**ỨNG DỤNG PHÂN TÁN**

**SEMINAR CHỦ ĐỀ:**

**“HIVE”**

**MỤC LỤC**

[I. THÔNG TIN THÀNH VIÊN NHÓM 3](#_Toc452158481)

[II. PHÂN CÔNG CÔNG VIỆC 3](#_Toc452158482)

[III. NỘI DUNG 4](#_Toc452158483)

[1. Giới thiệu. 4](#_Toc452158484)

[2. Hive là gì? 4](#_Toc452158485)

[3. Kiến trúc của Hive 5](#_Toc452158486)

[4. Quy trình làm việc của Hive 6](#_Toc452158487)

[5. Các kiểu dữ liệu của Hive 7](#_Toc452158488)

[6. HiveSQL 8](#_Toc452158489)

[7. So sánh Hive và SQL 11](#_Toc452158490)

[8. Cài đặt HIVE 12](#_Toc452158491)

# THÔNG TIN THÀNH VIÊN NHÓM

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **MSSV** | **HỌ TÊN** | **EMAIL** | **SĐT** |
|  | 1212354 | Nguyễn Minh Tân | [nmtan.fit@gmail.com](mailto:Nmtan.fit@gmail.com) | 0975048757 |
|  | 1212060 | Nguyễn Tiến Dũng | [nguyentiendung880@gmail.com](mailto:Nguyentiendung880@gmail.com) | 0978232410 |
|  | 1212048 | Diệp Mỹ Dung | [dmdung94@gmail.com](mailto:dmdung94@gmail.com) | 01675929083 |

# PHÂN CÔNG CÔNG VIỆC

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **THÀNH VIÊN** | **CÔNG VIỆC** | **ĐÁNH GIÁ** |
|  | 1212354  Nguyễn Minh Tân |  |  |
|  | 1212060  Nguyễn Tiến Dũng |  |  |
|  | 1212048  Diệp Mỹ Dung |  |  |

# NỘI DUNG

## **Giới thiệu.**

Như đã biết thuật ngữ “Big Data” được sử dụng để nói đến tập dữ liệu lớn trong đó hàng ngày nó gia tăng về cả khối lượng, tốc độ và đa dạng về kiểu dữ liệu. Việc quản lý và xử lý Big Data tạo ra một thách thức vô cùng lớn.Và Apache đã tạo ra một framework để quản lý và xử lý các thách thức mà big data mang lại, đó là Hadoop.

Hadoop có thể hiểu là một framework mã nguồn mở sử dụng để lưu trữ và xử lý dữ liệu lớn. Nó bao gồm hai thành phần chính là: MapReduce và HDFS (Hadoop Distributed File System)

* MapReduce: Đây là một mô hình lập trình song song, nó xử lý dữ liệu có cấu trúc, bán cấu trúc, và không có cấu trúc.
* HDFS: Hệ thống HDFS là nơi được sử dụng để lưu trữ và xử lý dữ liệu.

Câu hỏi đặt ra lúc này là: Làm thế nào để làm việc với Hadoop? Tất nhiên là Apache cũng cung cấp cho chúng ta các công cụ để có thể làm việc được với Hadoop một cách dễ dàng nhất như Sqoop, Pig, Hive.

* Sqoop: Dùng để chuyển đổi dữ liệu qua lại giữa RDBMS (dữ liệu quan hệ) với HDFS.
* Pig: Nền tảng là ngôn ngữ thủ tục được sử dụng để phát triển một kịch bản cho các hoạt động MapReduce.
* Hive: Nền tảng là SQL Script để làm hoạt động MapReduce.

