

**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**  
**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**  
**KHOA HỆ THỐNG THÔNG TIN**



**BÁO CÁO ĐO ÁN**

**Đề tài: PHÂN TÍCH DỮ LIỆU VỀ SỨC TÀN PHÁ CỦA LỐC XOÁY Ở HOA KỲ**

**Giảng viên hướng dẫn:** Th.S Nguyễn Thị Kim Phụng

**Lớp** : IS217.N22.HTCL

**Nhóm sinh viên thực hiện:**

1. Lâm Trà My                           MSSV: 20521622

2. Lê Thị Đoan Trang                   MSSV: 20522038

**Ngày: Tp. Hồ Chí Minh, 04/2022**

# NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

....., ngày ..... tháng ..... năm 20...

## **Người nhận xét**

(Ký tên và ghi rõ họ tên)

## LỜI CẢM ƠN

Lời đầu tiên, nhóm em xin chân thành cảm ơn sự hướng dẫn, dẫn dắt từ cô Nguyễn Thị Kim Phụng, giảng viên khoa Hệ thống Thông tin. Đồng thời nhóm em xin gửi lời cảm ơn đến Trường Đại học Công nghệ Thông tin đã đưa bộ môn “Kho dữ liệu và OLAP” vào chương trình đào tạo của sinh viên.

Nhờ môn học này và sự dẫn dắt của cô đã mang lại cho nhóm em những hiểu biết vô cùng mới mẻ, tạo điều kiện tốt nhất và cung cấp những kiến thức cần thiết giúp nhóm em có thể vận dụng được những kiến thức đã học kết hợp với việc tìm tòi, tiếp thu những kiến thức từ thầy cô, bạn bè để hoàn thành đồ án báo cáo của môn học Kho dữ liệu và OLAP với đề tài “Phân tích dữ liệu về sức tàn phá của lốc xoáy ở Hoa Kỳ”. Tuy nhiên, vì tầm nhìn và kiến thức còn hạn chế nên nội dung báo cáo của nhóm sẽ không tránh khỏi những thiếu sót và lỗi sai. Nhóm em rất mong nhận được những nhận xét, góp ý từ cô để nhóm có thể rút ra kinh nghiệm, tiếp thu thêm những ý kiến mới giúp nhóm em có thể hoàn thiện các đồ án khác trong tương lai cũng như trong việc học tập một cách tốt hơn.

Cuối cùng, nhóm em một lần nữa xin chân thành cảm ơn cô và nhà trường, kính chúc cô luôn mạnh khỏe, vui vẻ và hạnh phúc để cùng đồng hành với sinh viên trong nhiều học kỳ tới, để có thể truyền đạt tới sinh viên những bài học hay và những kinh nghiệm quý giá.

Tp. Hồ Chí Minh, tháng 4 năm 2023

Nhóm sinh viên thực hiện

## MỤC LỤC

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN .....	2
LỜI CẢM ƠN.....	3
MỤC LỤC .....	4
CHƯƠNG I: TỔNG QUAN VỀ ĐỀ TÀI .....	6
1. Lý do chọn đề tài .....	6
2. Giới thiệu về dataset.....	6
3. Mô tả thuộc tính bảng Fact, các bảng Dimension.....	7
3.1. <i>Lược đồ kho dữ liệu .....</i>	7
3.2. <i>Bảng Fact_Tornado.....</i>	8
3.3. <i>Bảng Dim_Date .....</i>	8
3.4. <i>Bảng Dim_Location.....</i>	9
3.5. <i>Bảng Dim_Size .....</i>	9
CHƯƠNG II: XÂY DỰNG KHO DỮ LIỆU – QUÁ TRÌNH SSIS.....	10
1. Tạo cơ sở dữ liệu và import file .csv vào SQL Server.....	10
1.1. <i>Tạo mới một cơ sở dữ liệu .....</i>	10
1.2. <i>Quá trình đưa file .csv vào SQL Server .....</i>	11
2. Tạo project mới và thiết lập kết nối .....	15
3. Quá trình đổ dữ liệu từ file Excel vào Database của các bảng Dim .....	20
4. Quá trình đổ dữ liệu từ file Excel vào Database của bảng Fact.....	27
CHƯƠNG III: PHÂN TÍCH KHO DỮ LIỆU – QUÁ TRÌNH SSAS .....	39
1. Tạo project SSAS .....	39
1.1 <i>Tạo Data Source .....</i>	40
1.2 <i>Tạo Data Source Views .....</i>	43
1.3 <i>Tạo Cubes .....</i>	45
1.4 <i>Chỉnh sửa Dimension.....</i>	51
1.5 <i>Process .....</i>	52
2. Thực thi truy vấn .....	53
2.1 <i>Roll up .....</i>	53
2.2 <i>Drill down .....</i>	54

2.3 Slide and Dice .....	55
2.4 Pivot .....	58
3. Ngôn ngữ MDX .....	60
3.1 Tổng số lượng lốc xoáy theo từng tháng của mỗi năm .....	60
3.2 Tổng số lượng lốc xoáy theo từng năm .....	61
3.3 Số lần xảy ra lốc xoáy ở từng bang trong năm 2020 .....	61
3.4 Số lần xảy ra lốc xoáy ở từng bang theo từng tháng trong năm 2020 .....	61
3.5 Tìm top 3 bang có tổng số người tử vong cao nhất năm 1953 .....	62
3.6 Tìm các bang với số lượng lốc xoáy tại bang đó có mức độ sức tàn phá từ "Nghiêm trọng" trở lên .....	62
3.7 Tìm các năm có các cơn lốc xoáy có độ đo F/EF cao nhất và gây tử vong.....	63
3.8 Số người bị thương do các cơn lốc xoáy có độ đo F/EF ở mức độ nhẹ đến trung bình .....	63
3.9 Trung bình một ngày có bao nhiêu người chết và bị thương ở bang MA .....	64
3.10 Tìm các bang có các cơn lốc xoáy gây thương tích nhưng không gây tử vong trong năm 2001 .....	64
3.11 Tên các tiểu bang có số lượng người bị thương >5500 người.....	65
3.12 Thống kê số lượng lốc xoáy theo từng tháng và từng loại độ đo F/EF .....	65
3.13 Thống kê top 5 bang xảy ra lốc xoáy ít nhất.....	66
3.14 Thống kê tổng số người bị thương theo từng năm và từng bang.....	66
4. Power BI – Excel .....	67
4.1 Tạo project Power BI.....	67
4.2 Tạo project Excel .....	69
4.3 Truy vấn .....	72
<b>CHƯƠNG IV: DATA MINING .....</b>	<b>101</b>
1. Random Forest .....	101
2. Decision Tree .....	113
<b>TÀI LIỆU THAM KHẢO .....</b>	<b>127</b>

## CHƯƠNG I: TỔNG QUAN VỀ ĐỀ TÀI

### 1. Lý do chọn đề tài

Lốc xoáy là một hiện tượng thời tiết nguy hiểm, làm cho hàng triệu người phải sơ hãi và đối mặt với những hậu quả nghiêm trọng. Lốc xoáy là một cơn bão xoáy, với một trục trung tâm mạnh mẽ xoay quanh trục này. Tốc độ gió bên trong lốc xoáy có thể đạt đến hàng trăm dặm một giờ và phá hủy mọi thứ trong đường đi của nó.

Hoa Kỳ là quốc gia có nhiều lốc xoáy nhất trên thế giới, với trung bình khoảng 1.200 cơn mỗi năm. Các bang trung tâm và đông nam của Hoa Kỳ, bao gồm Oklahoma, Kansas, Nebraska, Texas và Missouri, là nơi có nguy cơ lốc xoáy cao nhất. Các vùng khác như các bang miền Đông, miền Nam và miền Trung Tây Hoa Kỳ cũng thường xuyên bị ảnh hưởng bởi lốc xoáy. Thang đo Fujita và thang đo Fujita Nâng cao là hai phương pháp đánh giá độ tàn phá của lốc xoáy được sử dụng bởi các nhà khí tượng học. Thang đo Fujita được phát triển bởi một nhà khoa học người Nhật Bản, Tetsuya Theodore Fujita vào những năm 1970, đánh giá độ tàn phá của lốc xoáy dựa trên các chỉ số từ F0 đến F5. Thang đo Fujita Nâng cao được sử dụng từ năm 2007, được cải tiến với việc bổ sung các yếu tố khác như chiều dài và chiều rộng của lốc xoáy, giúp cho việc đánh giá độ tàn phá của lốc xoáy trở nên chính xác hơn và giúp các nhà khí tượng học cảnh báo kịp thời cho các cộng đồng có nguy cơ bị ảnh hưởng.

Để hiểu rõ hơn về sức tàn phá của lốc xoáy, nhóm em chọn đề tài “Phân tích dữ liệu về sức tàn phá của lốc xoáy ở Hoa Kỳ”.

### 2. Giới thiệu về dataset

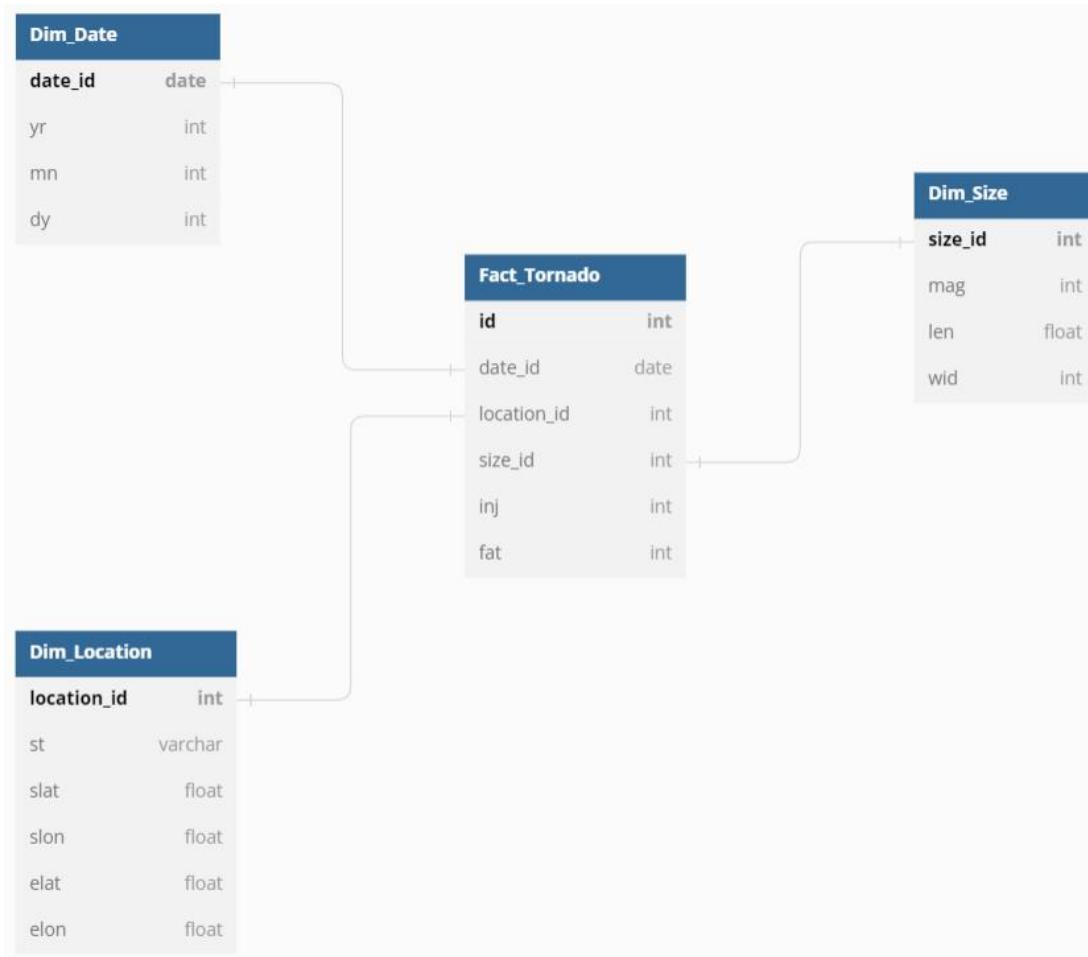
- Tên dataset: US Tornado Dataset 1950-2021
- Đây là bộ dữ liệu mô tả về hơn 65,000 thông tin về tọa độ và thời gian khác nhau xảy ra lốc xoáy từ năm 1950 đến 2021, được cung cấp bởi tài khoản Kaggle có tên RANDOM DRAW.
- Link dataset: <https://s.net.vn/rKRR>
- Mô tả dữ liệu ban đầu:

Mỗi dòng dữ liệu là thông tin về một trận lốc xoáy bao gồm: Ngày tháng năm xảy ra, kích thước lốc xoáy, số lượng người bị thương hoặc tử vong, ... Bộ dữ liệu có 67559 dòng và 14 thuộc tính.

STT	Tên thuộc tính	Ý nghĩa	Kiểu dữ liệu
1	yr	Năm xảy ra	Int
2	mn	Tháng xảy ra	Int
3	dy	Ngày của tháng xảy ra	Int
4	date	Ngày xảy ra	Date
5	st	Bang nơi cơn lốc bắt nguồn	Varchar
6	mag	Độ lớn đo trên thang F/EF (-9: lốc xoáy yếu), ( F0: nhẹ nhất),( F1: trung bình), (F2: đáng kể),(F3: nghiêm trọng),(F4: hủy diệt),(F5: siêu hủy diệt)	Int
7	inj	Số lượng người bị thương	Int
8	fat	Số lượng người tử vong	Int
9	slat	Vĩ độ bắt đầu	Float
10	slon	Kinh độ bắt đầu	Float
11	elat	Vĩ độ kết thúc	Float
12	elon	Kinh độ kết thúc	Float
13	len	Chiều dài cơn lốc (tính bằng dặm)	Float
14	wid	Chiều rộng cơn lốc (tính bằng thước)	Int

### 3. Mô tả thuộc tính bảng Fact, các bảng Dimension

#### 3.1. Lược đồ kho dữ liệu



### 3.2. Bảng Fact\_Tornado

Khóa	Tên thuộc tính	Kiểu dữ liệu	Mô tả
PK	id	Int	Mã thứ tự
FK	date_id	Date	Mã ngày
FK	location_id	Int	Mã vị trí
FK	size_id	Int	Mã kích thước
	inj	Int	Số lượng người bị thương
	fat	Int	Số lượng người tử vong

### 3.3. Bảng Dim\_Date

Khóa	Tên thuộc tính	Kiểu dữ liệu	Mô tả
PK	date_id	Date	Mã ngày
	yr	Int	Năm xảy ra
	mn	Int	Tháng xảy ra
	dy	Int	Ngày nào của tháng xảy ra

### 3.4. Bảng Dim\_Location

Khóa	Tên thuộc tính	Kiểu dữ liệu	Mô tả
PK	location_id	Int	Mã vị trí
	st	Varchar	Bang nơi cơn lốc bắt đầu
	slat	Float	Vĩ độ bắt đầu
	slon	Float	Kinh độ bắt đầu
	elat	Float	Vĩ độ kết thúc
	elon	Float	Kinh độ kết thúc

