**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP.HCM**

**KHOA CÔNG NGHỆ THÔNG TIN**

**----🙣🕮🙡----**

****

**ĐIỆN TOÁN ĐÁM MÂY**

**Giáo viên hướng dẫn: Huỳnh Xuân Phụng**

**Sinh viên thực hiện: Nhóm**

**Uông Thị Thanh Thủy 17133064**

**Trần Gia Bảo 17133002**

**Hoàng Thị Cẩm Tú 17133071**

**🙦🙤🙥🙧🙦🙤**

**TP. Hồ Chí Minh – Năm 2020**

# **Lời cảm ơn**

Để hoàn thành bài nghiên cứu này, nhóm em xin tỏ lòng biết ơn sâu sắc đến thầy Huỳnh Xuân Phụng vì đã tận tình hướng dẫn trong suốt quá trình làm bài, trang bị những kiến thức, kỹ năng cơ bản cần có để hoàn thành đề tài này, đồng thời giúp nhóm em hoàn thiện hơn trong quá trình học tập.Tuy nhiên, trong quá trình nghiên cứu đề tài, kiến thức chuyên ngành còn hạn chế nên nhóm vẫn còn nhiều thiếu sót về tìm hiểu, đánh giá và trình bày đề tài. Rất mong nhận được sự quan tâm, góp ý của thầy để báo cáo của nhóm được đầy đủ và hoàn chỉnh hơn.

Nhóm em xin chân thành cảm ơn!

**MỤC LỤC**

[Lời cảm ơn 2](#_Toc60757905)

[CHƯƠNG 1: GIỚI THIỆU CHUNG 4](#_Toc60757906)

[1.1. Kiến trúc Hive 4](#_Toc60757907)

[1.2. Mô hình dữ liệu của Hive 7](#_Toc60757909)

[1.3. Các kiễu dữ liệu trong Hive 8](#_Toc60757911)

[1.4. Ngôn ngữ truy vấn HiveSQL 11](#_Toc60757912)

[1.5. Cài đặt Hive trong hệ điều hành Linux: 11](#_Toc60757913)

[CHƯƠNG 2: CHỌN TẬP DỮ LIỆU VÀ XỬ LÝ DATA 11](#_Toc60757914)

[CHƯƠNG 3: XÂY DỰNG DATA WAREHOUSE 19](#_Toc60757915)

[3.1. Khởi tạo Database 19](#_Toc60757916)

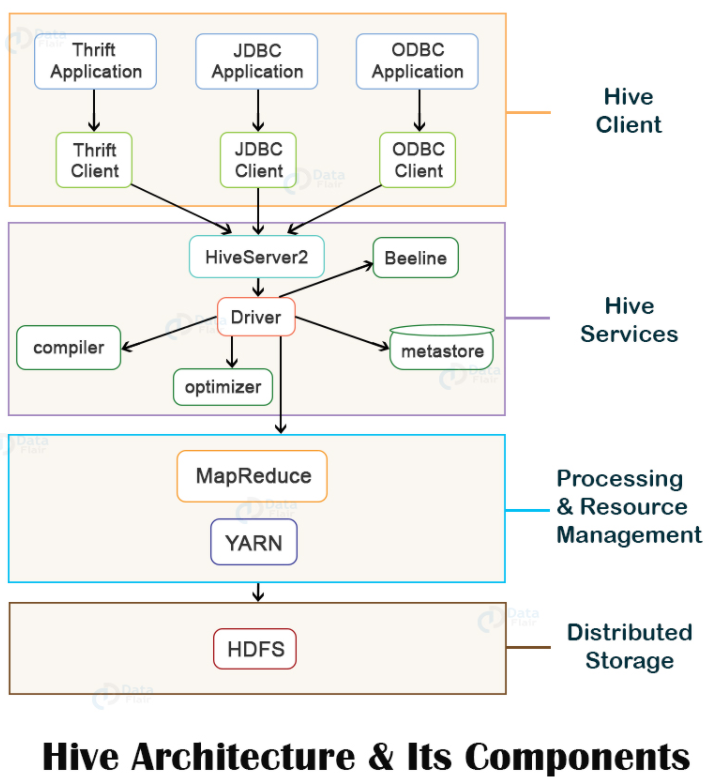
[3.2. Khởi tạo Data Mart 21](#_Toc60757917)

[3.3. Các câu lệnh truy vấn 23](#_Toc60757918)

[TÀI LIỆU THAM KHẢO 26](#_Toc60757919)

# **CHƯƠNG 1: GIỚI THIỆU CHUNG**

* 1. **Kiến trúc Hive**

****

* **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 Services:**

Để thực hiện tất cả các truy vấn, Hive cung cấp các dịch vụ khác nhau như Hive server2, Beeline, v.v. Các dịch vụ được cung cấp bởi Hive là:

* *Beeline:*

Beeline là một command shell được hỗ trợ bởi HiveServer2, nơi người dùng có thể gửi các truy vấn và lệnh của nó tới hệ thống. Nó là một ứng dụng JDBC client dựa trên SQLLINE CLI (tiện ích dựa trên bảng điều khiển Java thuần túy để kết nối với cơ sở dữ liệu quan hệ và thực thi các truy vấn SQL).

* *Hive server 2:*

HiveServer2 là sự kế thừa của HiveServer1. HiveServer2 cho phép khách hàng thực hiện các truy vấn chống lại Hive. Nó cho phép nhiều khách hàng gửi yêu cầu đến Hive và lấy kết quả cuối cùng. Về cơ bản, nó được thiết kế để cung cấp hỗ trợ tốt nhất cho các ứng dụng API mở như JDBC và ODBC.

***Note:*** Hive server1, còn được gọi là máy chủ Thrift, được xây dựng trên giao thức Apache Thrift để xử lý giao tiếp đa nền tảng với Hive. Nó cho phép các ứng dụng khách khác nhau gửi yêu cầu đến Hive và lấy kết quả cuối cùng.

* *Hive drive:*

Trình điều khiển Hive nhận các câu lệnh HiveQL do người dùng gửi thông qua command shell. Nó tạo các xử lý phiên cho truy vấn và gửi truy vấn đến compiler.

* *Hive compiler:*

Hive compiler phân tích cú pháp truy vấn. Nó thực hiện phân tích ngữ nghĩa và kiểm tra kiểu trên các khối truy vấn khác nhau và biểu thức truy vấn bằng cách sử dụng meta data được lưu trữ trong metastore và tạo ra một kế hoạch thực thi.

Kế hoạch thực thi được tạo bởi trình biên dịch là DAG (Directed Acyclic Graph), trong đó mỗi giai đoạn là một công việc bản đồ / thu nhỏ, hoạt động trên HDFS, một hoạt động siêu dữ liệu.

* *Optimizer:*

Trình tối ưu hóa thực hiện các hoạt động chuyển đổi trên kế hoạch thực thi và phân chia nhiệm vụ để cải thiện hiệu quả và khả năng mở rộng.

* *Excution engine:*

Công cụ thực thi, sau các bước biên dịch và tối ưu hóa, thực hiện kế hoạch thực thi do trình biên dịch tạo ra theo thứ tự phụ thuộc của chúng bằng Hadoop.

* *Metastore:*

Metastore là một kho lưu trữ trung tâm lưu trữ thông tin siêu dữ liệu về cấu trúc của bảng và partitions, bao gồm thông tin về cột và kiểu cột.

Nó cũng lưu trữ thông tin của bộ tuần tự và bộ giải mã, cần thiết cho hoạt động đọc / ghi và các tệp HDFS nơi dữ liệu được lưu trữ. Di căn này thường là một cơ sở dữ liệu quan hệ.

Metastore cung cấp giao diện Thrift để truy vấn và thao tác với siêu dữ liệu Hive.

* *Hcatalog:*

HCatalog là lớp quản lý bảng và lưu trữ cho Hadoop. Nó cho phép người dùng với các công cụ xử lý dữ liệu khác nhau như Pig, MapReduce, ... dễ dàng đọc và ghi dữ liệu trên lưới.

* *WebHCat:*

WebHCat là API REST cho HCatalog. Nó là một giao diện HTTP để thực hiện các hoạt động siêu dữ liệu Hive. Nó cung cấp một dịch vụ cho người dùng để chạy các công việc Hadoop MapReduce (hoặc YARN), Pig, Hive.

* **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. **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 thể 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 partion vs bucket:

* Partion thực hiện phân chia dữ liệu trong 1 table theo value của column key, mỗi partion 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 partion lại chỉ có 1 key duy nhất.

* 1. **Các kiễu dữ liệu trong Hive**

Kiểu dữ liệu trong Hive chỉ định loại cột / trường trong bảng Hive. Nó xấc 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. Numberic type (kiểu số): bao gồm
   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

**INTERGER:** 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. 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.

