**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TPHCM**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO ĐỒ ÁN**

**MÔN: ĐIỆN TOÁN ĐÁM MÂY**

**ĐỀ TÀI:**

**TÌM HIỂU APACHE HIVE**

**VÀ VIẾT ỨNG DỤNG DEMO**

|  |  |  |
| --- | --- | --- |
| **GVHD: Huỳnh Xuân Phụng**  *Nhóm sinh viên thực hiện* | | |
| **Stt** | **Tên** | **MSSV** |
| 1 | Phan Tấn Thịnh | 17133060 |
| 2  3 | Nguyễn Hiền Nhung  Nguyễn Tiến Duy | 17133044  17133008 |





**TP.HCM, 1/2021**

**Mục lục**

[**1.** **Tổng quan về Hive:** 3](#_Toc46882580)

[**2.** **Kiến trúc của Hive:** 3](#_Toc46882581)

[**3.** **Hoạt động của Hive:** 5](#_Toc46882582)

[**4.** **Các chế độ hoạt động trong Hive.** 6](#_Toc46882583)

[**5.** **Đặc trưng của Hive** 7](#_Toc46882584)

[**6.** **Mô hình dữ liệu trong Hive:** 7](#_Toc46882585)

[**9.** **Ngôn ngữ truy vẫn HiveQL** 11](#_Toc46882586)

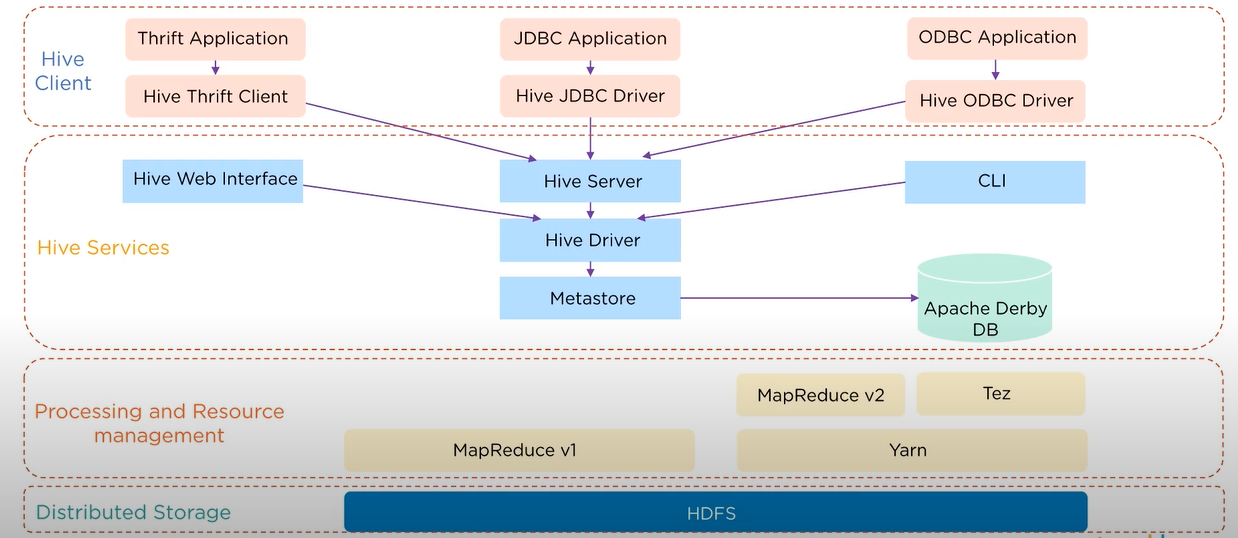
1. **Tổng quan về Hive:**

* 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 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.

