

**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HCM**  
**KHOA CÔNG NGHỆ THÔNG TIN**



**HCMUTE**

**ĐỒ ÁN MÔN HỌC**  
**ĐIỆN TOÁN ĐÁM MÂY**

**ĐỀ TÀI:**

**TÌM HIỂU VỀ APACHE HIVE VÀ ỨNG DỤNG DEMO**  
**DATAWAREHOUSE**

**GVHD: TS. HUỖNH XUÂN PHỤNG**

**Sinh viên thực hiện:**

**TRẦN NHƯ THUẬN      18133054**

**LƯƠNG UY LONG      18133026**

**Tp. Hồ Chí Minh, tháng 5 năm 2021**

## LỜI CẢM ƠN

Em xin gửi lời cảm ơn sâu sắc tới TS. Huỳnh Xuân Phụng hướng dẫn em trong suốt thời gian thực hiện đề tài. Em xin chân thành cảm ơn Thầy đã truyền đạt cho em những kiến thức quý báu, những bài học giá trị trong những năm học vừa qua, giúp em có một nền tảng lý thuyết vững chắc để phục vụ cho đam mê của em sau này.

Mặc dù đã cố gắng trong suốt quá trình thực tập và làm đồ án, tuy nhiên do còn gặp nhiều khó khăn trong quá trình tiếp cận thực tế, hạn chế về kiến thức và kinh nghiệm chuyên môn nên đồ án không tránh được những sai sót. Vì vậy em rất mong được sự góp ý từ Thầy và các bạn để đồ án tốt nghiệp của em được hoàn chỉnh hơn.

Em xin chân thành cảm ơn!

Sinh viên thực hiện

Trần Như Thuận                      18133054

Lương Uy Long                      18133026

## Mục Lục

<b>LỜI CẢM ƠN</b> .....	2
<b>CHƯƠNG 1: GIỚI THIỆU CHUNG</b> .....	4
1.1. Khái niệm và đặc trưng .....	4
1.2. Kiến trúc Hive .....	4
1.3. Hoạt động của Hive.....	7
1.4. Mô hình dữ liệu của Hive.....	8
1.5. Các kiểu dữ liệu trong Hive .....	9
1.6. Ngôn ngữ truy vấn HiveSQL .....	12
<b>CHƯƠNG 2: ỨNG DỤNG THỰC HIỆN DATA WAREHOUSE</b> .....	13
2.1. Xác định ý tưởng hình thành ý tưởng phân tích của Kho dữ liệu .....	13
2.2. Mô tả hệ thống các DB gốc liên quan phân tích sản xuất và thương mại lâm nghiệp.....	13
2.3. Phân tích các DB gốc xác định yêu cầu phân tích kho dữ liệu của sản xuất và thương mại lâm nghiệp.....	14
2.4. Thiết kế DB mới tổ chức phân tích Kho dữ liệu.....	15
2.5. Lập các Views tính toán cần thiết để nạp dữ liệu từ DB gốc vào các Factors của DWH.....	15
<b>CHƯƠNG 3: ÁP DỤNG HIVE PHÂN TÍCH KHO DỮ LIỆU</b> .....	29
3.1. Nạp kho dữ liệu vào Hive.....	29
3.2. Tạo kho dữ liệu trong hive .....	30
3.3. Truy vấn phân tích trong kho dữ liệu .....	32
<b>CHƯƠNG 4: KẾT LUẬN</b> .....	36
4.1. Vấn đề tồn tại .....	36
4.2. Hướng phát triển.....	36
<b>TÀI LIỆU THAM KHẢO</b> .....	37

# CHƯƠNG 1: GIỚI THIỆU CHUNG

## 1.1. Khái niệm và đặc trưng

- 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.
- Ban đầu Hive được phát triển bởi Facebook, sau đó Quỹ Phần mềm Apache đã lấy và phát triển nó thành một nguồn mở dưới tên Apache Hive. Nó được sử dụng bởi các công ty khác nhau. Ví dụ: Amazon sử dụng nó trong Amazon Elastic MapReduce.

### Hive không phải là:

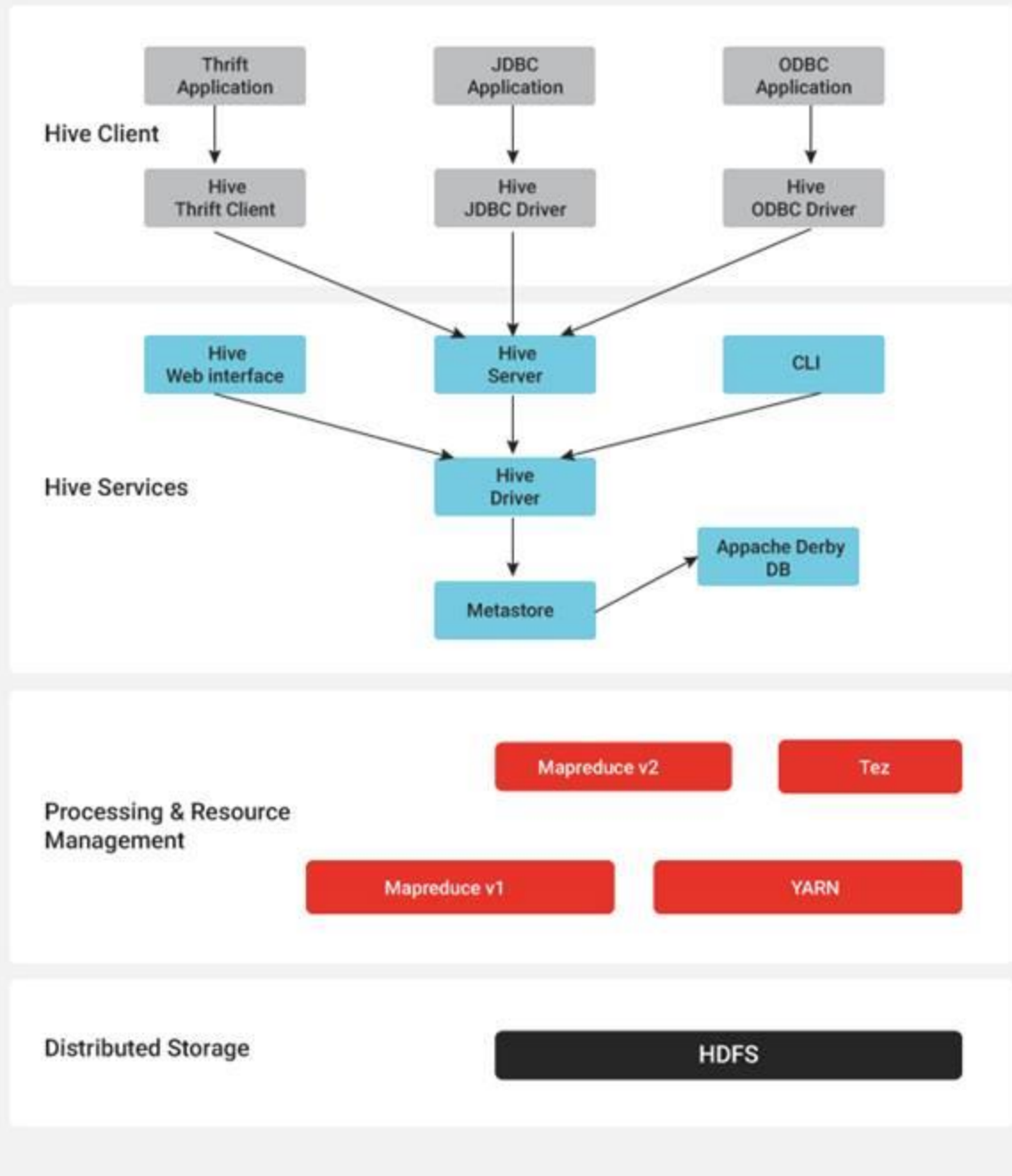
- Một CSDL quan hệ
- Một thiết kế để xử lý giao dịch Online (OnLine Transaction Processing - OLTP)
- Một ngôn ngữ cho các truy vấn thời gian thực và cập nhật cấp hàng

### Đặc trưng của Hive:

- Nó lưu trữ lược đồ trong cơ sở dữ liệu và xử lý dữ liệu vào HDFS.
- Nó được thiết kế cho OLAP.
- Nó cung cấp ngôn ngữ kiểu SQL để truy vấn được gọi là HiveQL hoặc HQL.
- Nó là quen thuộc, nhanh chóng, có khả năng mở rộng.

## 1.2. Kiến trúc Hive

## HIVE ARTCHITECTURE & ITS COMPONENTS



- **Hive Client:**

Hive hỗ trợ các ứng dụng được viết bằng bất kỳ ngôn ngữ nào như Python, Java, C++, Ruby, v.v. sử dụng trình điều khiển JDBC, ODBC và Thrift, để thực hiện các

truy vấn trên Hive. Do đó, người ta có thể dễ dàng viết một ứng dụng Hive client bằng bất kỳ ngôn ngữ nào mà mình lựa chọn.

Hive client được phân loại thành 3 loại.

- Thrift Client: Apache hive server dựa trên thrift để nó có thể phục vụ yêu cầu từ tất cả các ngôn ngữ hỗ trợ thrift.
- JDBC Client: Apache hive cho phép các ứng dụng Java kết nối với nó bằng trình điều khiển JDBC
- ODBC Client: Trình điều khiển ODBC cho phép các ứng dụng hỗ trợ giao thức ODBC kết nối với hive.

- **Hive Services:**

Hive cung cấp các loại service khác nhau như web UI, Command-line Interface (CLI) để thực hiện các truy vấn trên dữ liệu.

1. CLI: giao diện dạng shell cho phép người sử dụng tương tác trực tiếp qua command line.

2. Hive Web Interface: giao diện Web cho phép người sử dụng thực hiện các truy vấn thông qua giao diện Web.

3. Hive Thrift Server: cho phép các client từ nhiều ngôn ngữ lập trình khác nhau có thể thực hiện tương tác với Hive.

4. 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 partition từ metastore để sinh ra các execution plan.
- Execute engine: thành phần thực thi các execution plan được tạo bởi compiler (submit các job tới MapReduce). Ngoài ra thành phần execution engine này thực hiện việc quản lý các dependencies của các bước trong mỗi execution plan, thực thi từng bước này.

5. Hive Metastore: thành phần lưu trữ các metadata của Hive: table, partition, buckets bao gồm cả thông tin về các column trong mỗi table, các serializers và deserializers 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.

- **Processing Framework and Resource Managment**

Nội bộ Hive sử dụng MapReduce framework làm công cụ defacto để thực hiện các truy vấn.