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

1. 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 đề cập đến thời gian trong ngày.

1. 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.

1. 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].

1. **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’]

1. **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.

1. **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. **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;*

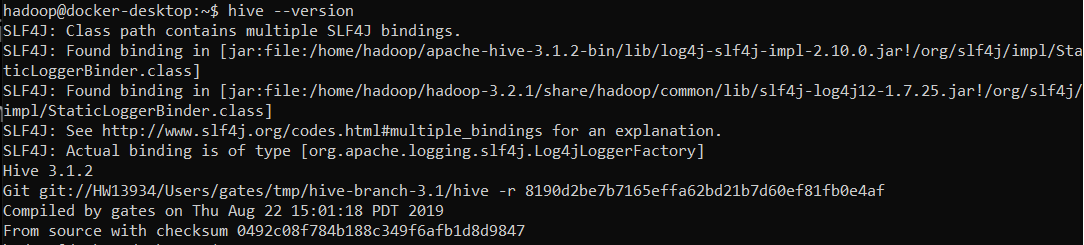
*SELECT category, count(1)*

*FROM products*

*GROUP BY category;*

Nó tương tự như SQL.

* 1. **Cài đặt Hive trong hệ điều hành Linux:**
* Phiên bản Hive:



* Link cài đặt ubuntu đã cài đặt hive từ docker :

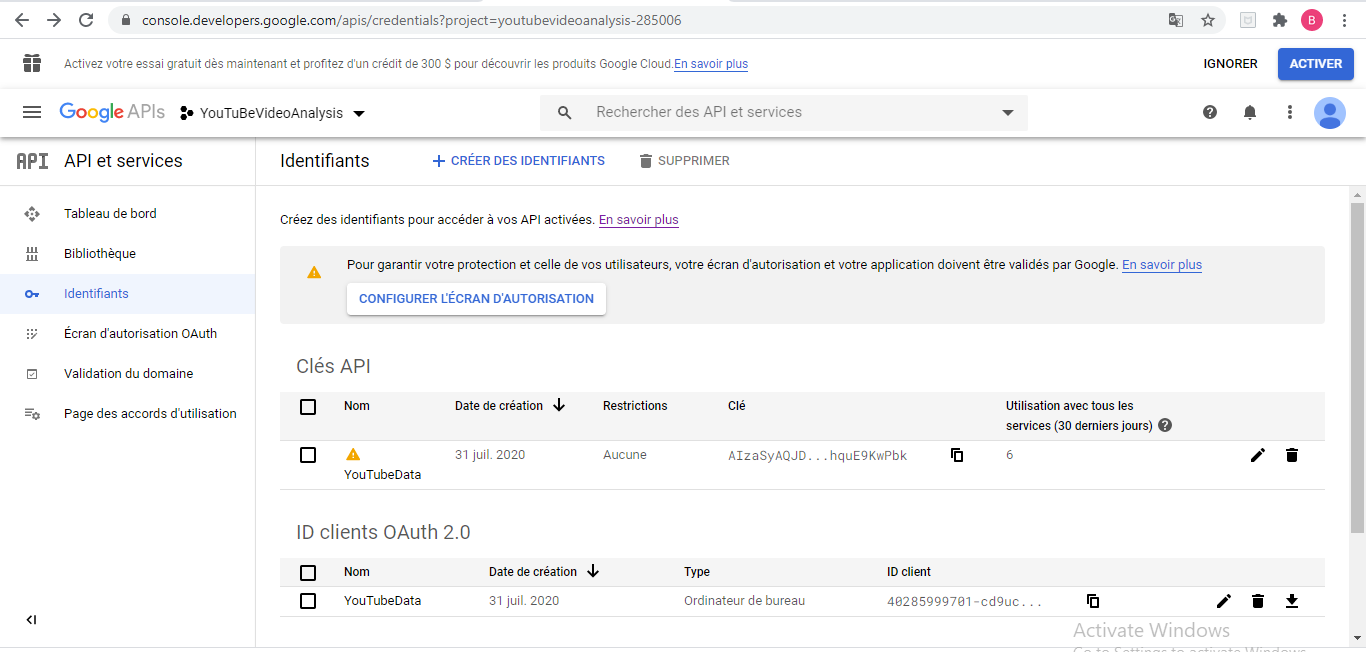
**docker pull thuyuong/ubuntu:18.04**

# **CHƯƠNG 2: CHỌN TẬP DỮ LIỆU VÀ XỬ LÝ DATA**

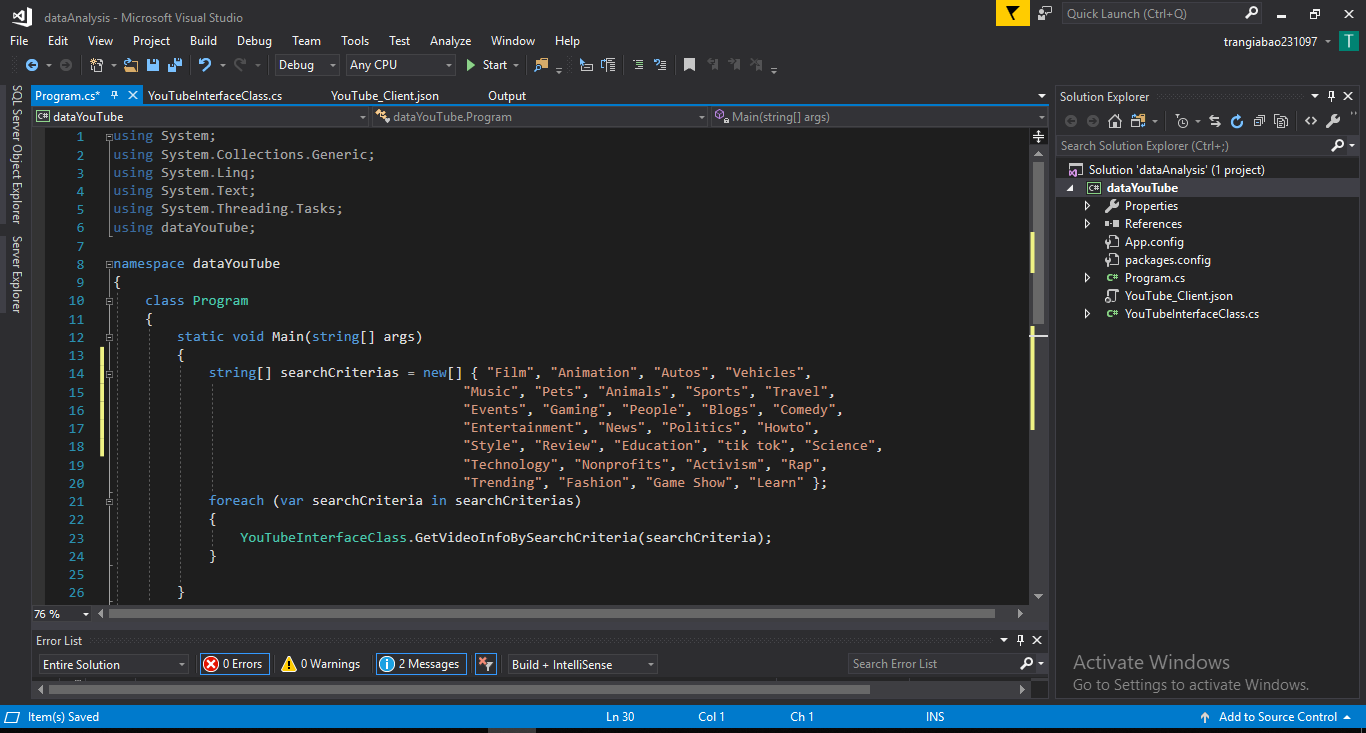
**Quá trình tạo dữ liệu gồm có 2 bước:**