Việc có nhiều các công cụ để làm việc với Hadoop cũng đặt cho chúng ta một câu hỏi là: khi nào thì sử dụng Sqoop, Pig, Hive. Câu trả lời là chúng ta sẽ lựa chọn chúng dựa trên dữ liệu phân tích. Với những dữ có cấu trúc rõ ràng thì Hive là lựa chọn tốt. Với những dữ liệu có cấu trúc và bán cấu trúc thì Pig sẽ dễ dàng tếp cận trong việc tạo kịch bản cho MapReduce. Còn với những dữ liệu đa dạng (có cấu trúc, bán cấu trúc, phi cấu trúc) thì các tiếp cận là tạo chương trình Java MapReduce truyền thống.

1. **Hive là gì?**

|  |  |
| --- | --- |
|  | Hive là hạ tầng kho dữ liệu cho Hadoop. Nhiệm vụ chính là cung cấp sự tổng hợp dữ liệu, truy vấn và phân tích. Nó hỗ trợ phân tích các tập dữ liệu lớn được lưu trong HDFS của Hadoop cũng như trên Amazon S3. Điểm hay của Hive là hỗ trợ truy xuất giống SQL đến dữ liệu có cấu trúc, được biết với tên HiveSQL (hoặc HQL) cũng như phân tích Big Data với MapReduce. Hive không được xây dựng để hồi đáp nhanh các câu truy vấn nhưng nó được xây dựng cho các ứng dụng khai thác dữ liệu (data mining). Các ứng dụng khai thác dữ liệu có thể mất nhiều phút đến nhiều giờ để phân tích dữ liệu và Hive được dùng chủ yếu. |

1. **Kiến trúc của Hive**

****

Sơ đồ mô tả kiến trúc của Hive

|  |  |
| --- | --- |
| User Interface | Hive cung cấp một giao diện web để tương tác với HDFS. Các giao diện người dùng mà Hive hỗ trợ là Hive Web UI, Hive Command Line, và Hive HD Insight. |
| Meta Store | Lưu thông tin cơ bản về cấu trúc dữ liệu. Các thông tin gồm ID của database (schema), ID của table, ID của index, định dạng của table, ... |
| HiveQL Process Engine | Tương tự như SQL, dùng để truy vấn dữ liệu dựa trên thông tin metastore cung cấp. Đây là phương pháp thay thế cho việc phải viết trương trình mapreduce truyền thống. |
| Execution Engine | Chuyển hóa các lệnh Hive QL thành MapReduce |
| HDFS or HBASE | HDFS, HBASE dùng để lưu trữ dữ liệu |

1. **Quy trình làm việc của Hive**

****

|  |  |
| --- | --- |
|  | **Execute Query**  Giao diện Hive như Command Line hay giao diện Web gửi câu truy vấn đến driver (bất kỳ trình điều khiển cơ sở dữ liệu như JDBC, ODBC,…) để thực thi. |
|  | **Get Plan**  Dirver với sự hỗ trợ của các trình biên dịch truy vấn phân tích các truy vấn để kiểm tra các cú pháp, kế hoạch truy vấn và yêu cầu truy vấn. |
|  | **Get Metadata**  Các trình biên dịch gửi Metadata đến Metastore. |
|  | **Send Metadata**  Metastore gửi Metadata phản hồi trình biên dịch. |
|  | **Send Plan**  Trình biên dịch sẽ kiểm tra các yêu cầu và gửi lại kế hoạch cho driver. Đến đây, các phân tích và biên dịch của một truy vấn là hoàn tất. |
|  | **Execute Plan**  Driver sẽ gửi kế hoạch thực hiện cho các execution engine. |
|  | **Execute Job**  MapReduce thực hiện quá trình execution. Các execution engine gửi các job đến JobTracker. |
| 7.1 | **Metadata Ops**  Trong khi execution, các execution engine có thể execute metadata với Metastore. |
|  | **Fetch Result**  Các execution engine nhận được kết quả từ Data nodes. |
|  | **Send Results**  Execution engine gửi giá trị kết quả đến Driver. |
|  | **Send Results**  Driver gửi kết quả về Hive Interfaces. |

