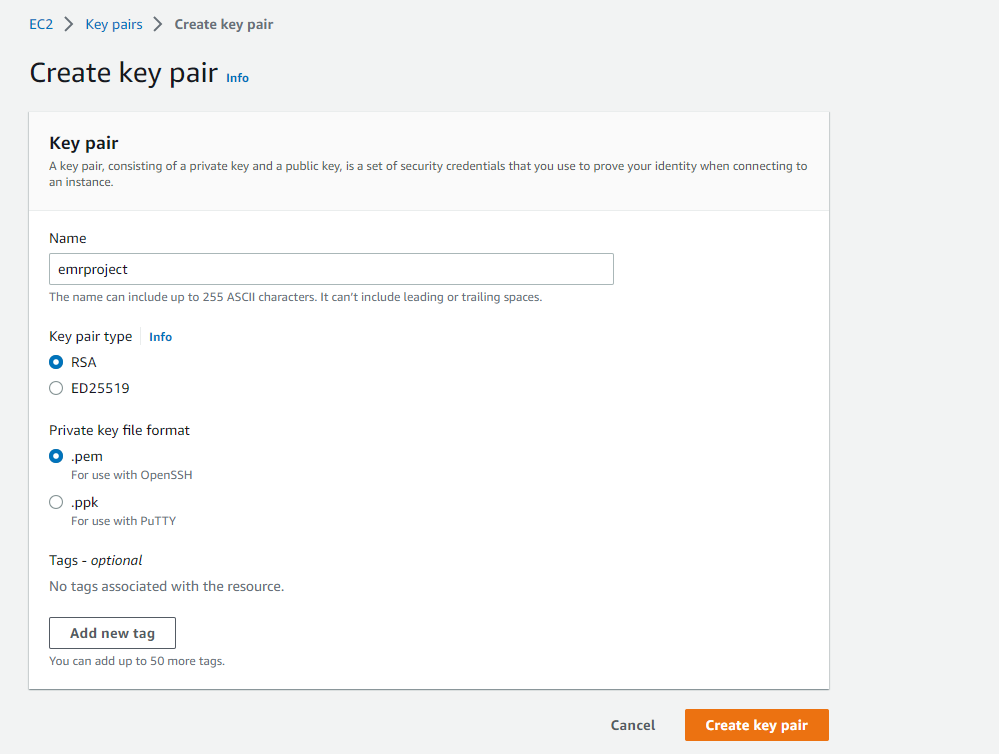


***- Hardware Configuration***

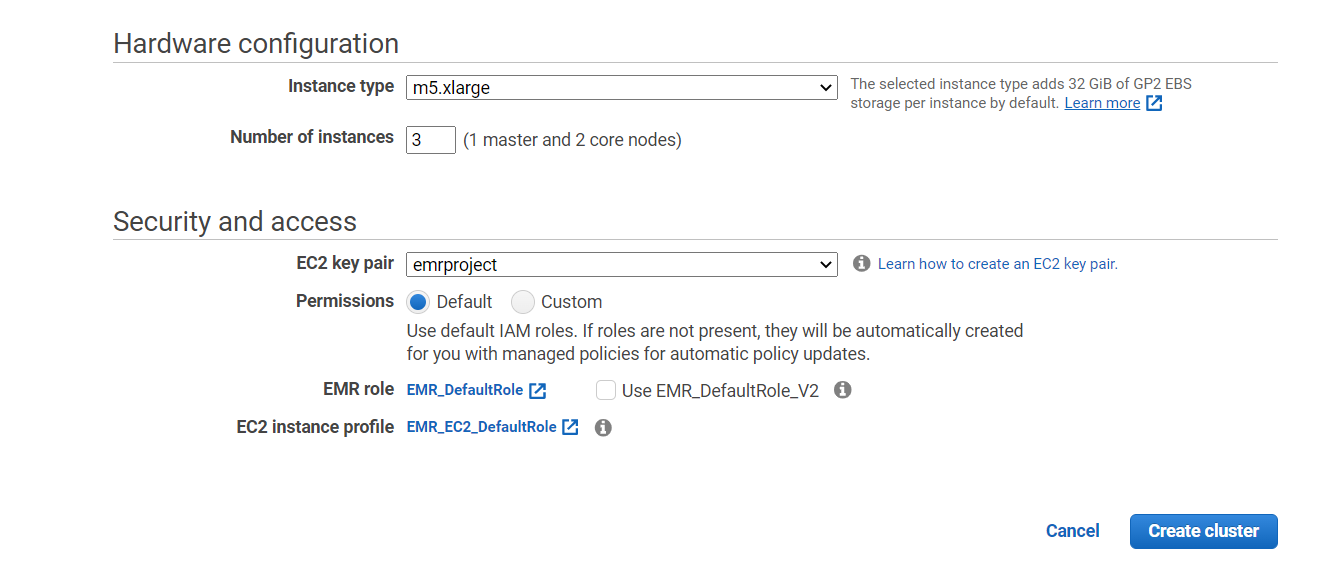
Instance type thì nhóm em chọn m5.xlagre và chọn 3 instance trong đó có 1 master và core nodes

***- Security and Access***

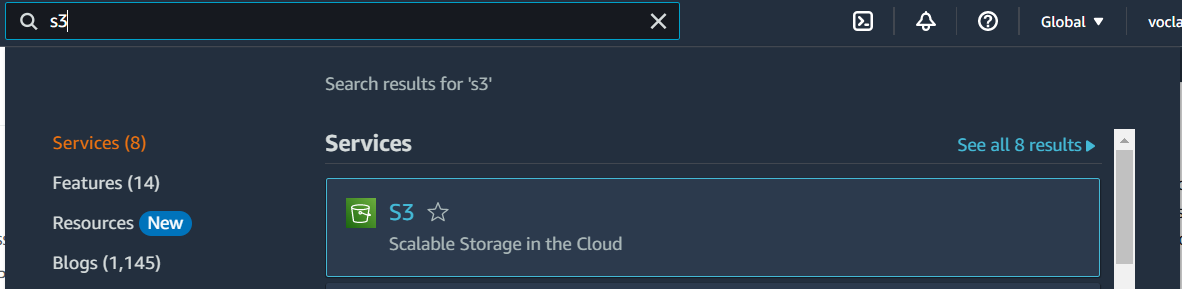
Chọn key pair là cái file emrproject.pem đã tạo từ trước.