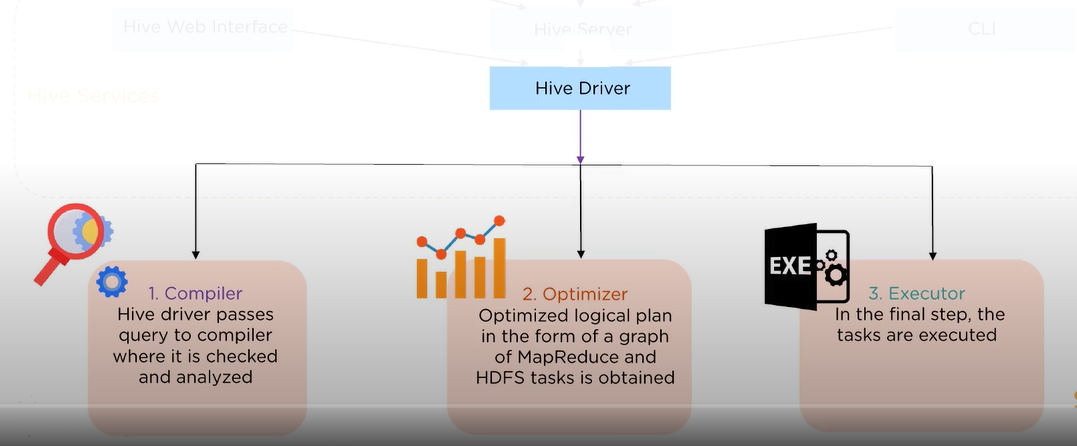
* HCatalog là một thành phần của Hive. Đây là lớp quản lý lưu trữ  cho Hadoop (table and management layer), nó cho phép người dùng với các công cụ xử lý dữ liệu khác nhau bao gồm cả Pig và MapReduce thực thi hoạt động đọc, ghi một cách dễ dàng hơn.
* WebHCat cung cấp một dịch vụ cho phép bạn có thể thực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

1. **Kiến trúc của Hive:**

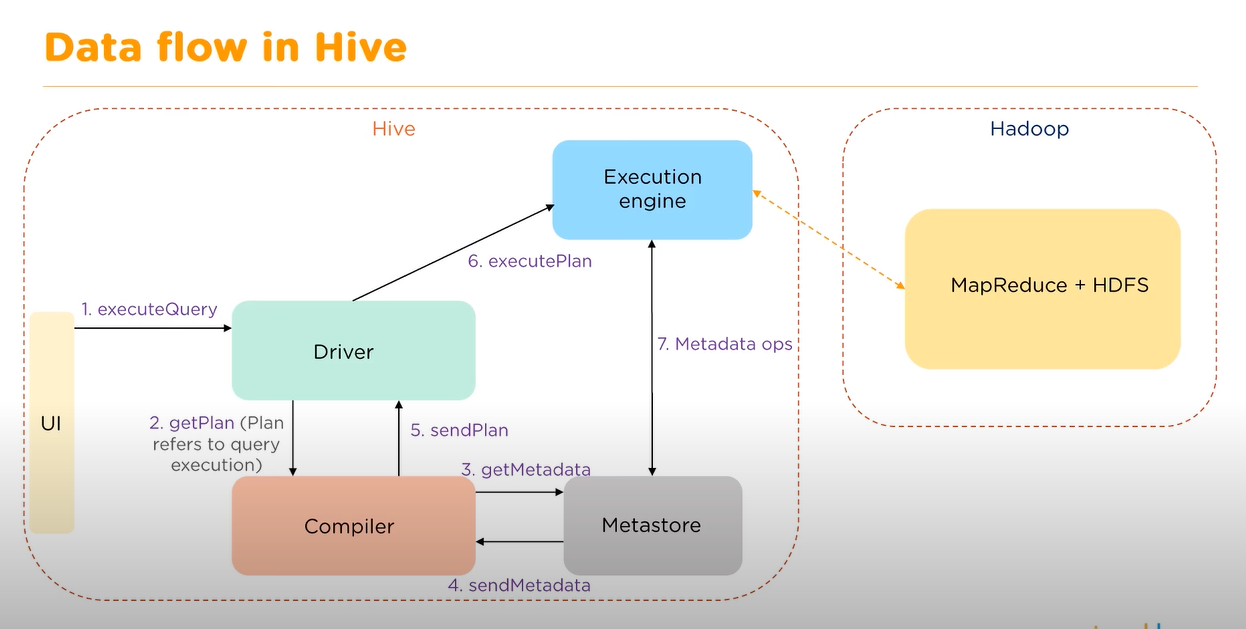


Hive có các thành phần chính là:

* 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.