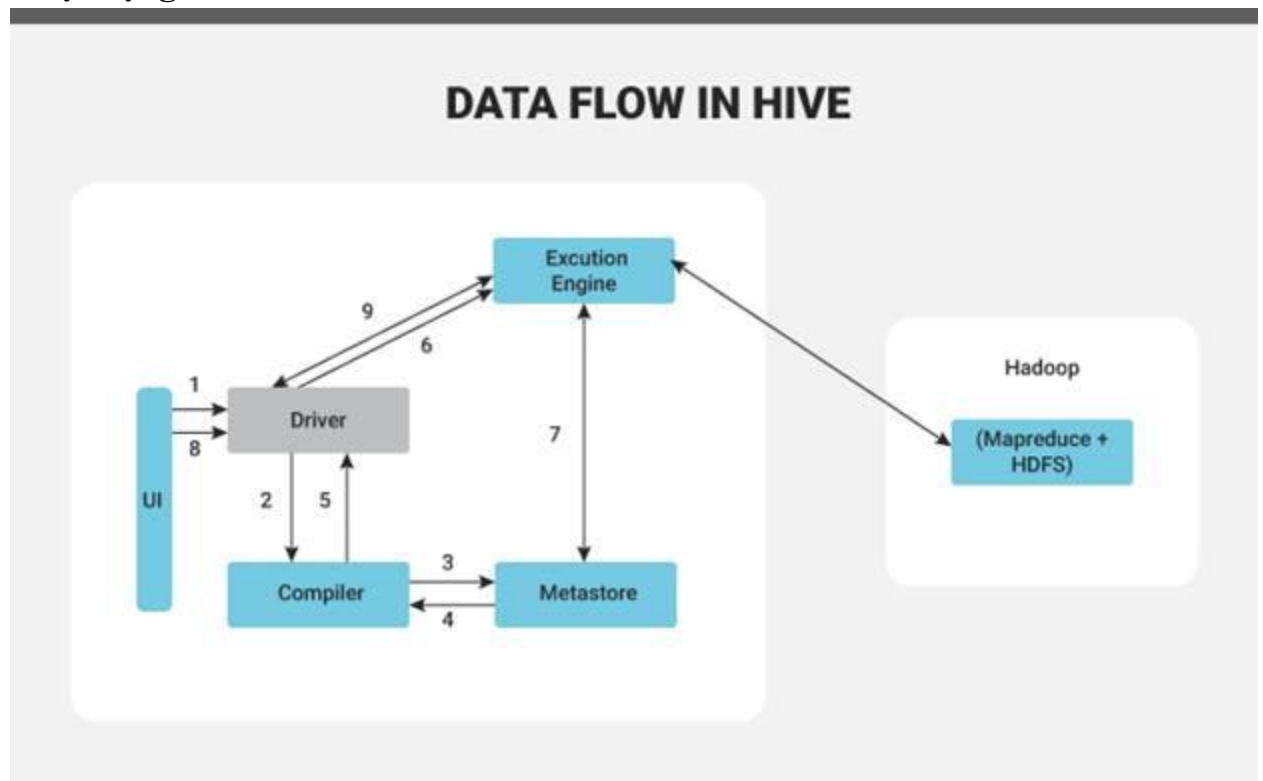
MapReduce là software framework để viết các ứng dụng xử lý một lượng lớn dữ liệu song song trên các cụm phần cứng hàng hóa lớn.

MapReduce hoạt động bằng cách chia nhỏ dữ liệu thành nhiều phần, được xử lý bằng các tác vụ map-reduce.

- **Distributed Storage**

Hive được xây dựng dựa trên Hadoop, vì vậy nó sử dụng Hệ thống tệp phân tán Hadoop cơ bản cho bộ nhớ phân tán.

### 1.3. 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
2. Driver tạo ra mới 1 session cho truy vấn này và gửi query tới compiler để nhận lấy Execution Plan

3. Compiler 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, temporary 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

#### **1.4. Mô hình dữ liệu của Hive**

Hive được biết như là một kho dữ liệu nguồn mở và được xây dựng để có thể phân tích và lưu trữ ngay cả các bộ dữ liệu lớn, được lưu trữ trong các tệp Hadoop. Apache Hive có thể lưu trữ dữ liệu theo ba dạng sau:

- Table
  - Partition
  - Bucket
- Table :

Các bảng Apache Hive tương tự như các bảng trong cơ sở dữ liệu quan hệ. Các bảng của Hive bao gồm dữ liệu và bố cục của chúng được mô tả với sự trợ giúp của metadata liên quan. Có thể được thực hiện các phép toán join, union, ... trên các bảng này.

Thông thường, trong Hadoop, dữ liệu được lưu trữ trong HDFS nhưng Hive lưu trữ metadata trong cơ sở dữ liệu quan hệ thay vì HDFS. Hai loại bảng tồn tại là:

    - Managed tables

Các managed tables của Hive cũng được gọi là các bảng nội bộ và là các bảng mặc định. Nếu không chỉ định loại người dùng bảng này, thì nó sẽ thuộc loại nội bộ.

Tất cả các managed tables được tạo hoặc lưu trữ trong HDFS và dữ liệu của các bảng được tạo hoặc lưu trữ trong thư mục / user / hive / repository của HDFS

Nếu xóa bảng thì cả dữ liệu bảng và metadata sẽ bị xóa khỏi HDFS.
    - External tables



Các external tables được sử dụng cho người dùng bên ngoài khi dữ liệu bảng nằm bên ngoài Hive. Các bảng này thường được sử dụng khi bạn muốn xóa metadata khỏi bảng và giữ dữ liệu bảng như hiện tại. Chỉ có lược đồ bảng bị xóa.

- Partition

Bảng Hive được tổ chức trong các partition (phân vùng) bằng cách nhóm các loại dữ liệu giống nhau dựa trên bất kỳ cột hoặc khóa phân vùng nào. Mỗi bảng có một khóa phân vùng để nhận dạng. Các phân vùng có thể tăng tốc quá trình truy vấn và slicing. Phân vùng làm tăng tốc độ truy vấn bằng cách giảm độ trễ vì nó chỉ quét dữ liệu có liên quan thay vì quét toàn bộ dữ liệu.

- Buckets

Trong Hive, bảng hoặc phân vùng được chia thành các buckets dựa trên hàm băm của một cột trong bảng để cung cấp thêm cấu trúc cho dữ liệu có thể được sử dụng cho các truy vấn hiệu quả hơn.

Ví dụ, bucketing bởi userID có nghĩa là chúng ta cho thực hiện việc tính toán nhanh hơn trên mỗi query của người sử dụng thay vì thực hiện nó trên 1 tập dữ liệu được sắp xếp 1 cách ngẫu nhiên.

Sự khác nhau giữa partition vs bucket:

- Partition thực hiện phân chia dữ liệu trong 1 table theo value của column key, mỗi partition key sẽ có 1 không gian lưu trữ của riêng nó.

Bucketing: thực hiện việc phân phối các key tới từng bucket khác nhau và mỗi partition lại chỉ có 1 key duy nhất.

## 1.5. Các kiểu dữ liệu trong Hive

Kiểu dữ liệu trong Hive chỉ định loại cột / trường trong bảng Hive. Nó xác định loại giá trị có thể được chèn vào cột được chỉ định.

Có hai loại dữ liệu trong Hive là *primitive data type* (kiểu dữ liệu nguyên thủy) và *complex data type* (kiểu dữ liệu phức tạp).

- Primitive data type:

1. Numeric type (kiểu số): bao gồm

- 1.1. Integral data type

**TINYINT:** 1 byte thể hiện cho số nguyên từ -127 đến 128

**SMALLINT:** 2 byte thể hiện cho số nguyên từ -32, 768 đến 32, 767

**INTEGER:** 4 byte thể hiện cho số nguyên từ -2, 147, 483, 648 đến 2, 147, 483, 647

**BIGINT:** 8 byte thể hiện cho số nguyên từ -9, 223, 372, 036, 854, 775, 808 đến 9, 223, 372, 036, 854, 775, 807

- 1.2. Floating data type

**FLOAT:** chiếm 4 byte và có thể có tới 6 con số phần sau dấu thập phân

**DOUBLE:** chiếm 8 bytes và có thể có tới 10 con số phần thập phân

**DOUBLE PRECISION:** là một bí danh cho DOUBLE. Chỉ có sẵn bắt đầu với Hive 2.2.0

**DECIMAL:** được giới thiệu trong Hive 0.11.0. Nó dựa trên Java BigDecimal. Các loại DECIMAL hỗ trợ cả các ký hiệu khoa học và phi khoa học. Trong Hive 0.11.0 và 0.12, độ chính xác của loại DECIMAL được cố định và giới hạn ở 38 chữ số. DECIMAL cung cấp các giá trị chính xác hơn và phạm vi lớn hơn DOUBLE.

**NUMERIC:** bắt đầu với Hive 3.0.0. Kiểu dữ liệu NUMERIC giống như kiểu DECIMAL.

## 2. Date/Time data type:

**TIMESTAMP:** được giới thiệu trong Hive 0.8.0. Nó hỗ trợ timestamp UNIX truyền thống với độ chính xác nano giây tùy chọn. Định dạng timestamp được hỗ trợ là yyyy-mm-dd hh:mm:ss[.f...] trong các tập tin văn bản.

**DATE:** được giới thiệu trong Hive 0.12.0. Giá trị DATE mô tả một năm / tháng / ngày cụ thể dưới dạng YYYY-MM-DD. Nó không có thành phần thời gian trong DATE. Phạm vi giá trị được hỗ trợ cho loại DATE là 0000-01-01 đến 9999-12-31.

**INTERVAL:** Các kiểu dữ liệu Hive Interval chỉ khả dụng sau khi bắt đầu với Hive phiên bản 1.2 trở lên. Hive chấp nhận cú pháp khoảng với thông số kỹ thuật đơn vị. Chúng ta phải xác định các đơn vị cùng với giá trị khoảng. Ví dụ: INTERVAL '1 DAY' để cập đến thời gian trong ngày.

## 3. String data type:

**STRING:** Trong Hive, chuỗi ký tự chuỗi được biểu thị bằng dấu ngoặc đơn ( ' ') hoặc bằng dấu ngoặc kép ( " ").

**VARCHAR:** Trong Hive, các kiểu dữ liệu VARCHAR có độ dài khác nhau, nhưng chúng ta phải chỉ định số lượng ký tự tối đa được phép trong chuỗi ký tự. Nếu giá trị chuỗi được gán cho varchar nhỏ hơn độ dài tối đa, thì khoảng trống còn lại sẽ được giải phóng. Ngoài ra, nếu giá trị chuỗi được gán lớn hơn độ dài tối đa, thì chuỗi sẽ tự bị cắt. Độ dài của varchar nằm trong khoảng (1 đến 65535). Whitespace rất quan trọng trong VARCHAR và sẽ ảnh hưởng đến kết quả so sánh.

**CHAR:** Các kiểu dữ liệu CHAR có độ dài cố định. Các giá trị ngắn hơn chiều dài đã chỉ định được đệm bằng khoảng trắng. Không giống như

VARCHAR, khoảng trắng không đáng kể trong các loại CHAR trong khi so sánh. Độ dài tối đa của CHAR được cố định ở 255.

#### 4. Miscellaneous data type:

**BOOLEAN:** Các loại Boolean trong Hive lưu trữ đúng hoặc sai.

**BINARY:** Kiểu BINary trong Hive là một mảng byte.

- Complex data type:

##### 1. Arrays:

Mảng trong Hive là một chuỗi có thứ tự gồm các phần tử loại tương tự có thể lập chỉ mục bằng cách sử dụng các số nguyên dựa trên zero.

Mảng trong Hive tương tự như các mảng trong JAVA.

Cú pháp: `array<datatype>`

Ví dụ: `array('Data','Flair')`. Phần tử thứ hai được truy cập theo `array[1]`.

##### 2. Maps:

Map trong Hive là một tập hợp các cặp key-value, trong đó các trường được truy cập bằng cách sử dụng các ký hiệu mảng của các khóa.

Cú pháp: `map<primitive_type, data_type>`

Ví dụ: 'first' -> 'John', 'last' -> 'Deo', biểu diễn dưới dạng code `map('first', 'John', 'last', 'Deo')`. Bây giờ 'John' có thể được truy cập với `map['first']`

##### 3. Structs:

STRUCT trong Hive tương tự như STRUCT trong ngôn ngữ C.

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

Cú pháp:

`STRUCT <col_name : data_type [ COMMENT col_comment], ...>`

Ví dụ: cho cột c3 thuộc kiểu STRUCT {c1 INTEGER; c2 INTEGER}, trường c1 được truy cập bởi biểu thức `c3.c1`.

##### 4. Union:

Kiểu UNION trong Hive tương tự như UNION trong C.

Các loại UNION tại bất kỳ thời điểm nào cũng có thể chứa chính xác một loại dữ liệu từ các loại dữ liệu được chỉ định.

Kiểu dữ liệu UNIONTYPE trong Hive vẫn chưa đầy đủ.