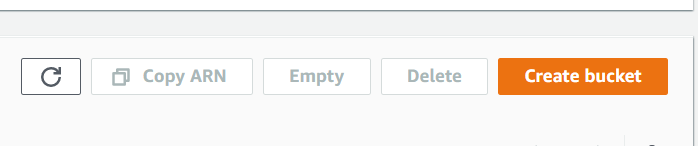
Rồi sau đó nhất Create cluster



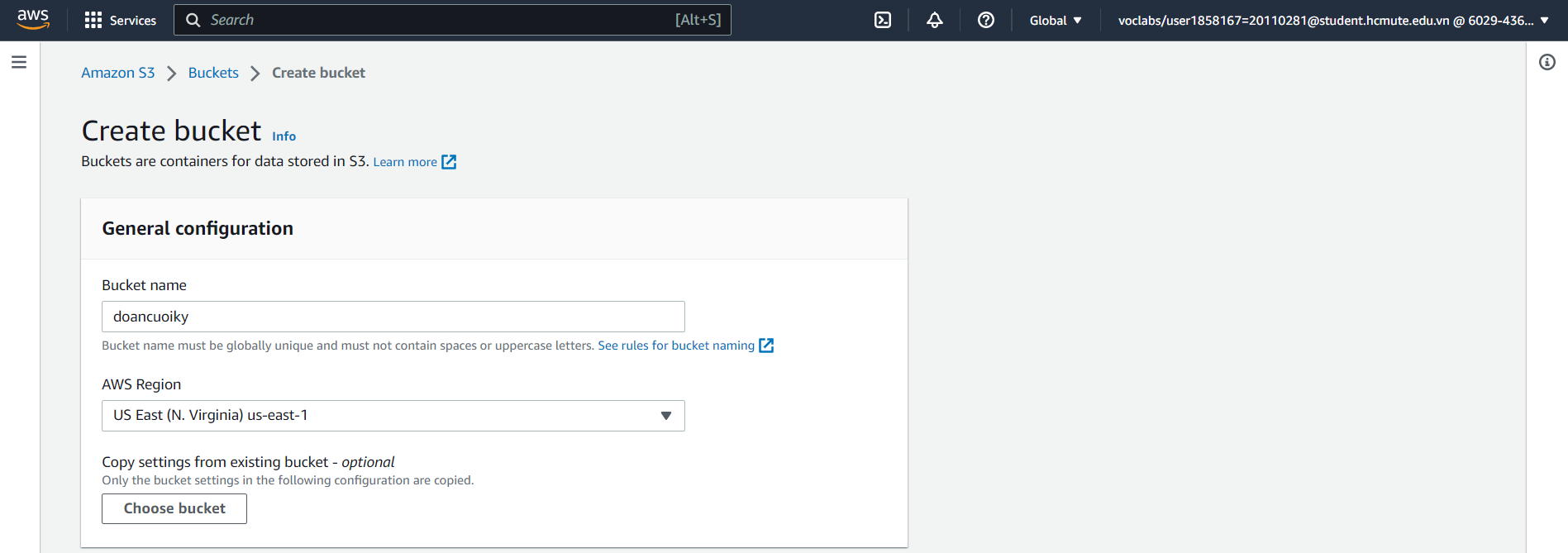
- Sau đó tìm kiếm service có tên là S3

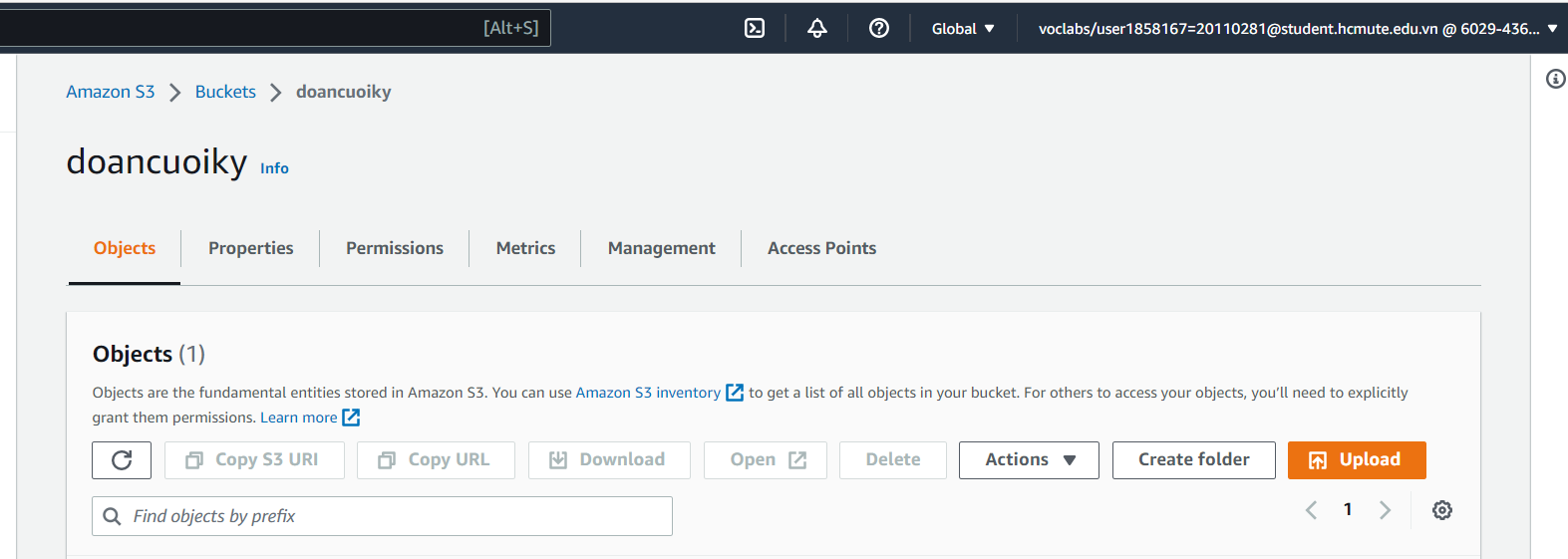


- Chọn Create bucket

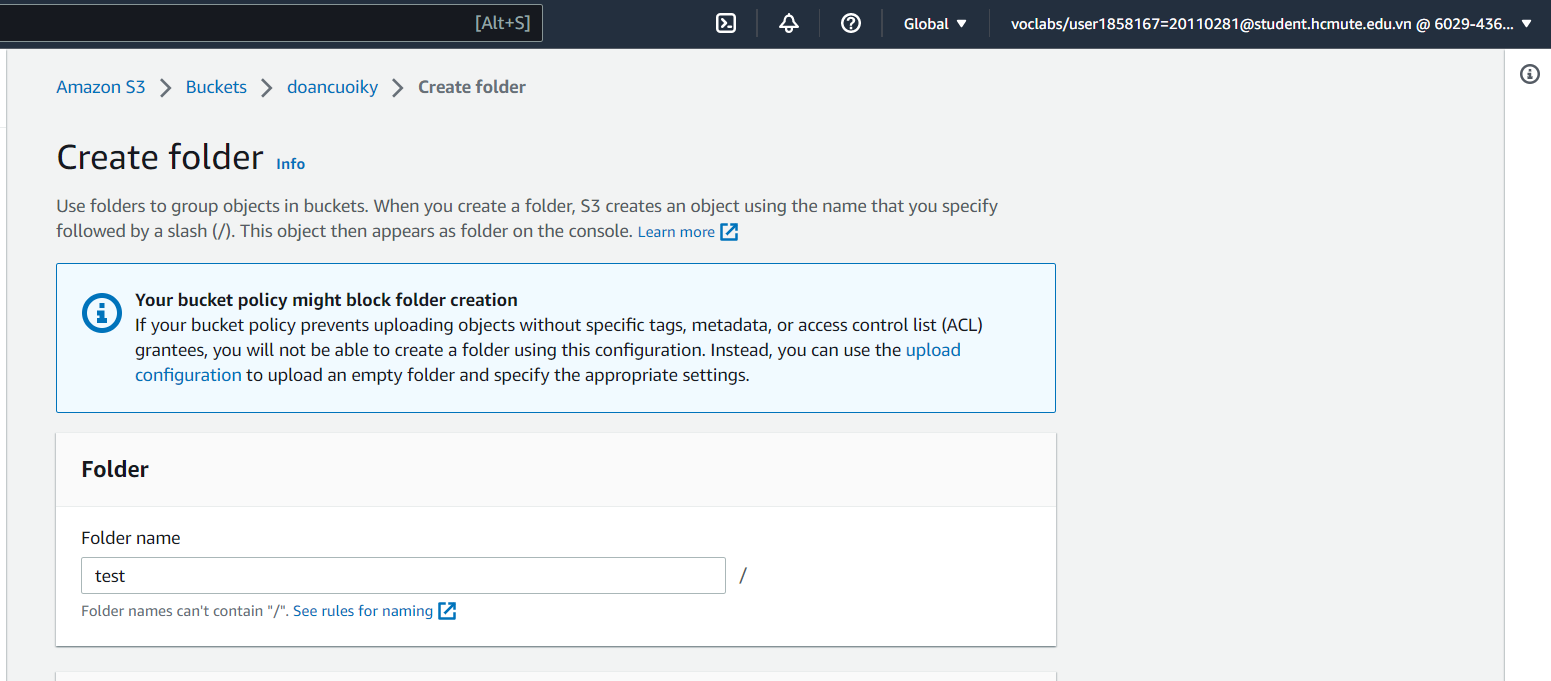


- Tạo 1 bucket có tên: doancuoiky





- Trong bucket doancuoiky tạo 1 folder có tên: test

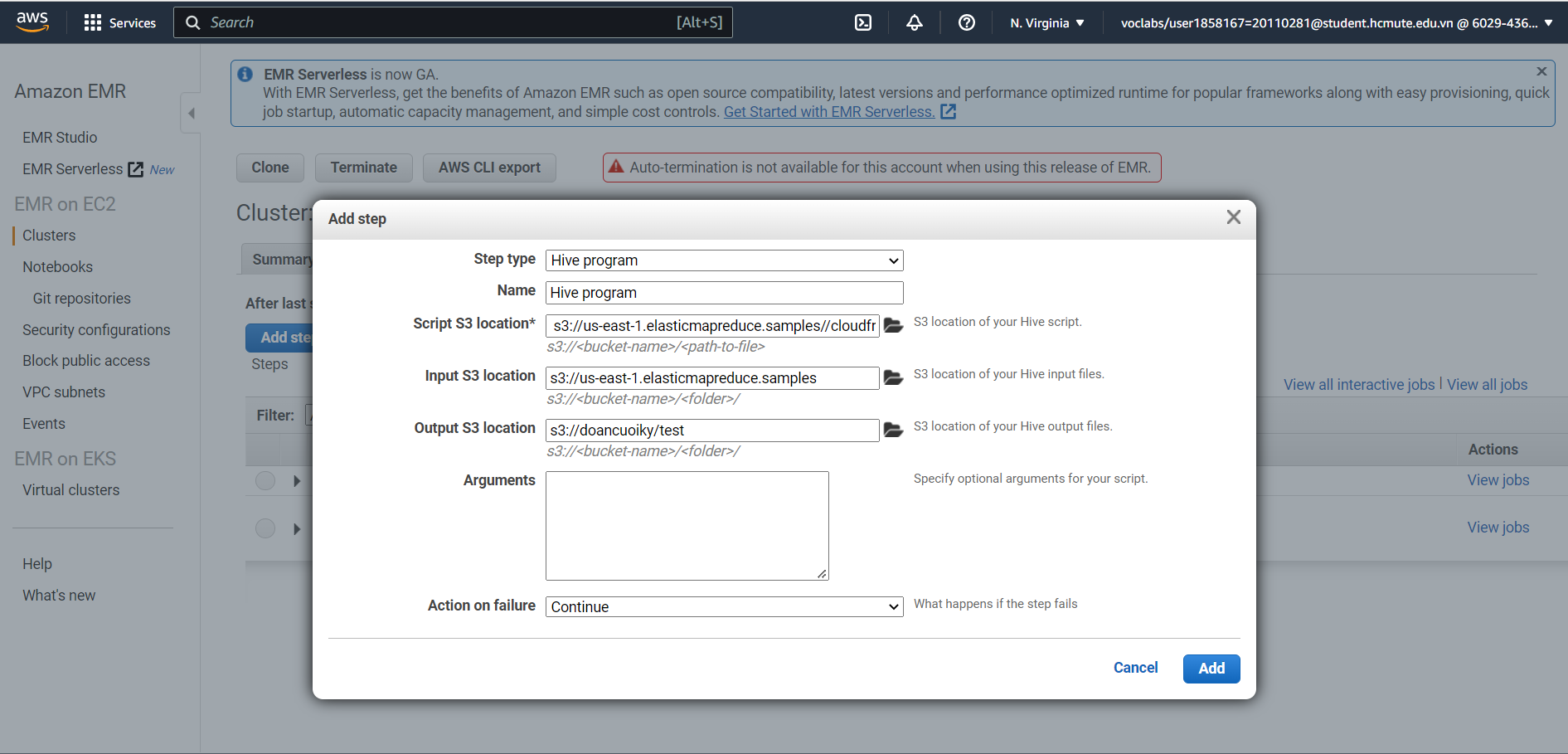


- Sau khi tạo xong cluster ở bước tiếp theo chọn Add step

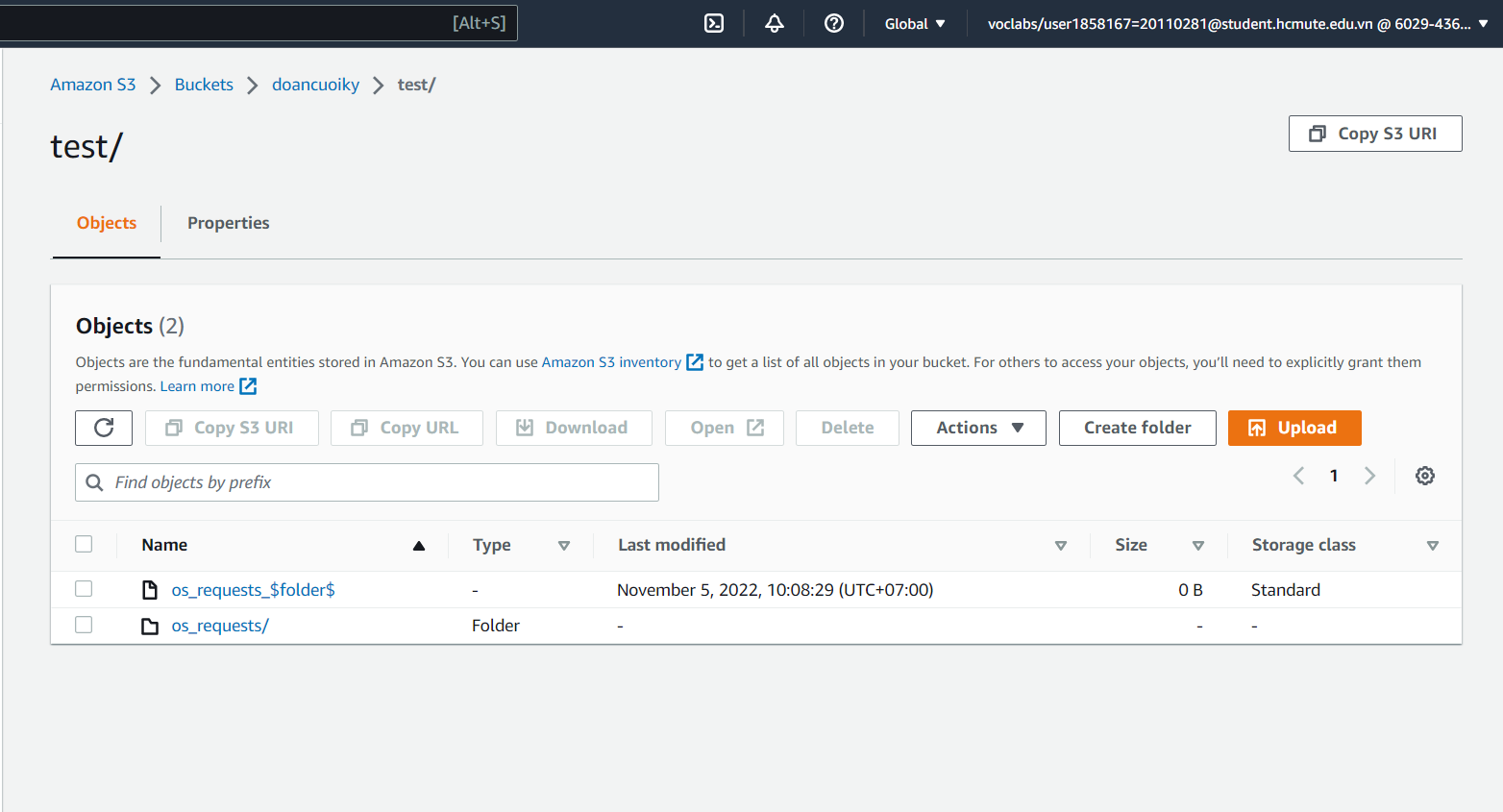
s3://us-east-1.elasticmapreduce.samples/cloudfront/code/Hive\_CloudFront.q

s3://us-east-1.elasticmapreduce.samples

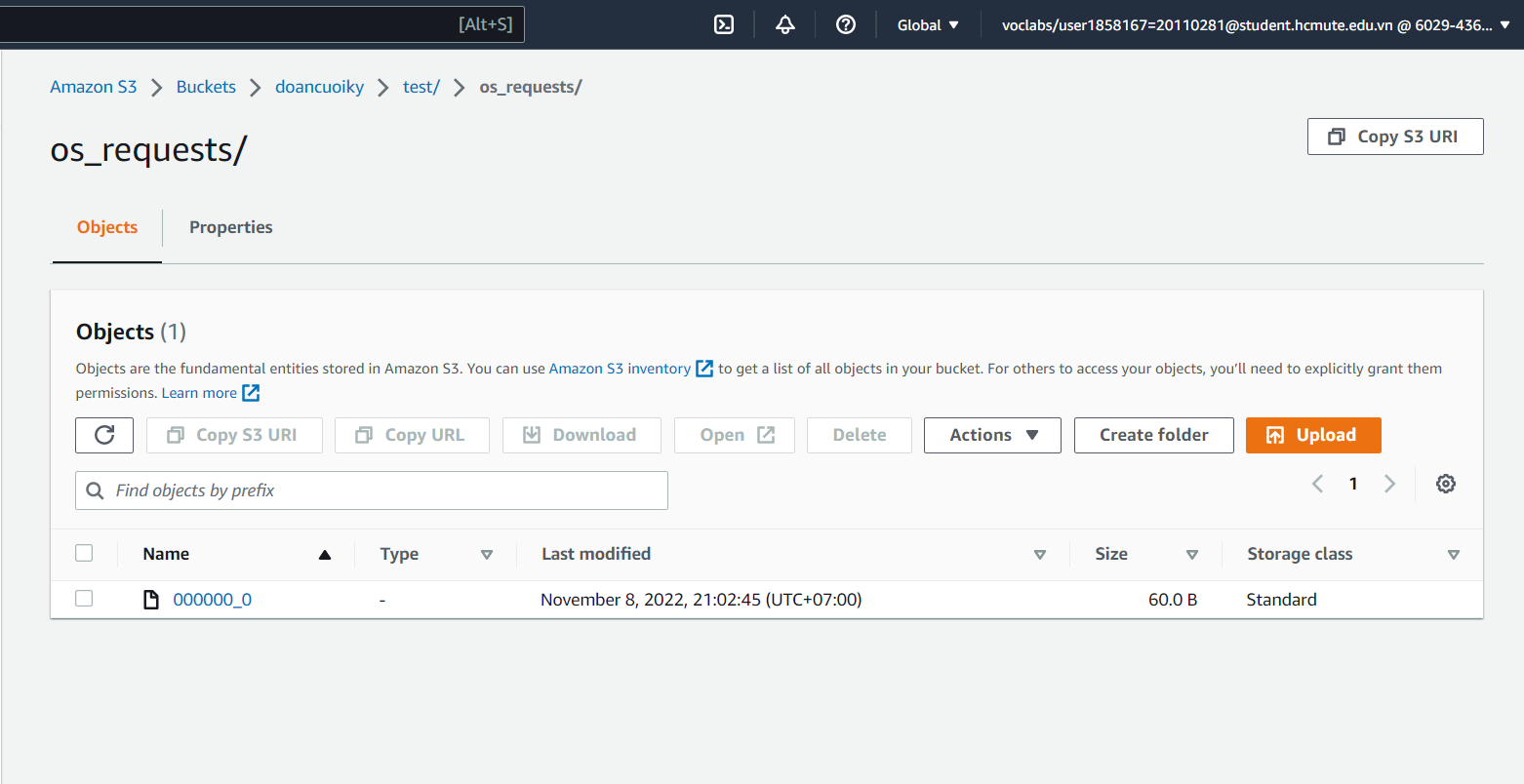
s3://doancuoiky/test

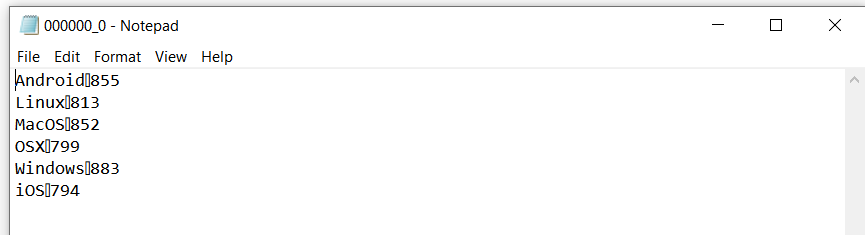


- Sau khi tạo xong step ta phải chờ vài phút để nó hoàn thành, sau đó ta vào trong thư mục test/ trong bucket doancuoiky/ sẽ thư mục os\_request/