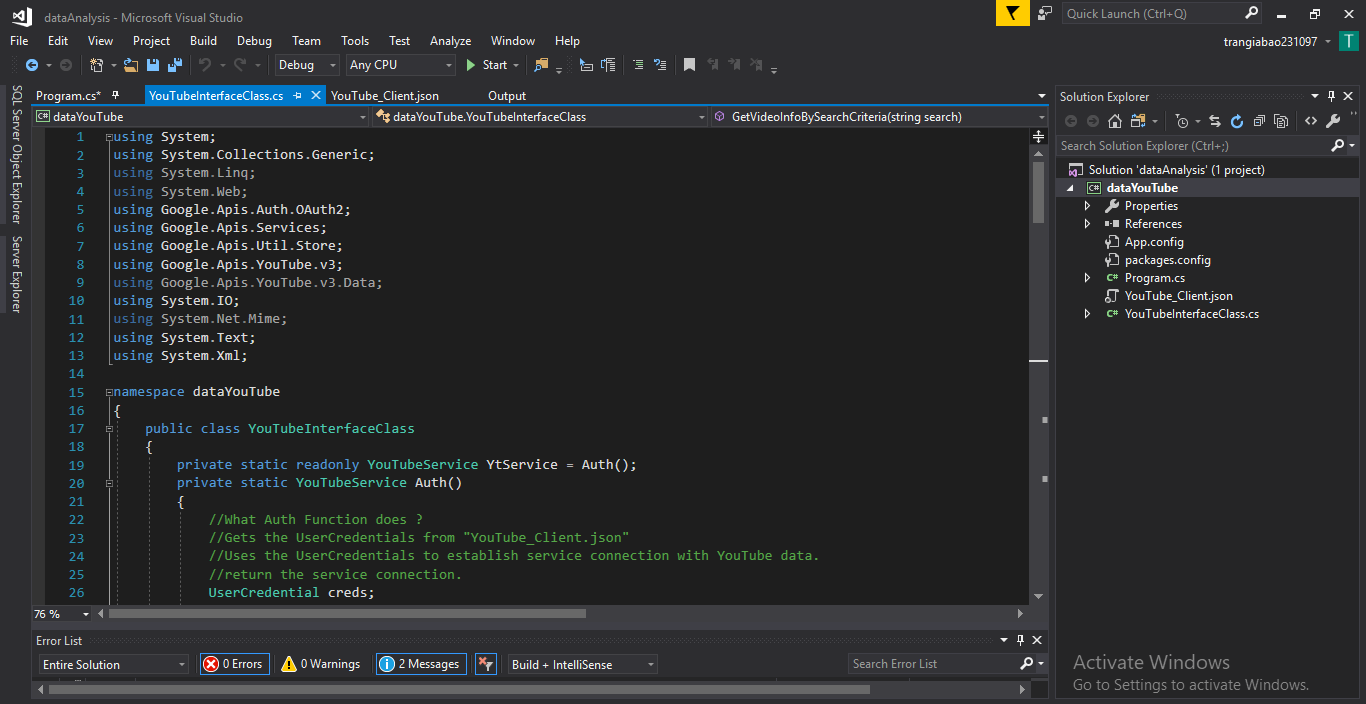
**Bước 1:** Tạo dữ liệu

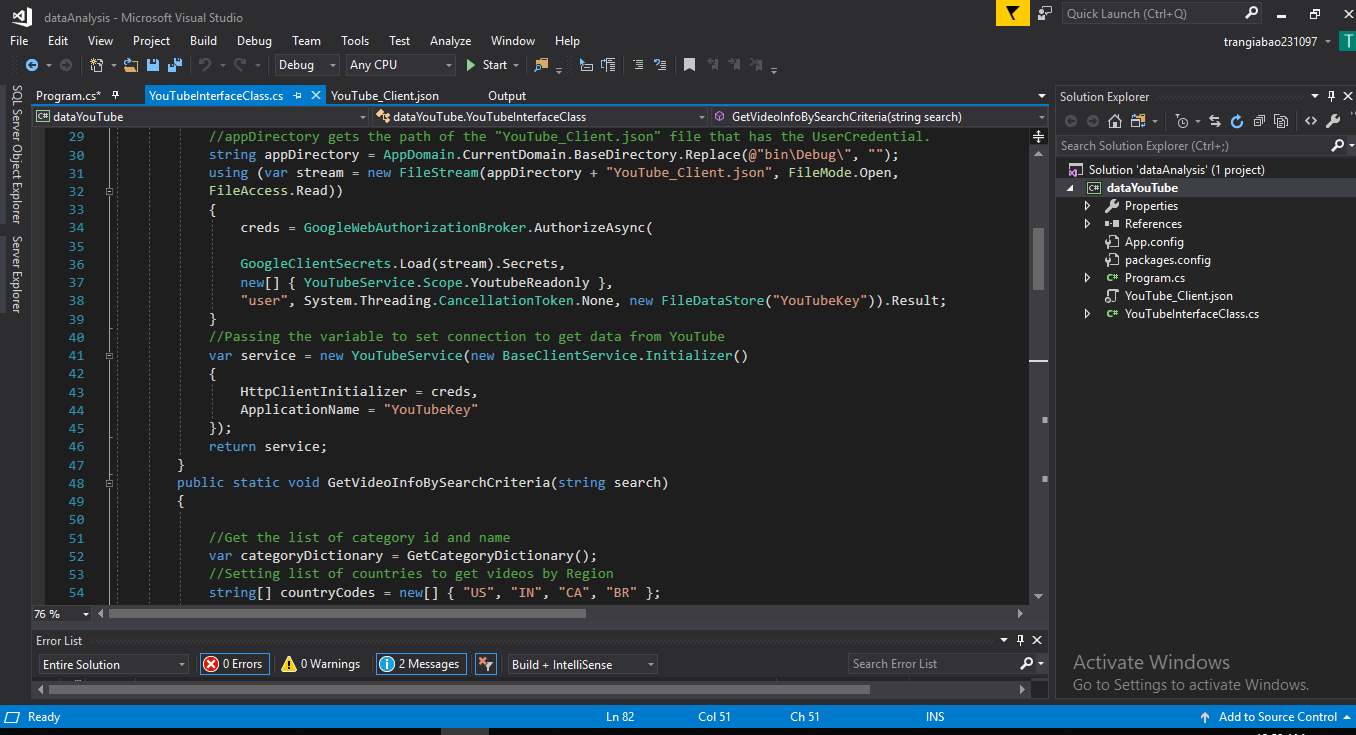
Đăng ký và xin cấp quyền từ API Key to Fetch YouTube Data (ID client OAuth)