### 3.5. Bảng Dim\_Size

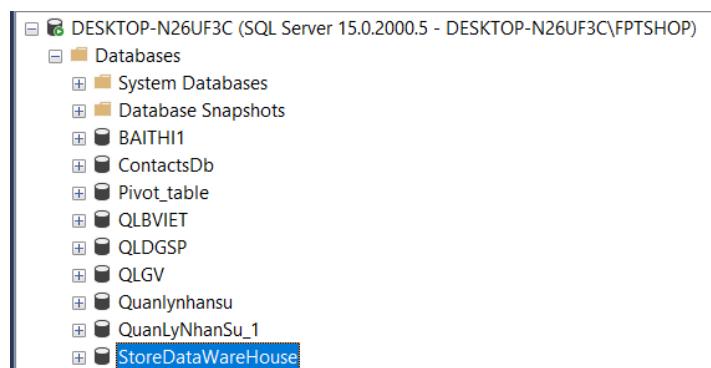
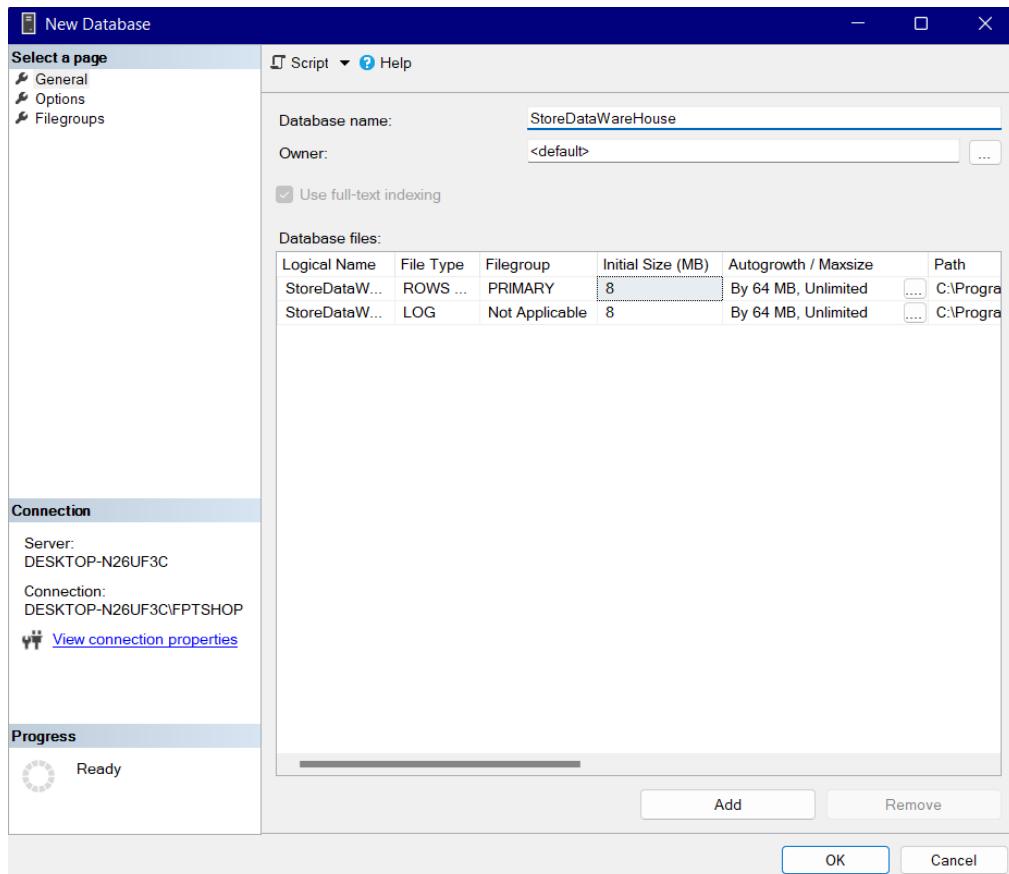
Khóa	Tên thuộc tính	Kiểu dữ liệu	Mô tả
PK	size_id	Int	Mã kích thước
	mag	Int	Độ lớn đo trên thang F/EF
	len	Float	Chiều dài cơn lốc (tính bằng dặm)
	wid	Int	Chiều rộng cơn lốc (tính bằng thước)

## CHƯƠNG II: XÂY DỰNG KHO DỮ LIỆU – QUÁ TRÌNH SSIS

### 1. Tạo cơ sở dữ liệu và import file .csv vào SQL Server

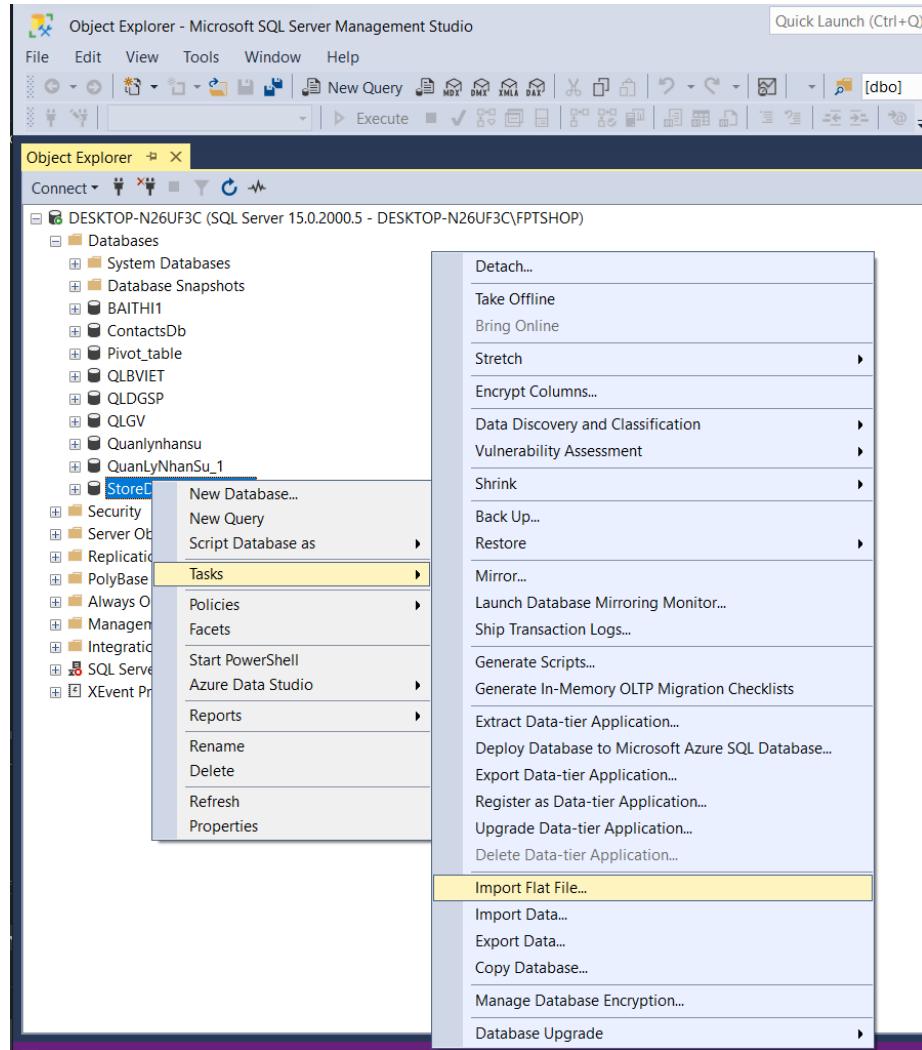
#### 1.1. Tạo mới một cơ sở dữ liệu

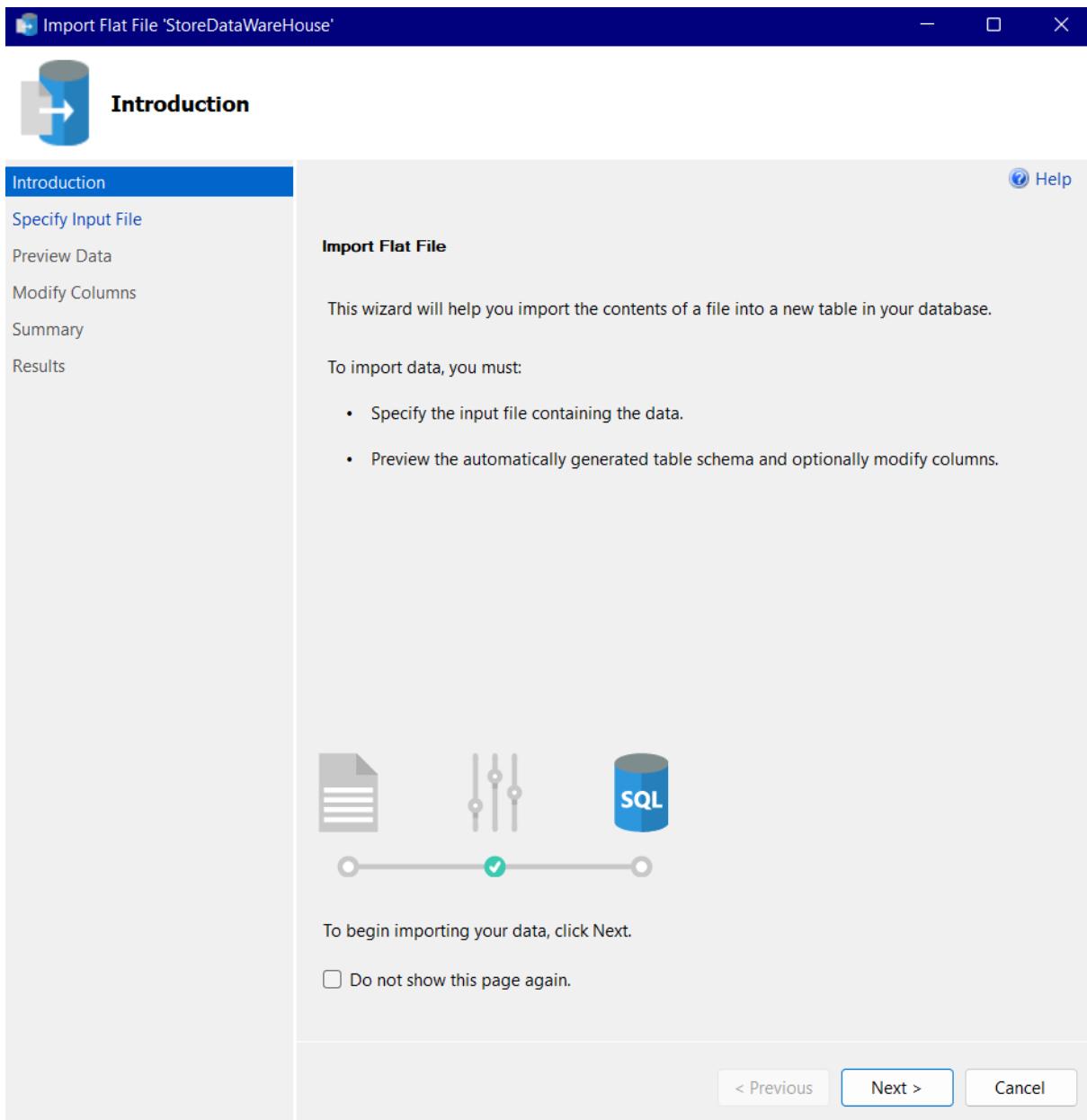
Thực hiện tạo một database trong SQL Server để lưu data với tên là StoreDataWareHouse



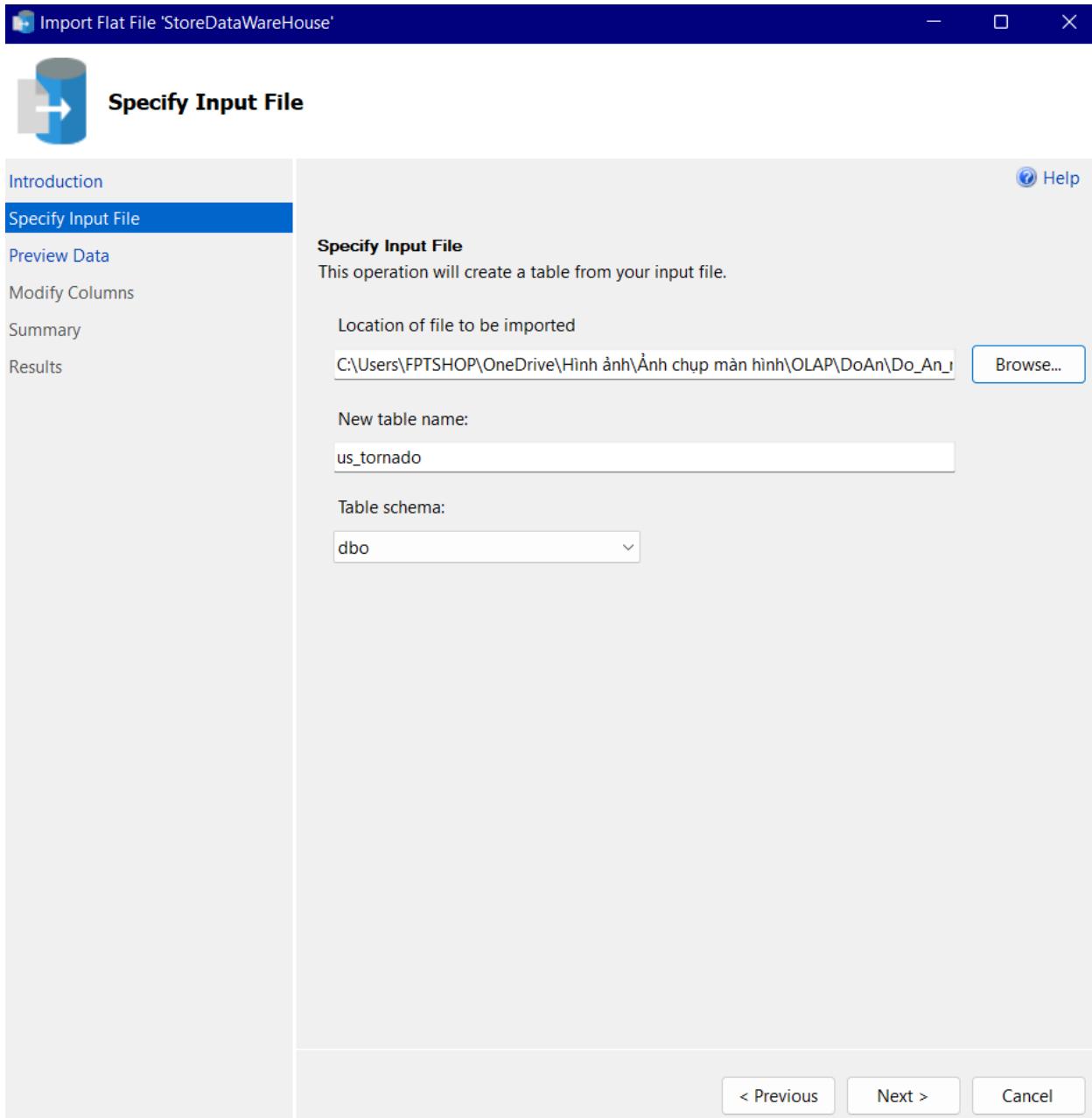
### 1.2. Quá trình đưa file .csv vào SQL Server

- Trong database chọn StoreDataWareHouse, click chuột vào Task và chọn Import Flat File

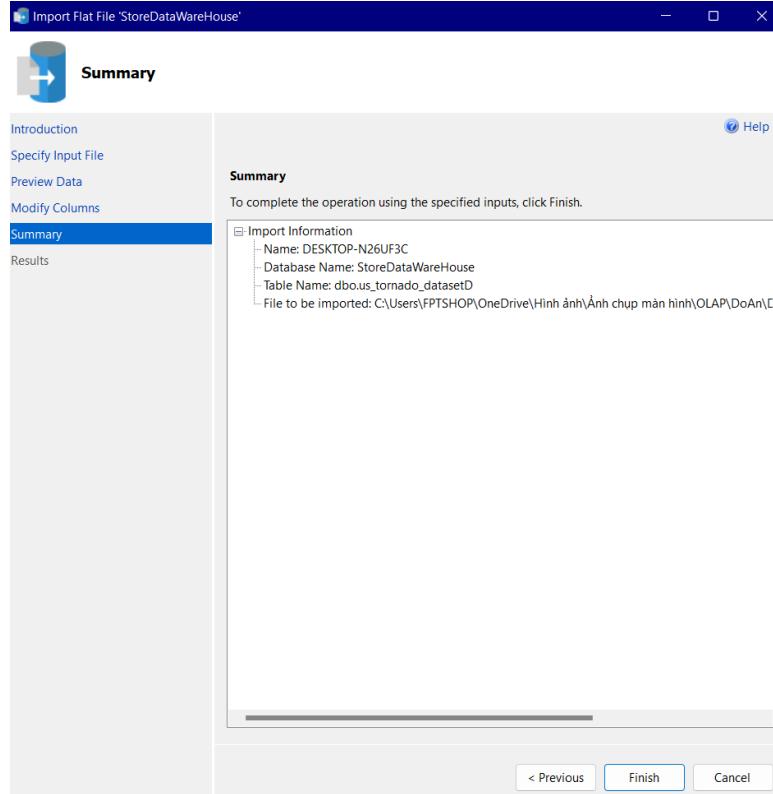




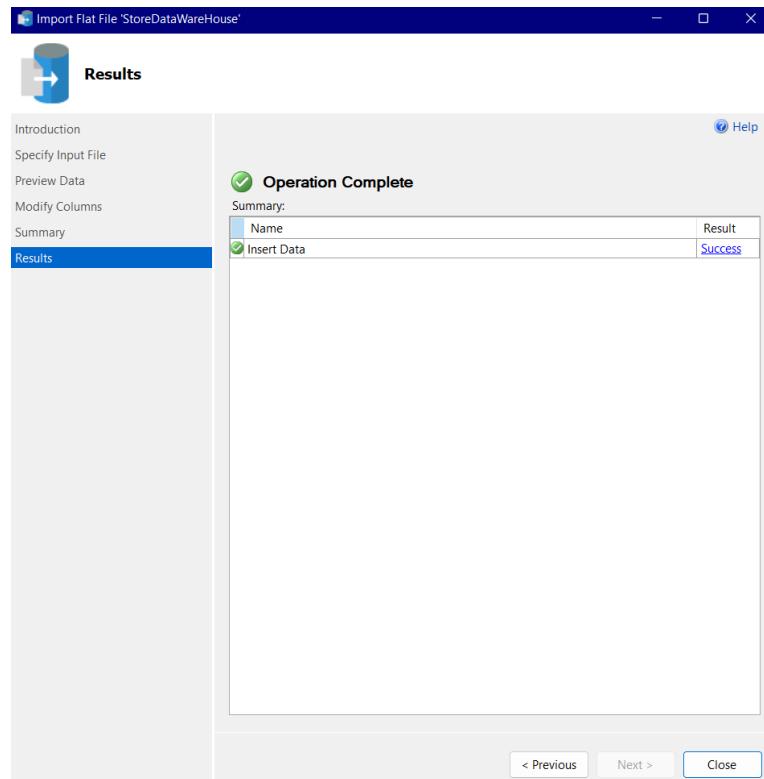
- Chọn Next -> Chọn Browse -> Tìm và chọn file .csv muốn đưa dữ liệu vào -> Chọn Next