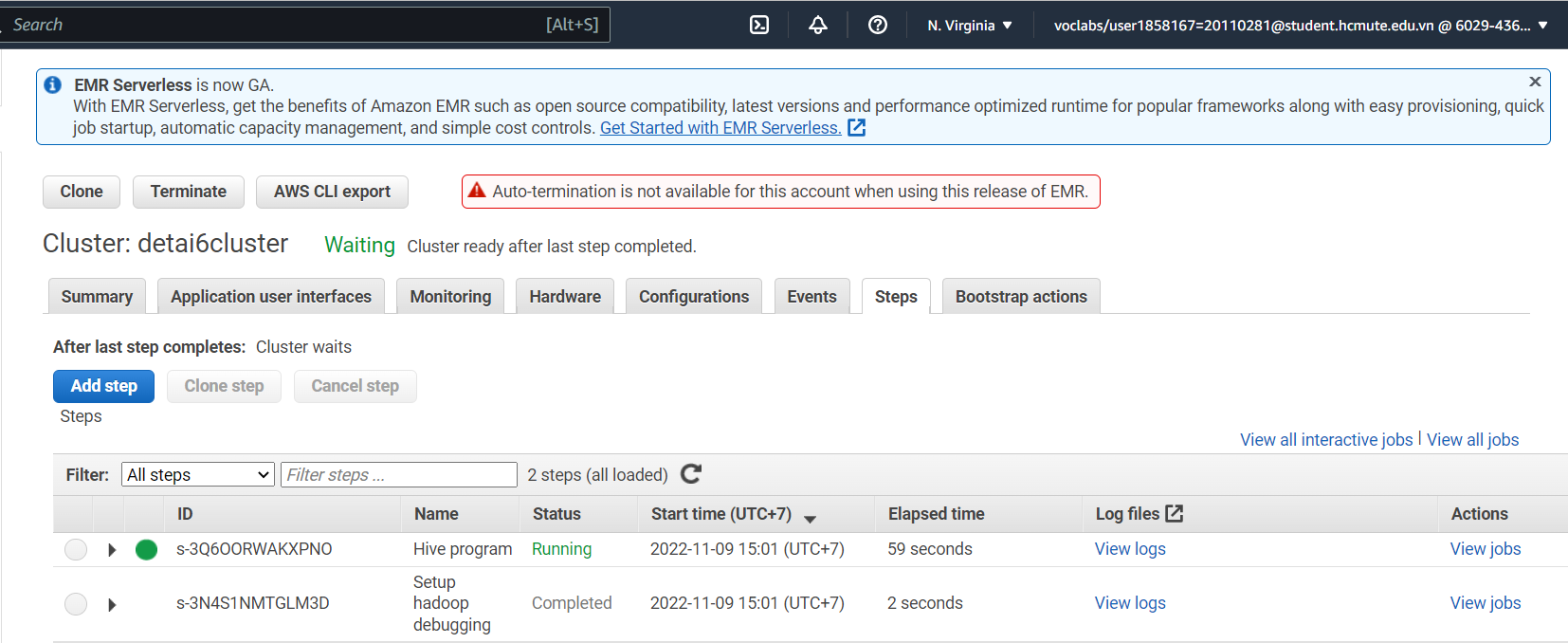


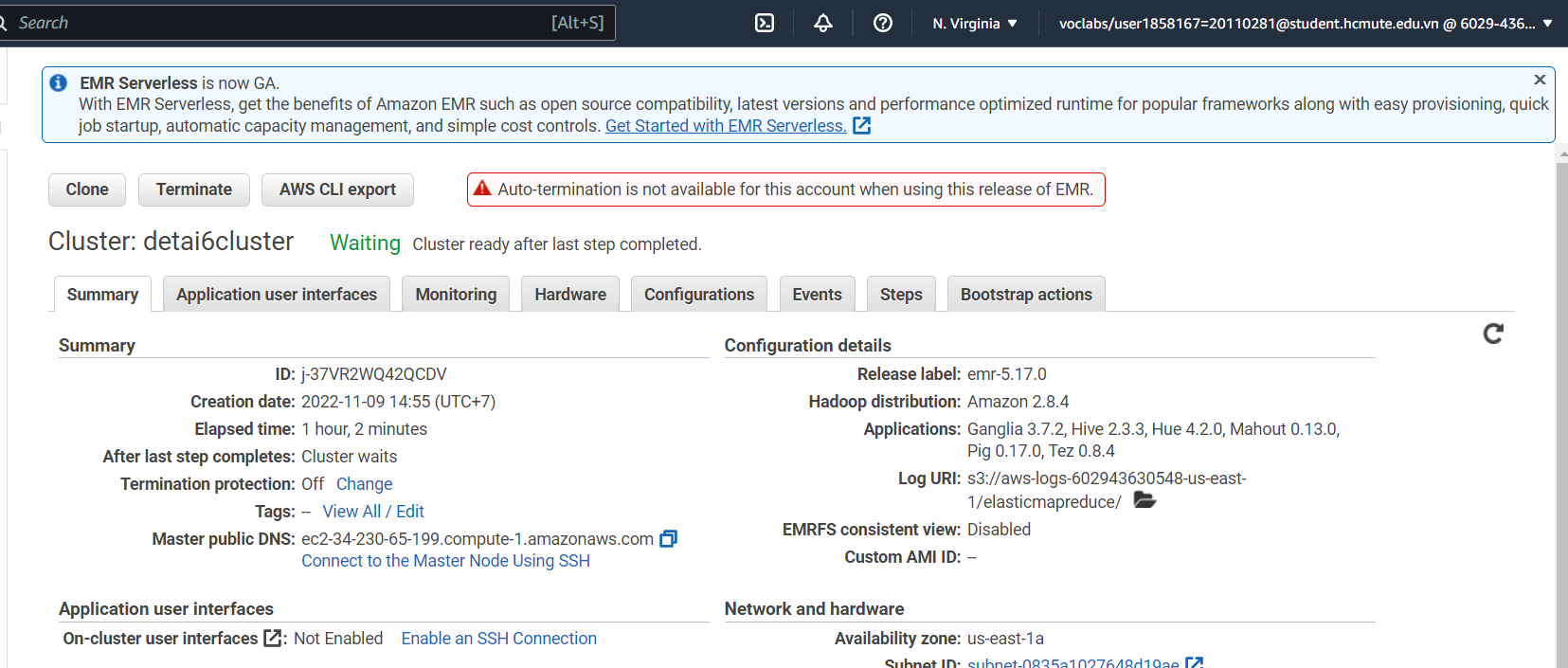
- Nếu ra được như hình thì chứng tỏ là step đã chạy thành công

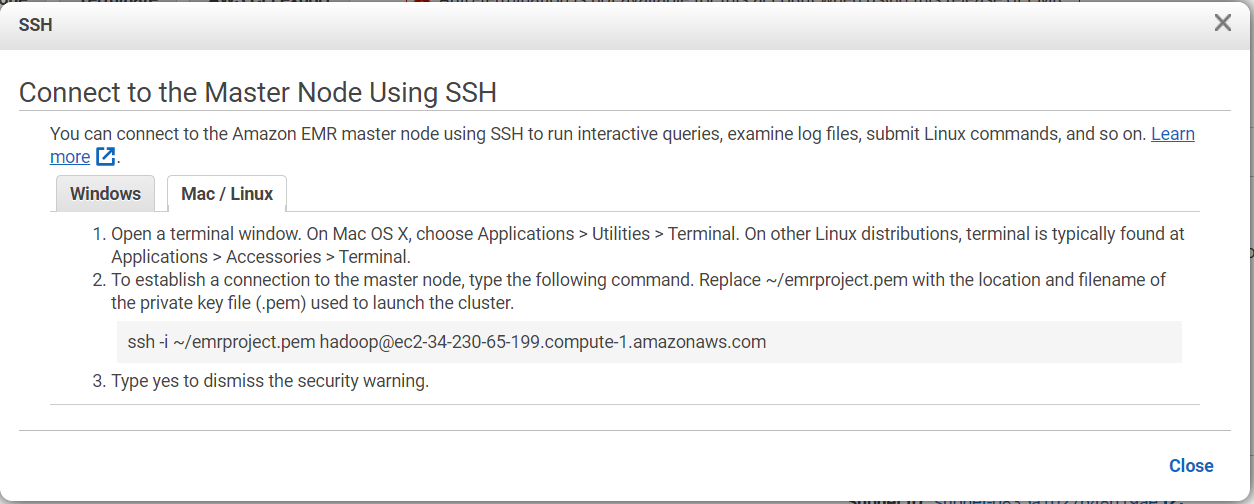




- Sau khi tiến trình chuyển sang Completed thì ta chuyển sang tab Summary để lấy chuỗi kết nối

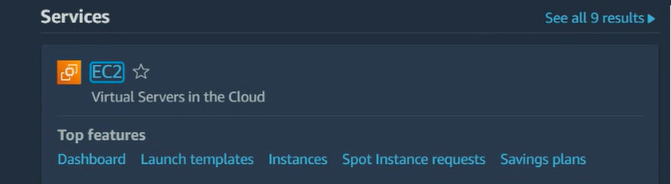




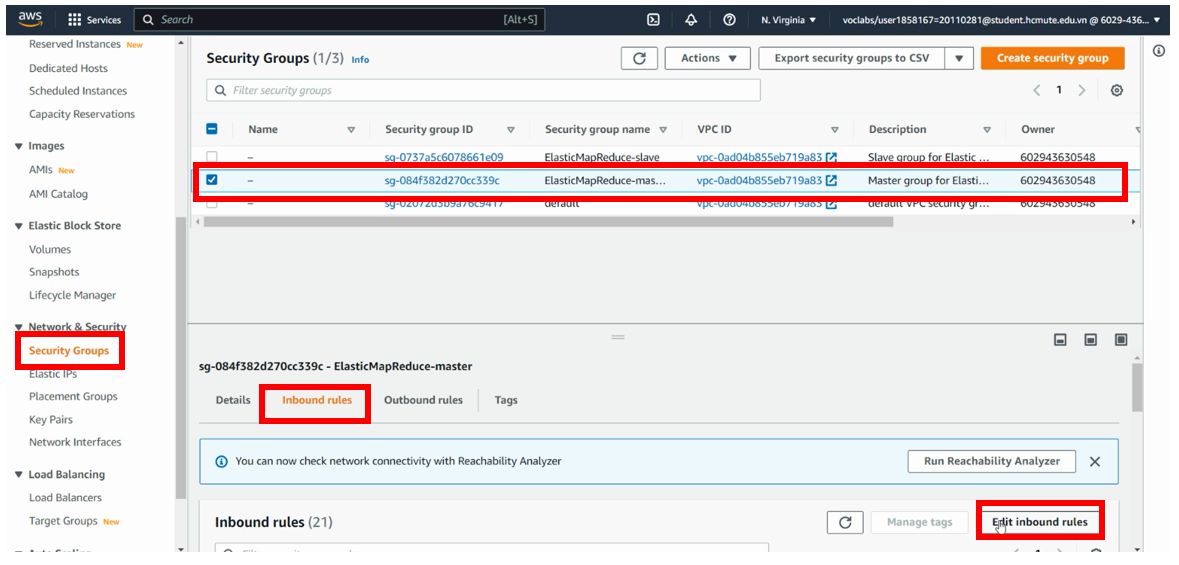


- Giờ ta sẽ chạy kết nối tới Node Master sử dụng SSH

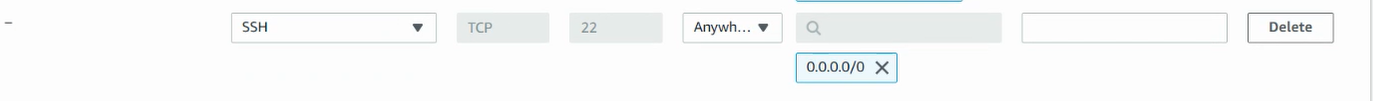
Trước đó ta vào EC2



Chọn security group => Inbound rules => Edit inbound rules



Thêm 1 inbound rule SSH với port là 22 ở chế độ public



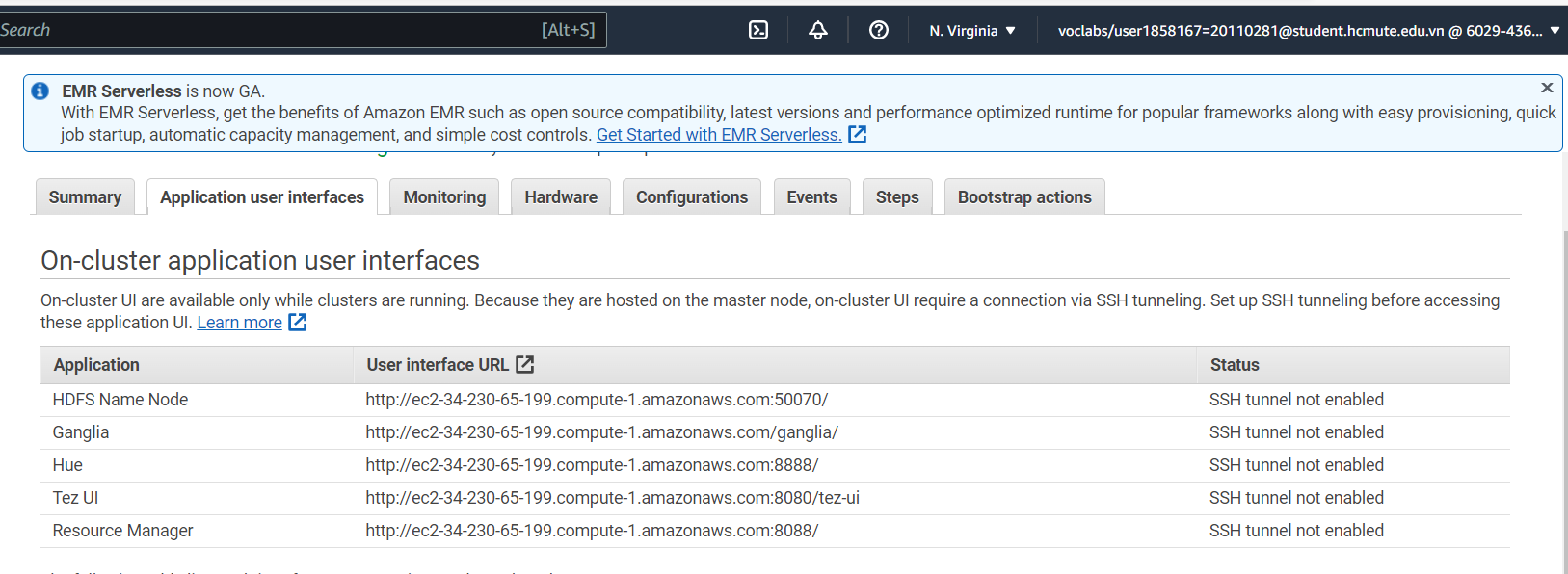
ssh -i emrproject.pem hadoop@ec2-34-230-65-199.compute-1.amazonaws.com

#note: mỗi lần tạo lại cluster thì câu lệnh trên sẽ khác ip.