1. **Các kiểu dữ liệu của Hive**
   * + Các kiểu dữ liệu của Hive được chia thành 4 loại chính:
   * Column Types
   * Literals
   * Null Values
   * Complex Types
     + Column Types:
   * Integral Types

|  |  |  |
| --- | --- | --- |
| Type | Postfix | Example |
| TINYINT | Y | 10Y |
| SMALLINT | S | 10S |
| INT | - | 10 |
| BIGINT | L | 10L |

* + String Types: Sử dụng dấu nháy đơn ( '') hoặc dấu ngoặc kép ( ""). Bao gồm 2 loại dữ liệu: VARCHAR và CHAR.

|  |  |
| --- | --- |
| Data Type | Length |
| VARCHAR | 1 to 65355 |
| CHAR | 255 |

* + Timestamp: Có hai định dạng là: “YYYY-MM-DD HH:MM:SS.fffffffff” và “yyyy-mm-dd hh:mm:ss.ffffffffff”.
  + Date: Với định đạng là:{{YYYY-MM-DD}}
  + Decimal
  + Union Types
    - Literals
  + Floating Point Types: Loại dữ liệu này bao gồm các kiểu dữ liệu DOUBLE.
  + Decimal Type
    - Complex Types
  + Arrays: ARRAY<data\_type>
  + Map: MAP<primitive\_type, data\_type>
  + Structs: STRUCT<col\_name : data\_type [COMMENT col\_comment], ...>

1. **HiveSQL**

**6.1. Tạo CSDL**

* + - Cú pháp:

**CREATE DATABASE** IF **NOT** **EXISTS** <Database name>;

[**COMMENT** database\_comment]

[**LOCATION** hdfs\_path]

[**WITH DBPROPERTIES** (property\_name=property\_value, ...)];

[**COMMENT:** Mô tả ngắn về CSDL]

[**LOCATION:** Xác định vị trí của CSDL]

[**WITH DBPROPERTIES:** Xác định bất kỳ thuộc tính của CSDL với hình thức (key, value)];

* + - Ví dụ:

**CREATE DATABASE** IF **NOT EXISTS** test\_db

**COMMENT** "Test Database created for tutorial"

**WITH DBPROPERTIES**(

'Date' = '2016-24-05’,

'Creator' = 'nguyentiendung’,

'Email' = 'bala@somewhere.com');

**6.2. Xóa CLDL**

* + - Cú pháp:

**DROP** **DATABASE** IF **EXISTS** <Database name> [**RESTRICT |** **CASCADE**];

**RESTRICT:** Tùy chọn này không cho phép CSDL bị xóa cho đến khi toàn bộ các bảng bị xóa

**CASCADE:** Xóa tất cả các bảng trước rồi sau đó mới xóa CSDL

* + - Ví dụ:

**DROP** **DATABASE** IF **EXISTS** test\_db;

**6.3. Thay đổi CSDL**

* + - Cú pháp:

**ALTER DATABASE** <Database name>

**SET DBPROPERTIES** (property\_name=property\_value, ...);

* + - Ví dụ:

**ALTER DATABASE** <Database name>

**SET DBPROPERTIES** ('Creator' = 'mydung’);

**6.4. Tạo bảng**

* + - Cú pháp:

**CREATE** **TABLE** IF **[NOT** **EXISTS] [db\_name.]**<Database name>

**[(col\_name** data\_type**)[COMMNET** col\_commnet**],..)]**

**[COMMNET** table\_commnet**]**

**[ROW FORMAT** row\_format**]**

**[STORE AS** file\_format**]**

* + - Ví dụ:

**CREATE** **TABLE** IF **NOT** **EXISTS** test\_db.employees(

name STRING **COMMENT** 'Employee name',

salary FLOAT **COMMENT** 'Employee salary',

address STRING **COMMENT** 'Home address')

**COMMENT** 'Description of the table'

TBLPROPERTIES ('creator'='me', 'created\_at'='2012-01-02 10:00:00', ...)