- Chọn Finish để hoàn thành Import Flat File

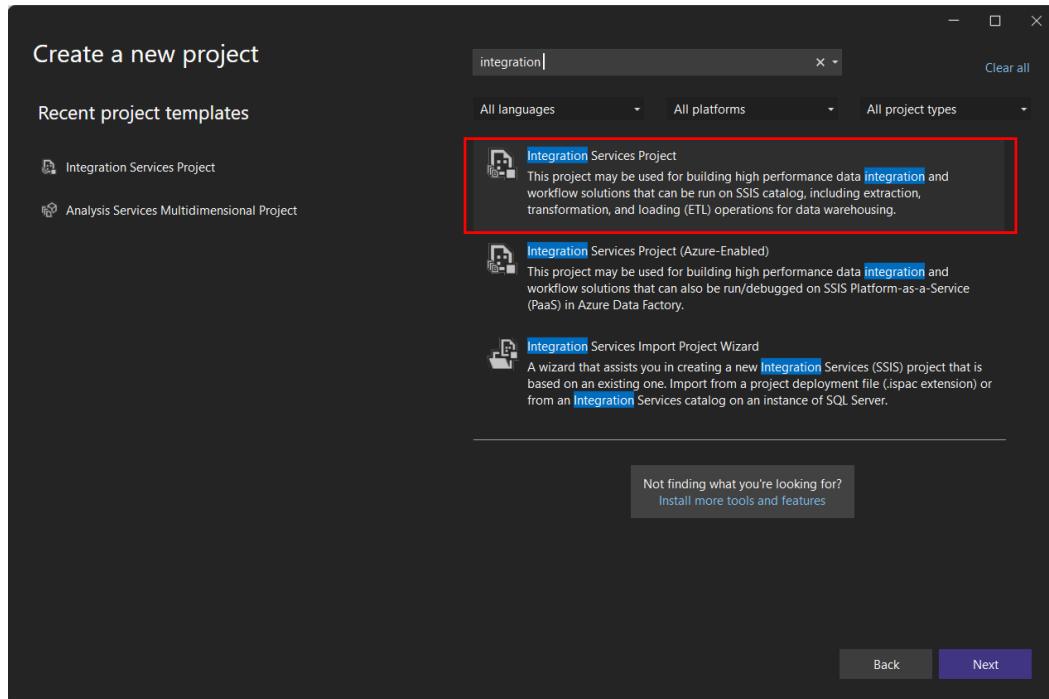


- Import dữ liệu thành công vào SQL Server

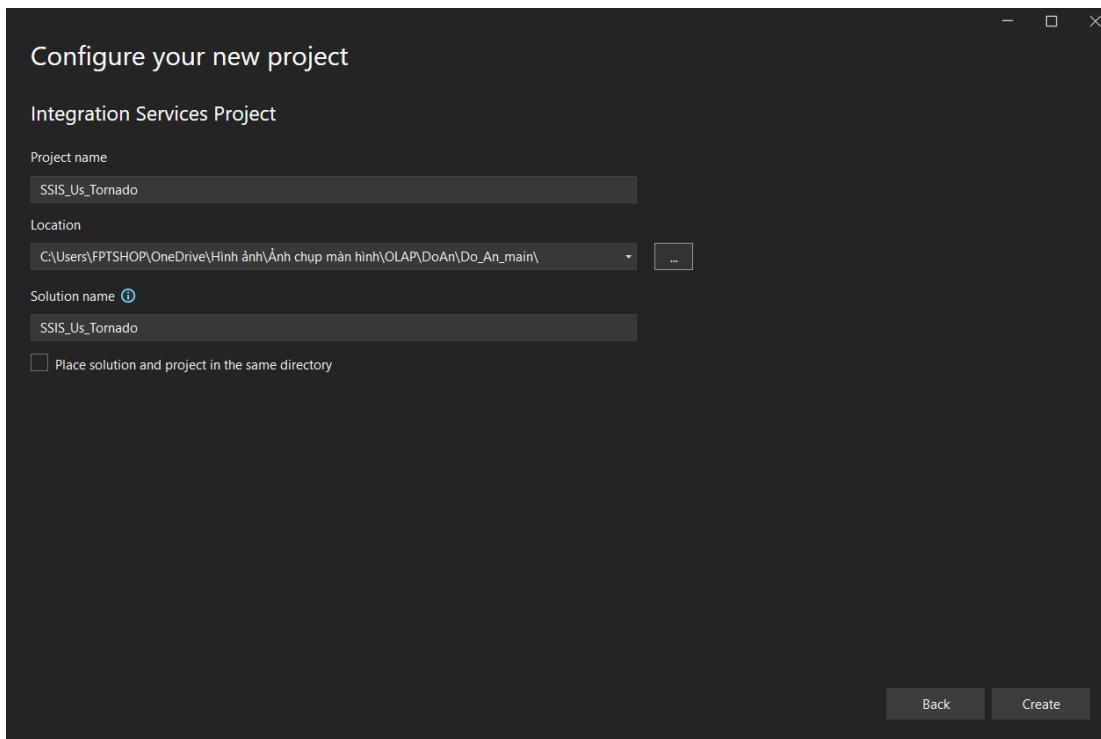


## 2. Tạo project mới và thiết lập kết nối

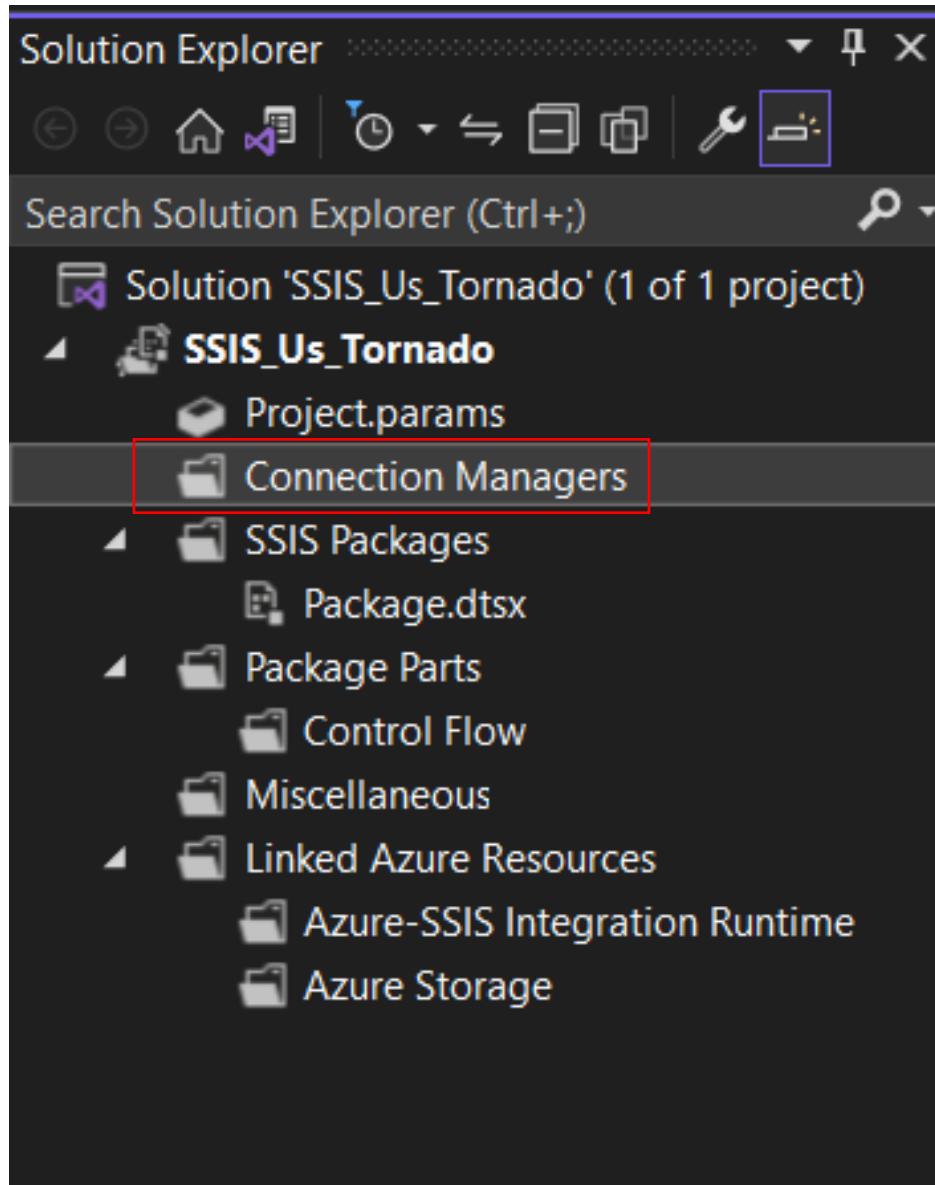
- Mở Visual -> chọn Create a new project -> chọn Integration Service Project



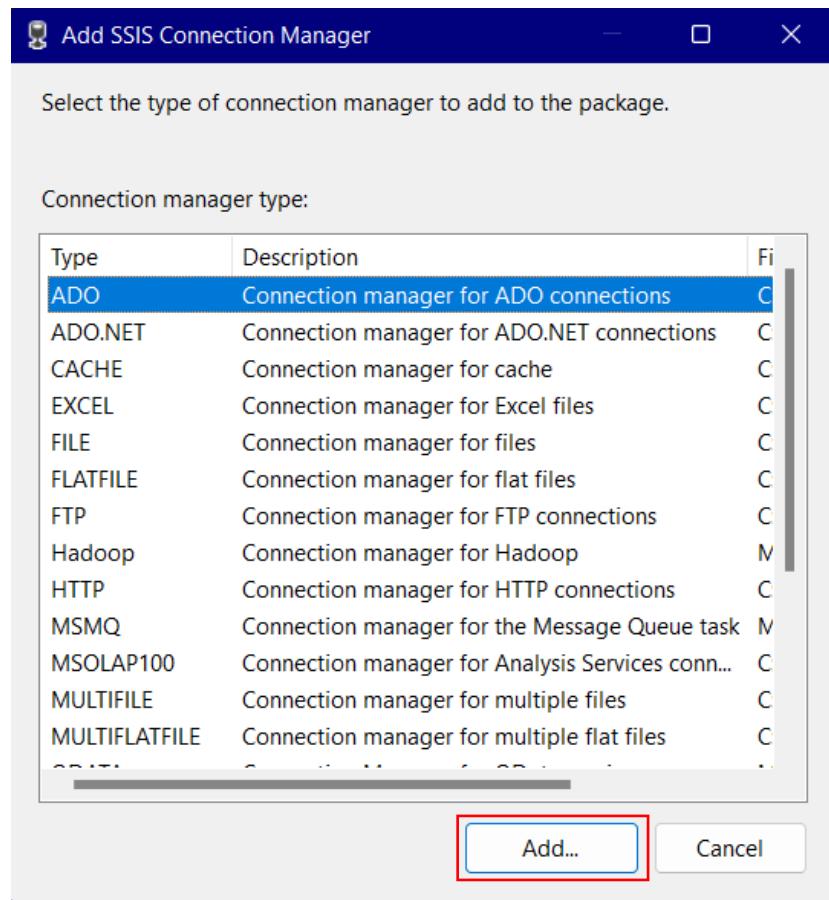
- Đặt tên file -> chọn nơi lưu trữ -> chọn Create



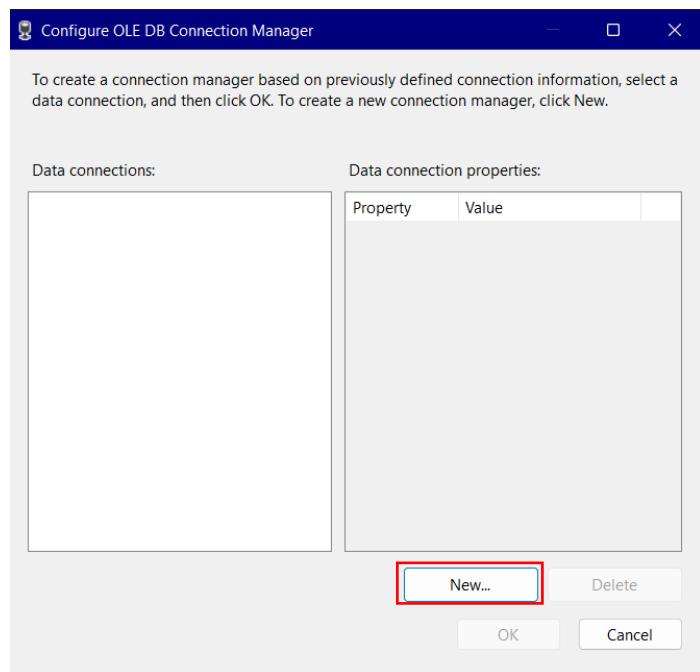
- Tại cửa sổ Solution Explorer -> chọn Connection Managers -> click chuột phải chọn New Connection Managers



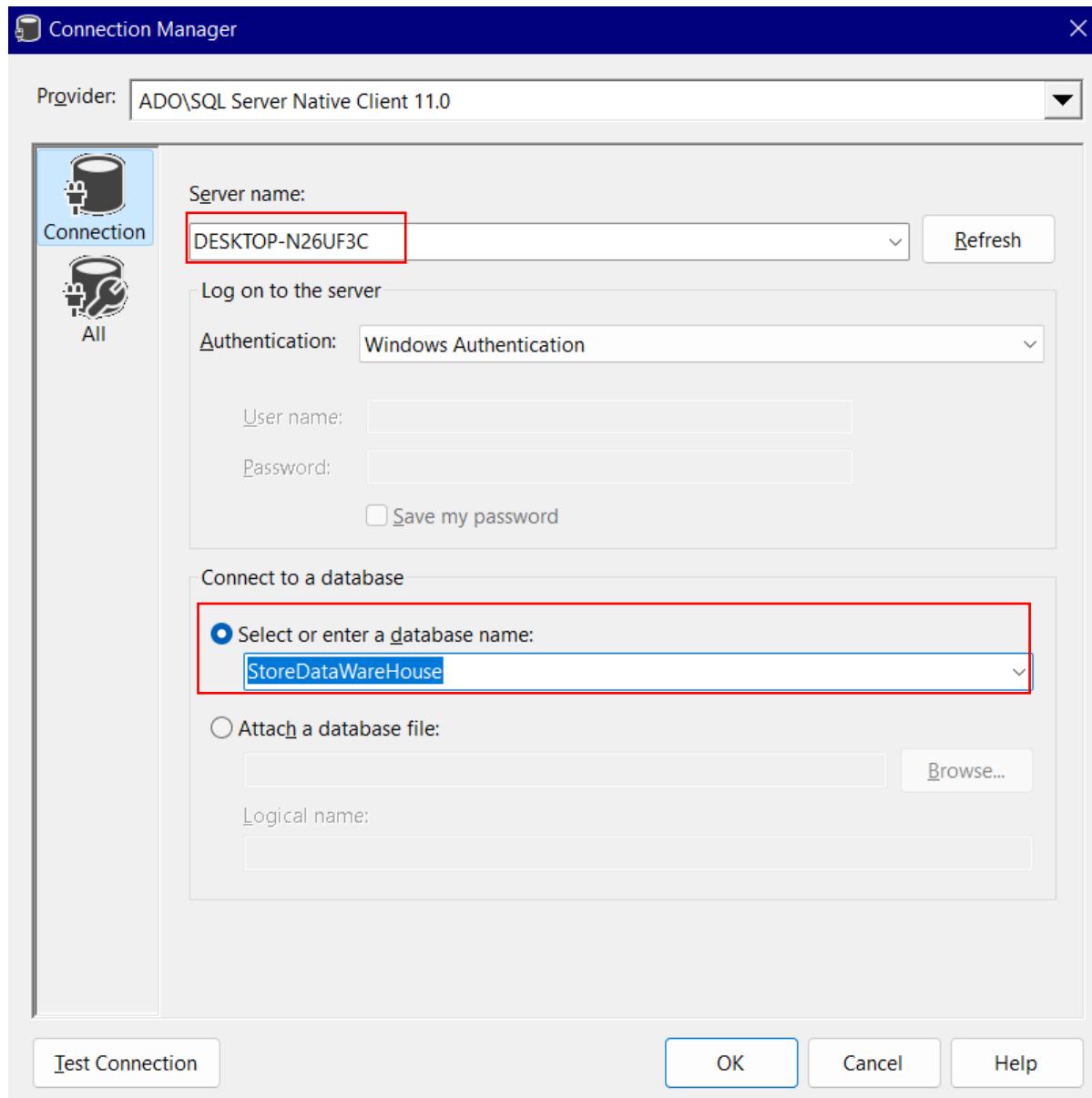
- Chọn Add để tạo kết nối mới tới SQL Server