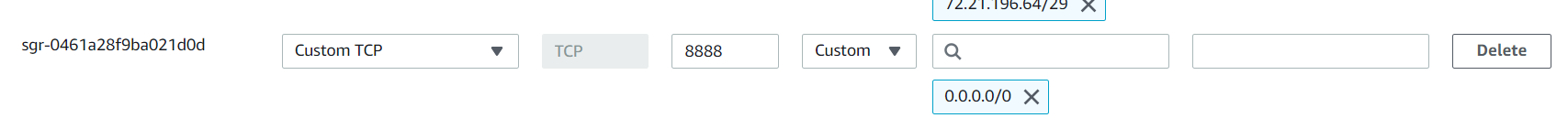
#note: nhớ bỏ ~/ trong câu lệnh từ trang aws



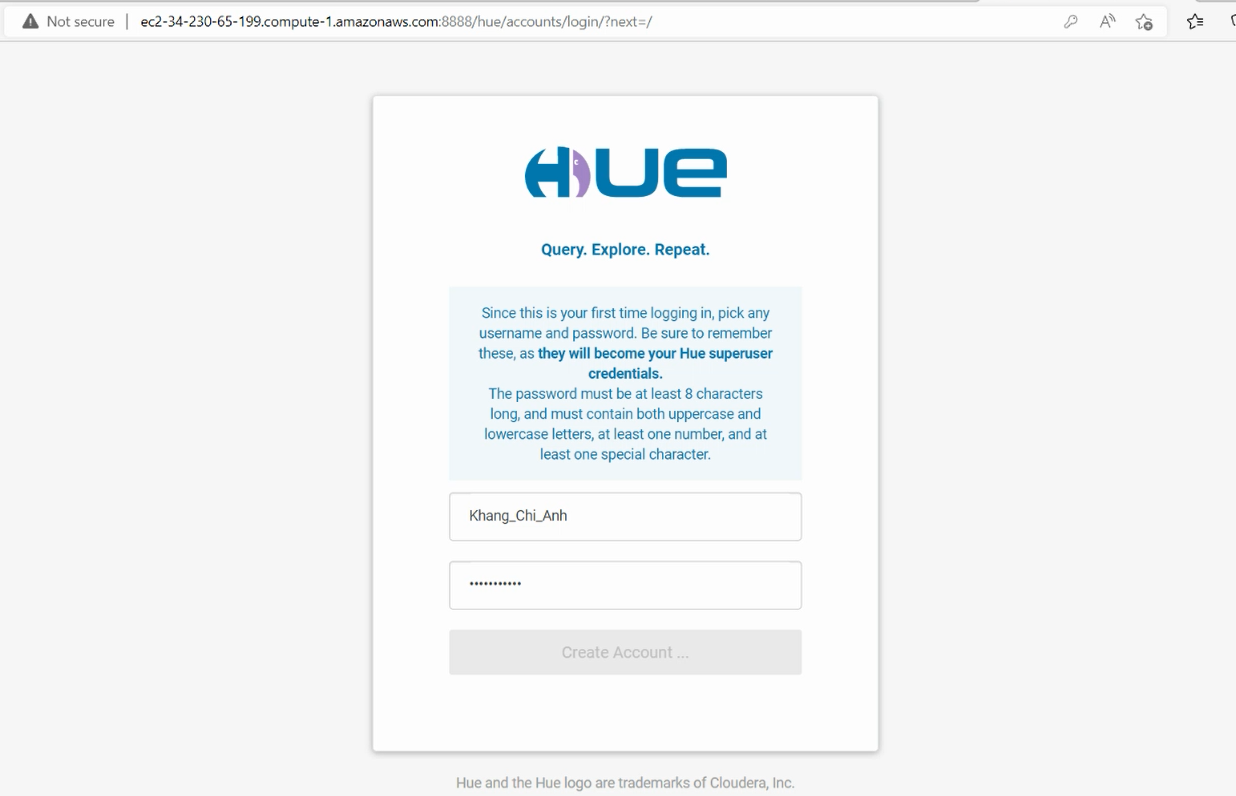
- Để chạy trên server ta cần mở các port tương ứng với từng application

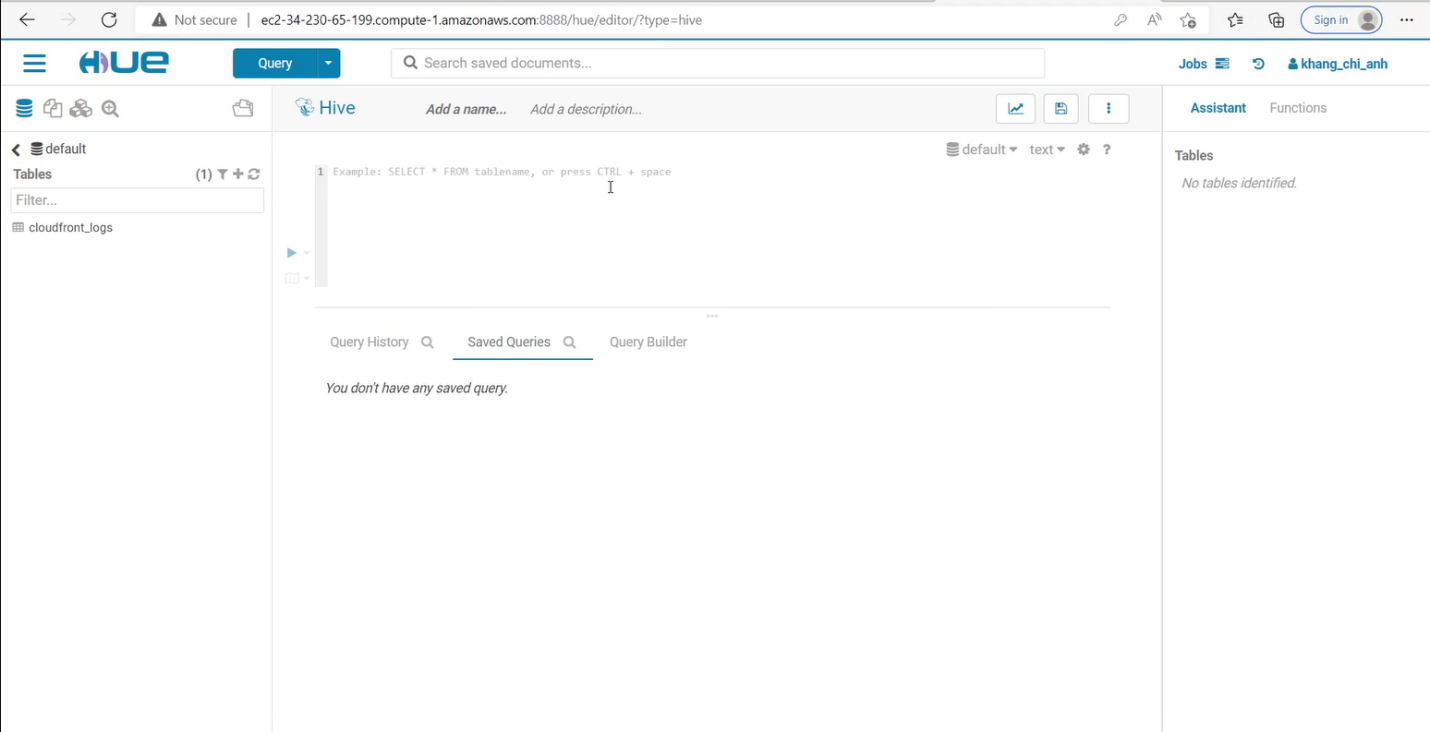


- Giờ ta mở HUE theo đường dẫn từ trang Application user interfaces, nhờ là trong phần đặt port 8888 trong inbound rules trước

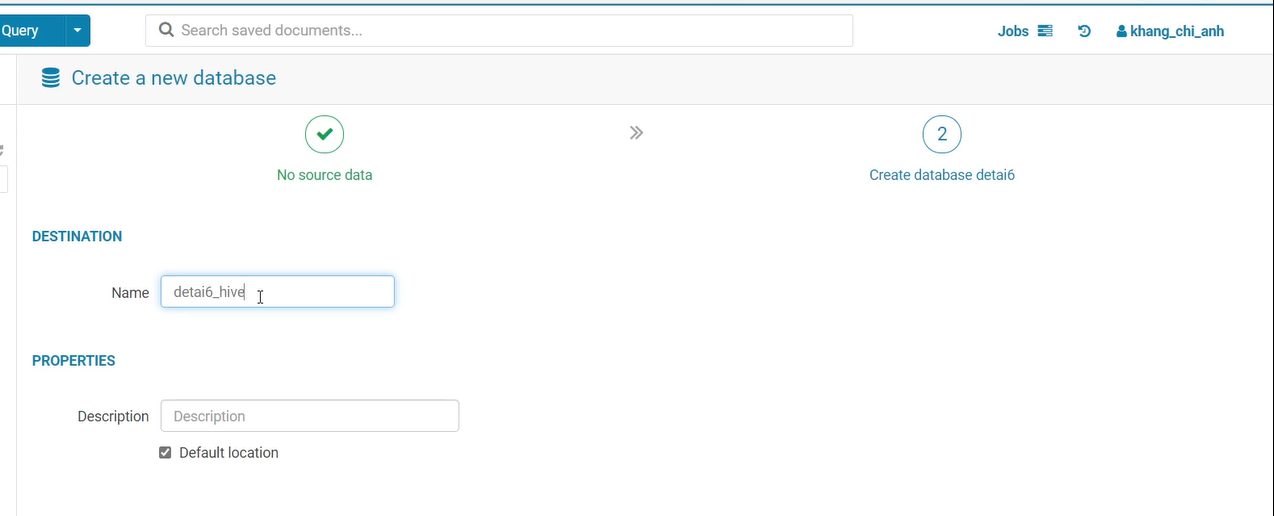


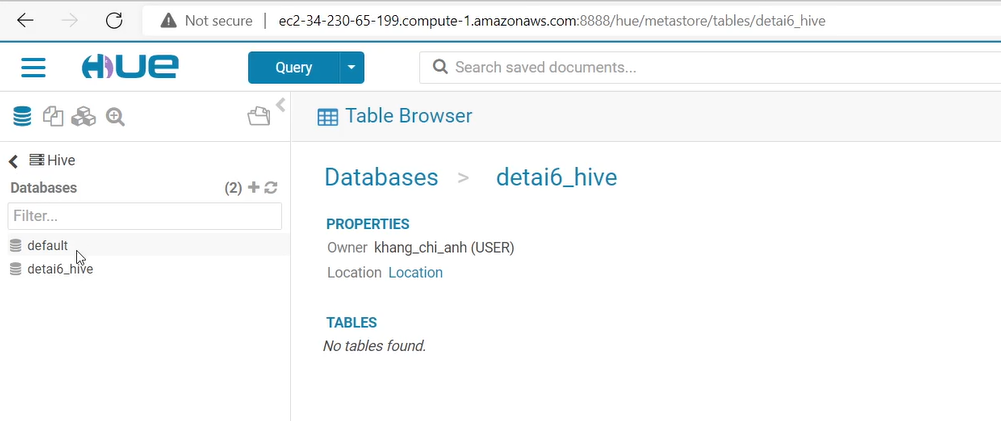
- Sau khi vào được Hue thì nó sẽ ra giống như hình