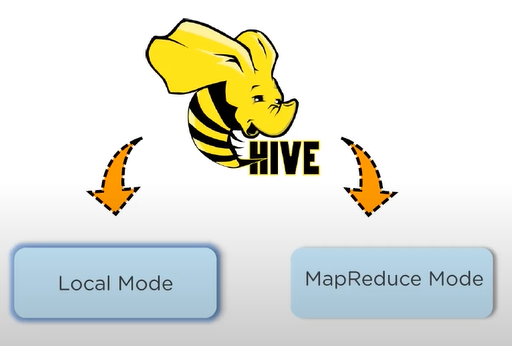
Hình 2.1. Kiến trúc của Hive

1. **Hoạt động của Hive:**

Hình 3.1. Mô hình hoạt động của Hive

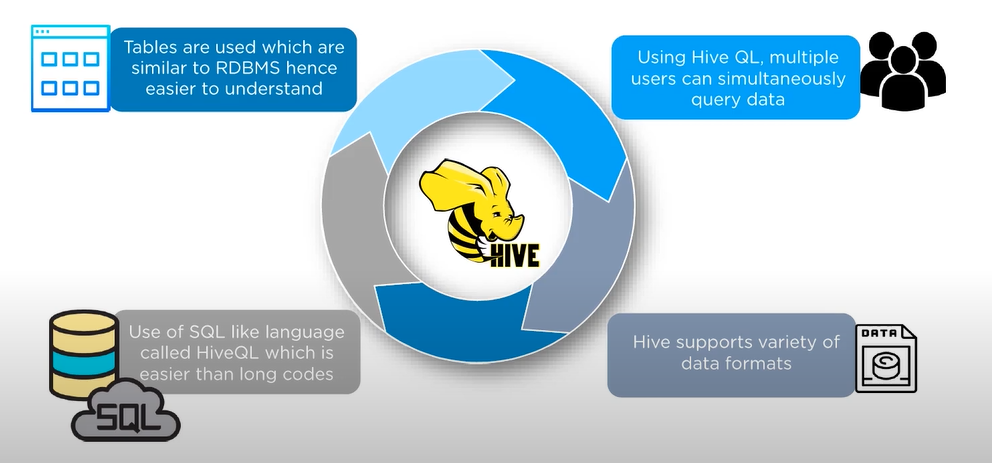
Quy trình hoạt động của Hive có thể được mô tả theo các bước sau:

1. Các truy vấn tới từ User Interface (CLI, Hive Web Interface, ThirftServer) được gửi tới thành phần Driver (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)
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.**



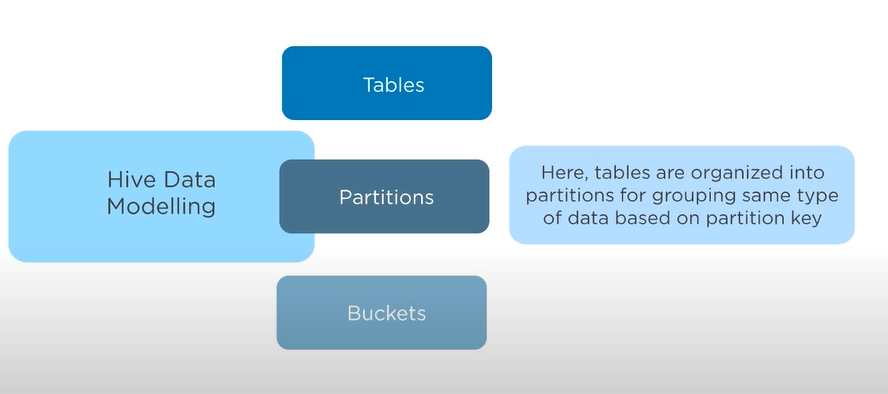
* **Local Mode:**
* **Được sử dụng khi hadoop có một nút dữ liệu và dữ liệu nhỏ.**
* **Việc xử lý sẽ rất nhanh trên các bộ dữ liệu nhỏ hơn có trong máy cục bộ.**
* **MapReduce Mode:**
* **Được sử dụng khi hadoop có nhiều nút dữ liệu và dữ liệu được trải trên các ghi chú dữ liệu khác nhau.**
* **Xử lý bộ dữ liệu lớn có thể hiệu quả hơn khi sử dụng chế độ này.**

1. **Đặc trưng của Hive**



* **Bảng được sử dụng tương tự như RDBMS do đó dễ hiểu hơn.**
* **Sử dụng hiveQL, nhiều người dùng có thể truy vấn dữ liệu đồng thời.**
* **Sử dụng ngôn ngữ như sql gọi là hiveql dễ hơn mã dài.**
* **Hive hỗ trợ nhiều định dạng dữ liệu.**