Cú pháp: `UNIONTYPE<data_type, data_type, ...>`

- NULL value:

Trong các kiểu dữ liệu Hive, các giá trị bị thiếu được biểu thị bằng giá trị đặc biệt NULL.

## 1.6. Ngôn ngữ truy vấn HiveSQL

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;
```

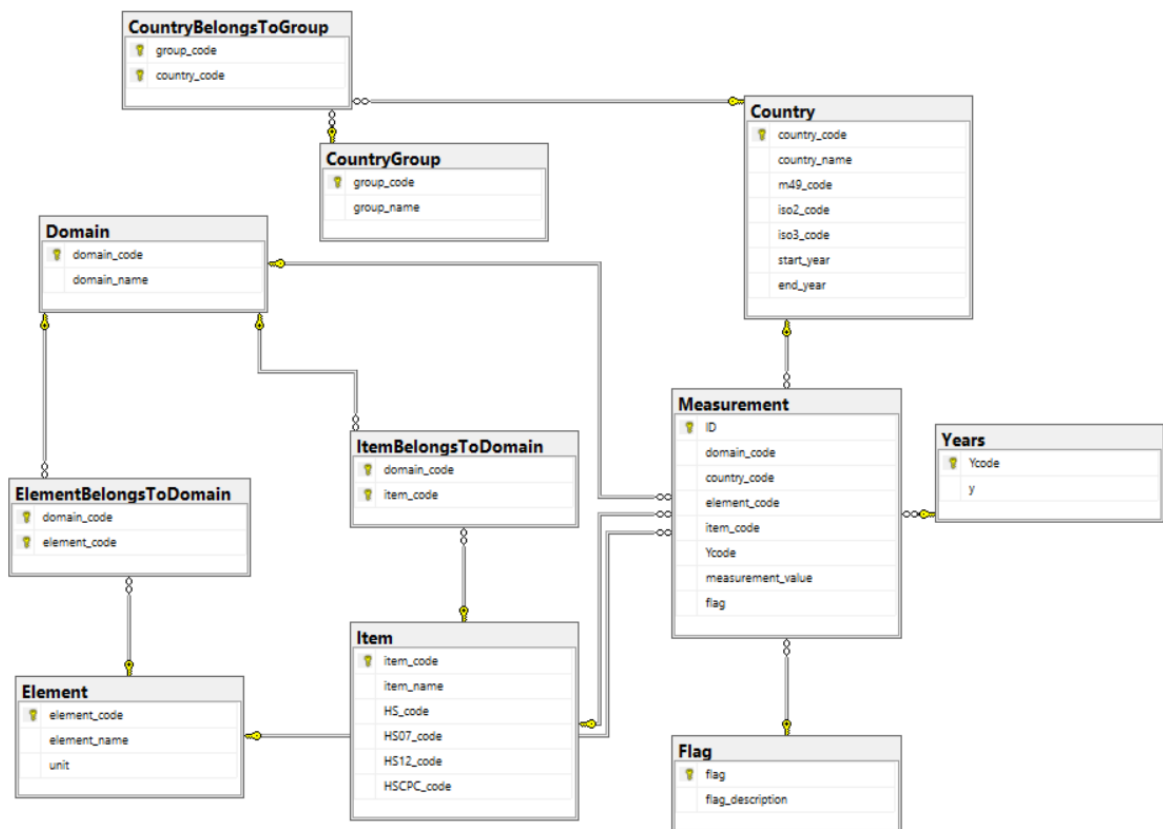
Nó tương tự như SQL.

## CHƯƠNG 2: ỨNG DỤNG THỰC HIỆN DATA WAREHOUSE

### 2.1. Xác định ý tưởng hình thành ý tưởng phân tích của Kho dữ liệu

- Tập dữ liệu lương thực và nông nghiệp toàn cầu khai thác sử dụng tại các cơ quan quản lý và kinh doanh lương thực và nông nghiệp.
- Ý tưởng dựa trên tập data cụ thể Forestry Production and Trade (2000-2019) bao gồm phân tích tổng sản lượng sản phẩm (giấy tờ gia dụng vệ sinh, giấy báo in, thùng bìa carton, tập giấy tờ, các loại giấy khác chủ yếu dùng đóng gói) trên mỗi quốc gia, tổng value của nhập khẩu cũng như xuất khẩu trên mỗi quốc gia.

### 2.2. Mô tả hệ thống các DB gốc liên quan phân tích sản xuất và thương mại lâm nghiệp.



#### Mô tả các quan hệ

+ Domain: lĩnh vực được thống kê

- + Item: các ngành, sản phẩm thuộc lĩnh vực được thống kê.
- + Element: các độ đo được dùng để đo các thông số.
- + Country: các quốc gia vùng lãnh thổ hoặc các vùng quốc gia, lục địa, tổ chức,...
- + CountryGroup: các vùng quốc gia, châu lục hoặc các tổ chức.
- + Years: các năm được thống kê.
- + Flag: cờ chỉ nguồn dữ liệu, ghi chú cho dữ liệu.
- + Measurement: các chỉ số được thống kê.
- + CountryBelongsToGroup: thể hiện mối quan hệ M-N của Country và CountryGroup.
- + ItemBelongsToDomain: thể hiện mối quan hệ M-N của Item và Domain.
- + ElementBelongsToDomain: thể hiện mối quan hệ M-N của Element và Domain.

### **2.3. Phân tích các DB gốc xác định yêu cầu phân tích kho dữ liệu của sản xuất và thương mại lâm nghiệp.**

#### **Xác định các Facts và Dims**

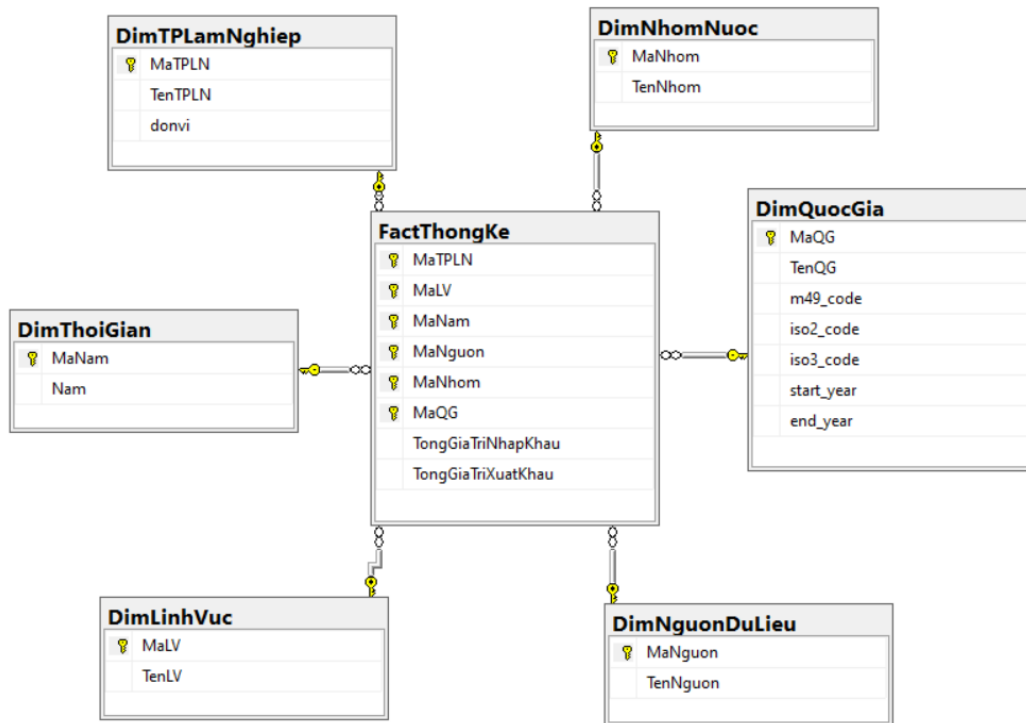
- **Facts:** + tổng giá trị nhập khẩu (số lượng nhập khẩu \* giá trị nhập khẩu)  
                   + tổng giá trị xuất khẩu (số lượng xuất khẩu \* giá trị xuất khẩu)
- **Dims:**
  - + Thành phần lâm nghiệp (DimTPLamNghiep): sản lượng sản xuất, số lượng giá trị xuất nhập khẩu
  - + Lĩnh vực (DimLinhVuc): sản xuất và thương mại lâm nghiệp (FO)
  - + Thời gian (DimThoiGian): từ 2000 đến nay.
  - + Quốc gia (DimQuocGia): các quốc gia trên thế giới.

+ Các nhóm nước (DimNhomNuoc): Châu Á, Châu Mỹ, Châu Âu, nước đang phát triển/phát triển,...

+ Các nguồn dữ liệu (DimNguonDuLieu): chính thức, không chính thức, đang tính toán,...

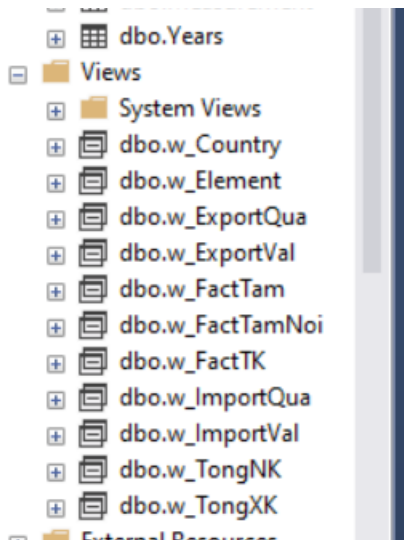
## 2.4. Thiết kế DB mới tổ chức phân tích Kho dữ liệu

Kho dữ liệu phân hệ cá nhân em có 6 dim (DimTPLamNghiep, DimNhomNuoc, DimThoiGian, DimQuocGia, DimLinhVuc, DimNguonDuLieu) và 1 Fact (FactThongKe) tính tổng value xuất nhập khẩu



## 2.5. Lập các Views tính toán cần thiết để nạp dữ liệu từ DB gốc vào các Factors của DWH

Lập các view tính toán (gồm 11 view)

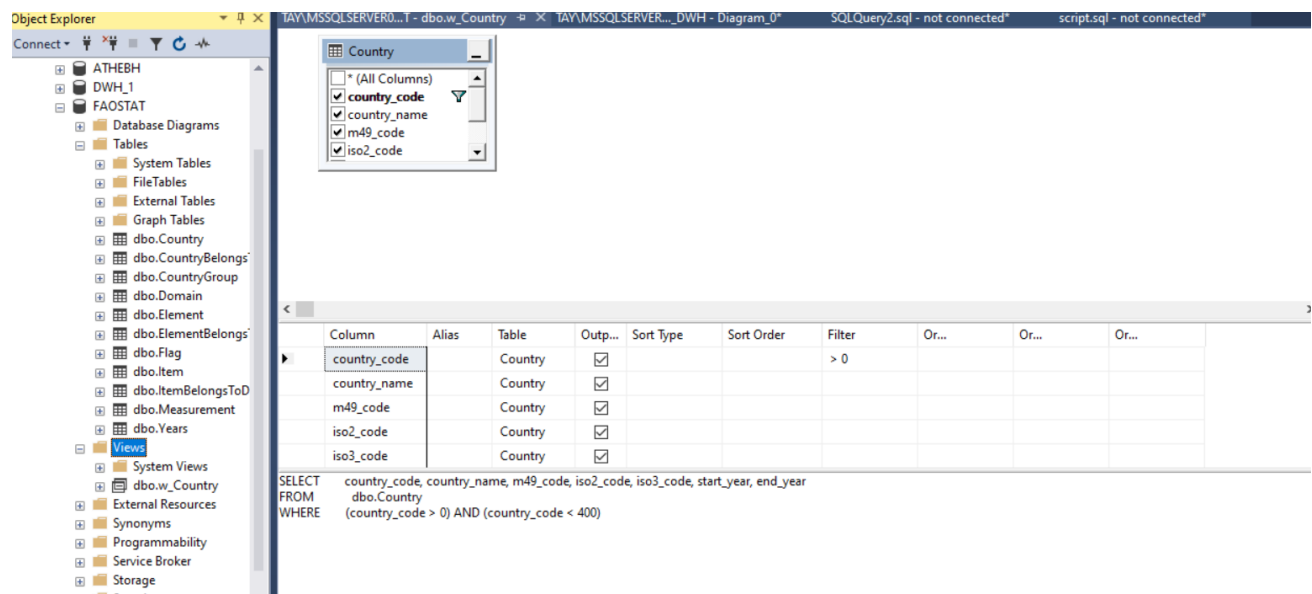


**W\_Country: lấy các nước và loại bỏ khu vực + dùng table Country**

SELECT country\_code, country\_name, m49\_code, iso2\_code, iso3\_code,  
start\_year, end\_year

FROM dbo.Country

WHERE (country\_code > 0) AND (country\_code < 400)



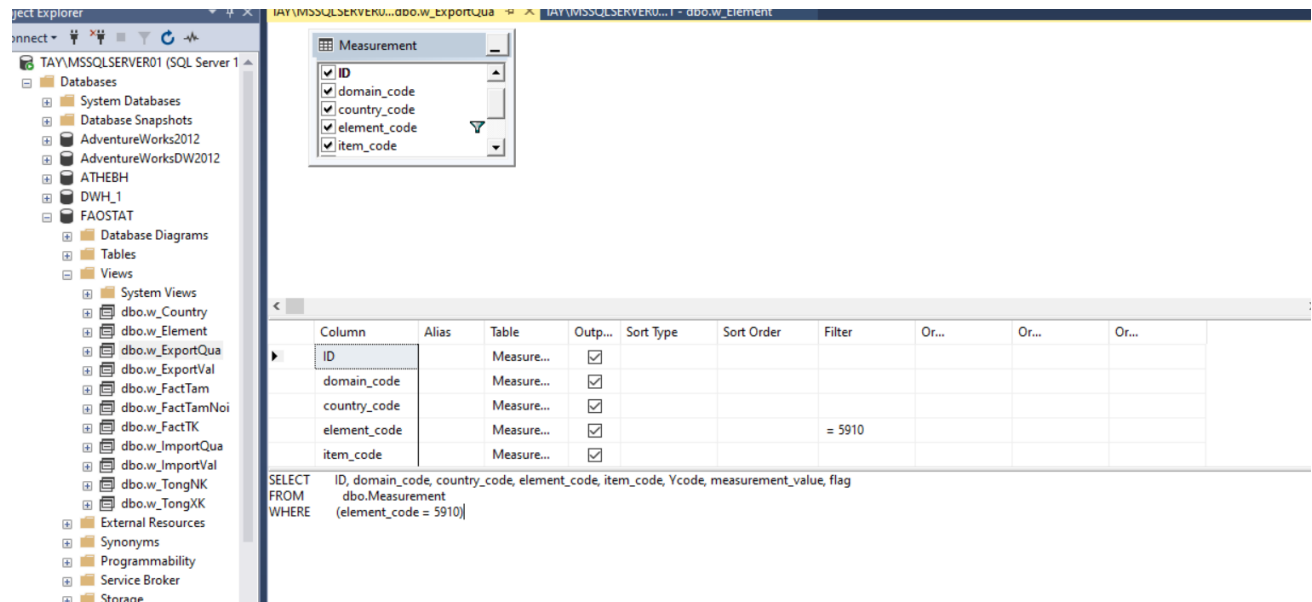
**W\_ExportQua: lấy các giá trị như bên dưới mà element\_code là số lượng xuất khẩu + dùng table Measurement**