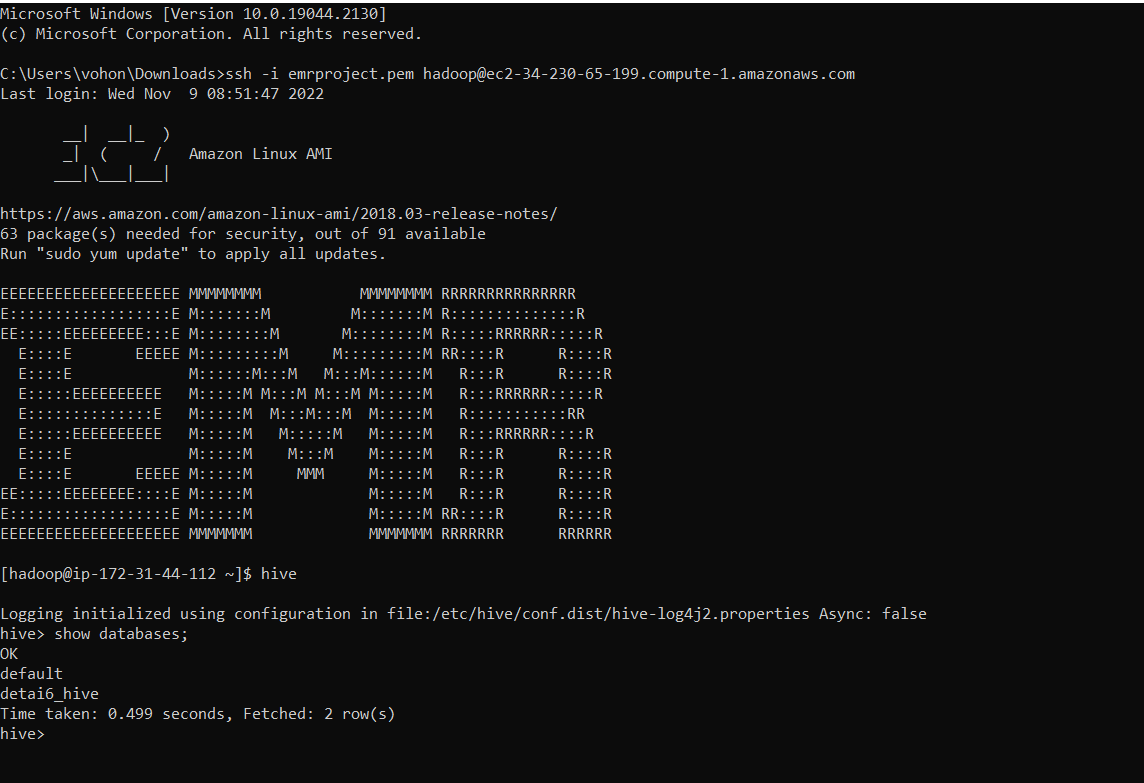


- Tiếp đến ta tạo database có detai6\_hive

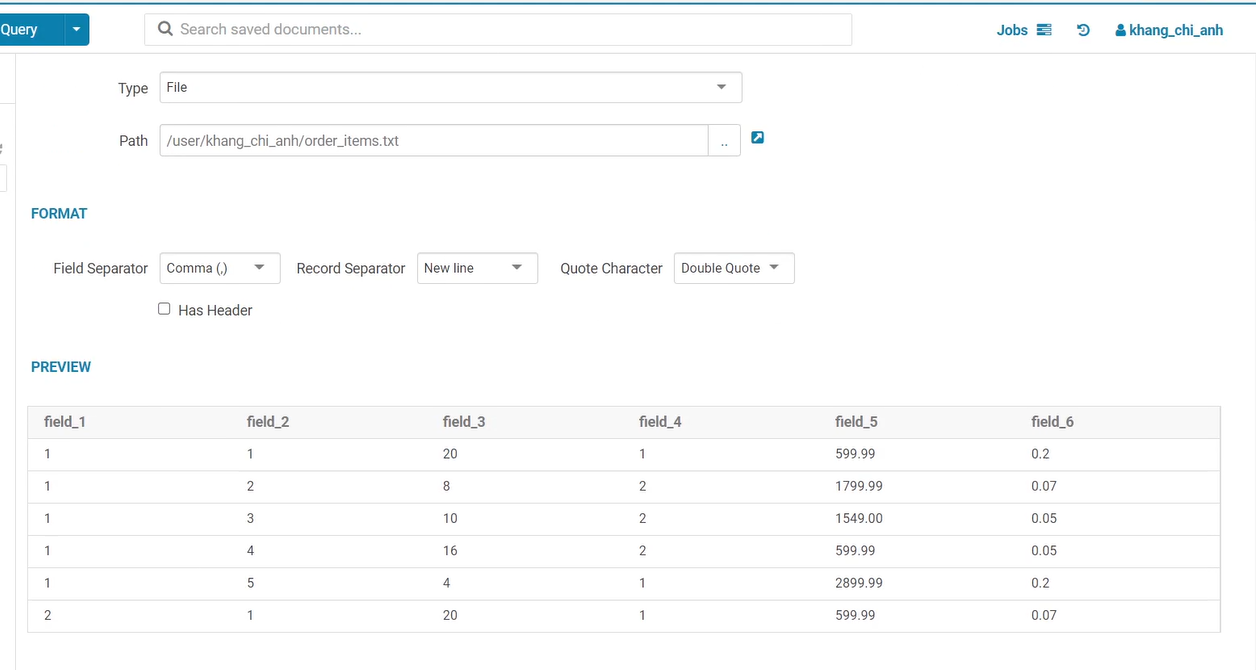




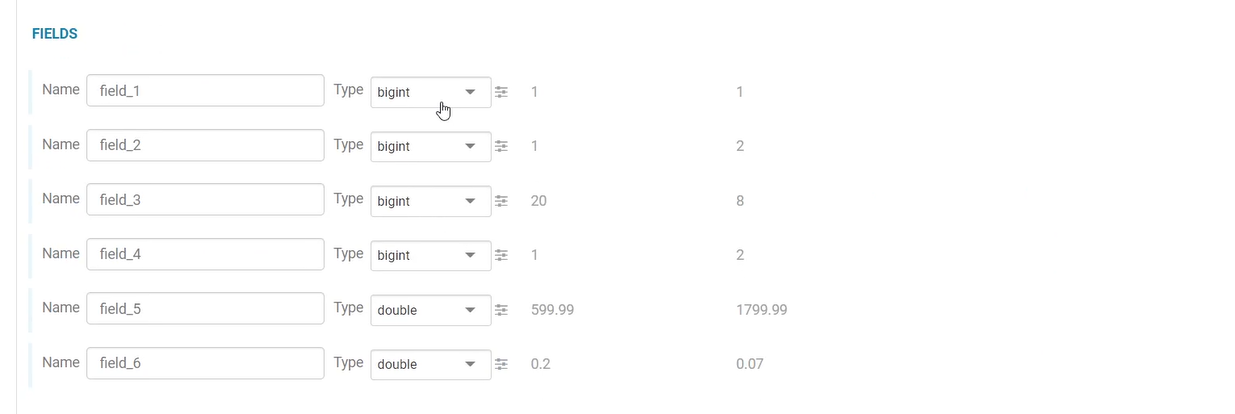
- Kiểm tra dưới command line

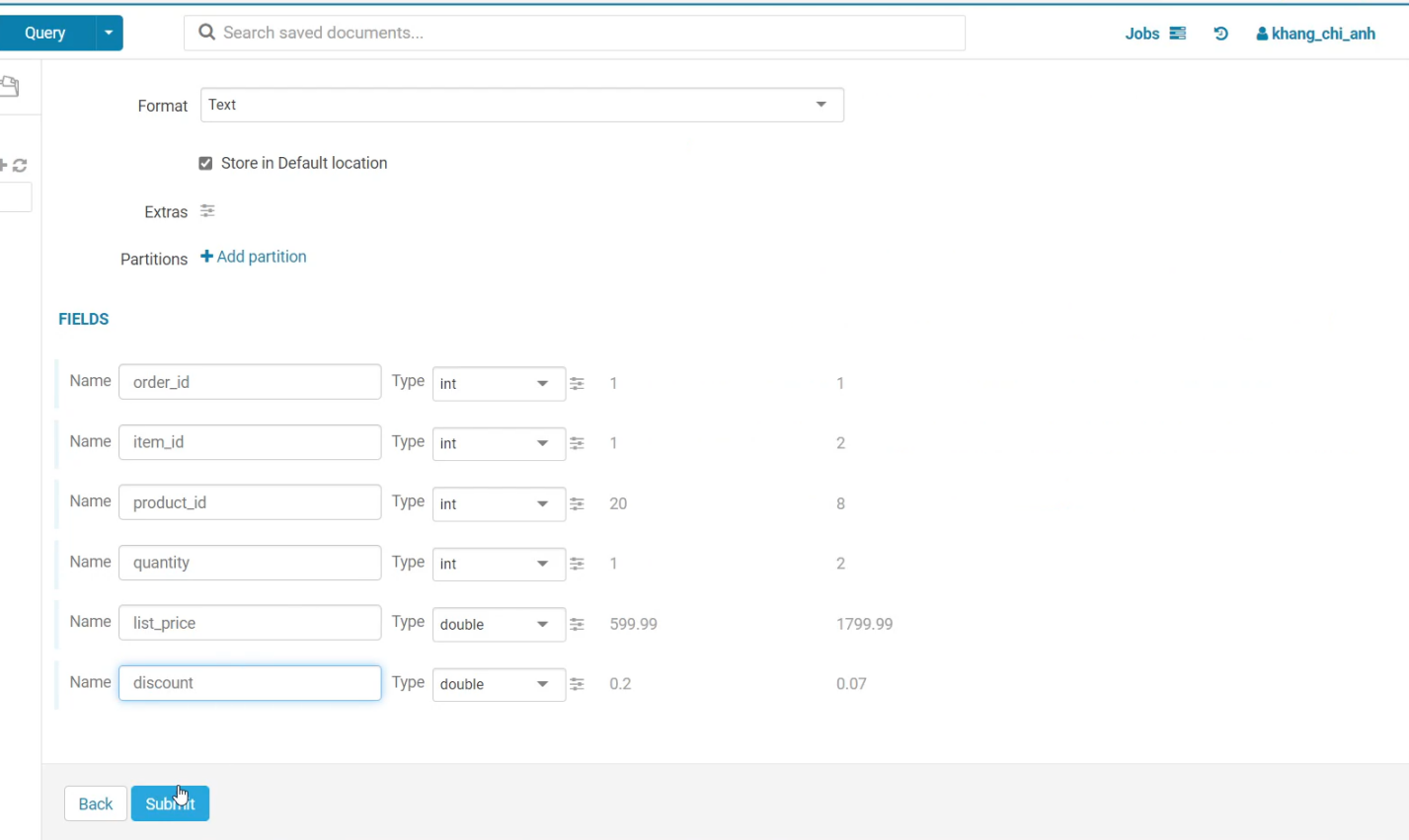


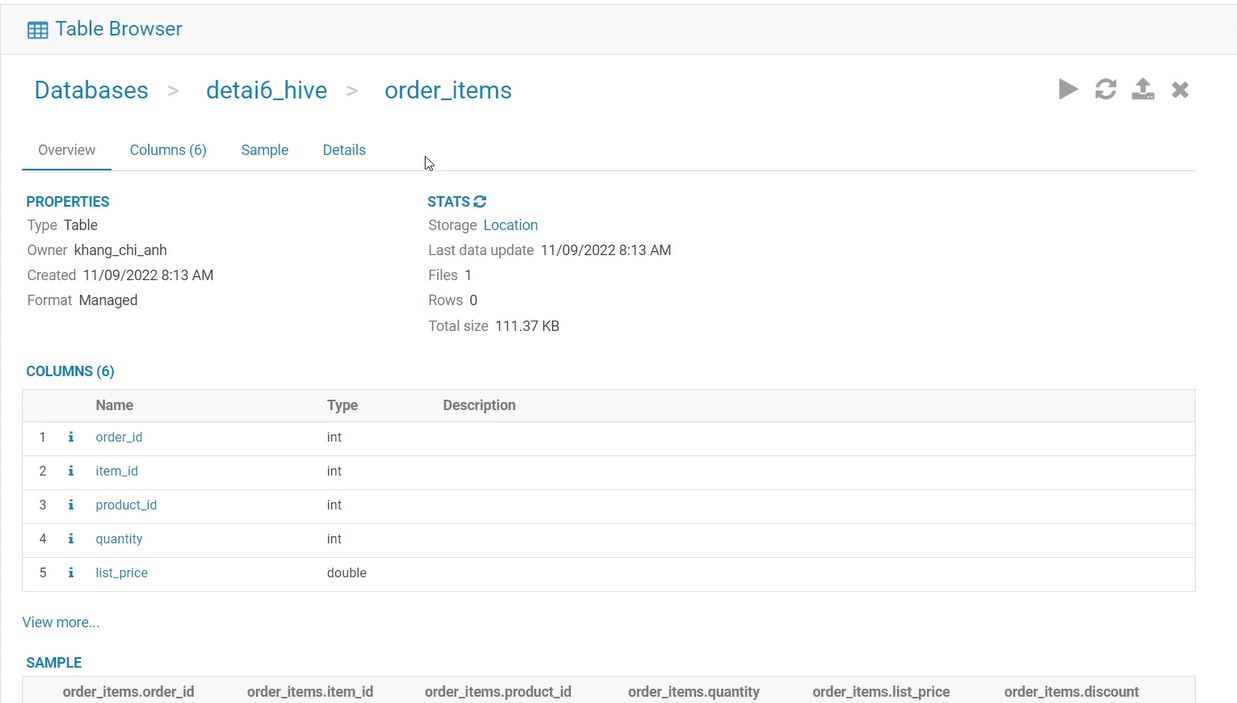
- Tạo 1 table và import dữ liệu vào table bằng file txt



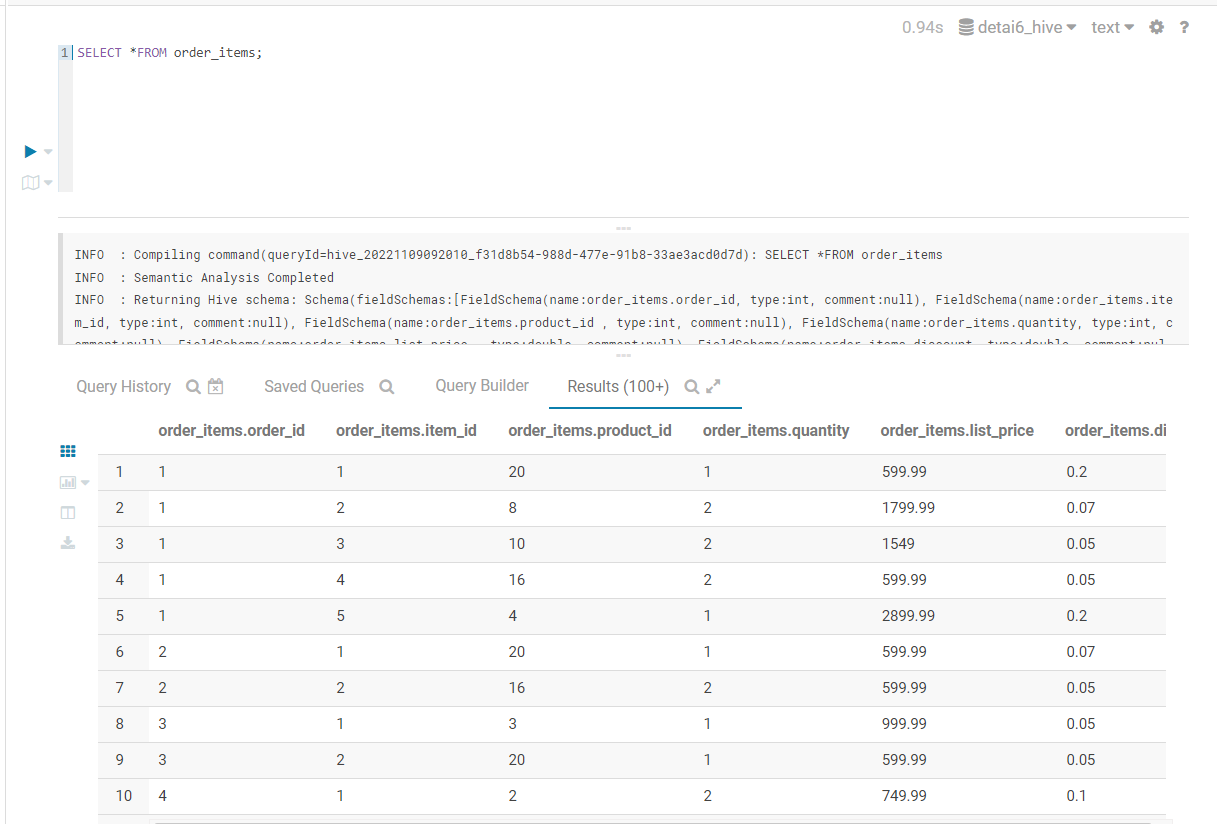
- Sau khi chọn file txt sẽ ra giao diện như hình để ta có thể nhìn tổng quát file, tiếp đên ta sẽ chỉnh lại tên các thuộc tính do file mà ta upload là chỉ có data nhưng ko có định dạng thuộc tính nên nó sẽ mặc định là field\_1, field\_2, …