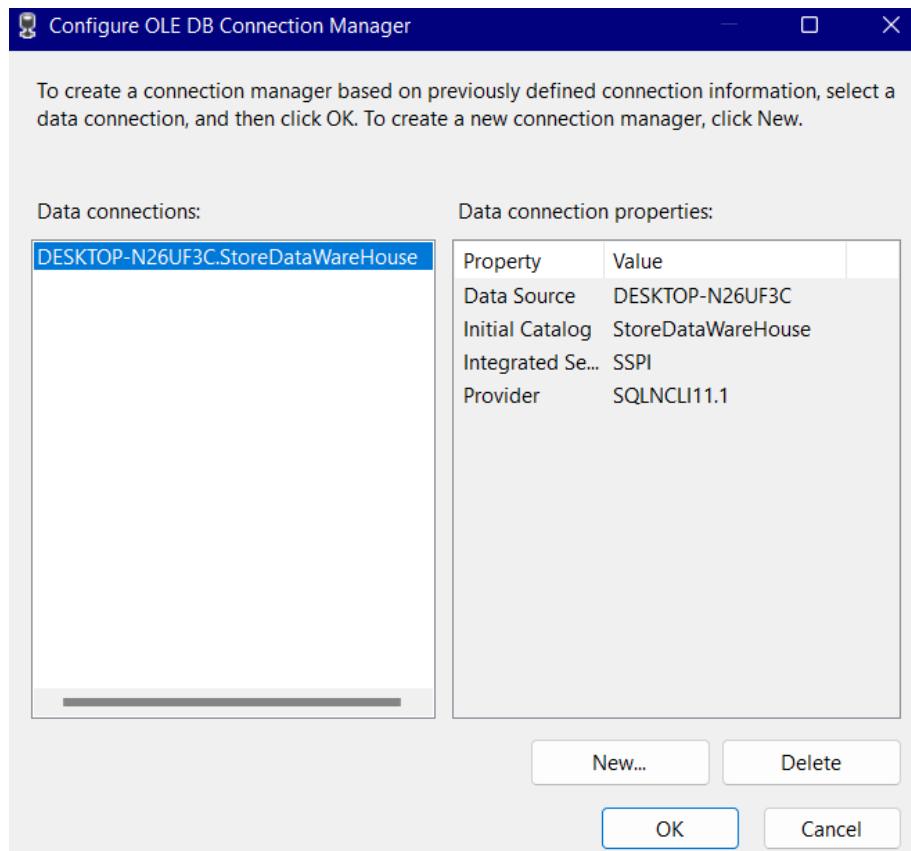
- Chọn New



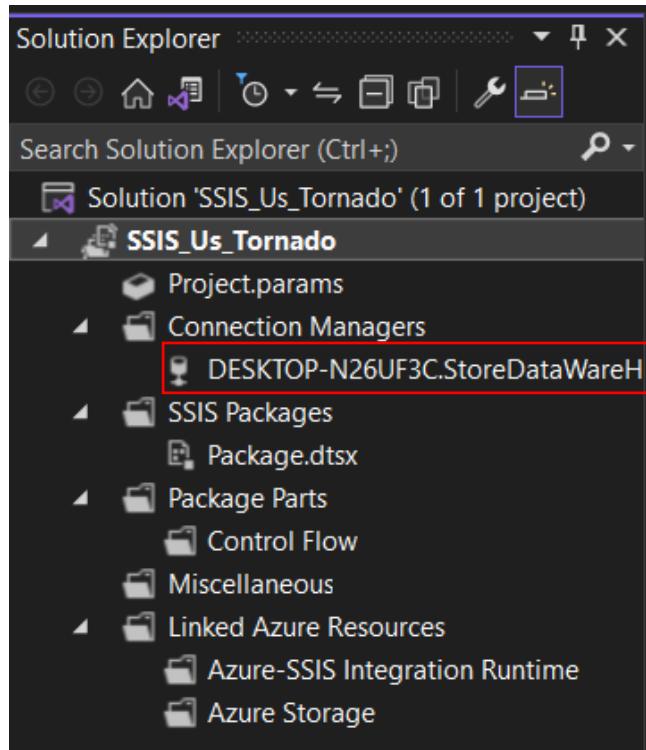
- Ở mục Server name điền tên Server name của SQL Server -> Chọn database StoreDataWareHouse ở mục “Select or enter a database name” -> Chọn OK



- Chọn OK

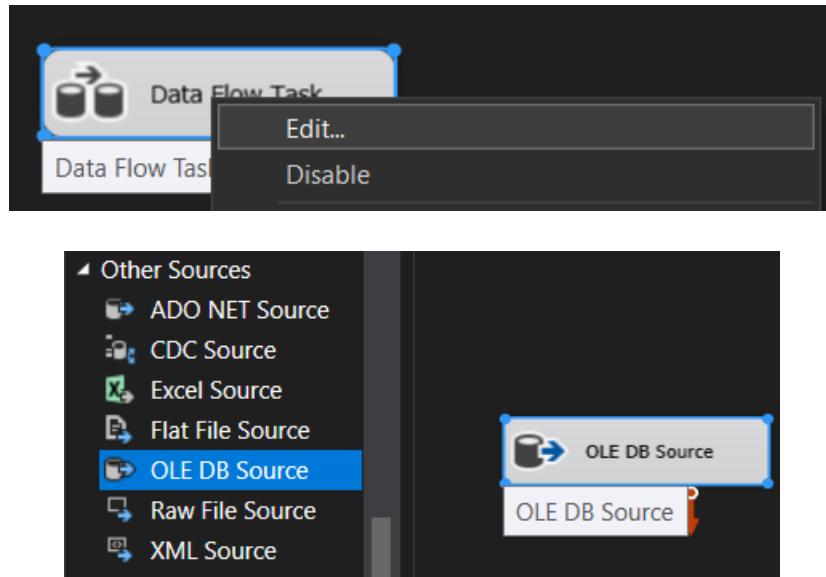


- Kết nối thành công tới Database StoreDataWareHouse

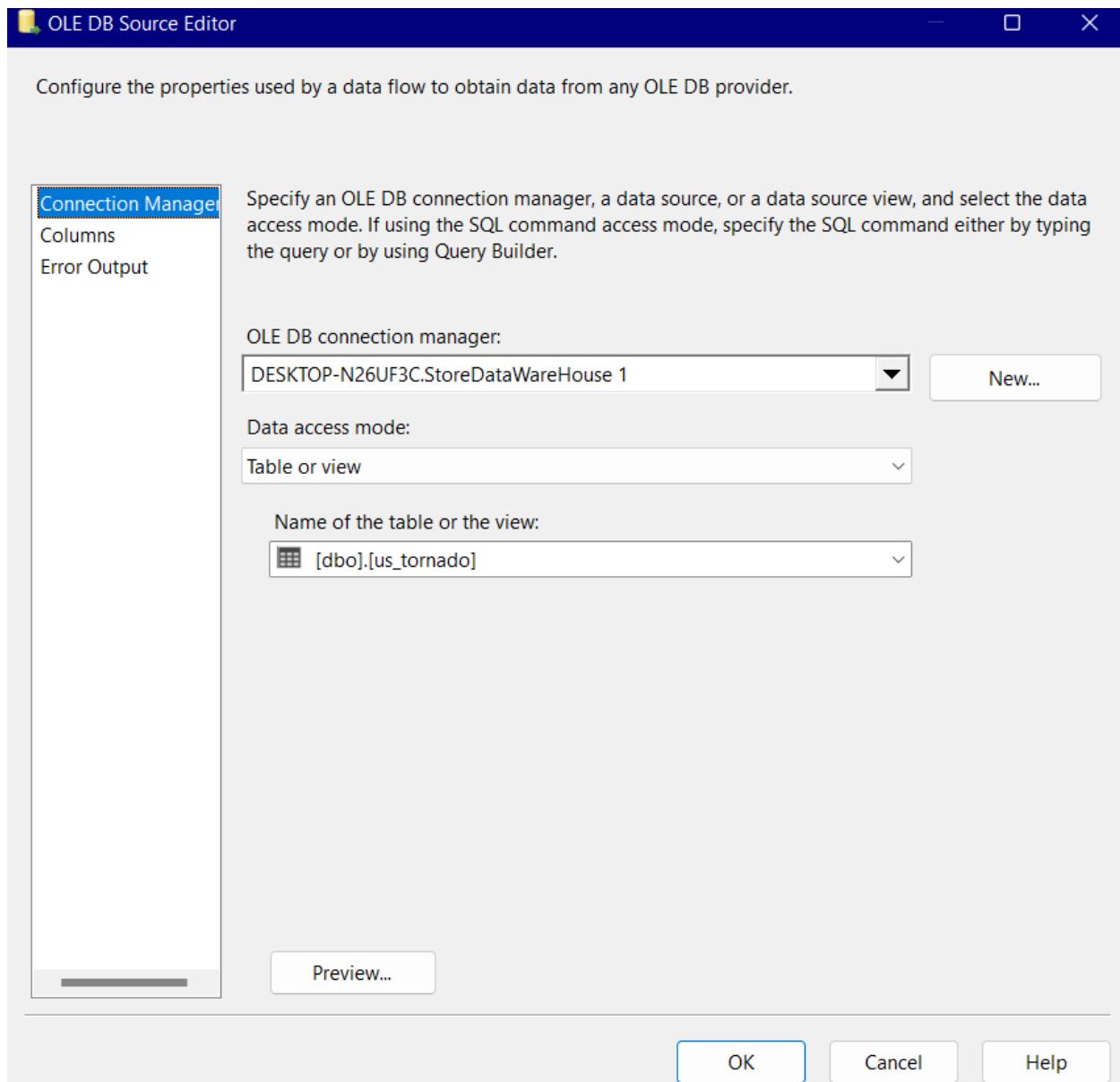


### 3. Quá trình đổ dữ liệu từ file Excel vào Database của các bảng Dim

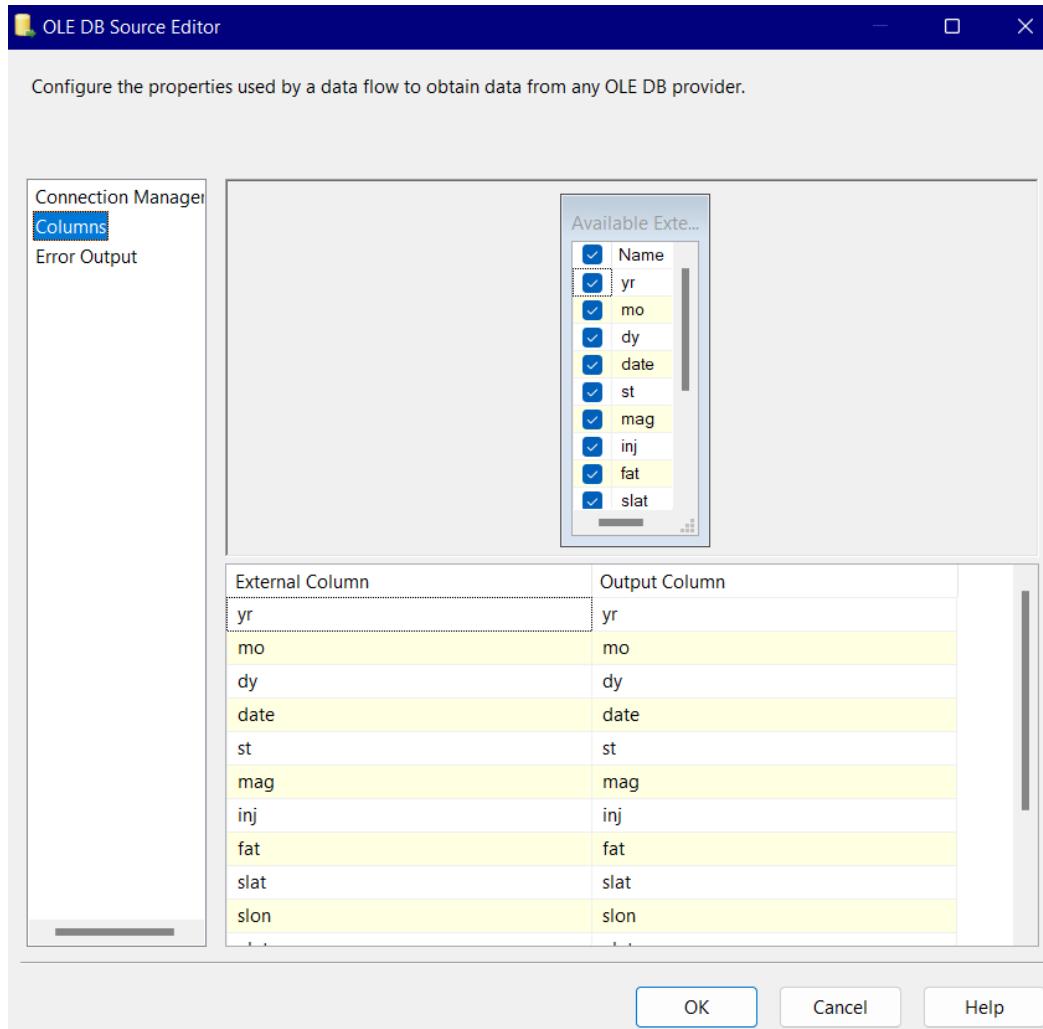
- Data Flow Task
  - Kéo thả Data Flow Task -> Chuột phải chọn Edit -> Kéo thả OLE DB Source



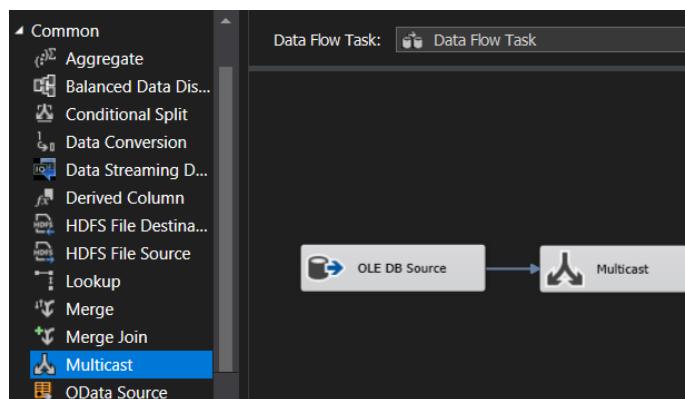
- Chuột phải OLE DB Source chọn Edit -> OLE DB connection manager chọn database StoreDataWareHouse -> Ở phần “Name of the table or the view” chọn bảng “us\_tornado” đã import file ở SQL trước đó