```
SELECT      ID, domain_code, country_code, element_code, item_code, Ycode,
measurement_value, flag
```

```
FROM        dbo.Measurement
```

```
WHERE       (element_code = 5910)
```

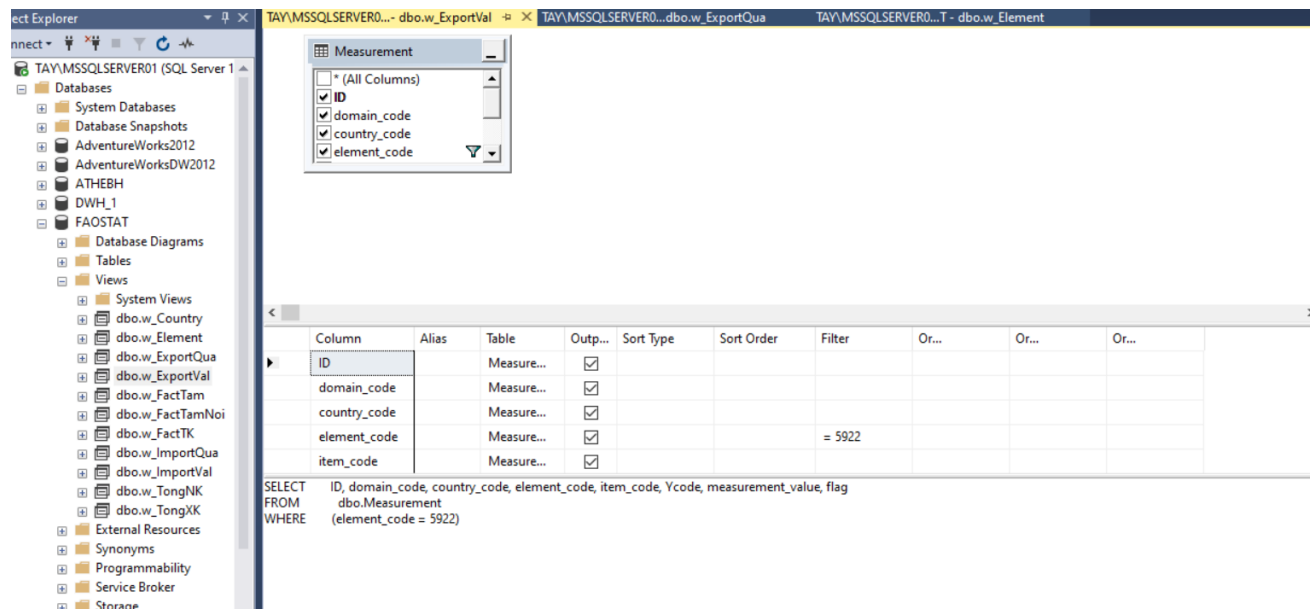


**W\_ExportVal: lấy các giá trị như bên dưới mà element\_code là giá trị xuất khẩu + dùng table Measurement**

```
SELECT      ID, domain_code, country_code, element_code, item_code, Ycode,
measurement_value, flag
```

```
FROM        dbo.Measurement
```

```
WHERE       (element_code = 5922)
```



Measurement

Columns: ID, domain\_code, country\_code, element\_code, item\_code

Filter: element\_code = 5922

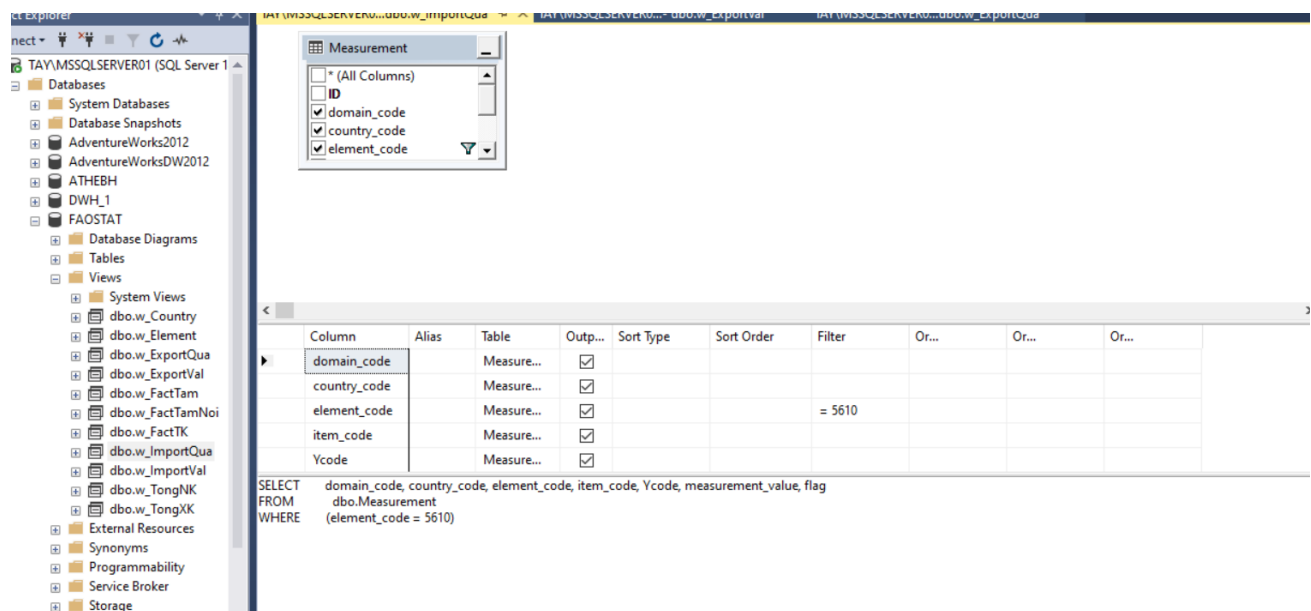
```
SELECT ID, domain_code, country_code, element_code, item_code, Ycode, measurement_value, flag
FROM dbo.Measurement
WHERE (element_code = 5922)
```

**W\_ImportQua: lấy các giá trị như bên dưới mà element\_code là số lượng nhập khẩu + dùng table Measurement**

SELECT domain\_code, country\_code, element\_code, item\_code, Ycode, measurement\_value, flag

FROM dbo.Measurement

WHERE (element\_code = 5610)



Measurement

Columns: domain\_code, country\_code, element\_code, item\_code, Ycode

Filter: element\_code = 5610

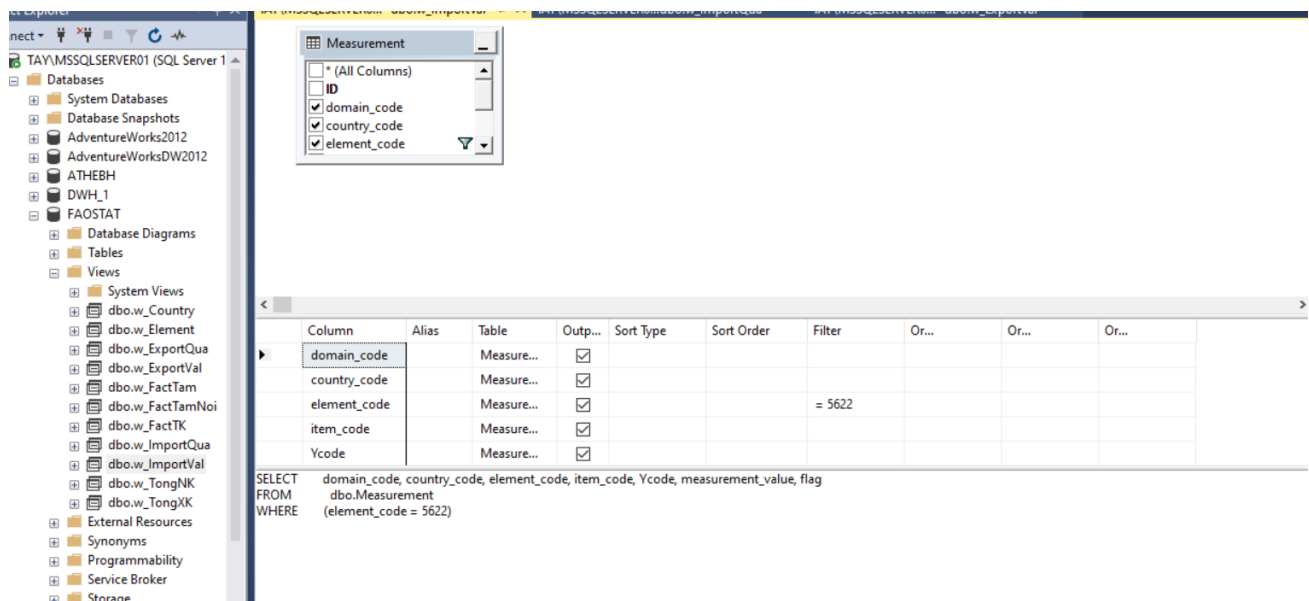
```
SELECT domain_code, country_code, element_code, item_code, Ycode, measurement_value, flag
FROM dbo.Measurement
WHERE (element_code = 5610)
```

**W\_ImportVal:** lấy các giá trị như bên dưới mà element\_code là giá trị nhập khẩu + dùng table Measurement