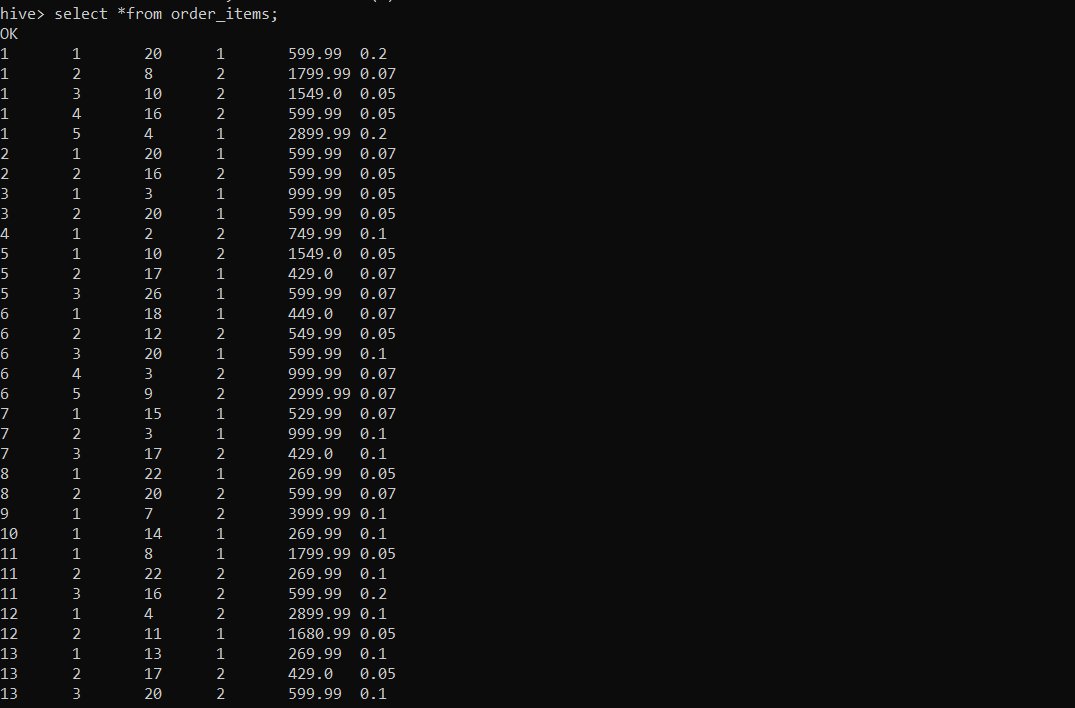




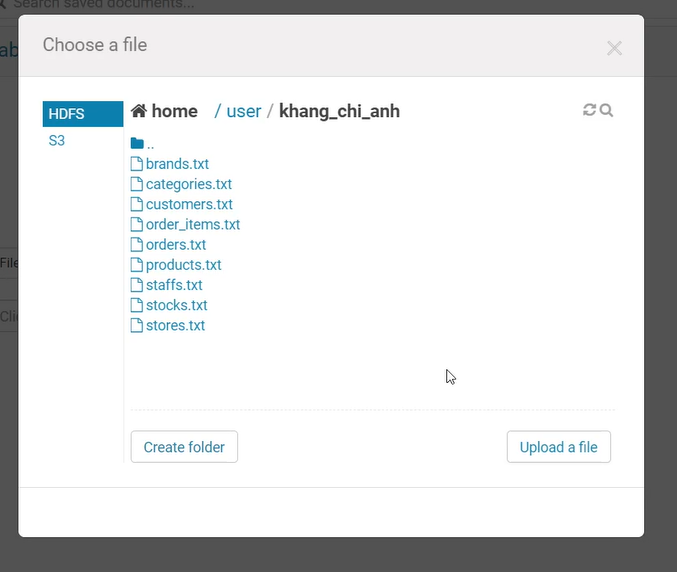
- Tiếp theo ta truy xuất dữ liệu từ bảng vừa tạo



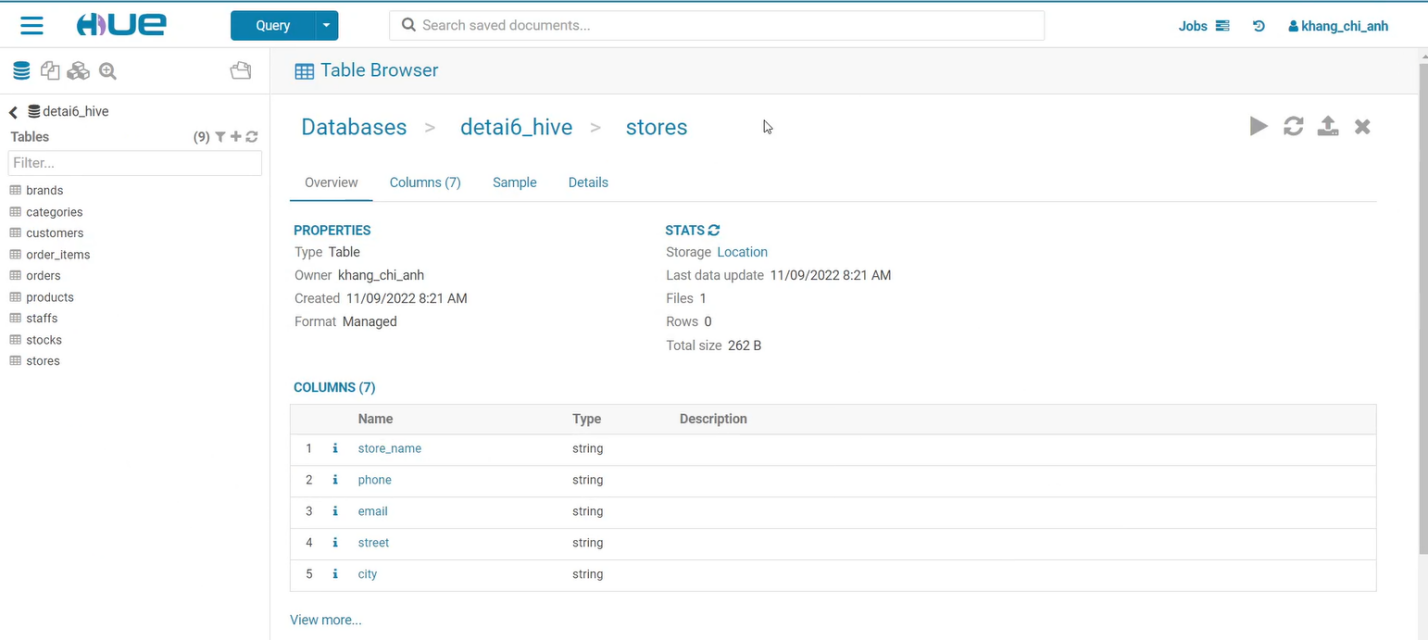
- Kiểm tra dưới command line



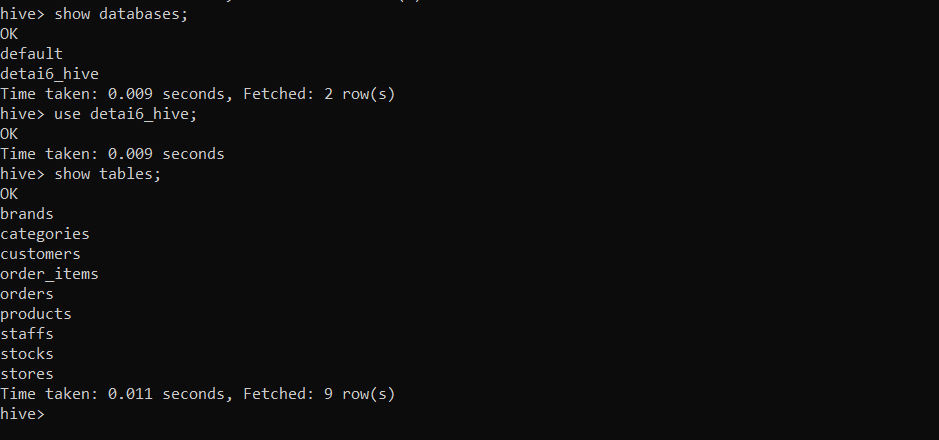
- Tương tự như tạo bảng order\_items, ta tạo thêm các bảng categories, customers, brands, orders, products, staffs, stocks, stores



- Sau khi tạo thành công các bảng



- Kiểm tra dưới commandl line



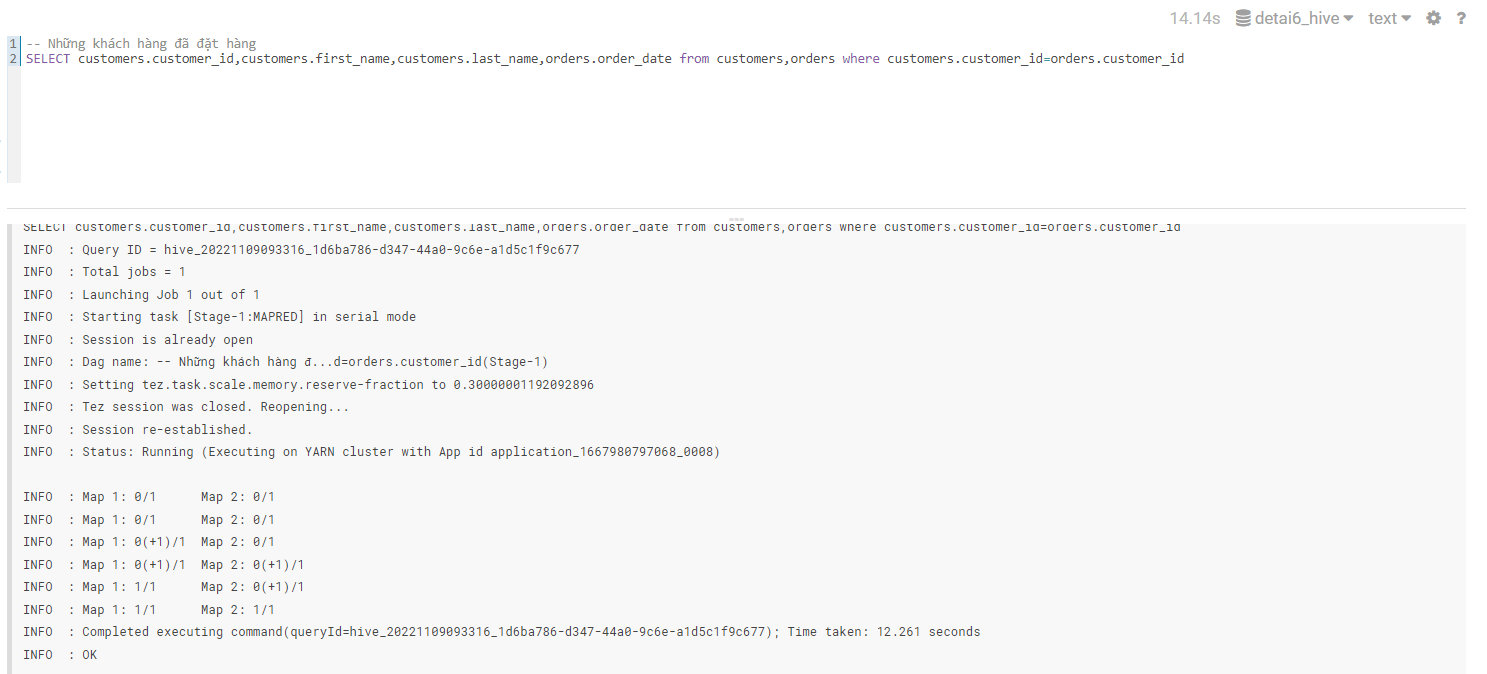
- Sau khi tạo xong các bảng ta thực hiện truy các truy vấn đơn giản

**+ Những khác hàng đã đặt hàng**

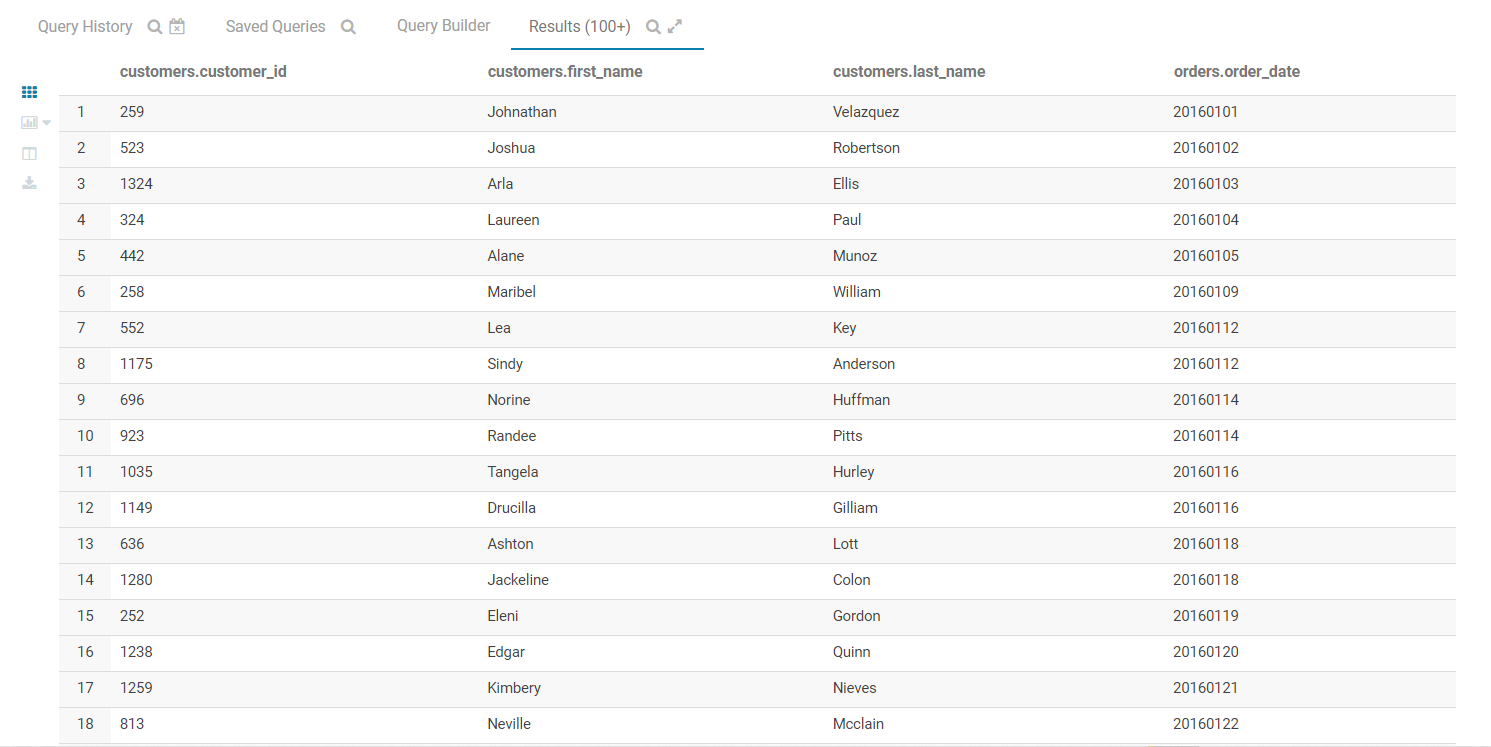
Truy vấn HQL:

SELECT customers.customer\_id, customers.first\_name, customers. last\_name, orders.order\_date from customers, orders

WHERE customers.customer\_id = orders.customer\_id



Kết quả:



Cột đầu tiên là mã khách hàng, cột thứ hai là tên đầu tiên của khách hàng, cột thứ 3 là tên cuối cùng của khách hàng, cột thứ 4 là ngày đặt hàng của khách hàng