**LOCATION** '/user/hive/warehouse/mydb.db/employees';

* + - Câu lệnh hiển thị tất cả các bảng có trong CSDL:

**SHOW** TABLES **IN** <Database name>;

**6.5. Sửa bảng**

* + - Cú pháp:

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

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

* + - Ví dụ:

**ALTER TABLE** employees **RENAME TO** emp;

**ALTER TABLE** emp **DROP COLUMN** deductions;

**ALTER TABLE** emp **ADD COLUMNS** (name STRING **COMMENT** 'Employee name');

**ALTER TABLE** emp **CHANGE** name emp\_name VARCHAR;

**ALTER TABLE** emp **REPLACE COLUMNS** (name STRING **COMMENT** 'Employee name');

**6.6. Xóa bảng**

* + - Cú pháp:

**DROP** TABLES **[IF EXISTS]** <table\_name>;

* + - Ví dụ:

**DROP** TABLES **IF EXISTS** emp;

**6.7. Partition**

* + - Tạo Partition:

**ALTER TABLE** table\_name

**ADD [IF NOT EXISTS] PARTITION (**p\_column = p\_col\_value,...**)**

**[LOCATION** ‘location’**]**

* + - Ví dụ:

**ALTER TABLE** emp

**ADD IF NOT EXISTS PARTITION (**address= “Ho Chi Minh”)

* + - Đổi tên Partition:

**ALTER TABLE** table\_name **PARTITION (**p\_column = p\_col\_value,...**)**

**RENAME TO PARTITION** (p\_column = p\_col\_value,...);

* + - Xóa Partition:

**ALTER TABLE** table\_name

**DROP [IF EXISTS] PARTITION** (p\_column = p\_col\_value,...);

**6.8. View**

* + - Tạo view:

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

**[COMMENT** table\_comment**]**

**AS SELECT**....

* + - Ví dụ:

**CREATE VIEW** emp\_salary AS

**SELECT** \* **FROM** emp

**WHERE** salary>5.000.000;

* + - Xóa view:

**DROP VIEW** view\_name;

VD: **CREATE VIEW** emp\_salary;

**6.9. Index**

* + - Tạo index:

**CREATE INDEX** index\_name

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

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

**[WITH DEFERRED REBUILD]**

**[IDXPROPERTIES** (property\_name=property\_value, ...)**]**

**[IN TABLE** index\_table\_name**]**

**[**

**[ ROW FORMAT ...] STORED AS ...**

**| STORED BY...**

**]**

**[LOCATION** hdfs\_path**]**

**[TBLPROPERTIES (...)];**

* + - Ví dụ:

**CREATE INDEX** index\_salar**y ON TABLE** emp(salary)

* + - Xóa index:

**DROP INDEX** index\_salary **ON** emp;

**6.10. Câu lệnh SELECT**

* + - Cú pháp:

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

**FROM** table\_reference

**[WHERE** where\_condition**]**

**[GROUP BY** col\_list**]**

**[HAVING** having\_condition**]**

**[CLUSTER BY** col\_list **| [DISTRIBUTE BY** col\_list**] [SORT BY** col\_list**]]**

**[LIMIT** number**];**

1. **So sánh Hive và SQL**
   * + Hive: Sử dụng những câu lệnh như SQL và biên dịch những câu lệnh này thành tập hợp các tác vụ MapReduce.