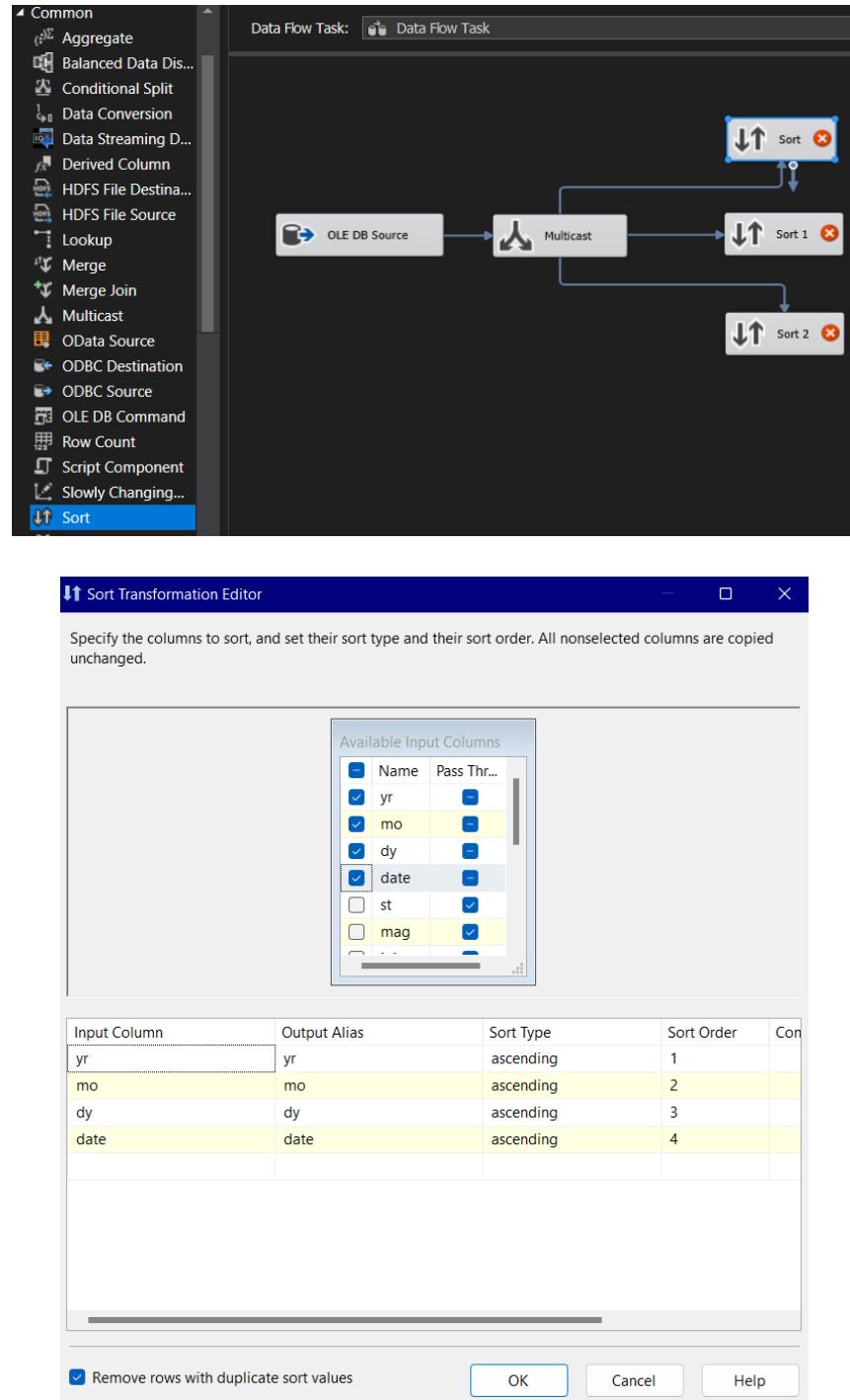
- Kiểm tra lại từng cột ở mục Columns -> OK -> Hoàn tất quá trình kết nối



- Thêm Multicast để đồ dữ liệu theo nhiều luồng

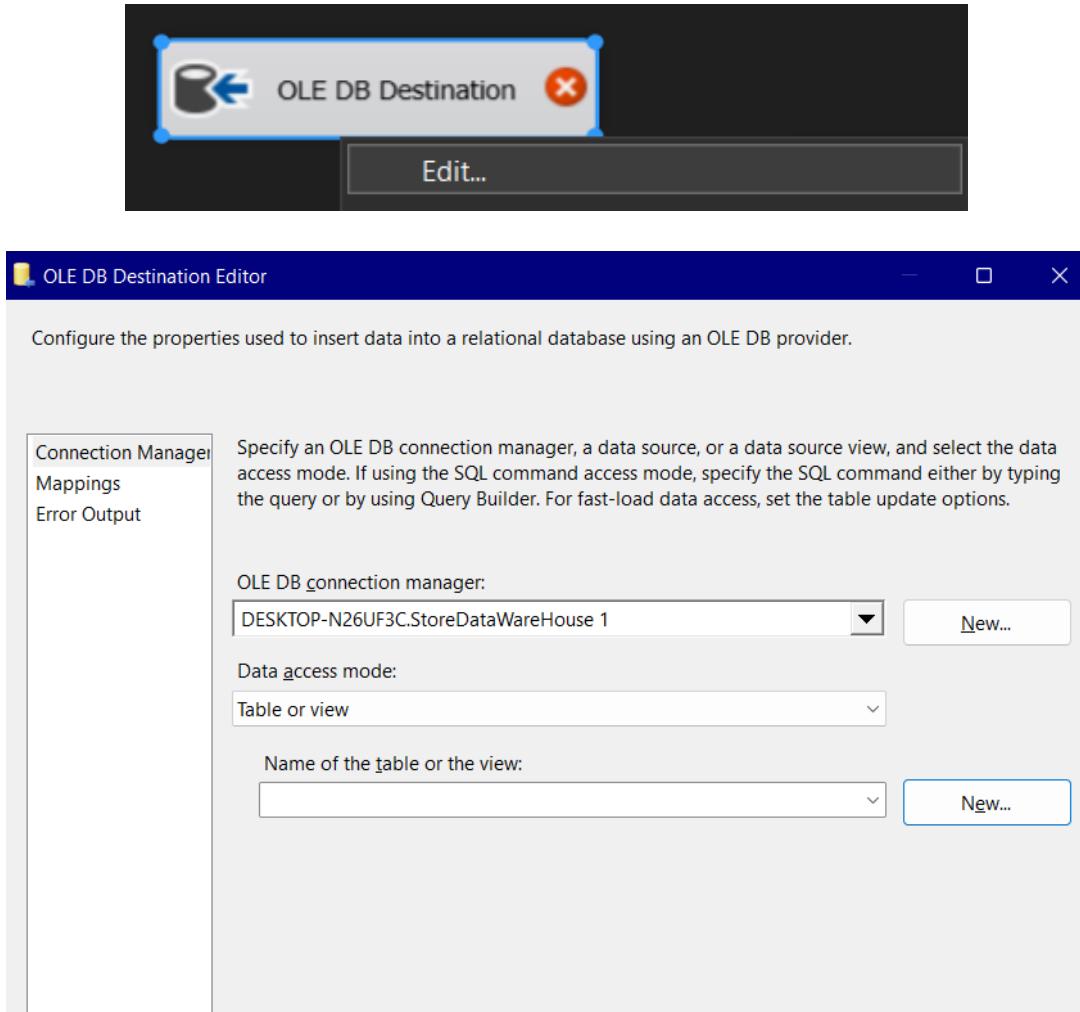


- Thêm Sort để có thể loại bỏ dữ liệu trùng lặp. Có 3 bảng Dim nên sẽ thêm 3 Sort

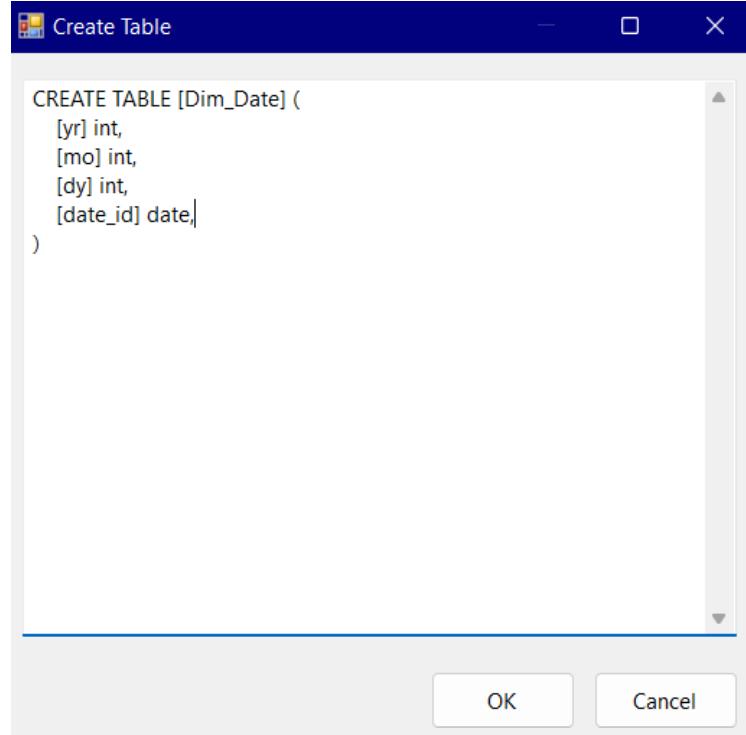


( Ví dụ Sort bảng Dim\_Date)

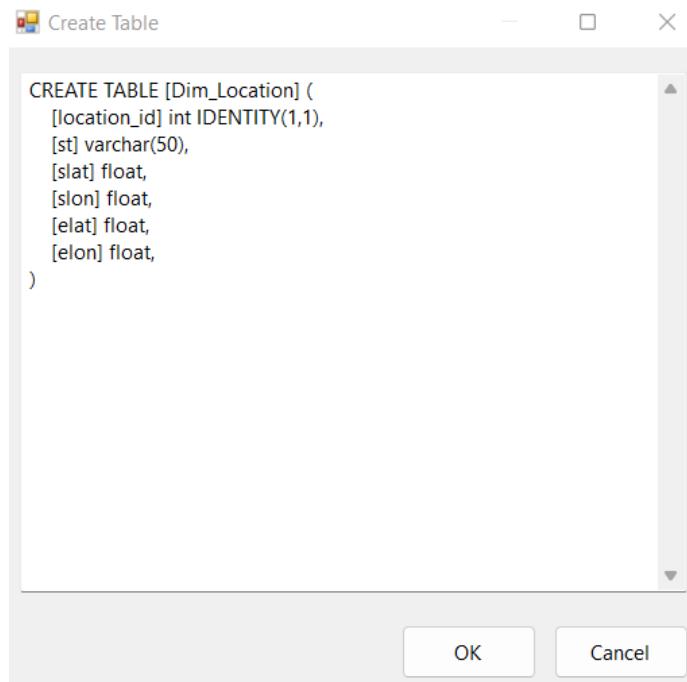
- Thêm OLE DB Destination để đổ dữ liệu ra các bảng Dim
- Chuột phải vào OLE DB chọn Edit -> Chọn New ở mục OLE DB connection -> Chọn Data connection là StoreDataWareHouse



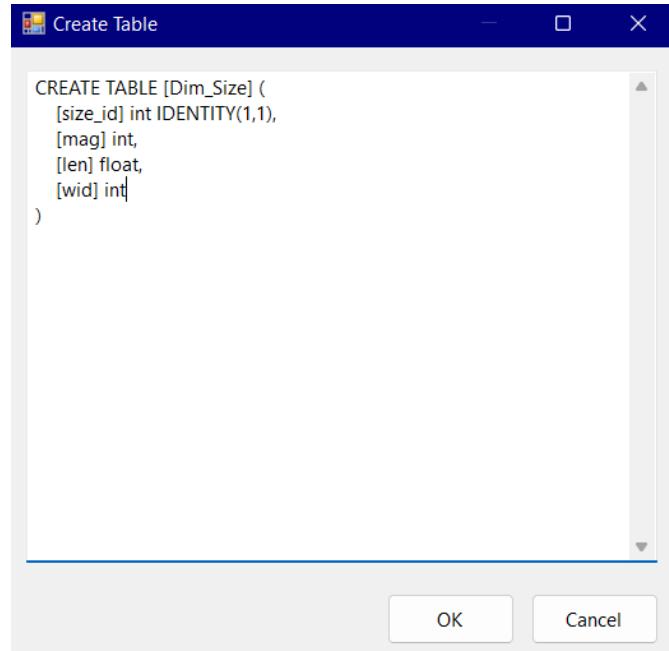
- Chọn New ở mục “Name of the table or the view”, giữ lại những thuộc tính cần thiết cho bảng còn lại thì xóa đi
- **Bảng Dim\_Date:** Giữ lại thuộc tính “yr” “mo” “dy” “date” và thuộc tính “date” chuyển thành “date\_id” để lập khóa ngoại



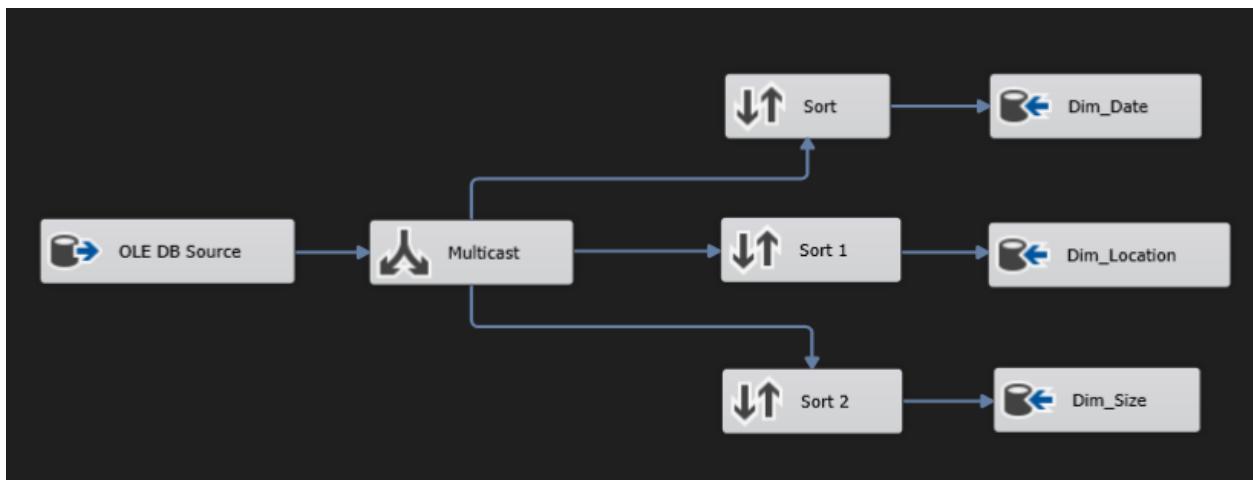
- **Bảng Dim\_Location:** Giữ lại thuộc tính “st” “slat” “slon” “elat” “elon” và thêm thuộc tính location\_id để lập khóa ngoại



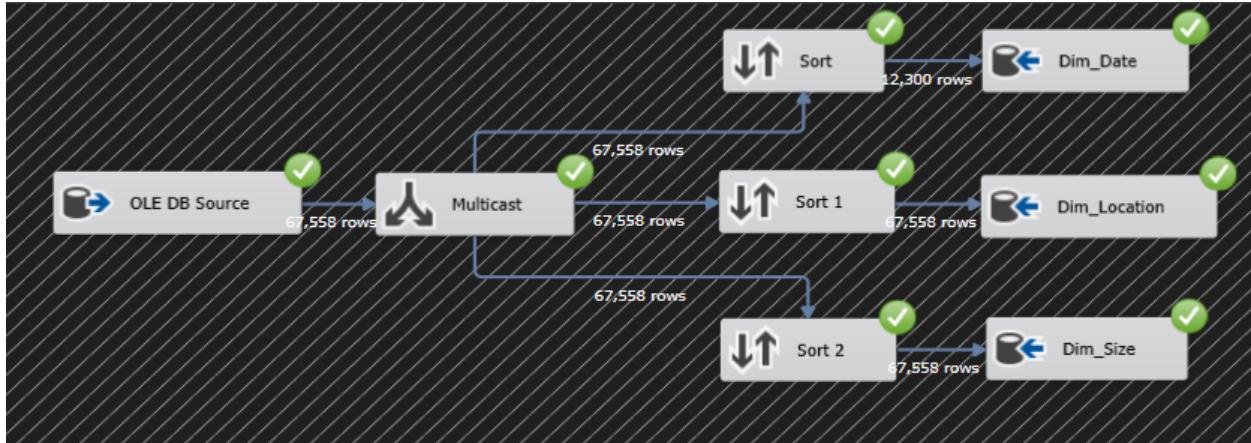
- **Bảng Dim\_Size:** Giữ lại thuộc tính “mag” “len” “wid” và thêm thuộc tính size\_id để lập khóa ngoại