```
SELECT      domain_code, country_code, element_code, item_code, Ycode,
measurement_value, flag
```

```
FROM        dbo.Measurement
```

```
WHERE       (element_code = 5622)
```



**W\_TongNK:** tính tổng value nhập khẩu (số lượng \* giá trị theo nhập khẩu) + dùng 2 view ImportQua, ImportVal

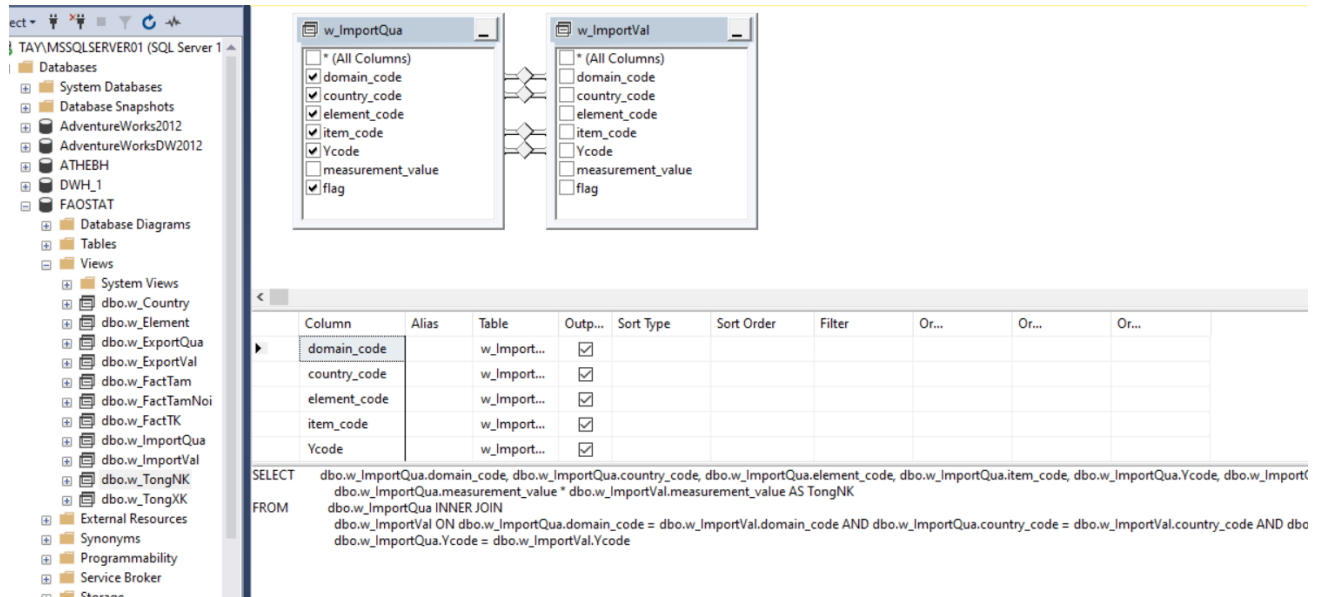
```
SELECT      dbo.w_ImportQua.domain_code, dbo.w_ImportQua.country_code,
dbo.w_ImportQua.element_code, dbo.w_ImportQua.item_code,
dbo.w_ImportQua.Ycode, dbo.w_ImportQua.flag,
```

```
            dbo.w_ImportQua.measurement_value *
dbo.w_ImportVal.measurement_value AS TongNK
```

```
FROM        dbo.w_ImportQua INNER JOIN
```

`dbo.w_ImportVal ON dbo.w_ImportQua.domain_code =  
 dbo.w_ImportVal.domain_code AND dbo.w_ImportQua.country_code =  
 dbo.w_ImportVal.country_code AND dbo.w_ImportQua.item_code =  
 dbo.w_ImportVal.item_code AND`

`dbo.w_ImportQua.Ycode = dbo.w_ImportVal.Ycode`



**W\_TongXK: tính tổng value xuất khẩu (số lượng \* giá trị theo xuất khẩu) + dùng 2 view ExportQua, ExportVal**

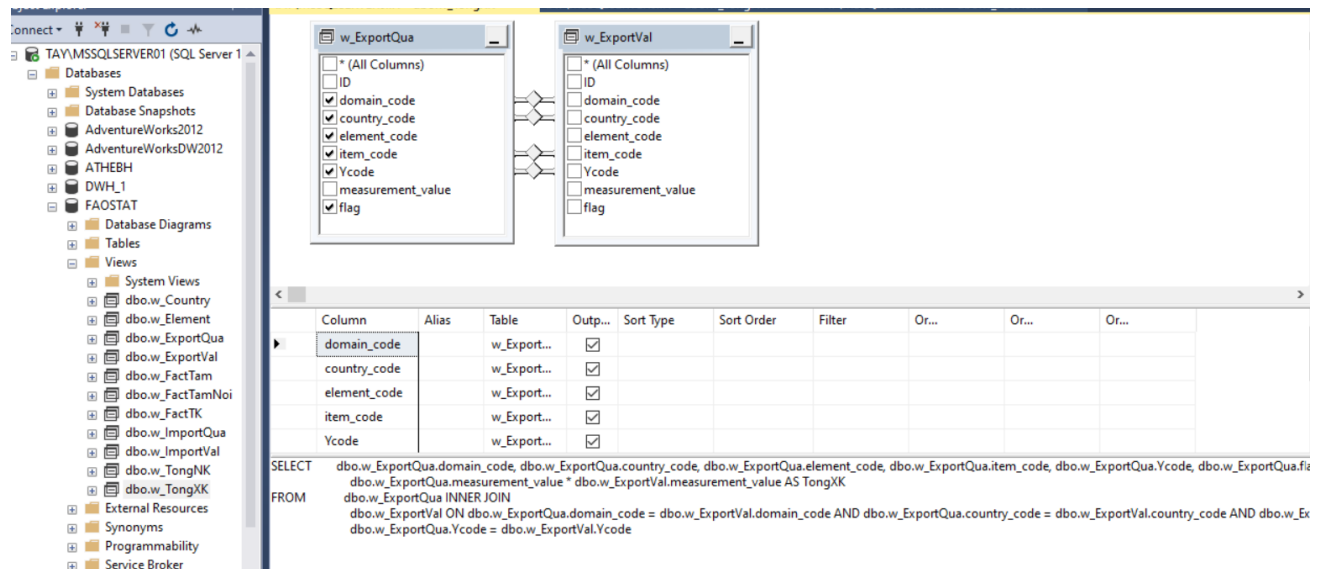
`SELECT dbo.w_ExportQua.domain_code, dbo.w_ExportQua.country_code,  
 dbo.w_ExportQua.element_code, dbo.w_ExportQua.item_code,  
 dbo.w_ExportQua.Ycode, dbo.w_ExportQua.flag,`

`dbo.w_ExportQua.measurement_value *  
 dbo.w_ExportVal.measurement_value AS TongXK`

`FROM dbo.w_ExportQua INNER JOIN`

`dbo.w_ExportVal ON dbo.w_ExportQua.domain_code =  
 dbo.w_ExportVal.domain_code AND dbo.w_ExportQua.country_code =  
 dbo.w_ExportVal.country_code AND dbo.w_ExportQua.item_code =  
 dbo.w_ExportVal.item_code AND`

`dbo.w_ExportQua.Ycode = dbo.w_ExportVal.Ycode`



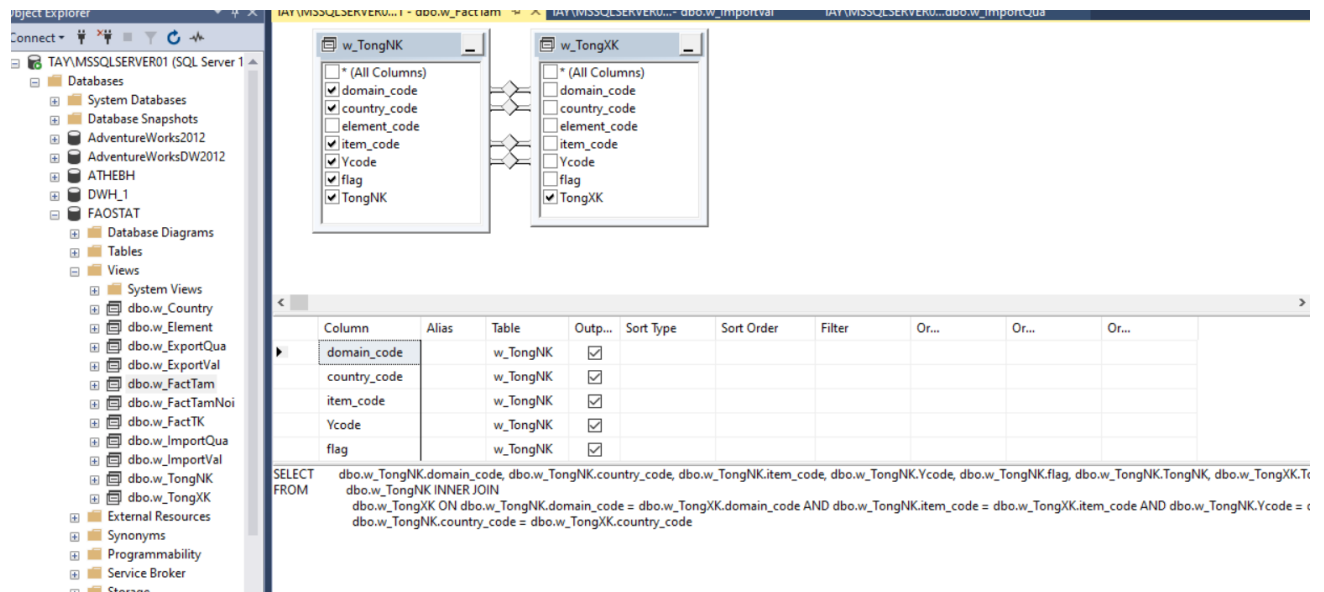
**W\_FactTam: lấy tất cả thuộc tính cần sử dụng import và FactThongKe + dùng 2 view TongNK, TongXK**

```
SELECT      dbo.w_TongNK.domain_code, dbo.w_TongNK.country_code,
            dbo.w_TongNK.item_code, dbo.w_TongNK.Ycode, dbo.w_TongNK.flag,
            dbo.w_TongNK.TongNK, dbo.w_TongXK.TongXK
```

```
FROM        dbo.w_TongNK INNER JOIN
```

```
            dbo.w_TongXK ON dbo.w_TongNK.domain_code =
            dbo.w_TongXK.domain_code AND dbo.w_TongNK.item_code =
            dbo.w_TongXK.item_code AND dbo.w_TongNK.Ycode =
            dbo.w_TongXK.Ycode AND
```