1. **Mô hình dữ liệu trong Hive:**



Hình 4.1. Hive Data Model

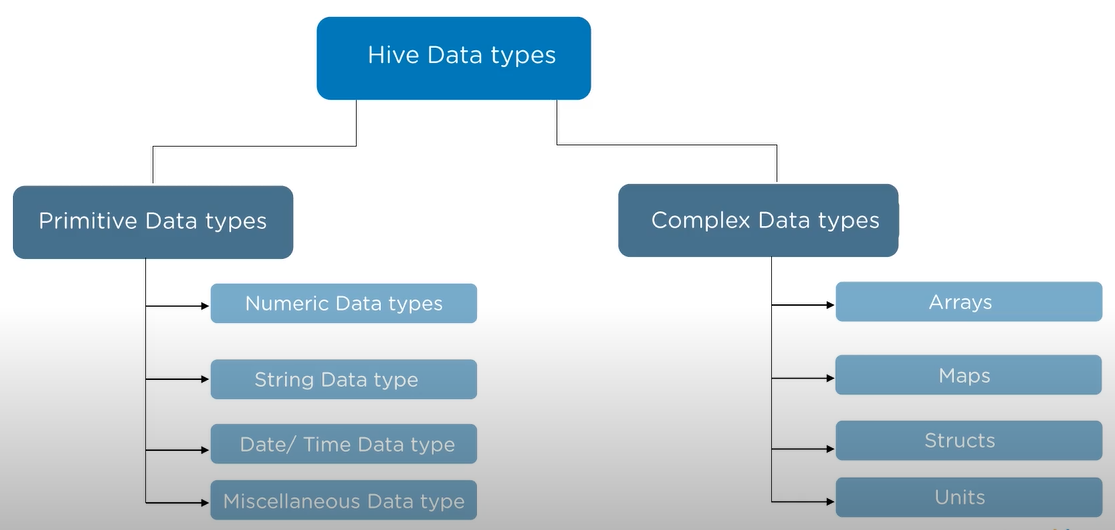
Dữ liệu trong Hive được tổ chức thành các kiểu sau:

* 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: ở đây, các bảng được tổ chức thành các phần để nhóm cùng loại dữ liệu dựa trên khóa phân vùng.

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 Hive:**



Primittive Data types:

* Numeric Data types:
  + Integer
  + Float
  + Decimal
* String Data types
  + Char
  + String
* Date/Time Data type
  + Timestamp
  + Date
  + Interval
* Miscellaneous Data type
  + Boolean
  + Binary

Complex Data types:

* Arrays: Kiểu mảng. (Vd: array<data\_type>)
* Structs: là kiểu dữ liệu mà mỗi phần tử bên trong đó có thể được truy cập thông qua việc sử dụng ký hiệu (.)

(Vd: struct<col\_name: data\_type[COMMEM col\_comment],…..>

* Maps: 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’]

(Vd: map<primitive\_type, data\_type>)

1. **Tạo các đối tượng trong Hive**

* Create Database Statement

CREATE DATABASE|SCHEMA [IF NOT EXISTS] <database name>

* Drop Database Statement

DROP DATABASE StatementDROP (DATABASE|SCHEMA) [IF EXISTS] database\_name [RESTRICT|CASCADE];

* **Create Table Statement**

**CREATE [TEMPORARY] [EXTERNAL] TABLE [IF NOT EXISTS] [db\_name.] table\_name**

**[(col\_name data\_type [COMMENT col\_comment], ...)]**

**[COMMENT table\_comment]**

**[ROW FORMAT row\_format]**

**[STORED AS file\_format]**

* **Load Data Statement**

**LOAD DATA [LOCAL] INPATH 'filepath' [OVERWRITE] INTO TABLE tablename**

* **Alter Table Statement**

**ALTER TABLE name RENAME TO new\_name**

**ALTER TABLE name ADD COLUMNS (col\_spec[, col\_spec ...])**

**ALTER TABLE name DROP [COLUMN] column\_name**

**ALTER TABLE name CHANGE column\_name new\_name new\_type**

**ALTER TABLE name REPLACE COLUMNS (col\_spec[, col\_spec ...])**

* **Drop Table Statement**

**DROP TABLE [IF EXISTS] table\_name;**

* **Creating a View**

**CREATE VIEW [IF NOT EXISTS] view\_name [(column\_name [COMMENT column\_comment], ...) ]**

**[COMMENT table\_comment]**

**AS SELECT ...**

* **Dropping a View**

**DROP VIEW view\_name**

* **Creating an Index**

**CREATE INDEX index\_name**

**ON TABLE base\_table\_name (col\_name, ...)**

**AS 'index.handler.class.name'**

* **Dropping an Index**

**DROP INDEX <index\_name> ON <table\_name>**

1. **Ngôn ngữ truy vẫn HiveQL**

* Ngôn ngữ truy vấn Hive (HiveQL) là ngôn ngữ truy vấn để Hive xử lý và phân tích dữ liệu có cấu trúc trong Metastore.
* 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];

Tài liệu tham khảo

<https://www.educba.com/install-hive/>

<https://viblo.asia/p/tim-hieu-ve-hadoop-bWrZn1XwKxw>

<https://www.tutorialspoint.com/hive/index.htm>

<https://hive.apache.org/>