- Nối OLE DB Source với Multicast, nối Multicast với Sort và nối Sort với OLE DB Destination



- Run project để đổ dữ liệu từ file Excel vào Database

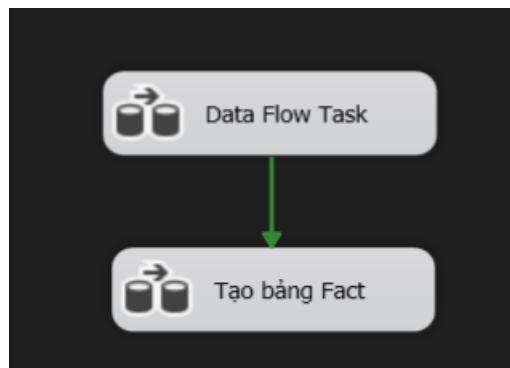


(Đỗ dữ liệu thành công)

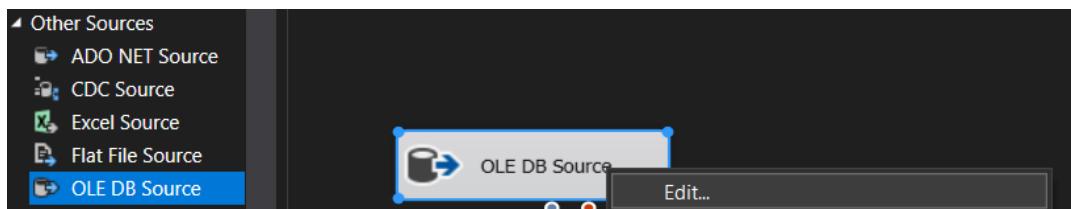
#### 4. Quá trình đỗ dữ liệu từ file Excel vào Database của bảng Fact

- Data Flow Task

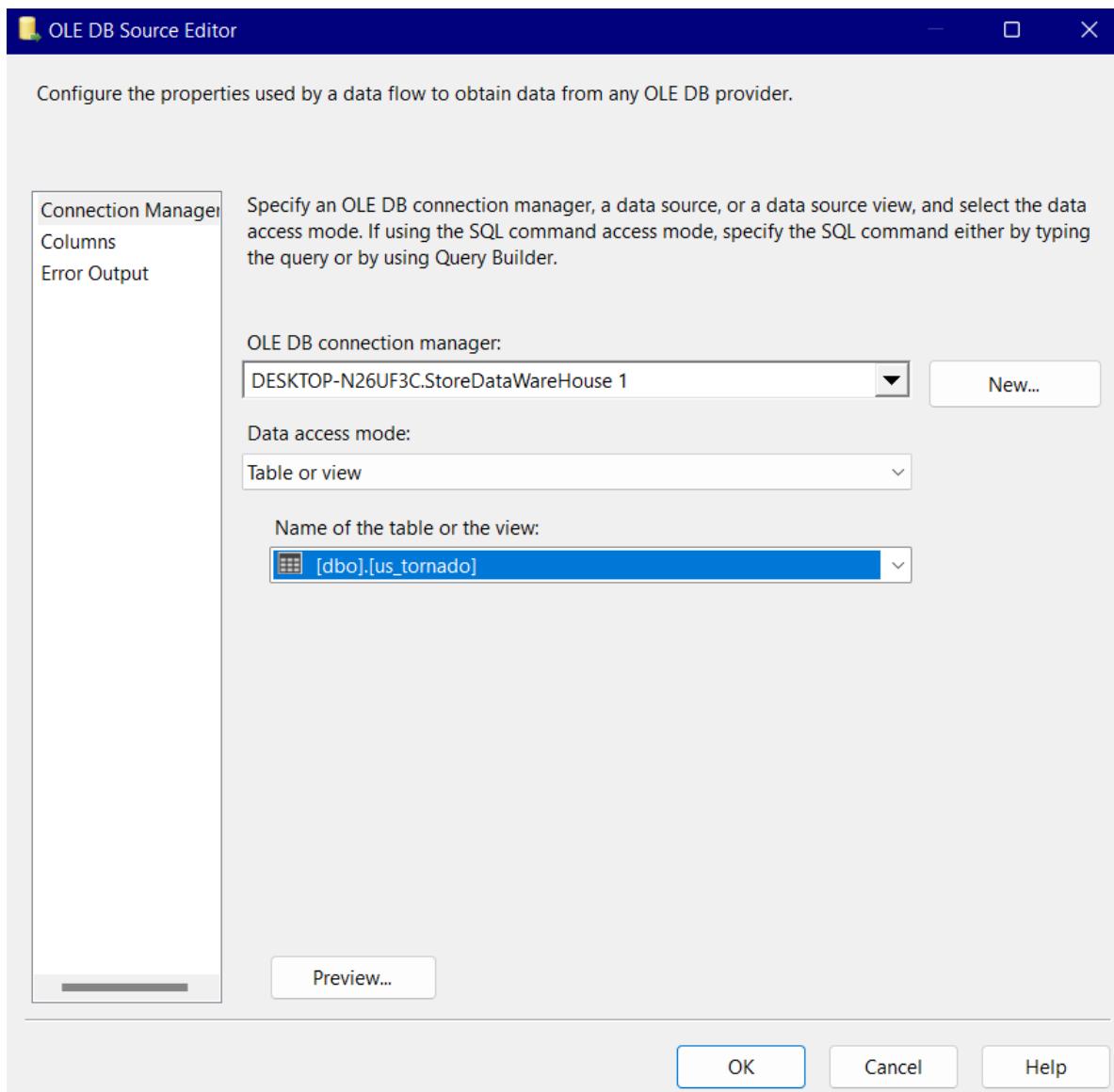
- Kéo thả Data Flow Task “Tạo bảng Fact” -> Nối Data Flow Task của Dim với Fact



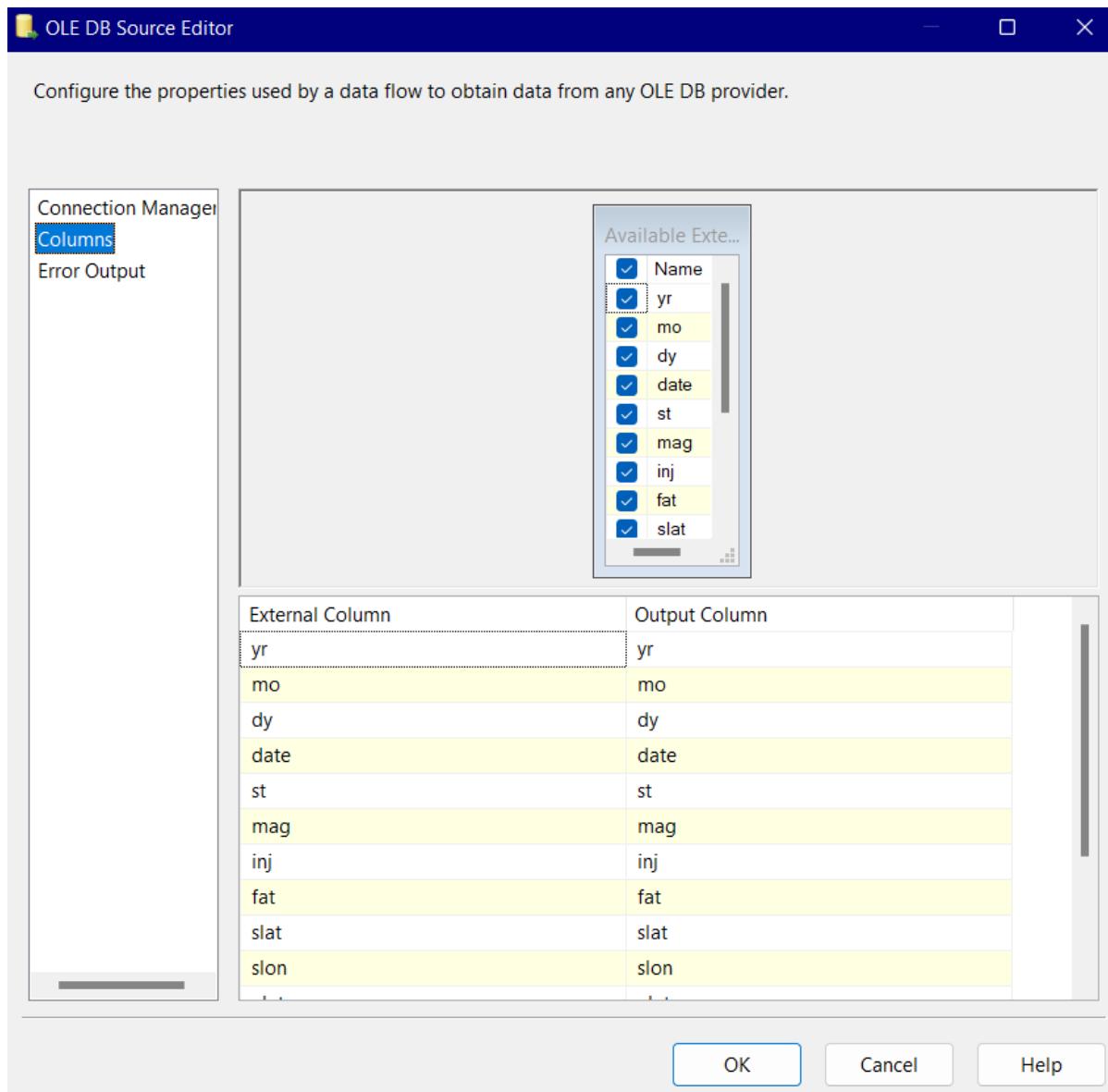
- Edit “Tạo bảng Fact” -> Kéo thả OLE DB Source -> Chuột phải chọn Edit



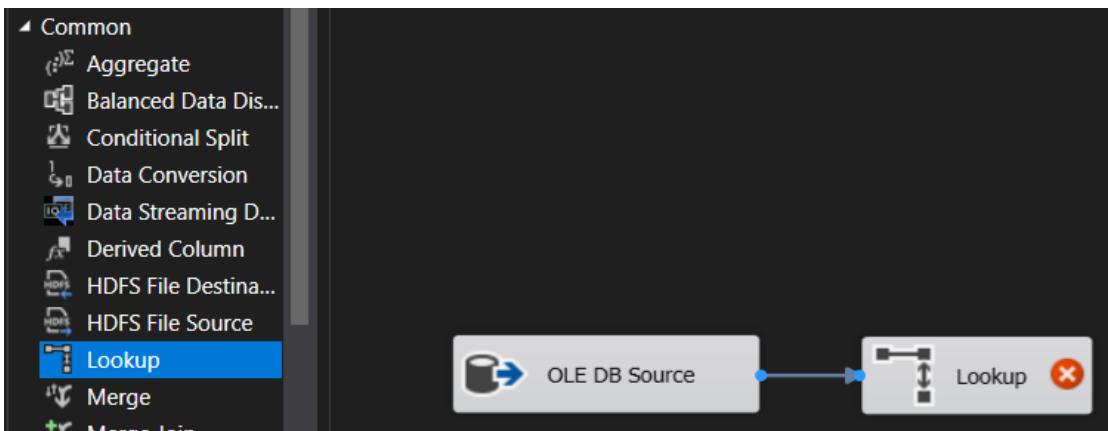
- Chọn kết nối tới database StoreDataWareHouse và chọn bảng us\_tornado



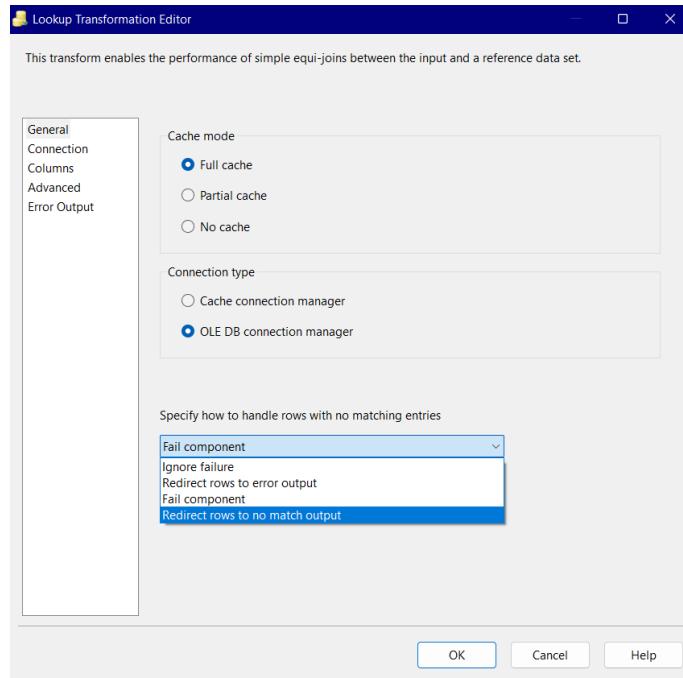
- Chọn các cột thuộc tính cần thiết -> OK



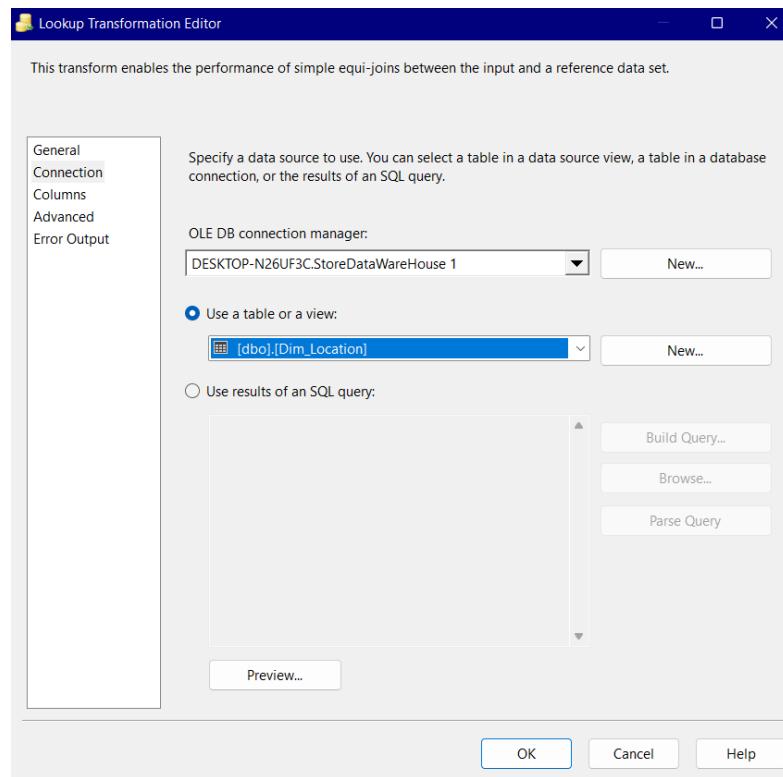
- Kéo thả và tạo mới Look Up các thuộc tính từ bảng Dim\_Location