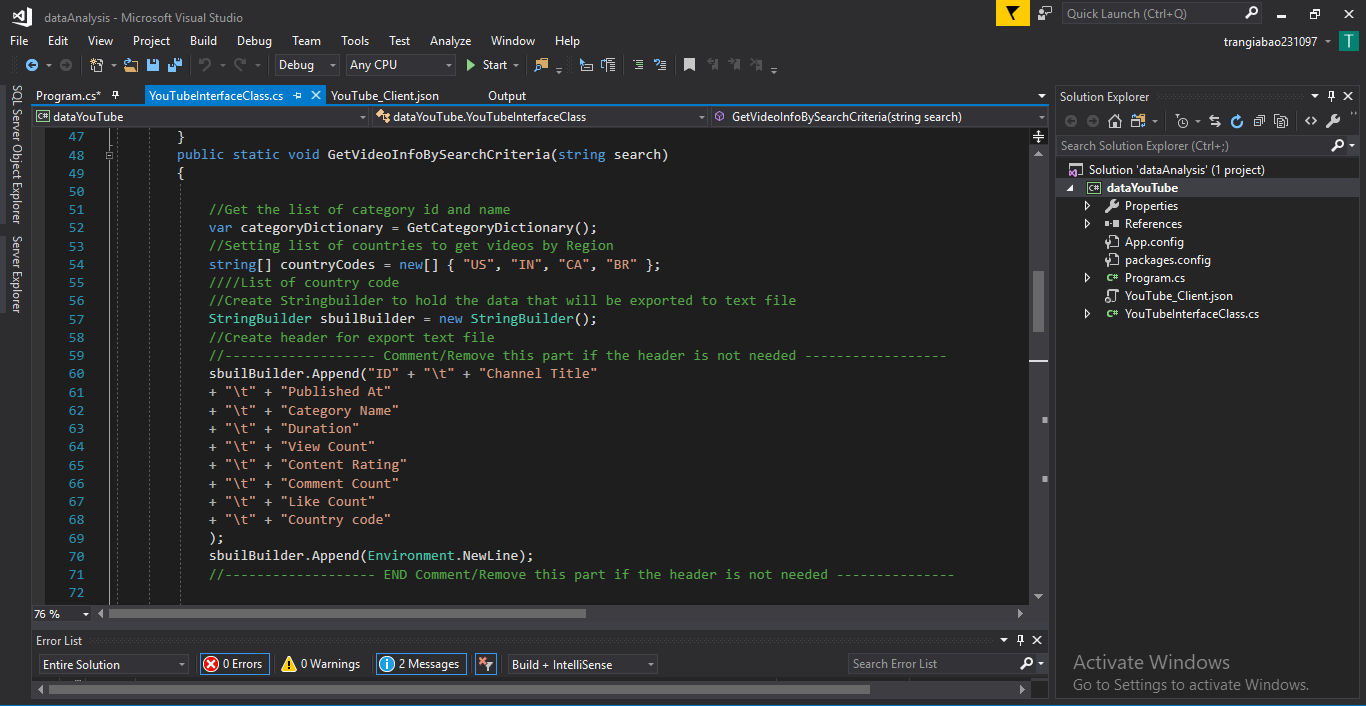


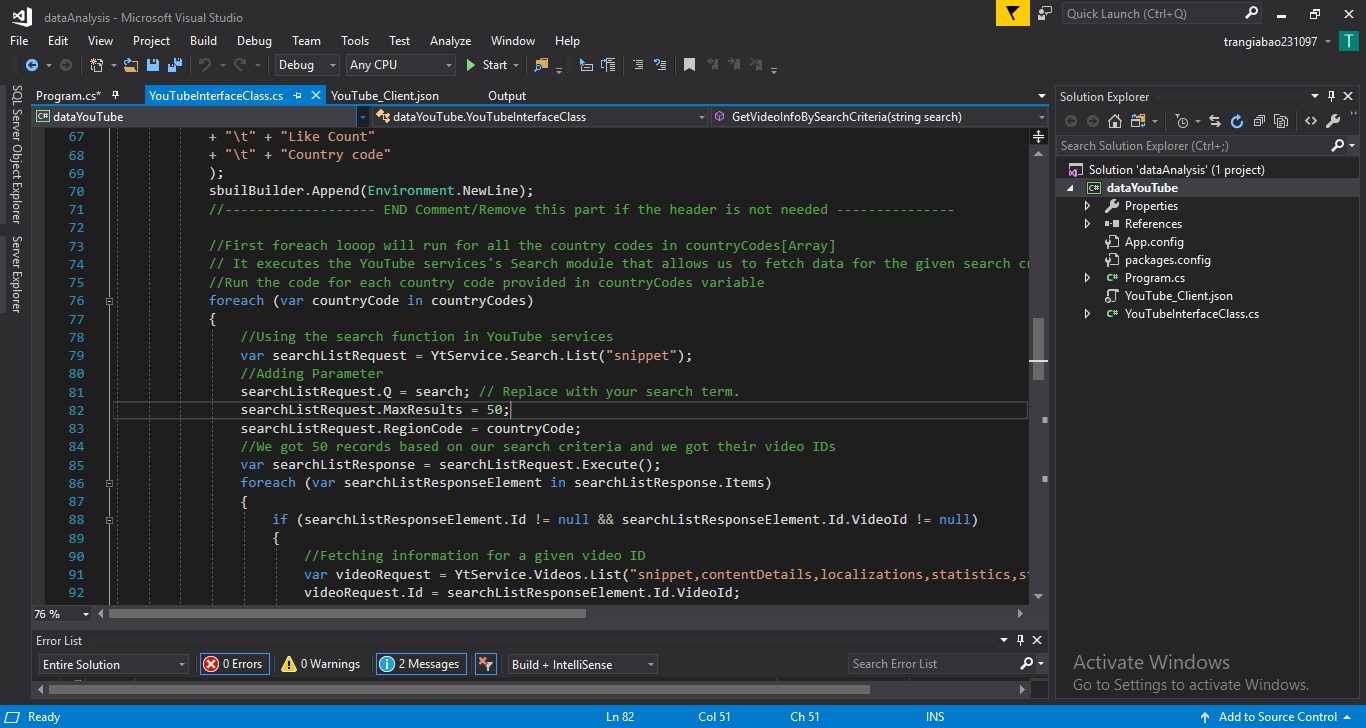
Viết project .Net(C#) Console Application để lấy dữ liệu từ YouTube API