```
            dbo.w_TongNK.country_code = dbo.w_TongXK.country_code
```



**W\_TamNoi: Lấy nối thêm thuộc tính nhóm quốc gia (group\_code), lĩnh vực (element\_code) + dùng 1 view w\_FactTam và 2 table CountryBelongToGroup, Element**

```
SELECT
    dbo.Element.element_code, dbo.w_FactTam.domain_code,
    dbo.w_FactTam.Ycode, dbo.w_FactTam.flag,
    dbo.CountryBelongsToGroup.group_code, dbo.w_FactTam.country_code,
    dbo.w_FactTam.TongNK,
```

```
        dbo.w_FactTam.TongXK
```

```
FROM
    dbo.w_FactTam INNER JOIN
```

```
        dbo.CountryBelongsToGroup ON dbo.w_FactTam.country_code =
    dbo.CountryBelongsToGroup.country_code CROSS JOIN
```

```
        dbo.Element
```

```
WHERE
    (dbo.Element.element_code = 5510)
```

Database Structure:

- w\_FactTam: \* (All Columns), domain\_code, country\_code, item\_code, Ycode, flag, TongNK, TongXK
- CountryBelongsTo: \* (All Columns), group\_code, country\_code
- Element: \* (All Columns), element\_code, element\_name, unit

Query:

```
SELECT
  dbo.Element.element_code, dbo.w_FactTam.domain_code, dbo.w_FactTam.Ycode, dbo.w_FactTam.flag, dbo.CountryBelongsTo.group_code, dbo.w_FactTam.country_code
FROM
  dbo.w_FactTam INNER JOIN
  dbo.CountryBelongsTo ON dbo.w_FactTam.country_code = dbo.CountryBelongsTo.country_code CROSS JOIN
  dbo.Element
WHERE
  (dbo.Element.element_code = 5510)
```

**W\_FactTK: dùng SUM() tính tổng value xuất nhập khẩu theo các quốc gia trên thế giới + dùng view w\_FactTamNoi**

SELECT element\_code, domain\_code, Ycode, flag, group\_code, country\_code, SUM(TongNK) AS TongValueNK, SUM(TongXK) AS TongValueXK

FROM dbo.w\_FactTamNoi

GROUP BY element\_code, domain\_code, Ycode, flag, group\_code, country\_code

Database Structure:

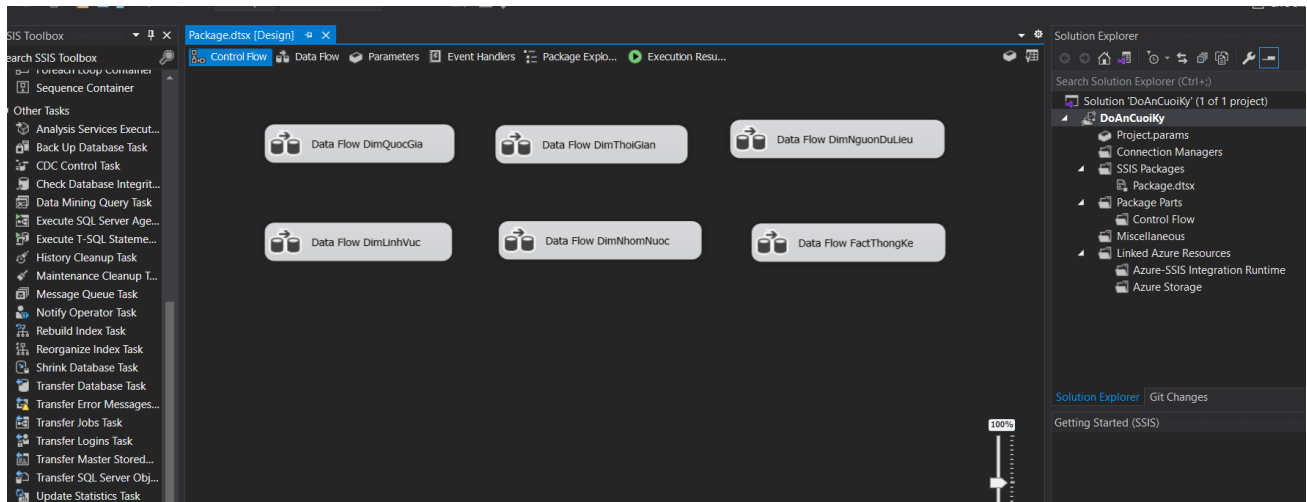
- w\_FactTamNoi: \* (All Columns), element\_code, domain\_code, Ycode, flag, group\_code, country\_code, TongNK, TongXK
- w\_FactTam: \* (All Columns), domain\_code, country\_code, item\_code, Ycode, flag, TongNK, TongXK
- CountryBelongsTo: \* (All Columns), group\_code, country\_code
- Element: \* (All Columns), element\_code, element\_name, unit

Query:

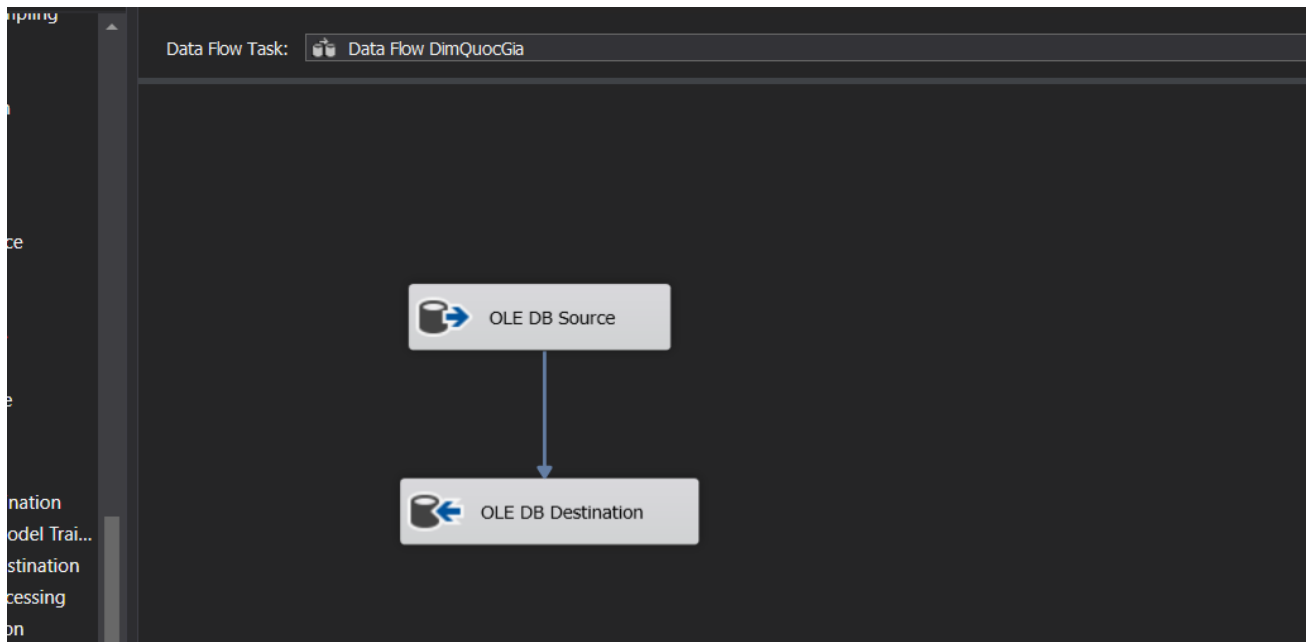
```
SELECT
  element_code, domain_code, Ycode, flag, group_code, country_code, SUM(TongNK) AS TongValueNK, SUM(TongXK) AS TongValueXK
FROM
  dbo.w_FactTamNoi
GROUP BY
  element_code, domain_code, Ycode, flag, group_code, country_code
```

## Nạp dữ liệu vào DataWareHouse

Để nạp dữ liệu vào DataWarehouse ta sử dụng Integration service → tạo các data flow như hình



Ví dụ đối với Data Flow DimQuocGia, ta tạo các OLE DB Source và OLE DB Destination như bên dưới



Thực hiện thiết lập các connection, name table view cần thiết import vào DWH



OLE DB Source Editor

Configure the properties used by a data flow to obtain data from any OLE DB provider.

**Connection Manager**  
Columns  
Error Output

Specify an OLE DB connection manager, a data source, or a data source view, and select the data access mode. If using the SQL command access mode, specify the SQL command either by typing the query or by using Query Builder.

OLE DB connection manager:  
TAY\MSSQLSERVER01.FAOSTAT.sa New...

Data access mode:  
Table or view

Name of the table or the view:  
[dbo].[w\_Country]

OLE DB Destination Editor

Configure the properties used to insert data into a relational database using an OLE DB provider.

**Connection Manager**  
Mappings  
Error Output

Specify an OLE DB connection manager, a data source, or a data source view, and select the data access mode. If using the SQL command access mode, specify the SQL command either by typing the query or by using Query Builder. For fast-load data access, set the table update options.

OLE DB connection manager:  
TAY\MSSQLSERVER01.FAOSTAT\_DWH.sa New...

Data access mode:  
Table or view - fast load

Name of the table or the view:  
[dbo].[DimQuocGia] New...

☐ Keep identity ☒ Table lock  
☐ Keep nulls ☒ Check constraints

Rows per batch:

Thực hiện mapping các thuộc tính từ DB gốc tới DWH

Input Column	Destination Column
country_code	MaQG
country_name	TenQG
m49_code	m49_code
iso2_code	iso2_code
iso3_code	iso3_code
start_year	start_year
end_year	end_year

Sau đó ta thực hiện run → kiểm tra lại sql server, thành công nếu dữ liệu được nạp vào

Ví dụ như bên dưới DimQuocGia đã nạp dữ liệu thành công.

**DimQuocGia**

Object Explorer

Connect

Database Diagrams

Tables

Views

System Views

dbo.w\_Country

External Resources

Synonyms

Programmability

Service Broker

Storage

Security

FAOSTAT\_COPY

FAOSTAT\_DWH

Database Diagrams

Tables

System Tables

FileTables

External Tables

Graph Tables

dbo.DimLinhVuc

dbo.DimNguonDuLie

dbo.DimNhomNuoc

dbo.DimQuocGia

dbo.DimThoiGian

dbo.DimTPLamNghie

dbo.FactThongKe

Views

External Resources

Synonyms

Programmability

Service Broker

Storage

Security

SQLQuery1.sql - [A...:STAT\_DWH (sa (33))]

```

/***** Script for SelectTopNRows command from SSMS *****/
SELECT TOP (1000) [MaQG]
, [TenQG]
, [m49_code]
, [iso2_code]
, [iso3_code]
, [start_year]
, [end_year]
FROM [FAOSTAT_DWH].[dbo].[DimQuocGia]

```

100 %

Results Messages

	MaQG	TenQG	m49_code	iso2_code	iso3_code	start_year	end_year
1	1	Amenia	51	AM	ARM	1992	NULL
2	2	Afghanistan	4	AF	AFG	NULL	NULL
3	3	Albania	8	AL	ALB	NULL	NULL
4	4	Algeria	12	DZ	DZA	NULL	NULL
5	5	American Samoa	16	AS	ASM	NULL	NULL
6	6	Andorra	20	AD	AND	NULL	NULL
7	7	Angola	24	AO	AGO	NULL	NULL
8	8	Antigua and Barbuda	28	AG	ATG	NULL	NULL
9	9	Argentina	32	AR	ARG	NULL	NULL
10	10	Australia	36	AU	AUS	NULL	NULL
11	11	Austria	40	AT	AUT	NULL	NULL
12	12	Bahamas	44	BS	BHS	NULL	NULL
13	13	Bahrain	48	BH	BHR	NULL	NULL

Tương tự với các Dim và Fact còn lại

## DimThoiGian

TAYMSSQLSERVER01 (SQL Server 1

Databases

System Databases

Database Snapshots

AdventureWorks2012

AdventureWorksDW2012

ATHEBH

DWH\_1

FAOSTAT

FAOSTAT\_COPY

FAOSTAT\_DWH

Database Diagrams

Tables

System Tables

FileTables

External Tables

Graph Tables

dbo.DimLinhVuc

dbo.DimNguonDuLie

dbo.DimNhomNuoc

dbo.DimQuocGia

dbo.DimThoiGian

dbo.DimTPLamNghie

dbo.FactThongKe

Views

External Resources

Synonyms

Programmability

Service Broker

Storage

Security

myfirst\_copy

QLKCBHYT\_DWH

SELECT TOP (1000) [MaNam]

, [Nam]

FROM [FAOSTAT\_DWH].[dbo].[DimThoiGian]

100 %

Results Messages

	MaNam	Nam
1	1961	1961
2	1962	1962
3	1963	1963
4	1964	1964
5	1965	1965
6	1966	1966
7	1967	1967
8	1968	1968
9	1969	1969
10	1970	1970
11	1971	1971
12	1972	1972
13	1973	1973

## DimLinhVuc

SQL Explorer - TAY\MSSQLSERVER01 (SQL Server 1)

SQLQuery4.sql - TAY\STAT\_DWH (sa (61)) - TAY\MSSQLSERVER...\_DWH - Diagram\_0\*