|  |  |  |
| --- | --- | --- |
| **Tiêu chí so sánh** | **HIVE** | **SQL** |
| Phát triển bởi | Facebook | ISO/IEC (1986) |
| Mục đích sử dụng | Báo cáo, truy xuất nhanh dữ liệu, trả lời những câu hỏi cấp thiết, gần gũi với SQL | Quản lý dữ liệu quy mô vừa và nhỏ |
| Ngôn ngữ | HiveQL (SQL query) | SQL-92 standard |
| Kiểu dữ liệu | Có thêm hỗ trợ array, map và struct | Int, float, text, string, boolean, datetime... |
| JOIN | Hỗ trợ outer join và Equi-join. | Hỗ trợ đầy đủ inner join, outer join, cross join, seft join. |
| Lược đồ quan hệ (Schema) | Tường minh (Explicit) | Có |
| Chức năng CRUD | Ban đầu chỉ có Insert, không có Update, Delete. Sau này bổ sung đầy đủ chức năng. | Đầy đủ |
| Hỗ trợ transaction | Có (Không) | Có |
| Hỗ trợ stored procedure | Không | Có |
| Đánh chỉ số Index | Có (Không) | Có |
| Kích thước dữ liệu | Petabyte (1,000,000 Gigabyte) | Terabyte (1,000 Gigabyte) |
| Kích thước mỗi câu truy vấn | Petabyte (1,000,000 Gigabyte) | Gigabyte |
| Sử dụng bởi | Analysts | Programmers |
| Làm việc trên | Server side (Thrift) trên một cluster | Client side trên một cluster |
| User defined function | Java | SQL User-Defined Functions |
| MapReduce | Có | Không |
| Web interface | Có | Có |
| Cấu trúc dữ liệu | Có | Có |
| JDBC/ODBC | Giới hạn | Có (đầy đủ) |
| Trường hợp sử dụng | Dùng Big Data xử lý và phân tích dữ liệu nhanh chóng. Là lựa chọn tốt cho truy vấn, phân tích tập dữ liệu lịch sử | cung cấp khả năng truy vấn, cập nhật và bảo đảm tính toàn vẹn của dữ liệu một cách tối đa, phù hợp cho doanh nghiệp vừa và nhỏ. |

* + - Store Procedure: dù Hive không hỗ trợ viết Store Procedure trực tiếp trên Hive nhưng ta có thể viết ứng dụng ngôn ngữ khác để chạy Store Procedure.
    - Đặc điểm của Hive:
* Không hỗ trợ kiểu dữ liệu data, timestamp và các hàm thao tác ngày giờ.
* Thao tác trên 1 bảng.
* Không dử dụng IN, CONCAT, UNION, INTERSECT.
* Null là null và empty là \n.
* Dù HQL được xem là tương tự SQL nhưng Hive được xem là phần mở rộng của ETL hơn là một kho dữ liệu.

1. **Cài đặt HIVE** 
   * + Để cài đặt HIVE cần phải cài đặt JAVA và HADOOP trước

**8.1. Cài đặt JAVA**

* + - Để kiểm tra máy cài java chưa, có thể mở Terminal và gõ java –version, nếu kết quả trả về phiên bản java thì máy đã cài java, nếu chưa cài thì cần download phiên bản Java JDK về và cài theo các bước sau:
  + Giải nén file java vừa tải về.
  + Chuyển nội dung trong forder vừa giải nén vào forder “/usr/local” trong máy.
  + Tại thư mục “/usr/local” gõ lệnh “sudo getdit ~/.bashrc”, gõ lại password của user và chèn thêm đoạn code sau vào cuối file (jdk1.7.0.71 là tên file java mà bước trước bạn giải nén java vào)
* export JAVA\_HOME=/usr/local/jdk1.7.0\_71
* export PATH=$PATH:$JAVA\_HOME/bin
* PATH=$PATH:$JAVA\_HOME/bin:$JRE\_HOME/bin  
  export JAVA\_HOME  
  export JRE\_HOME  
  export PATH
  + Khai báo các dòng sau
* # alternatives --install /usr/bin/java java usr/local/java/bin/java 2
* # alternatives --install /usr/bin/javac javac usr/local/java/bin/javac 2
* # alternatives --install /usr/bin/jar jar usr/local/java/bin/jar 2
* # alternatives --set java usr/local/java/bin/java
* # alternatives --set javac usr/local/java/bin/javac
* # alternatives --set jar usr/local/java/bin/jar
* Như thế quá trình cài đặt java đã hoàn thành, có thể gõ java –version để kiểm tra lại.