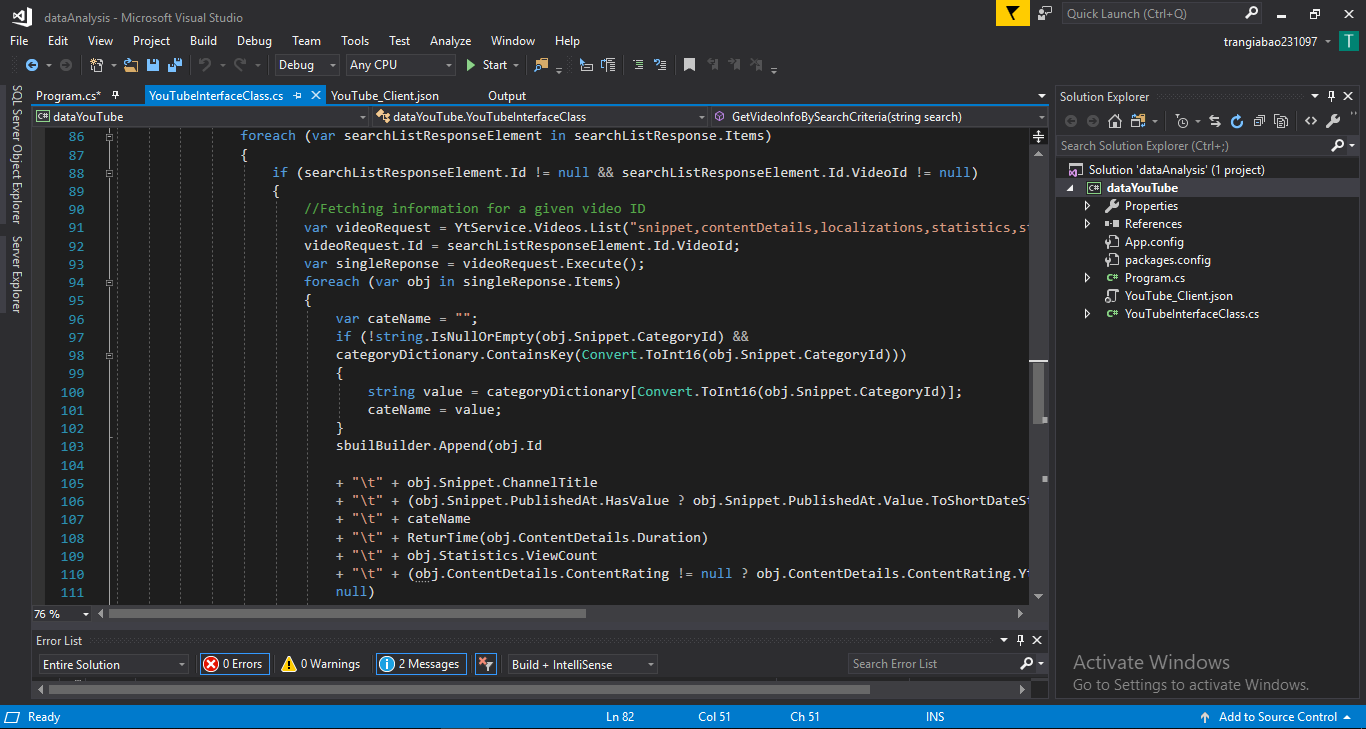


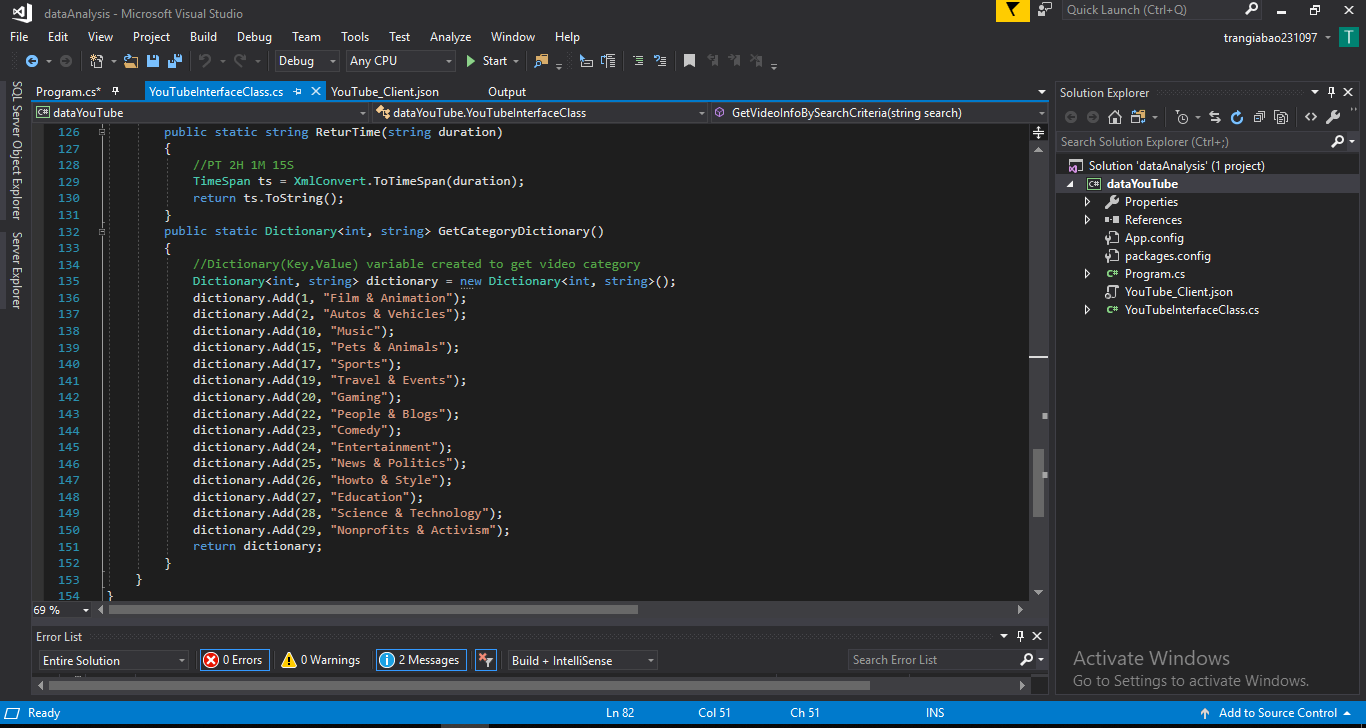


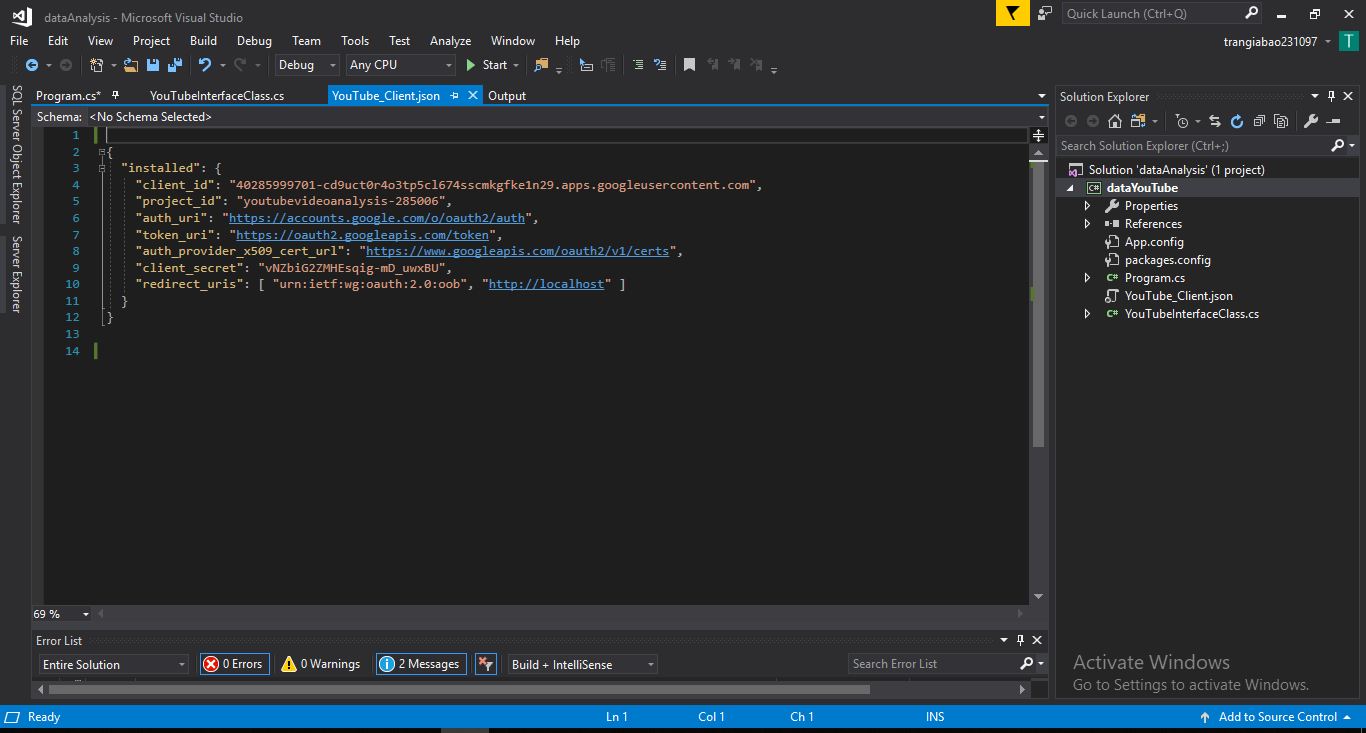




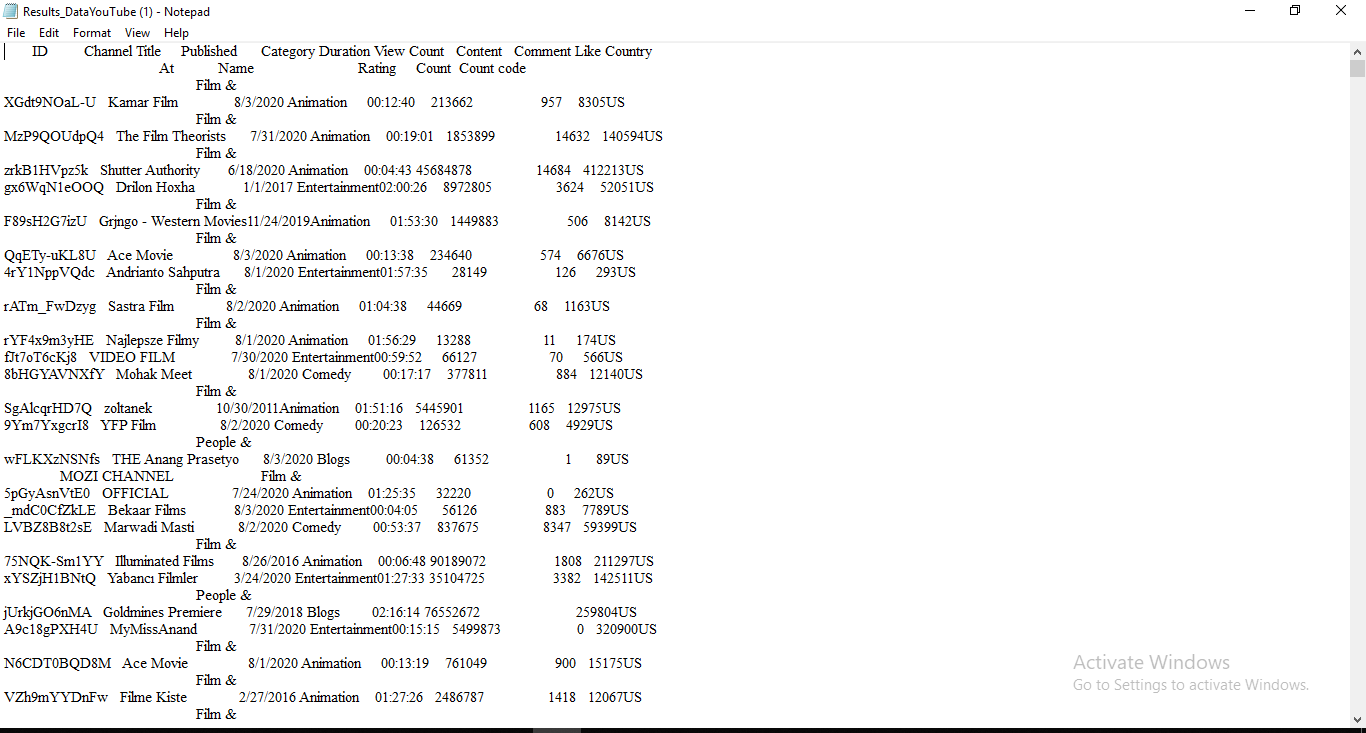




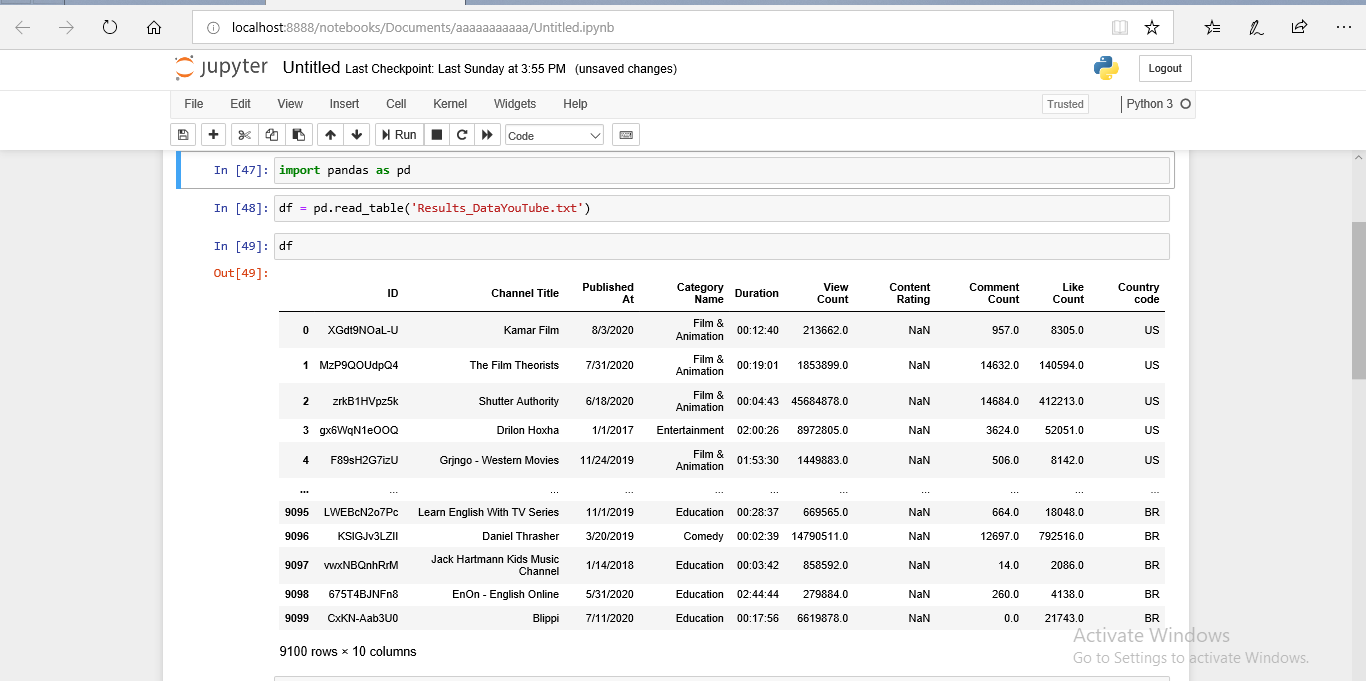