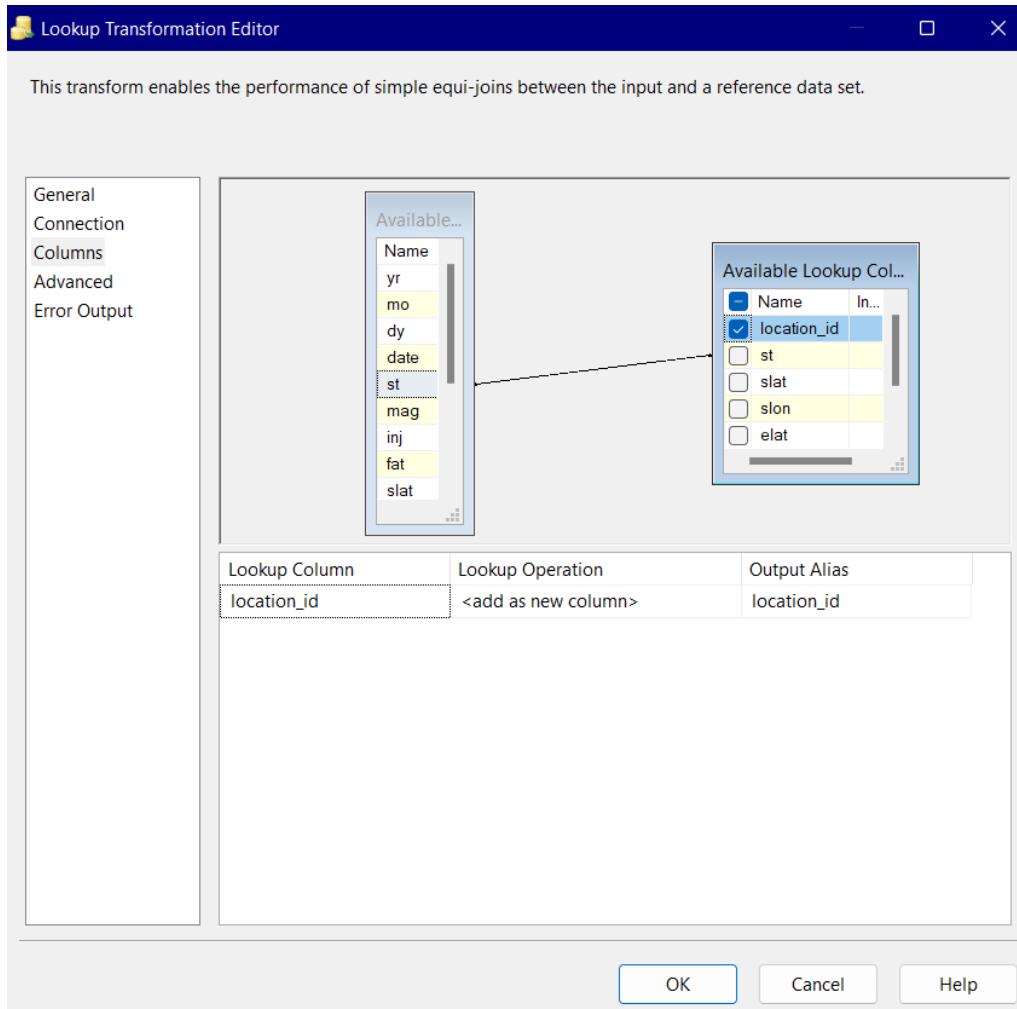
- Chuột phải chọn Edit -> Chọn “Redirect rows to no match output”



- Ở mục Connection chọn connection tới database StoreDataWareHouse và chọn bảng Dim\_Location

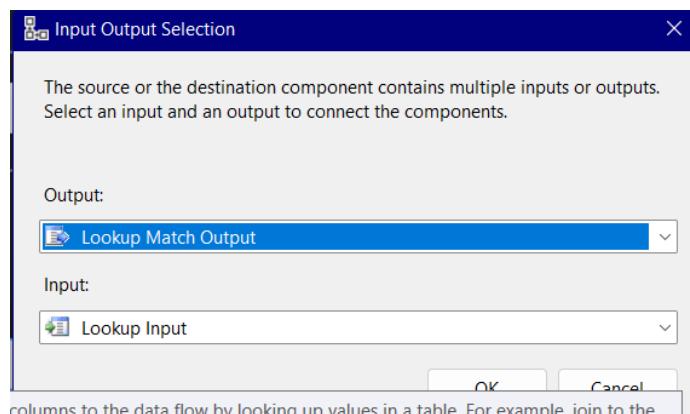


- Chọn thuộc tính để Look Up -> Chọn OK

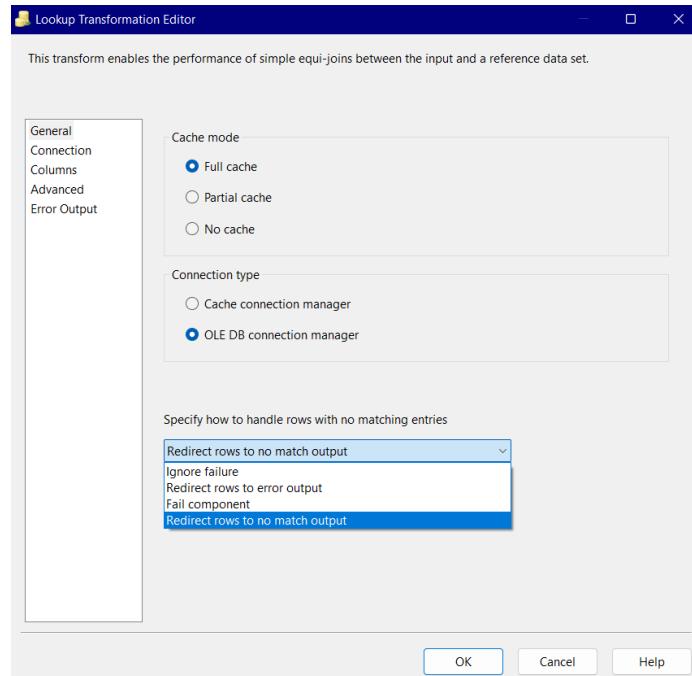


- Kéo thả và tạo mới Look Up các thuộc tính từ bảng Dim\_Size

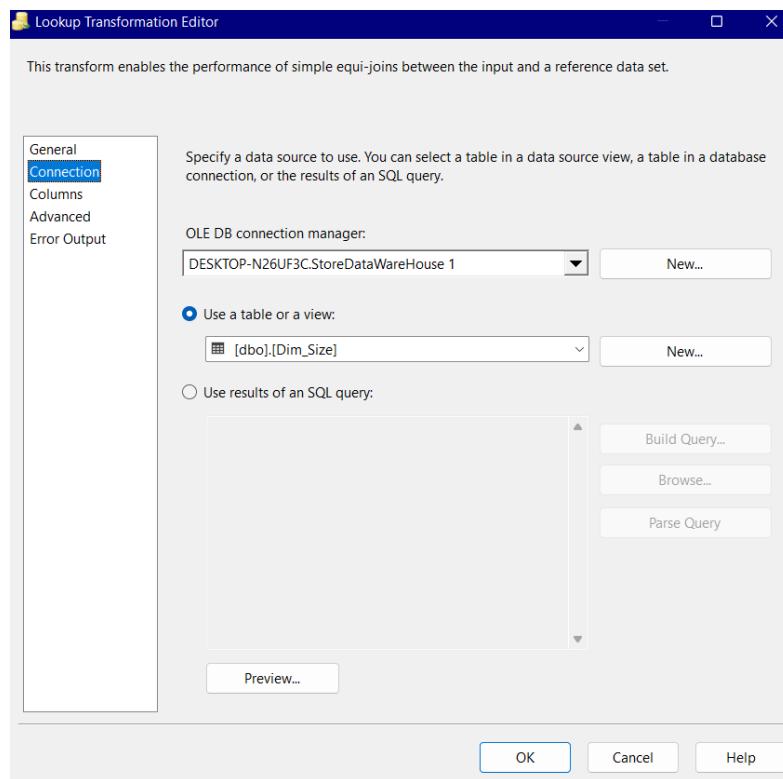
- Chọn Output là “Lookup Match Output” -> OK



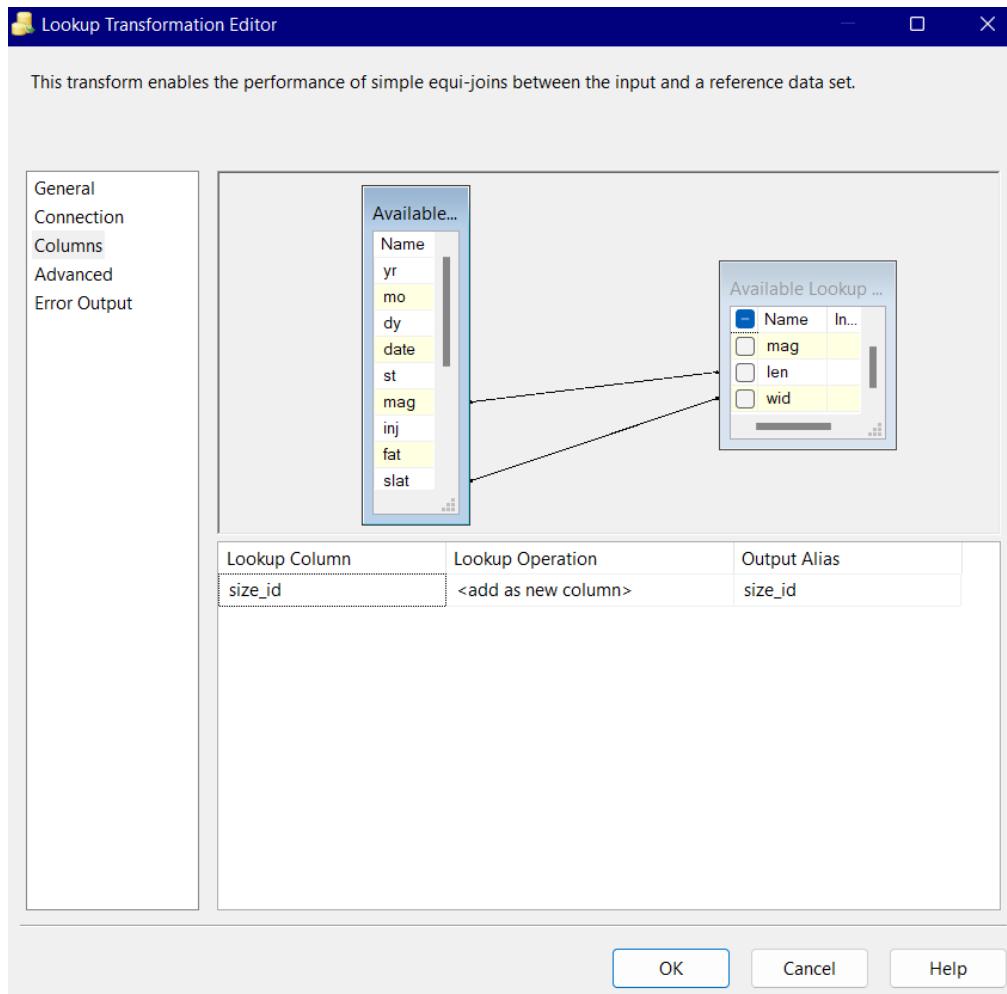
- Chuột phải chọn Edit -> Chọn “Redirect rows to no match output”



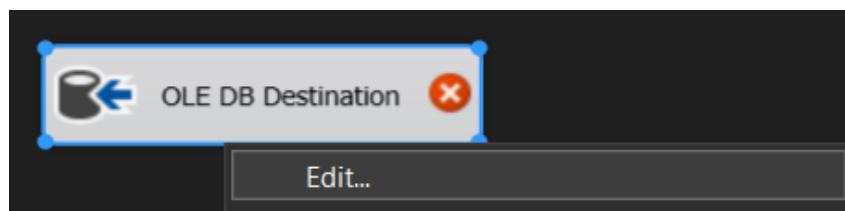
- Ở mục Connection chọn connection tới database StoreDataWareHouse và chọn bảng Dim\_Size

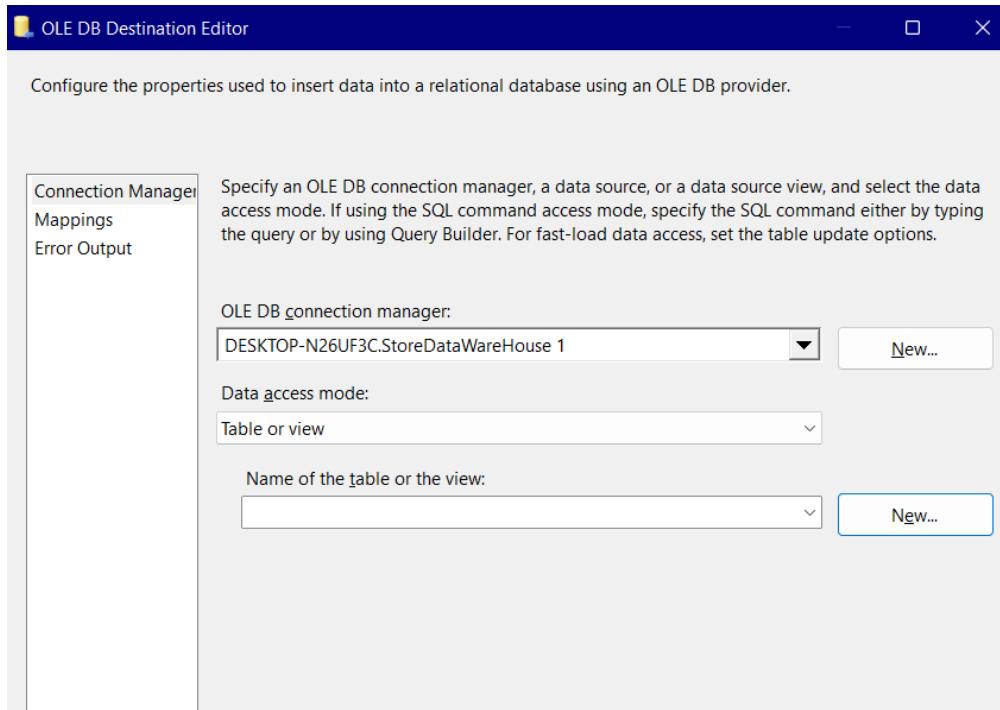


- Chọn thuộc tính để Look Up -> Chọn OK

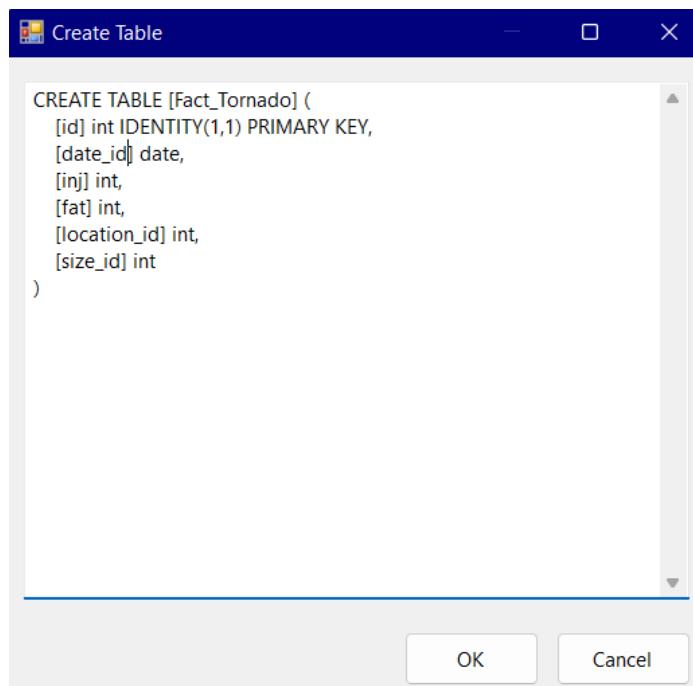


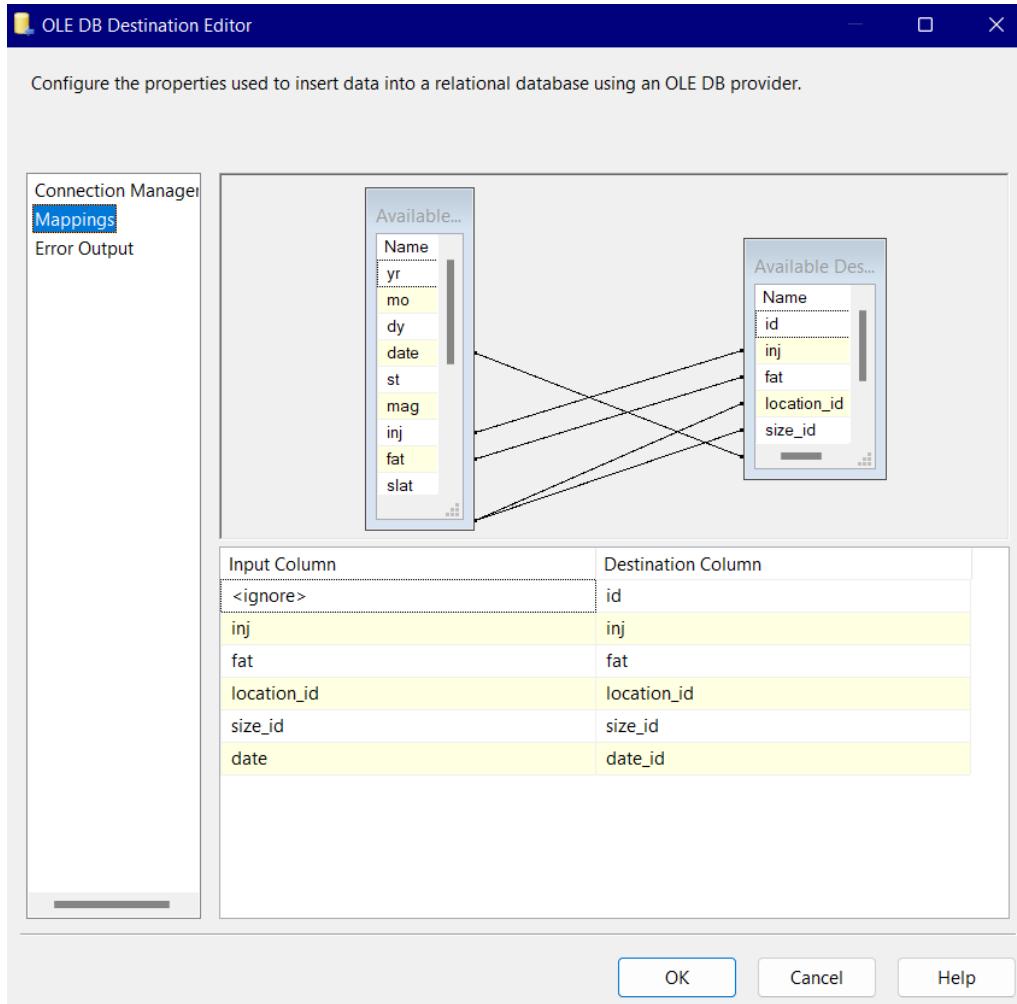
- Thêm OLE DB Destination để đổ dữ liệu ra bảng Fact
- Chuột phải vào OLE DB chọn Edit -> Chọn New ở mục OLE DB connection -> Chọn Data connection là StoreDataWareHouse