**+ Tổng số lượng đơn hàng của mỗi nhân viên đã bán**

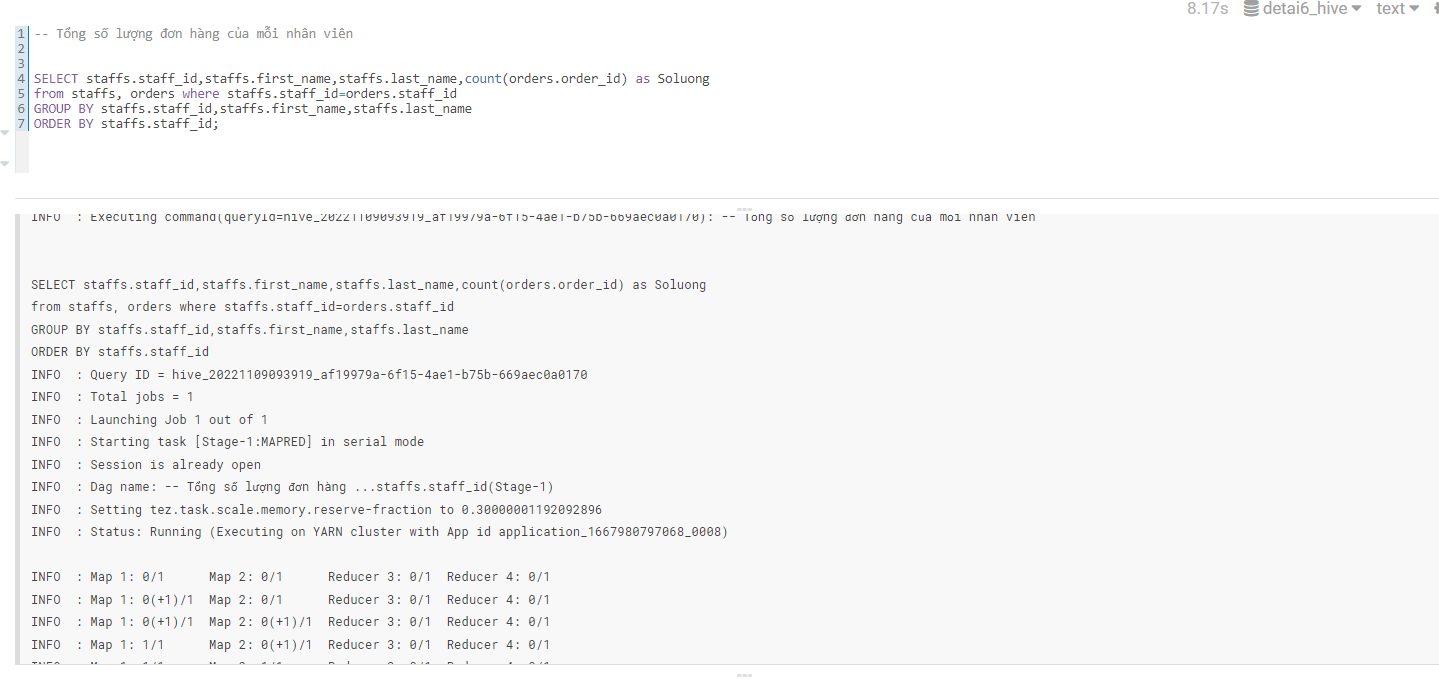
Truy vấn HQL:

SELECT staffs.staff\_id, staffs.first\_name, staffs.last\_name, count(orders.order\_id) as Soluong

FROM staffs, orders where staffs.staff\_id = orders.staff\_id

GROUP BY staffs.staff\_id, staffs.first\_name, staffs.last\_name

ORDER BY staffs.staff\_id;



Kết quả:



Cột đầu tiên là mã nhân viên, cột thứ hai là tên đầu tiên của nhân viên, cột thứ 3 là tên cuối cùng của nhân viên, cột thứ 4 là số lượng đơn hàng mà nhân viên đã bán.

Nhìn vào kết quả truy vấn, ta có thể biêt được nhân viên nào bán được nhiều hàng nhất, từ đó có thể khen thưởng để khích lệ nhân viên.

**+ Tổng số lượng đơn hàng của mỗi nhân viên trong tháng 1 năm 2016 theo thứ tự số lượng giảm dần**

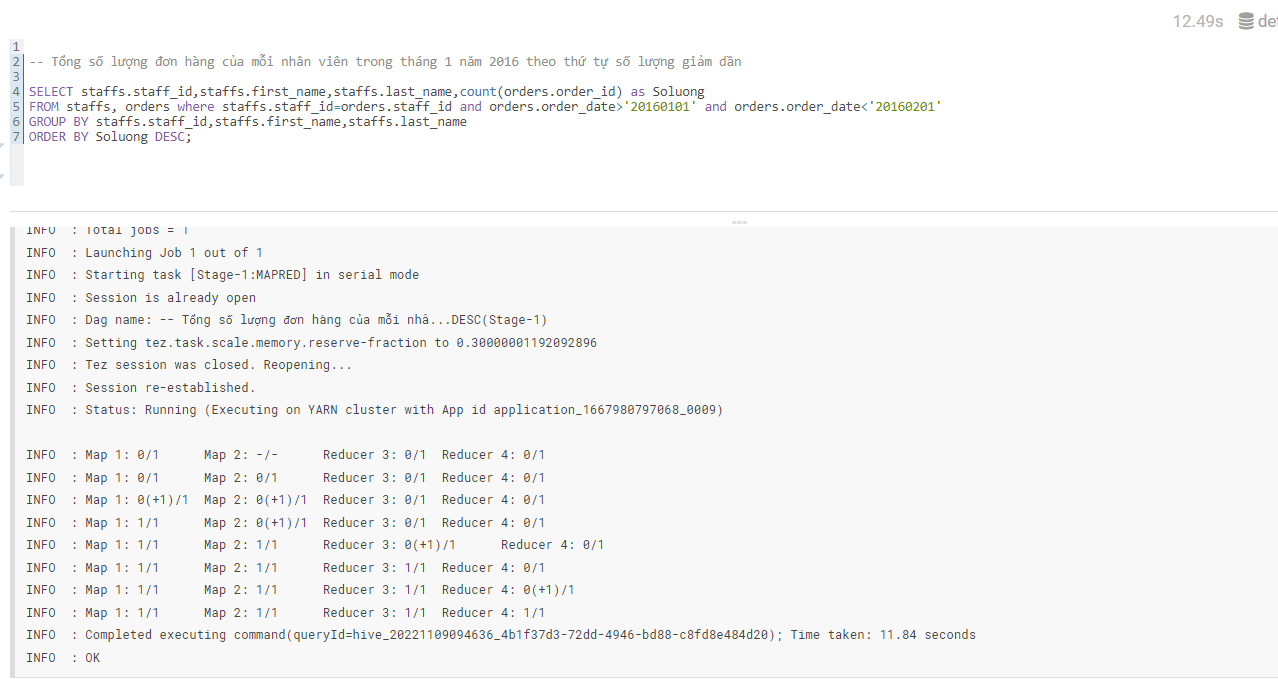
Truy vấn HQL:

SELECT staffs.staff\_id, staffs.first\_name, staffs.last\_name, count(orders.order\_id) as Soluong

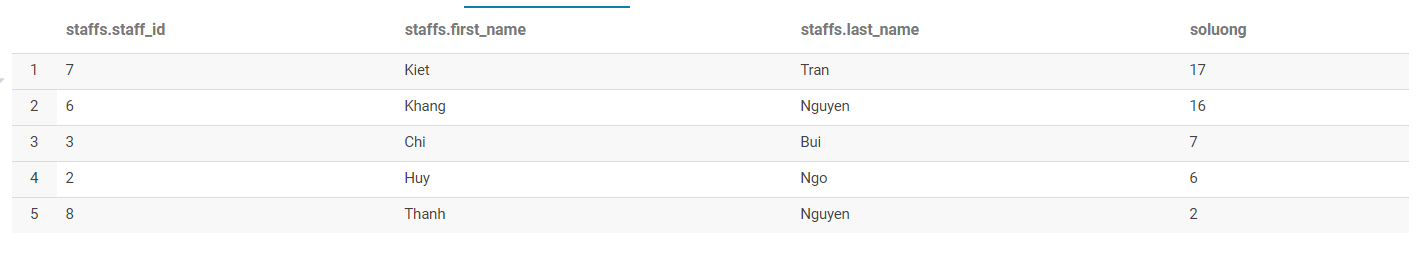
FROM staffs, orders where staffs.staff\_id = orders.staff\_id and orders.order\_date>'20160101' and orders.order\_date<'20160201'

GROUP BY staffs.staff\_id,staffs.first\_name,staffs.last\_name

ORDER BY Soluong DESC;



Kết quả:



Cột đầu tiên là mã nhân viên, cột thứ hai là tên đầu tiên của nhân viên, cột thứ 3 là tên cuối cùng của nhân viên, cột thứ 4 là số lượng đơn hàng mà nhân viên đã bán.

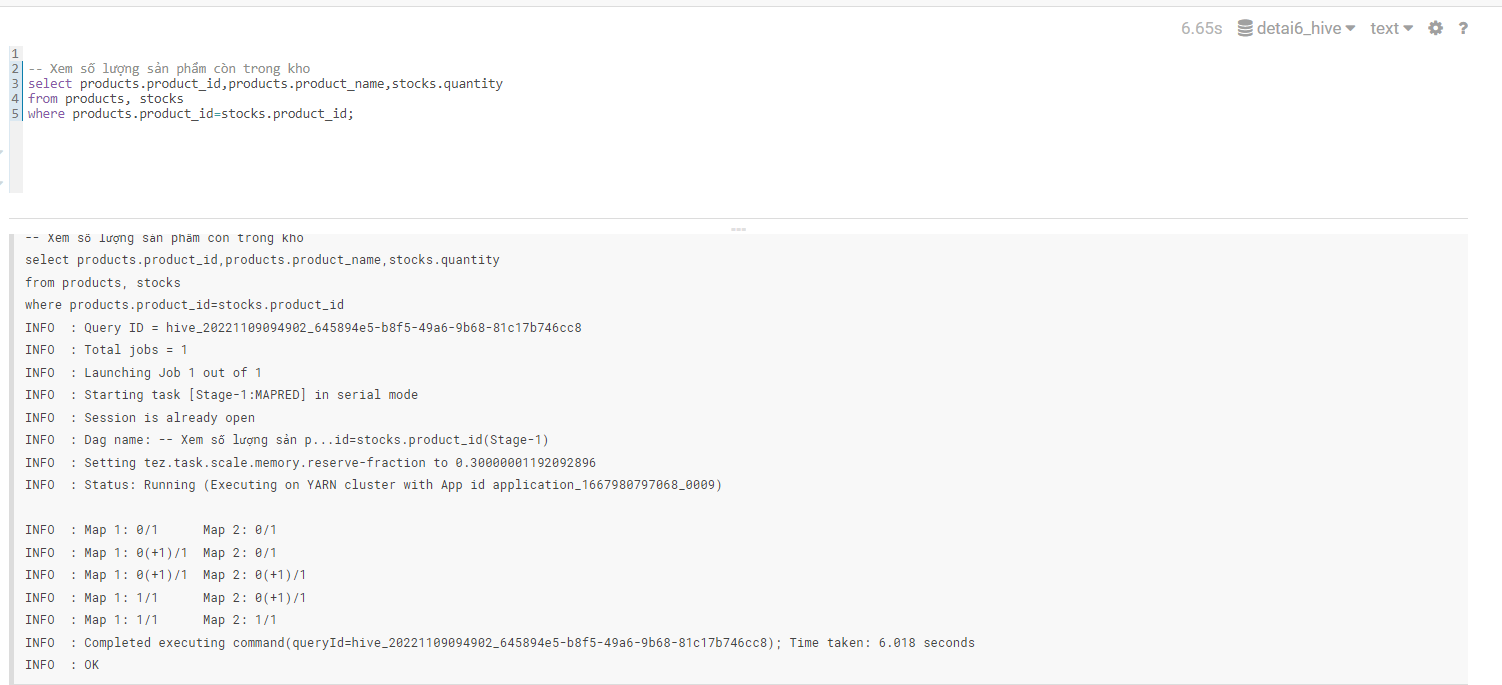
**+ Xem số lượng sản phẩm còn trong kho**

Truy vấn HQL:

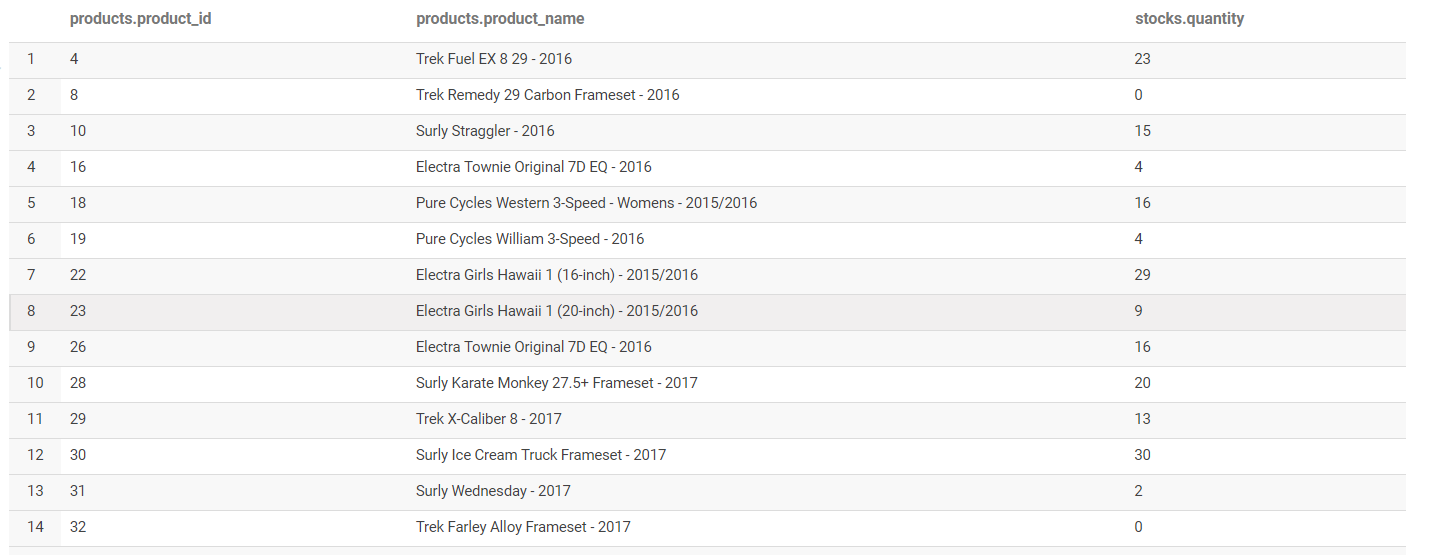
select products.product\_id,products.product\_name,stocks.quantity

from products, stocks

where products.product\_id=stocks.product\_id;