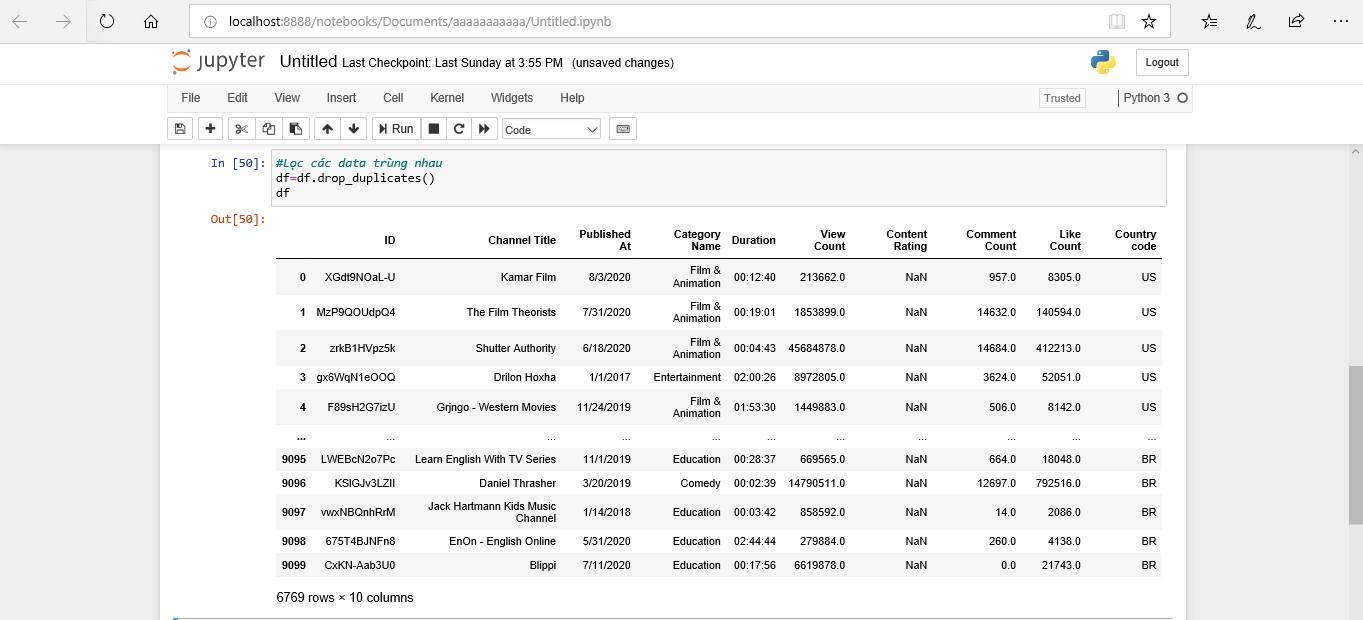
Kết quả



**Bước 2:** Làm sạch dữ liệu bằng Jupyter Notebook



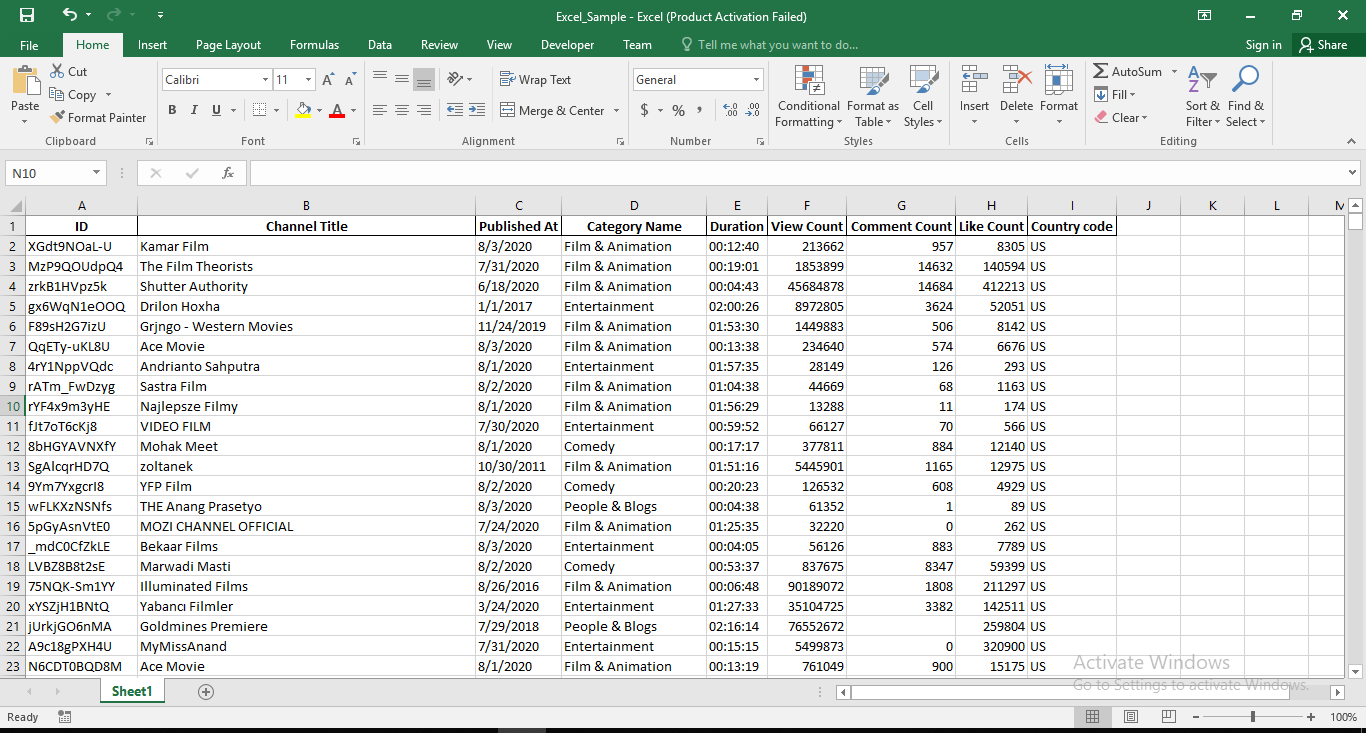
Kết quả:



Xuất File:



Kết quả của File:



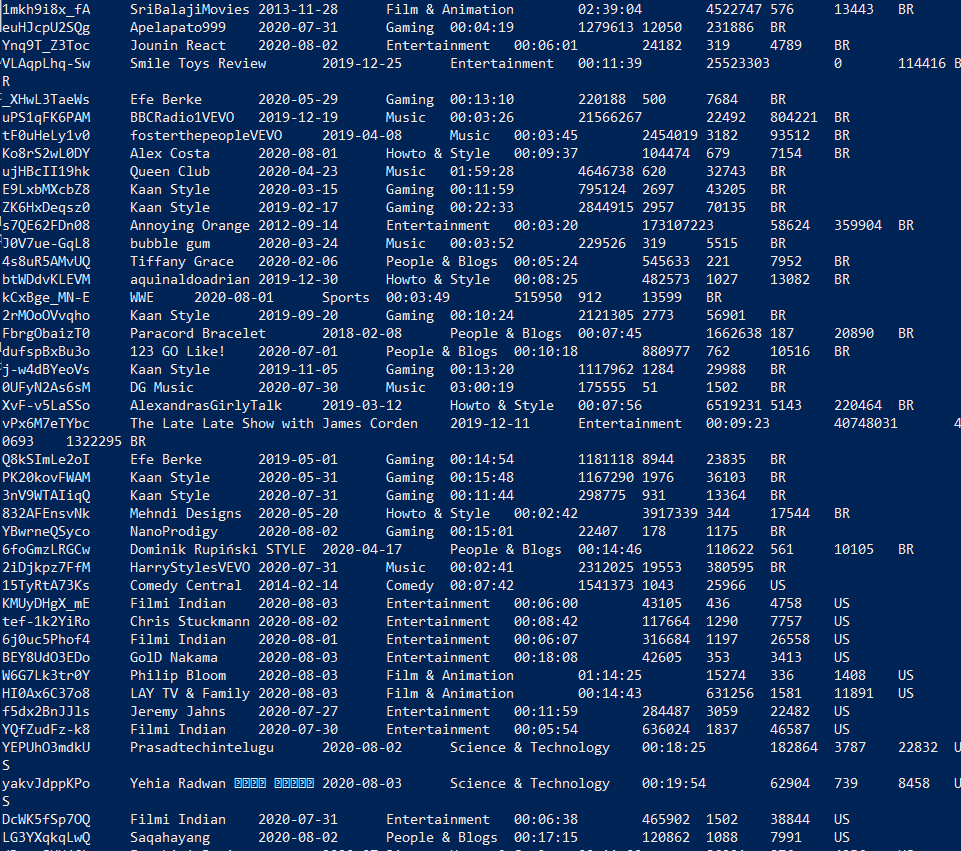
Mô tả dữ liệu:

* Dữ liệu gồm 6769 dòng và 9 cột.
* Các cột lần lượt từ trái sang phải gồm: ID(mã video), Channel Title(tiêu đề kênh), Published At(thời gian xuất bản video), Category Name(tên danh mục), Duration(thời lượng), View Count(số lượt xem), Comment Count(số nhận xét), Like Count(số lượt thích), Country code(mã quốc gia).
* Mổi một video sẽ có một mã video, tiêu đề kênh, thời gian xuất bản video, tên danh mục, thời lượng, số lượt xem và số nhận xét.
* Nhưng có số lượt thích khác nhau tùy vào mã quốc gia

# **CHƯƠNG 3: XÂY DỰNG DATA WAREHOUSE**

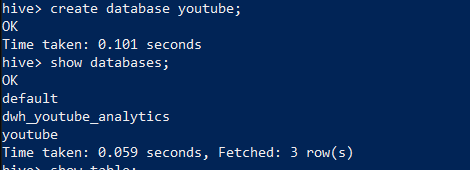
* 1. **Khởi tạo Database**
* Link down datayoutube.txt: <https://drive.google.com/file/d/1uJ72mxyNLnzapv_-7cJyjgM93nBsW_xh/view?usp=sharing>
* Câu lệnh download file datayoutube.txt: wget --no-check-certificate 'https://docs.google.com/uc?export=download&id=1uJ72mxyNLnzapv\_-7cJyjgM93nBsW\_xh' -O datayoutube.txt





* Tạo database:

Hive> CREATE DATABASE youtube;



* Tạo bảng dwh\_yt\_ext từ dữ liệu datayoutube.txt:

*CREATE EXTERNAL TABLE IF NOT EXISTS dwh\_youtube\_ext*

*(ID string,*

*Channel\_Title string,*

*Published\_At string,*

*Category\_Name string,*

*Duration int,*

*View\_Count int,*

*Comment\_Count int,*

*Like\_Count int,*

*Country\_code string*

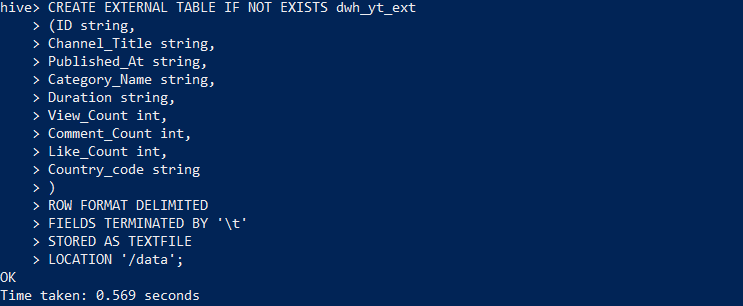
*)*

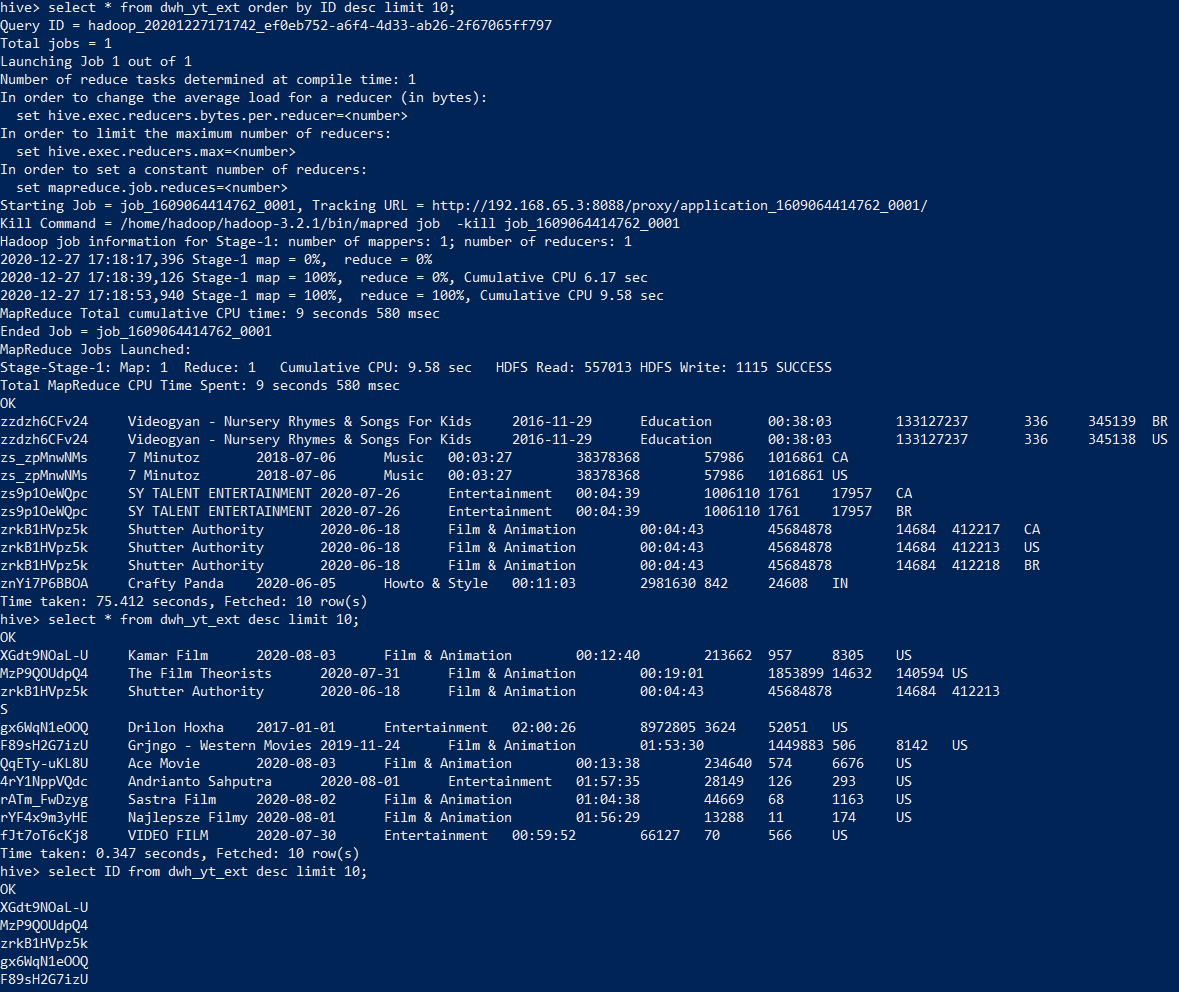
*ROW FORMAT DELIMITED*

*FIELDS TERMINATED BY '\t'*

*STORED AS TEXTFILE*

*LOCATION '/data';*





* 1. **Khởi tạo Data Mart**
* Bảng *youtube.trending\_channel* đếm số thời gian video, lượt view, lượt like và lượt comment của các *Channel\_Title* thuộc các *Category\_Name*