CÀI ĐẶT HADOOP

* + - Trước khi cài đặt Hadoop, trong Terminal gõ sudo apt-get install update để cập nhật những phần mềm mới nhất, như thế sẽ hạn chế những lỗi trong quá trình cài đặt Hadoop.
    - Gõ “hadoop version” để kiểm tra máy đã cài Hadoop chưa, nếu chưa thì download bản Hadoop trên mạng.
    - Các bước cài đặt Hadoop:
  + Tạo group, user mới và add user vào group.
  + Cài đặt openssh, đăng nhập vào user mới tạo và nhập *ssh-keygen –t rsa –P “”* để phát sinh key ssh phát sinh quyền cho user này.
  + Sau đó gõ lệnh cat để ủy quyền



* + Tiếp theo, tương tự như java, giải nén, di chuyển hadoop vừa tải về vào “/usr/local/hadoop” (trao quyền truy cập forder này cho user vừa tạo trên)
  + Chèn thêm code vào file bashrc
* export HADOOP\_HOME=/usr/local/hadoop
* export PATH=$PATH:$HADOOP\_HOME/bin
* export PATH=$PATH:$HADOOP\_HOME/sbin
* export HADOOP\_MAPRED\_HOME=$HADOOP\_HOME
* export HADOOP\_COMMON\_HOME=$HADOOP\_HOME
* export HADOOP\_HDFS\_HOME=$HADOOP\_HOME
* export YARN\_HOME=$HADOOP\_HOME
* export HADOOP\_COMMON\_LIB\_NATIVE\_DIR=$HADOOP\_HOME/lib/native
* export HADOOP\_OPTS="-Djava.library.path=$HADOOP\_HOME/lib
  + Sau đó, chỉnh sửa các file trong /usr/local/hadoop/etc/hadoop các file sau:
    - **hadoop-env.sh**
* export JAVA\_HOME=/usr/local/jdk1.7.0\_71
  + - **core-site.xml**
* <configuration>
* <property>
* <name>fs.default.name</name>
* <value>hdfs://localhost:9000</value>
* </property>
* </configuration>
  + - **hdfs-site.xml**
* <configuration>
* <property>
* <name>dfs.replication</name>
* <value>1</value>
* </property>
* <property>
* <name>dfs.name.dir</name>
* <value>file:///home/hadoop/hadoopinfra/hdfs/namenode </value>
* </property>
* <property>
* <name>dfs.data.dir</name>
* <value>file:///home/hadoop/hadoopinfra/hdfs/datanode </value >
* </property>
* </configuration>
  + - **yarn-site.xml**
* <configuration>
* <property>
* <name>yarn.nodemanager.aux-services</name>
* <value>mapreduce\_shuffle</value>
* </property>
* </configuration>
  + - Dùng lệnh sau để tạo file **mapred-site.xml** tương tự **mapred-site.xml.template**
* cp mapred-site.xml.template mapred-site.xml

Và chèn code sau:

<configuration>

<property>

<name>mapreduce.framework.name</name>

<value>yarn</value>

</property>

</configuration>

* + Tạo forder hadoop\_tmp trong /usr/local và tạo forder hdfs/namenode và hdfs/datanode.
* Quá trình cài đặt thành công, ta gõ hadoop version để kiểm tra.
  1. **Cài đặt HIVE**
     + Tương tự như Hadoop và Java, cần download Hive, Derby và giải nén, sao chép đến thư mục data. Khai báo trong bashrc. Với cài đặt Hive cần chèn code vào **hive-site.xml**:
* <property>
* <name>javax.jdo.option.ConnectionURL</name>
* <value>jdbc:derby://localhost:1527/metastore\_db;create=true </value>
* <description>JDBC connect string for a JDBC metastore </description>
* </property