Kết quả:

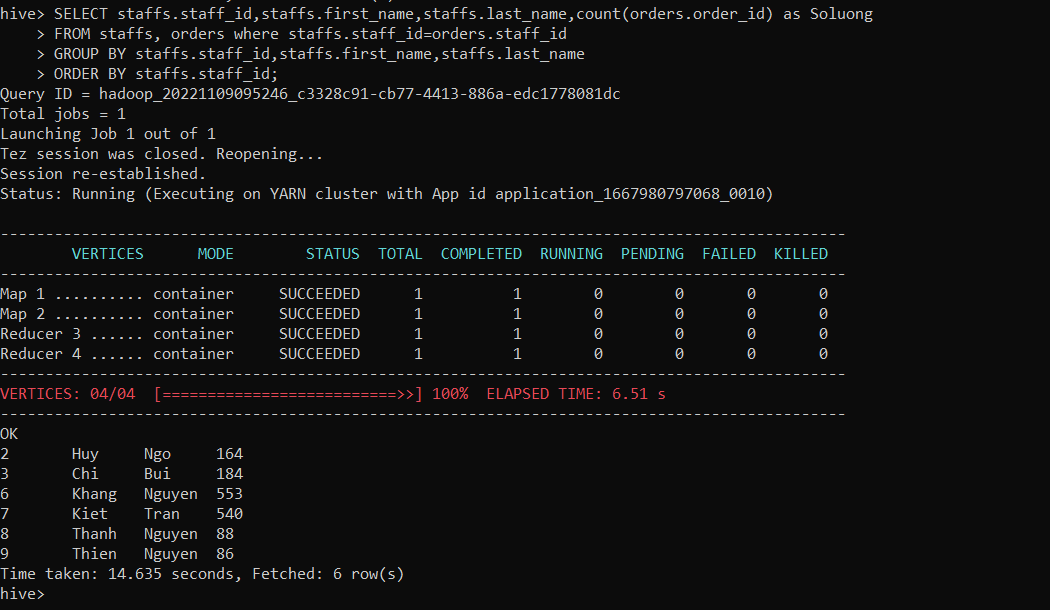


Cột thứ nhất là mã sản phẩm, cột thứ 2 là tên sản phẩm, cột thứ 3 là số lượng sản phẩm còn trong kho.

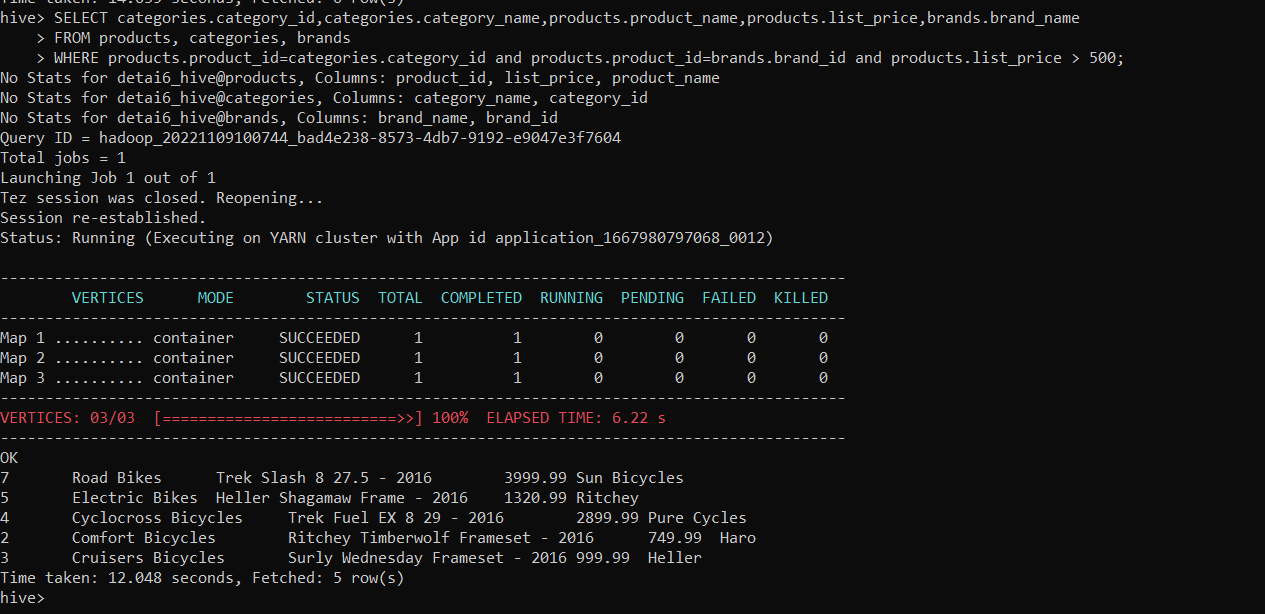
Nhìn vào kết quả truy vấn, ta có thể biêt được sản phẩm nào còn nhiều và sản phẩm nào còn ít, từ đó có thể điều chỉnh số lượng hàng nhập vào kho.

**Kiểm tra với command line**

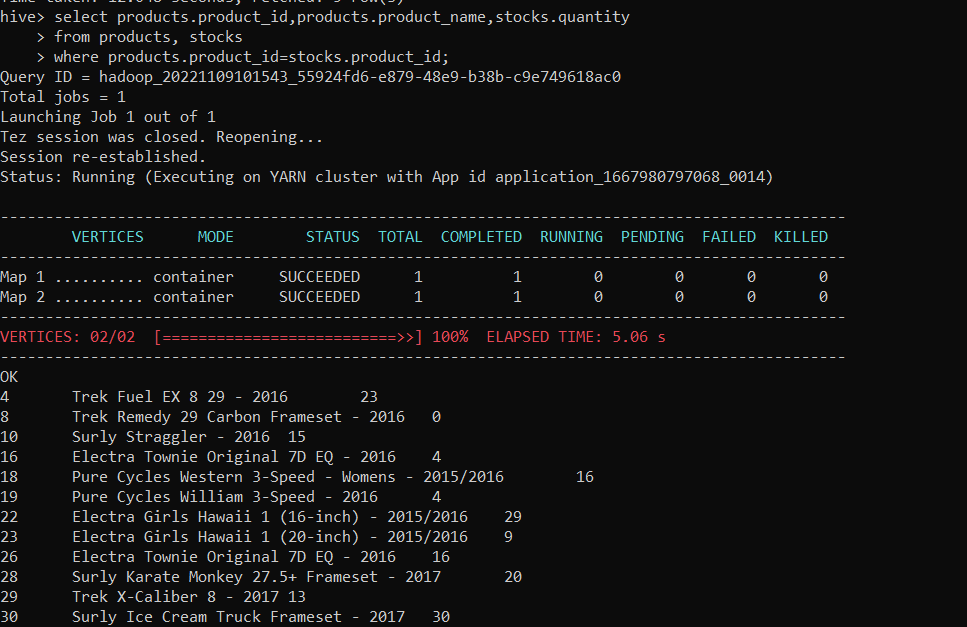
**+ Tổng số lượng đơn hàng của mỗi nhân viên đã bán**



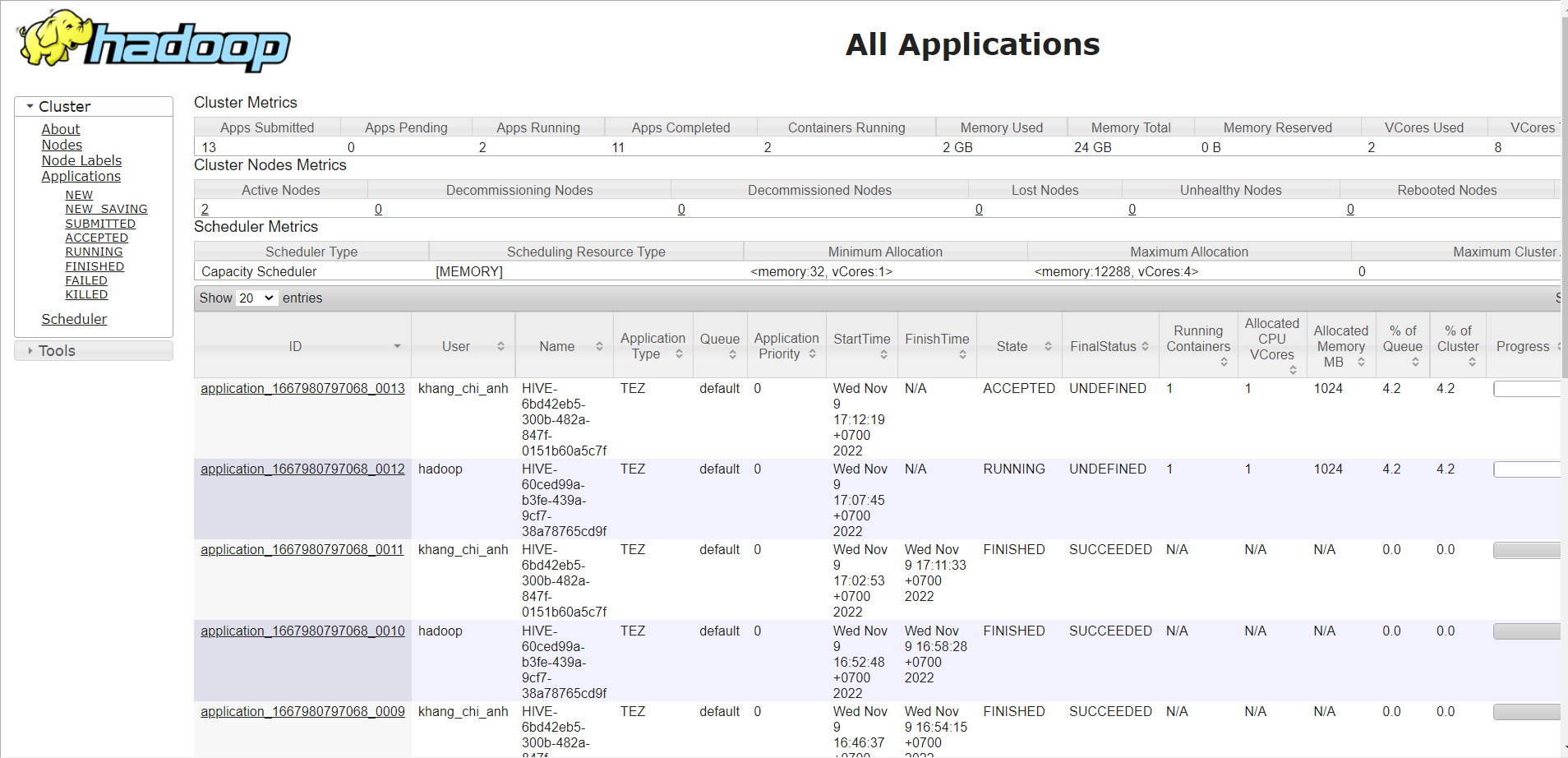
**+ Xem thông tin chi tiết của những sản phẩm có giá lớn hơn 500$**



**+ Xem số lượng sản phẩm còn trong kho**



- Hình dưới đây cho ta thấy các câu truy vấn đang thực hiện và đã kết thúc



**CHƯƠNG 3: PHẦN KẾT LUẬN**

## **Kết quả đạt được**

Sau một thời gian nghiên cứu và thực hiện đề tài “***Tìm hiểu Apache Hive và viết ứng dụng demo***”, nhóm chúng em đã đạt được những kết quả như sau:

### **1.1. Kiến thức tìm hiểu được**

Nắm bắt được các kiến thức cũng như những vấn đề liên quan trọng về Apache Hive, truy vấn với HQL và áp dụng kiến thức để thiết kế và xây dựng một Data warehouse. Biết được cách Hive hoạt đông trên Hadoop, luồng dữ liệu của Hive, đặc trưng, kiến trúc, cách tổ chức dữ liệu trong Hive.

Nắm bắt được quy trình xử lý Big Data, thiết kế và xây dựng các bảng fact, các dimension, thực hiện tích hợp dữ liệu bằng ETL, HQL để tạo truy vấn phân tích dữ liệu từ những câu truy vấn.

### **1.2. Chương trình đã làm được**

Xây dựng hoàn chỉnh một data warehouse bằng Hive với các chức năng cơ bản như:

* Quản lý các database, quản lý các bảng trong database.
* Thực hiện các câu lệnh truy vấn và tiến hành phân tích từ các câu lệnh truy vấn đó.
* Truy vấn Hive trên HUE sử dụng dịch vụ EMR của AWS

## **Ưu điểm**

* Hoạt động của Hive diễn ra một cách trơn tru và chính xác, không xảy ra tình trạng lỗi trong hệ thống.
* Lưu trữ được lượng dữ liệu lớn.
* Xử lý thông tin, truy vấn dữ liệu chính xác và nhanh chóng.

## **Nhược điểm**

* Chưa cấu hình được Hive Web Interface (HWI) do phiên bản nhóm cài đặt hiện tại quá cao, không hỗ trợ HWI.
* Để có thể sử dụng được HWI, cần phải sử dụng phiên bản Hive 2.2.0 trở xuống. Nhưng các phiên bản Hive 2.2.x không hỗ trợ Derby và Ant nên phải cài thêm. Trong quá trình cài đặt HWI, do bị giới hạn về mặt thời gian và con người nên nhóm vẫn chưa cài đặt được.
* Dịch vụ EMR bị terminated cụm Hadoop nên làm lại nhiều lần gặp nhiều khó khăn

## **Hướng phát triển**

* Tiếp tục hoàn thiện các chức năng còn thiếu.
* Xây dựng và quản lý data warehouse lớn hơn.
* Tìm hiểu và cài đặt giao diện HWI ở các phiên bản cũ hơn hoặc liên kết với các giao diện hỗ trợ ngôn ngữ khác.
* Thực hiện thêm nhiều các câu truy vấn.
* Đánh giá chi phí sử dụng trên AWS

**TÀI LIỆU THAM KHẢO**

[1]. <https://docs.aws.amazon.com/emr/latest/ManagementGuide/emr-what-is-emr.html>

[2]. <https://www.youtube.com/watch?v=u6NSowmu7Dc>

[3]. <https://trello.com/b/B7Mz7xOR/ti%CC%80m-hi%C3%AA%CC%89u-apache-hive-va%CC%80-vi%C3%AA%CC%81t-%C6%B0%CC%81ng-du%CC%A3ng-demo>

[4]. Apache Hive: <https://hive.apache.org>