*CREATE TABLE youtube.trending\_channel*

*AS*

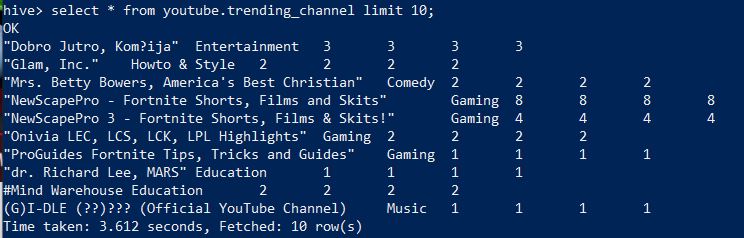
*(*

*SELECT Channel\_Tilte,Category\_Name,count(Duration) as total\_time,count(View\_Count) as total\_view,count(Like\_Count) as total\_like,count(Comment\_Count) as total\_comment*

*FROM dwh\_youtube\_ext*

*GROUP BY Channel\_Tilte,Category\_Name*

*);*



* Bảng *youtube.trending\_date* có mục đích tính tổng thời gian video, lượt view, lượt like và lượt comment trong từng thời gian *Published\_At* thuộc các *Category\_Name*

*CREATE TABLE youtube.trending\_date*

*AS*

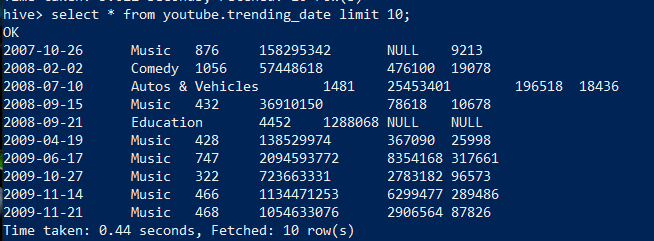
*(*

*SELECT to\_date(Published\_At) as day,Category\_Name,sum(Duration) as total\_time,sum(View\_Count) as total\_view,sum(Like\_Count) as total\_like,sum(Comment\_Count) as total\_comment*

*FROM dwh\_youtube\_ext*

*GROUP BY to\_date(Published\_At), Category\_Name*

*);*



* Bảng *youtube.trending\_categories\_country* có mục đích tính tổng thời gian video, lượt view, lượt like và lượt comment trong từng *Country\_Code* thuộc các *Category\_Name*

*CREATE TABLE youtube.trending\_categories\_country*

*AS*

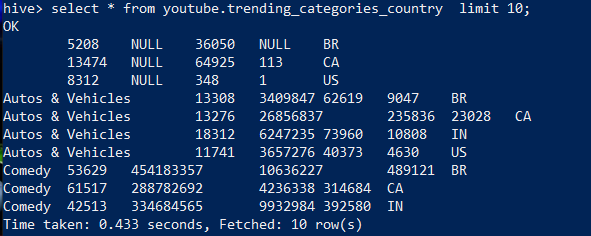
*(*

*SELECT Category\_Name,sum(Duration) as total\_time,sum(View\_Count) as total\_view,sum(Like\_Count) as total\_like,sum(Comment\_Count) as total\_comment,Country\_Code*

*FROM dwh\_youtube\_ext*

*GROUP BY Category\_Name, Country\_Code*

*);*



* 1. **Các câu lệnh truy vấn**
* Tổng thời gian video, lượt view, lượt like và lượt comment của thể loại video Education vào 10 ngày gần nhất

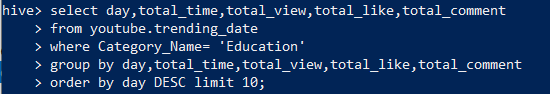
*select day,total\_time,total\_view,total\_like,total\_comment*

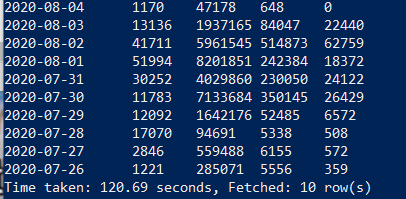
*from youtube.trending\_date*

*where Category\_Name= 'Education'*

*group by day,total\_time,total\_view,total\_like,total\_comment*

*order by day DESC limit 10;*





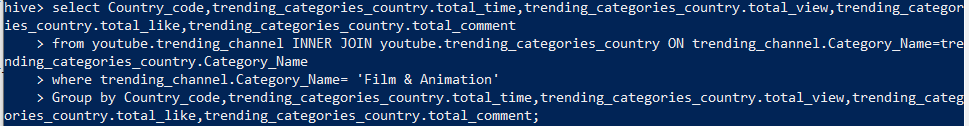
* Đếm thời gian video, lượt view, lượt like và lượt comment của thể loại Film & Animation ở các nước

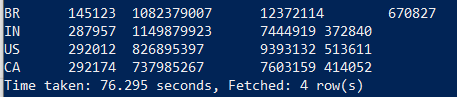
*select Country\_code, trending\_categories\_country.total\_time, trending\_categories\_country.total\_view, trending\_categories\_country.total\_like, trending\_categories\_country.total\_comment*

*from youtube.trending\_channel INNER JOIN youtube.trending\_categories\_country ON trending\_channel.Category\_Name=trending\_categories\_country.Category\_Name*

*where trending\_channel.Category\_Name= 'Film & Animation'*

*Group by Country\_code, trending\_categories\_country.total\_time, trending\_categories\_country.total\_view, trending\_categories\_country.total\_like, trending\_categories\_country.total\_comment;*





* Top 10 tên video và thể loại video được xem nhiều nhất ở nước Mĩ

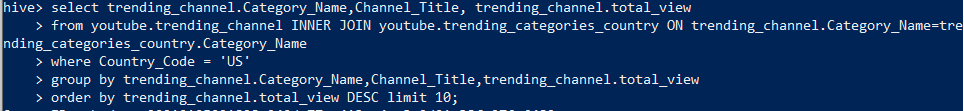
*select trending\_channel.Category\_Name,Channel\_Title, trending\_channel.total\_view*

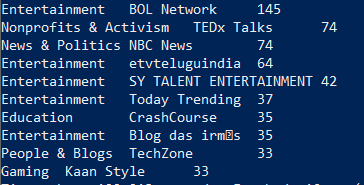
*from youtube.trending\_channel INNER JOIN youtube.trending\_categories\_country ON trending\_channel.Category\_Name=trending\_categories\_country.Category\_Name*

*where Country\_Code = 'US'*

*group by trending\_channel.Category\_Name,Channel\_Title,trending\_channel.total\_view*

*order by trending\_channel.total\_view DESC limit 10;*





* 10 tên video của thể loại Music vào ngày '2007-10-26' ở nước Mĩ

*select b.Channel\_Title*

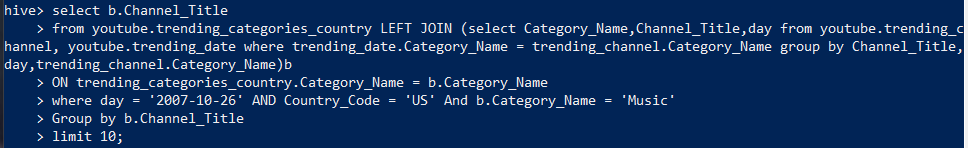
*from youtube.trending\_categories\_country LEFT JOIN (select Category\_Name,Channel\_Title,day from youtube.trending\_channel, youtube.trending\_date where trending\_date.Category\_Name = trending\_channel.Category\_Name group by Channel\_Title,day,trending\_channel.Category\_Name)b*

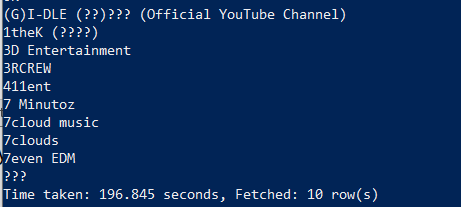
*ON trending\_categories\_country.Category\_Name = b.Category\_Name*

*where day = '2007-10-26' AND Country\_Code = 'US' And b.Category\_Name = 'Music'*

*Group by b.Channel\_Title*

*Limit 10;*





# **TÀI LIỆU THAM KHẢO**

* <https://viblo.asia/p/gioi-thieu-ve-hive-4P856kvaKY3>
* <https://viblo.asia/p/cac-loai-du-lieu-trong-hive-gDVK2z4wKLj>
* <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.statista.com/>
* <https://www.statista.com/topics/2019/>
* <http://www.pnrsolution.org/>.
* <http://www.pnrsolution.org/Datacenter/Vol3/Issue6/18.pdf>
* <http://common.books24x7.com/toc.aspx?bookid=114046>
* <http://common.books24x7.com/toc.aspx?bookid=62641>
* <http://share.edureka.co/>
* <http://share.edureka.co/pydio/data/public/hadoop>
* <https://www.appcoda.com>
* <https://www.appcoda.com/YouTube-api-ios-tutorial/>