```

/***** Script for SelectTopNRows command from SSMS *****/
SELECT TOP (1000) [MaLV]
, [TenLV]
FROM [FAOSTAT_DWH].[dbo].[DimLinhVuc]

```

100 %

	MaLV	TenLV
1	AE	ASTI R&D Indicators: ASTI-Expenditures
2	AF	ASTI R&D Indicators: ASTI-Researchers
3	BC	Food Balance: Commodity Balances - Crops Primary ...
4	BL	Food Balance: Commodity Balances - Livestock and...
5	CC	Food Balance: Food Supply - Crops Primary Equival...
6	CISP	Investment: Country Investment Statistics Profile
7	CL	Food Balance: Food Supply - Livestock and Fish Ph...
8	CP	Prices: Consumer Price Indices
9	CS	Macro-Statistics: Capital Stock
10	EA	Investment: Development Flows to Agriculture
11	EF	Agri-Environmental Indicators: Fertilizers indicators
12	EI	Agri-Environmental Indicators: Emissions intensities
13	EK	Agri-Environmental Indicators: Livestock Patterns

## DimNhomNuoc

SQL Explorer - TAY\MSSQLSERVER01 (SQL Server 1)

SQLQuery4.sql - TAY\STAT\_DWH (sa (61)) - TAY\MSSQLSERVER...\_DWH - Diagram\_0\*

```

/***** Script for SelectTopNRows command from SSMS *****/
SELECT TOP (1000) [MaNhom]
, [TenNhom]
FROM [FAOSTAT_DWH].[dbo].[DimNhomNuoc]

```

100 %

	MaNhom	TenNhom
1	420	Sub-Saharan Africa
2	429	North Africa (excluding Sudan)
3	5000	World
4	5100	Africa
5	5101	Eastern Africa
6	5102	Middle Africa
7	5103	Northern Africa
8	5104	Southern Africa
9	5105	Western Africa
10	5200	Americas
11	5203	Northern America
12	5204	Central America

## DimNguonDuLieu

Script for SelectTopNRows command from SSMS

```

SELECT TOP (1000) [MaNguon]
, [TenNguon]
FROM [FAOSTAT_DWH].[dbo].[DimNguonDuLieu]

```

	MaNguon	TenNguon
1		Official data
2	*	Unofficial figure
3	A	Aggregate, may include official, semi-official, esti...
4	Bk	Break in series
5	C	Calculated
6	Ce	Calculated data based on estimated data
7	Cv	Calculated through value
8	E	Expert sources from FAO (including other divisio...
9	F	FAO estimate
10	Fa	Other analytical methods
11	Fb	Data obtained as a balance
12	Fc	Calculated data
13	Fd	estimates - discontinued time series

## FactThongKe

Script for SelectTopNRows command from SSMS

```

SELECT TOP (1000) [MaTPLN]
, [MaLV]
, [MaNam]
, [MaNguon]
, [MaNhom]
, [MaQG]
, [TongGiaTriNhapKhu]
, [TongGiaTriXuatKhu]
FROM [FAOSTAT_DWH].[dbo].[FactThongKe]

```

	MaTPLN	MaLV	MaNam	MaNguon	MaNhom	MaQG	TongGiaTriNhapKhu	TongGiaTriXuatKhu
1	5510	FO	2000		420	107	1418400.00	0.00
2	5510	FO	2000		420	114	61460000.00	0.00
3	5510	FO	2000		420	137	26955500.00	2551100.00
4	5510	FO	2000		420	159	151200000.00	1048000.00
5	5510	FO	2000		420	202	6655779055.00	13050985584.00
6	5510	FO	2000		420	217	7800.00	0.00
7	5510	FO	2000		429	59	1934000.00	19440000.00
8	5510	FO	2000		429	143	1620714000.00	30272000.00
9	5510	FO	2000		429	222	83940000.00	0.00
10	5510	FO	2000		5000	10	75832738000.00	13197423000.00
11	5510	FO	2000		5000	21	95168115200.00	5135982520.00
12	5510	FO	2000		5000	40	984967974.00	5650627100.00
13	5510	FO	2000		5000	41	746577242767.00	3572695526.00

## CHƯƠNG 3: ỨNG DỤNG HIVE PHÂN TÍCH KHO DỮ LIỆU

### 3.1. Nạp kho dữ liệu vào Hive

- Dùng lệnh `hdfs dfs -put xuấtnhapLN.txt /user/hadoopuser/hive` để đẩy dữ liệu vào hdfs. Ta kiểm tra bằng lệnh như hình:

```
hadoopuser@thuan-master:~/apache-hive-2.3.7-bin$ cd
hadoopuser@thuan-master:~$ hdfs dfs -ls /user/hadoopuser/hive
21/06/11 17:48:35 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-j
va classes where applicable
Found 1 items
-rw-r--r--  2 hadoopuser supergroup      40712 2021-06-11 13:14 /user/hadoopuser/hive/xuấtnhapLN.txt
hadoopuser@thuan-master:~$
```

- Dùng lệnh `hdfs dfs -put Dimquocgia.txt /user/hadoopuser/dim1` để đẩy dữ liệu vào hdfs. Ta kiểm tra bằng lệnh như hình:

```
hadoopuser@thuan-master:~/apache-hive-2.3.7-bin$ cd
hadoopuser@thuan-master:~$ hdfs dfs -ls /user/hadoopuser/dim1
21/06/11 17:50:46 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-j
va classes where applicable
Found 1 items
-rw-r--r--  2 hadoopuser supergroup      4637 2021-06-11 15:11 /user/hadoopuser/dim1/Dimquocgia.txt
hadoopuser@thuan-master:~$
```

- Dùng lệnh `hdfs dfs -put Dimloaisp.txt /user/hadoopuser/dim2` để đẩy dữ liệu vào hdfs. Ta kiểm tra bằng lệnh như hình:

```
hadoopuser@thuan-master:~/apache-hive-2.3.7-bin$ cd
hadoopuser@thuan-master:~$ hdfs dfs -ls /user/hadoopuser/dim2
21/06/11 17:51:35 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-j
va classes where applicable
Found 1 items
-rw-r--r--  2 hadoopuser supergroup      127 2021-06-11 15:44 /user/hadoopuser/dim2/Dimloaisp.txt
hadoopuser@thuan-master:~$
```

### 3.2. Tạo kho dữ liệu trong hive

- Tạo Fact bằng câu lệnh dưới:  

```
create external table if not exists lamnghiep
(linhvuc string,
quocgia int,
loaisp int,
nam int,
kiemtradl string,
giatrinhap float,
giatrixuat float
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '\t'
STORED AS TEXTFILE
LOCATION '/user/hadoopuser/hive';
```

Kiểm tra dữ liệu có nạp được vào Fact hay không:

```

engine (i.e. spark, tez) or using Hive 1.X releases.
hive> select * from lamnghiep limit 10;
OK
FO      10      1671      2001      2.92306555E10      3603000.0
FO      10      1671      2002      3.20970977E10      267000.0
FO      10      1671      2004      5.93145E10      1400000.0
FO      10      1671      2005      7.3188E10      360000.0
FO      10      1671      2007      *      5.3619442E10      17248.0
FO      10      1671      2010      3.7804335E10      4.0191672E7
FO      10      1671      2013      *      3.50935322E9      7.4786309E9
FO      10      1671      2016      2.2480087E9      5.9286103E9
FO      10      1671      2018      4.42569792E8      1.73227704E10
FO      10      1671      2019      1.90163392E8      9.6000799E9
Time taken: 4.733 seconds, Fetched: 10 row(s)
hive>

```

- Tạo dim quốc gia
  - create external table if not exists Dimquocgia
    - ( maquocgia int,
    - tenquocgia string
    - )
    - ROW FORMAT DELIMITED
    - FIELDS TERMINATED BY '\t'
    - STORED AS TEXTFILE
    - LOCATION '/user/hadoopuser/dim1';

```

hive> select * from dimquocgia limit 10;
OK
1      Armenia
2      Afghanistan
3      Albania
4      Algeria
5      American Samoa
6      Andorra
7      Angola
8      Antigua and Barbuda
9      Argentina
10     Australia
Time taken: 0.113 seconds, Fetched: 10 row(s)
hive>

```

- Tạo dim loại sản phẩm
  - create external table if not exists Dimloaisp
    - ( maloaisp int,
    - tenloaisp string
    - )
    - ROW FORMAT DELIMITED
    - FIELDS TERMINATED BY '\t'
    - STORED AS TEXTFILE
    - LOCATION '/user/hadoopuser/dim2';

```
hive> select * from dimloaisp;
OK
1671    Newsprint
1676    Household and sanitary papers
1618    Cartonboard
1621    Wrapping papers
1622    Other papers mainly for packaging
Time taken: 0.111 seconds, Fetched: 5 row(s)
hive>
```

### 3.3. Truy vấn phân tích trong kho dữ liệu

- **Tính tổng value xuất nhập khẩu lâm nghiệp mặt hàng giấy báo từ năm 2000-2019 của tất cả các nước trên thế giới**

```
select nam, SUM(giattrinhap), SUM(giatrixuat)
from lamnghiep
where loaisp = 1671
group by nam;
```

```
2021-06-11 16:07:38,439 Stage-1 map = 0%, reduce = 0%
2021-06-11 16:07:42,620 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 2.34 sec
2021-06-11 16:07:47,803 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 4.14 sec
MapReduce Total cumulative CPU time: 4 seconds 140 msec
Ended Job = job_1623425352778_0001
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1 Reduce: 1 Cumulative CPU: 4.14 sec HDFS Read: 51223 HDFS Write: 1071 SUCCESS
Total MapReduce CPU Time Spent: 4 seconds 140 msec
OK
2000    4.369264652E10    3.3886039086E10
2001    2.02577729865E11    2.5485752587E10
2002    8.8923236598E10    3.6612368506E10
2003    1.3634355124E10    1694333.0
2004    1.40327440929E11    6.3926926138E10
2005    1.43076630219E11    9.2853962127E10
2006    1.18936566781E11    6.2101043999E10
2007    1.84899973518E11    8.3960320784E10
2008    1.27121705657E11    1.55789888231E11
2009    9.0538687514E10    5.84850423E10
2010    1.77043904086E11    1.46914828607E11
2011    8.1728772934E10    1.59506151339E11
2012    1.14121578152E11    2.1871272226E10
2013    1.39874135668E11    1.73127613653E11
2014    5.830173184E10    1.37961050147E11
2015    5.5412076564E10    5.322485001E9
2016    3.6146435242E10    1.18685729068E11
2017    3.2228015428E10    1.24430525123E11
2018    5.9879825954E10    1.780235614E10
2019    1.9240694947E10    1.22922632169E11
Time taken: 17.345 seconds, Fetched: 20 row(s)
```

- **Tính tổng value nhập khẩu lâm nghiệp mặt hàng giấy báo của các quốc gia trên thế giới của tất cả các năm**

```
select lamnghiep.quocgia, dimquocgia.tenquocgia, SUM(giattrinhap)
from lamnghiep INNER JOIN dimquocgia ON dimquocgia.maquocgia=
lamnghiep.quocgia
where lamnghiep.loaisp = 1671
```



group by lamnghiep.quocgia, dimquocgia.tenquocgia;

```
Total MapReduce CPU Time Spent: 3 seconds 970 msec
OK
10      Australia      2.91644124416E11
11      Austria 7.177934464E10
19      Bolivia (Plurinational State of)      3.46679686E8
23      Belize  453532.0
44      Colombia      4.98021376E10
46      Congo  699201.0
50      Cyprus  4.03933425E8
53      Benin  1425580.0
102     Iran (Islamic Republic of)      2.8392982144E10
103     Iraq  1.446059E7
104     Ireland 3.638960256E10
105     Israel  8.0592328704E10
107     Côte d'Ivoire  5.6428767E7
147     Namibia 1.11180962E8
149     Nepal  1.222447744E9
150     Netherlands      8.8300873728E11
151     Netherlands Antilles (former)  7.9410866E7
167     Czechia 3.18402688E9
191     Saint Vincent and the Grenadines      128344.0
193     Sao Tome and Principe  14.0
194     Saudi Arabia  4.8272607472E10
225     United Arab Emirates  1.6593115152E10
230     Ukraine 5.2492713088E10
244     Samoa  267885.0
255     Belgium 3.63316907008E11
Time taken: 23.278 seconds, Fetched: 25 row(s)
hive>
```

- **Top 5 quốc gia có tổng value nhập khẩu mặt hàng giấy báo tính từ 2000-2019**

```
select lamnghiep.quocgia, dimquocgia.tenquocgia, SUM(giatrinhap) as k
from lamnghiep INNER JOIN dimquocgia ON
dimquocgia.maquocgia= lamnghiep.quocgia
where lamnghiep.loaisp = 1671
group by lamnghiep.quocgia, dimquocgia.tenquocgia
order by k DESC limit 5;
```

```
Hadoop job information for Stage-3: number of mappers: 1; number of reducers: 1
2021-06-11 18:18:22,093 Stage-3 map = 0%, reduce = 0%
2021-06-11 18:18:26,228 Stage-3 map = 100%, reduce = 0%, Cumulative CPU 0.88 sec
2021-06-11 18:18:31,388 Stage-3 map = 100%, reduce = 100%, Cumulative CPU 2.4 sec
MapReduce Total cumulative CPU time: 2 seconds 400 msec
Ended Job = job_1623425352778_0006
MapReduce Jobs Launched:
Stage-Stage-2: Map: 1 Reduce: 1 Cumulative CPU: 4.27 sec HDFS Read: 53496 HDFS Write: 1081 SUCCESS
Stage-Stage-3: Map: 1 Reduce: 1 Cumulative CPU: 2.4 sec HDFS Read: 7416 HDFS Write: 291 SUCCESS
Total MapReduce CPU Time Spent: 6 seconds 670 msec
OK
150     Netherlands      8.8300873728E11
255     Belgium 3.63316907008E11
10      Australia      2.91644124416E11
105     Israel  8.0592328704E10
11      Austria 7.177934464E10
Time taken: 42.743 seconds, Fetched: 5 row(s)
hive>
```

- **Top 10 quốc gia có value xuất khẩu mặt hàng thùng carton lớn nhất thế giới tính từ 2000-2019**

```
select lamnghiep.quocgia, dimquocgia.tenquocgia, SUM(giatrixuat) as k
from lamnghiep INNER JOIN dimquocgia ON
dimquocgia.maquocgia= lamnghiep.quocgia
where lamnghiep.loaisp = 1618
group by lamnghiep.quocgia, dimquocgia.tenquocgia
order by k DESC limit 10;
```

```
set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
set mapreduce.job.reduces=<number>
Starting Job = job_1623425352778_0008, Tracking URL = http://thuan-master:8088/proxy/application_1623425352778_0008/
Kill Command = /home/hadoopuser/hadoop/bin/hadoop job -kill job_1623425352778_0008
Hadoop job information for Stage-3: number of mappers: 1; number of reducers: 1
2021-06-11 18:23:43,228 Stage-3 map = 0%, reduce = 0%
2021-06-11 18:23:47,351 Stage-3 map = 100%, reduce = 0%, Cumulative CPU 0.84 sec
2021-06-11 18:23:52,546 Stage-3 map = 100%, reduce = 100%, Cumulative CPU 2.47 sec
MapReduce Total cumulative CPU time: 2 seconds 470 msec
Ended Job = job_1623425352778_0008
MapReduce Jobs Launched:
Stage-Stage-2: Map: 1 Reduce: 1 Cumulative CPU: 3.79 sec HDFS Read: 53499 HDFS Write: 1003 SUCCESS
Stage-Stage-3: Map: 1 Reduce: 1 Cumulative CPU: 2.47 sec HDFS Read: 7348 HDFS Write: 493 SUCCESS
Total MapReduce CPU Time Spent: 6 seconds 260 msec
OK
150 Netherlands 2.191099666432E12
255 Belgium 9.61877497856E11
11 Austria 6.84658637312E11
10 Australia 2.40060457216E11
230 Ukraine 1.52311425024E11
101 Indonesia 5.922486644E10
194 Saudi Arabia 1.826656024E9
44 Colombia 1.297840939E9
104 Ireland 4.25167384E8
147 Namibia 2.75472854E8
Time taken: 42.211 seconds, Fetched: 10 row(s)
hive>
```

- **Tính trung bình value xuất nhập khẩu lâm nghiệp từ năm 2000-2019 của nước Australia**

```
Select lamnghiep.nam, AVG(giatriNhap), AVG(giatrixuat)
from lamnghiep INNER JOIN dimquocgia ON
dimquocgia.maquocgia= lamnghiep.quocgia
where dimquocgia.tenquocgia = 'Australia'
group by lamnghiep.nam;
```

```

2021-06-11 18:32:26,829 Stage-2 map = 100%, reduce = 0%, Cumulative CPU 2.22 sec
2021-06-11 18:32:31,982 Stage-2 map = 100%, reduce = 100%, Cumulative CPU 3.93 sec
MapReduce Total cumulative CPU time: 3 seconds 930 msec
Ended Job = job_1623425352778_0009
MapReduce Jobs Launched:
Stage-Stage-2: Map: 1 Reduce: 1 Cumulative CPU: 3.93 sec HDFS Read: 54355 HDFS Write: 1036 SUCCESS
Total MapReduce CPU Time Spent: 3 seconds 930 msec
OK
2000 1.0042968484E10 8.33628508E8
2001 1.7050314752E10 4.338021053333333E9
2002 9.676459392E9 6.80805726E8
2003 3.008399872E9 1.618366464E10
2004 1.8734649856E10 6.905932648E9
2005 2.5627706538666668E10 2729333.3333333335
2006 4.911750144E9 2.4273000448E10
2007 2.0253276544E10 1.2350267256E10
2008 1.263216996E9 1168500.0
2009 1.361791488E10 3.7251076096E10
2010 1.7043925674666666E10 3.8044630906666665E9
2011 4.367979689E9 6637036.0
2012 1.6934144E10 1.6068134912E10
2013 7.588203946666667E9 7.872636714333333E9
2014 2.6282424E7 12.0
2015 1.6125563392E10 1.471937792E10
2016 8.113574144E9 5.230739504E9
2017 1.965486E7 300.0
2018 1.5547114602666666E10 1.4019813376E10
2019 7.561686250666667E9 7.14618064E9
Time taken: 22.137 seconds, Fetched: 20 row(s)
hive>

```

- **Tìm 5 năm có AVG value xuất khẩu lâm nghiệp cao nhất tính từ 2000-2019 của Australia**

Select lamnghiep.nam, AVG(giatriNhap) as a, AVG(giatriXuat) as b  
 from lamnghiep INNER JOIN dimquocgia ON  
 dimquocgia.maquocgia= lamnghiep.quocgia  
 where dimquocgia.tenquocgia = 'Australia'  
 group by lamnghiep.nam  
 order by a DESC limit 5;

```

2021-06-11 18:33:40,976 Stage-3 map = 100%, reduce = 100%, Cumulative CPU 2.44 sec
MapReduce Total cumulative CPU time: 2 seconds 440 msec
Ended Job = job_1623425352778_0011
MapReduce Jobs Launched:
Stage-Stage-2: Map: 1 Reduce: 1 Cumulative CPU: 3.81 sec HDFS Read: 53448 HDFS Write: 816 SUCCESS
Stage-Stage-3: Map: 1 Reduce: 1 Cumulative CPU: 2.44 sec HDFS Read: 6951 HDFS Write: 354 SUCCESS
Total MapReduce CPU Time Spent: 6 seconds 250 msec
OK
2005 2.5627706538666668E10 2729333.3333333335
2007 2.0253276544E10 1.2350267256E10
2004 1.8734649856E10 6.905932648E9
2001 1.7050314752E10 4.338021053333333E9
2010 1.7043925674666666E10 3.8044630906666665E9
Time taken: 44.061 seconds, Fetched: 5 row(s)
hive>

```

## **CHƯƠNG 4: KẾT LUẬN**

### **4.1. Vấn đề tồn tại**

- Sau quá trình thực hiện Project, nhóm đã phần nào học được kiến thức 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 động trên Hadoop, luồng dữ liệu của Hive, đặc trưng, kiến trúc, cách tổ chức dữ liệu trong Hive, thiết kế và xây dựng các bảng fact.
- Bên cạnh những việc nhóm đã làm thì không thể thiếu các sơ xuất xảy ra và những khó khăn mắc phải. Chưa đủ để khai thác hết các business process cho kho dữ liệu, mô hình datawarehouse còn nhỏ, chưa sử dụng được nhiều truy vấn với HQL.

### **4.2. Hướng phát triển**

- Xây dựng mô hình Fact cũng như có thể phân tích sâu rộng hơn trong đề tài, trực quan được giao diện một cách cụ thể rõ ràng hơn. Đồng thời phát triển đồ án trên quy mô dữ liệu lớn hơn. Nhóm em mong muốn rằng trong thời gian sắp tới có thể cải thiện hơn về project, mong sự góp ý từ thầy và các bạn để nhóm đánh giá một cách khách quan về những gì đạt và chưa đạt được của nhóm.

## TÀI LIỆU THAM KHẢO

- <https://www.slideshare.net/Simplilearn/hive-tutorial-hive-architecture-hive-tutorial-for-beginners-hive-in-hadoop-simplilearn/Simplilearn/hive-tutorial-hive-architecture-hive-tutorial-for-beginners-hive-in-hadoop-simplilearn>
- <https://www.facebook.com/notes/c%E1%BB%99ng-%C4%91%E1%BB%93ng-big-data-vi%E1%BB%87t-nam/apache-hive-tutorial/516751379203717/>
- [https://drive.google.com/file/d/1uJ72mxyNLnzapv\\_-7cJyJgM93nBsW\\_xh/view?usp=sharing](https://drive.google.com/file/d/1uJ72mxyNLnzapv_-7cJyJgM93nBsW_xh/view?usp=sharing)
- <https://viblo.asia/p/gioi-thieu-ve-hive-4P856kvaKY3>
- <http://www.fao.org/faostat/en/?fbclid=IwAR1lyMxgbz2vOGzbaUmi8SaapQxssyK2G92FnsPC4YBTo14rXdqb7Yszayo#data>