- Chọn New ở mục “Name of the table or the view”, giữ lại những thuộc tính cần thiết cho bảng còn lại thì xóa đi
- **Bảng Fact\_Tornado:** Giữ lại thuộc tính “date\_id” “inj” “fat” “location\_id” “size\_id” và thêm thuộc tính “id” để lập khóa chính cho bảng

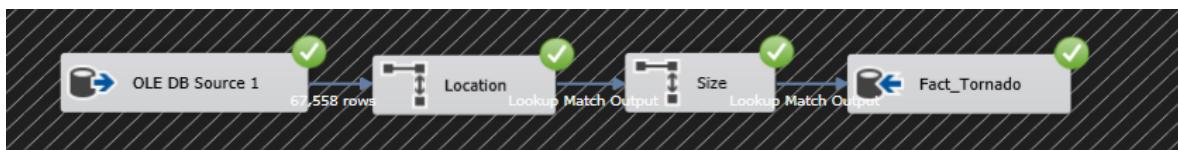




- Nối OLE DB Source với Lookup, nối Lookup với OLE DB Destination
- Sau khi thêm connection ta được như hình

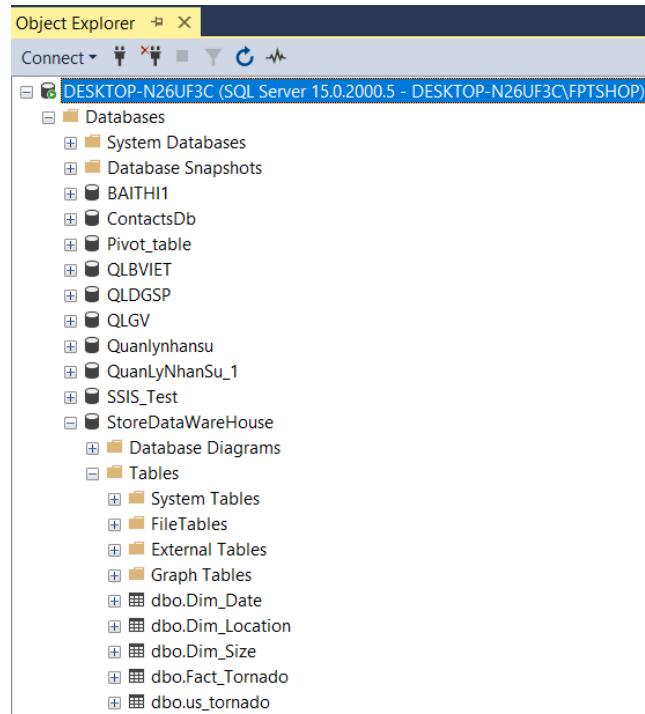


- Run project để đỗ dữ liệu từ file Excel vào Database



(Đỗ dữ liệu thành công)

- Sau khi đã xử lý thành công, qua SQL Server check lại dữ liệu



- Bảng Dim\_Date

SQLQuery6.sql - D...UF3C\FPTSHOP (63) Object Explorer

```
***** Script for SelectTopNRows command from 'Dim_Date' *****
SELECT TOP (1000) [yr]
      ,[mo]
      ,[dy]
      ,[date_id]
  FROM [StoreDataWareHouse].[dbo].[Dim_Date]
```

110 %

	yr	mo	dy	date_id
1	1950	1	13	1950-01-13
2	1950	1	25	1950-01-25
3	1950	1	26	1950-01-26
4	1950	1	3	1950-01-03
5	1950	10	1	1950-10-01
6	1950	10	9	1950-10-09
7	1950	11	20	1950-11-20
8	1950	11	4	1950-11-04
9	1950	12	2	1950-12-02
10	1950	2	11	1950-02-11
11	1950	2	12	1950-02-12
12	1950	2	13	1950-02-13

- Bảng Dim\_Location

```

SQLQuery5.sql - D...UF3C\FPTSHOP (63)  X SQLQuery4.sql - D...UF3C\FPTSHOP (65)*
***** Script for SelectTopNRows command from SSMS *****/
SELECT TOP (1000) [location_id]
      ,[st]
      ,[slat]
      ,[slon]
      ,[elat]
      ,[elon]
  FROM [StoreDataWareHouse].[dbo].[Dim_Location]

110 % ▾
Results Messages

```

	location_id	st	slat	slon	elat	elon
1	1	AK	55.27	-160.63	55.27	-160.63
2	2	AK	59.81	-144.59	59.81	-144.59
3	3	AK	60.8	-163.53	60.8	-163.53
4	4	AK	61.02	-161.75	61.02	-161.75
5	5	AL	30.2	-88.08	0	0
6	6	AL	30.23	-87.78	0	0
7	7	AL	30.23	-88.02	30.23	-88.02
8	8	AL	30.23	-88.02	30.23	-88.02
9	9	AL	30.23	-88.1	30.23	-88.1
10	10	AL	30.2376	-87.7571	30.2399	-87.7567
11	11	AL	30.246	-87.6926	30.2465	-87.6937
12	12	AL	30.2463	-88.1991	30.2488	-88.1997
13	13	AL	30.25	-87.65	30.4	-87.78
14	14	AL	30.25	-88.12	0	0
15	15	AL	30.25	-88.12	0	0
16	16	AL	30.25	-88.12	30.25	-88.12
17	17	AL	30.25	-88.13	0	0

- Bảng Dim\_Size

```

SQLQuery8.sql - D...UF3C\FPTSHOP (62)  X Object Explorer
***** Script for SelectTopNRows command from
SELECT TOP (1000) [size_id]
      ,[mag]
      ,[len]
      ,[wid]
  FROM [StoreDataWareHouse].[dbo].[Dim_Size]

110 % ▾
Results Messages

```

	size_id	mag	len	wid
1	1	0	0	0
2	2	0	0	10
3	3	0	0	16
4	4	0	0	20
5	5	0	0	25
6	6	0	0	30
7	7	0	0	4
8	8	0	0	40
9	9	0	0	50
10	10	0	0	55
11	11	0	0	75
12	12	0	0.01	1
13	13	0	0.01	10
14	14	0	0.01	100
15	15	0	0.01	15

- Bảng Fact\_Tornado

The screenshot shows a SQL query in the Object Explorer pane and its results in the Results pane.

```

SQLQuery15.sql -...UF3C\FPTSHOP (62) Object Explorer
***** Script for SelectTopNRows command from SSMS *****/
SELECT TOP (1000) [id]
    ,[date_id]
    ,[inj]
    ,[fat]
    ,[location_id]
    ,[size_id]
FROM [StoreDataWarehouse].[dbo].[Fact_Tornado]

```

The Results pane displays the following data:

	<b>id</b>	<b>date_id</b>	<b>inj</b>	<b>fat</b>	<b>location_id</b>	<b>size_id</b>
1	1	1950-01-03	3	0	15835	64972
2	2	1950-01-03	3	0	31519	64509
3	3	1950-01-03	1	0	44336	32002
4	4	1950-01-13	1	1	2363	64461
5	5	1950-01-25	0	0	15835	54867
6	6	1950-01-25	5	0	31519	56063
7	7	1950-01-26	2	0	54996	56018
8	8	1950-02-11	0	0	54996	56070
9	9	1950-02-11	5	0	54996	54867
10	10	1950-02-11	6	0	54996	55943
11	11	1950-02-11	12	1	54996	64822
12	12	1950-02-12	0	0	2363	54867

- Chạy query để tạo khóa chính khóa ngoại giữa bảng Fact và Dim

---

```

ALTER TABLE Fact_Tornado
ADD CONSTRAINT id_pk PRIMARY KEY (id);

ALTER TABLE Fact_Tornado
ADD CONSTRAINT fk_date_id FOREIGN KEY (date_id) REFERENCES Dim_Date (date_id);

ALTER TABLE Fact_Tornado
ADD CONSTRAINT fk_location_id FOREIGN KEY (location_id) REFERENCES Dim_Location (location_id);

ALTER TABLE Fact_Tornado
ADD CONSTRAINT fk_size_id FOREIGN KEY (size_id) REFERENCES Dim_Size (size_id);

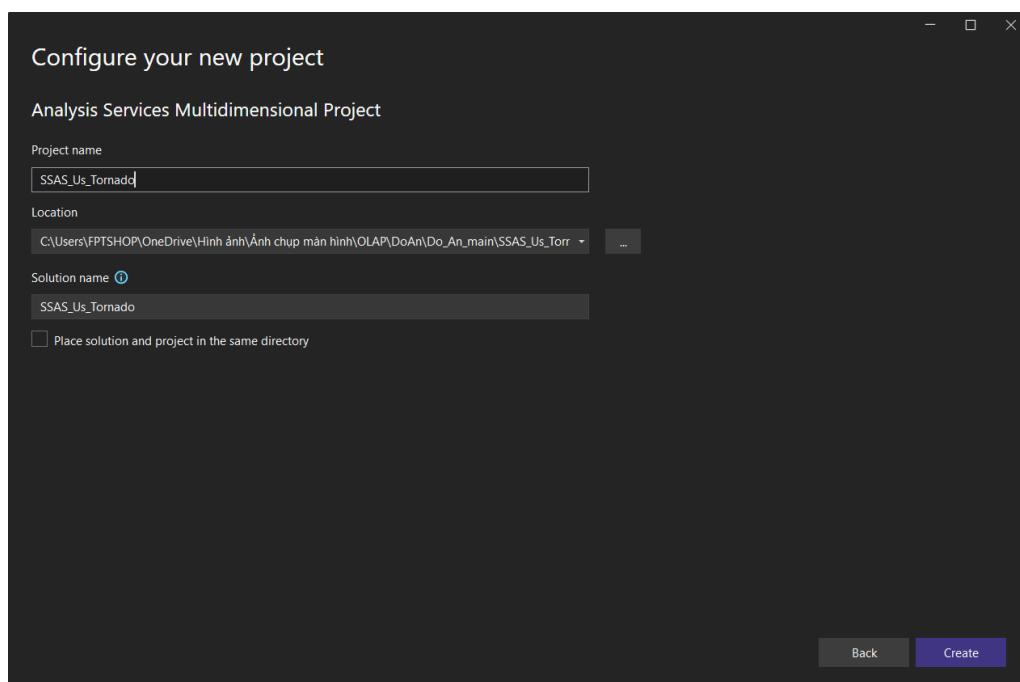
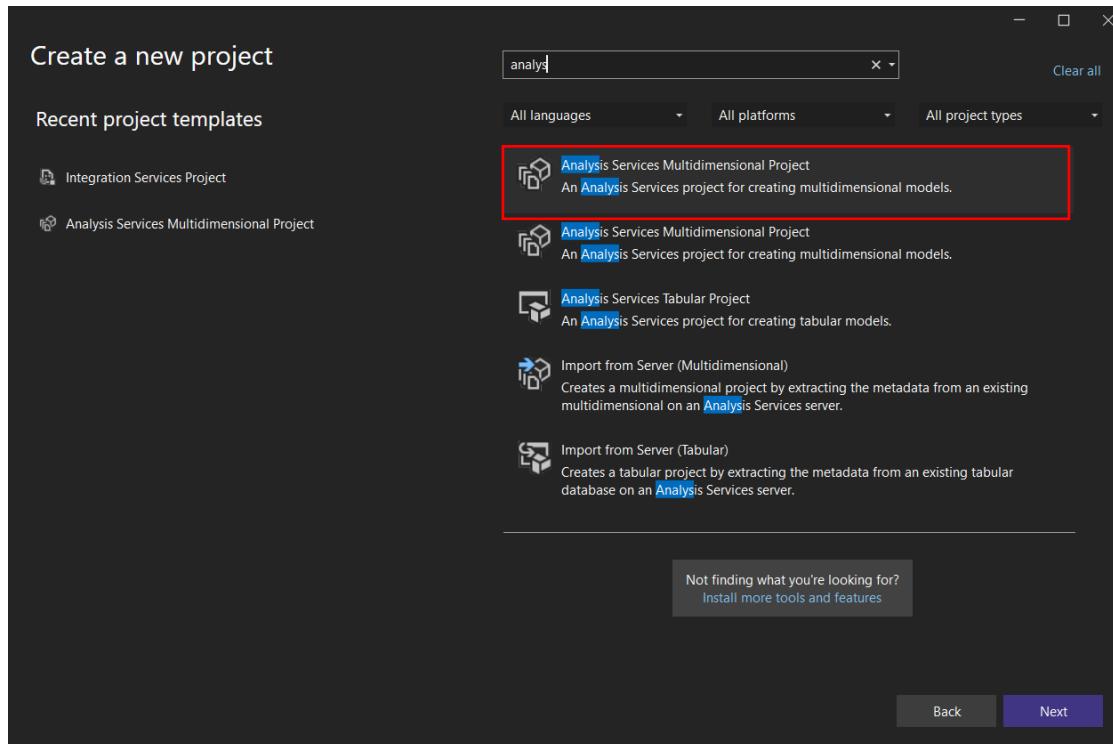
```

---

## CHƯƠNG III: PHÂN TÍCH KHO DỮ LIỆU – QUÁ TRÌNH SSAS

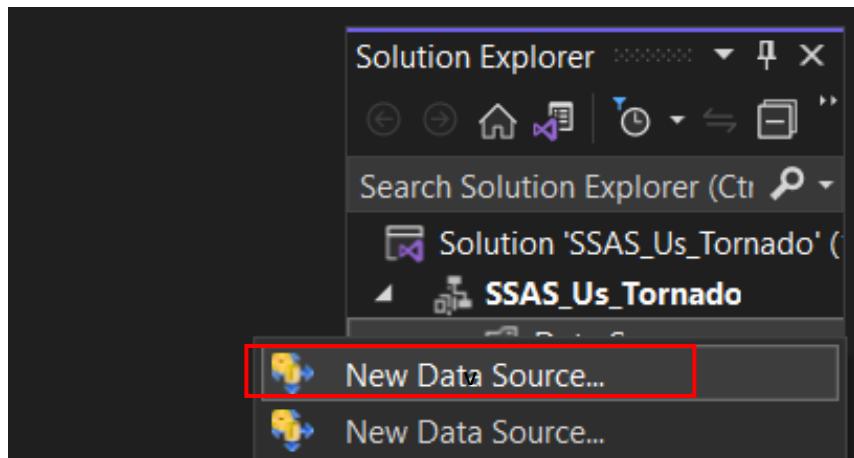
### 1. Tạo project SSAS

- Mở Visual → chọn Create a new project → chọn Analysis Services Multidimensional Project

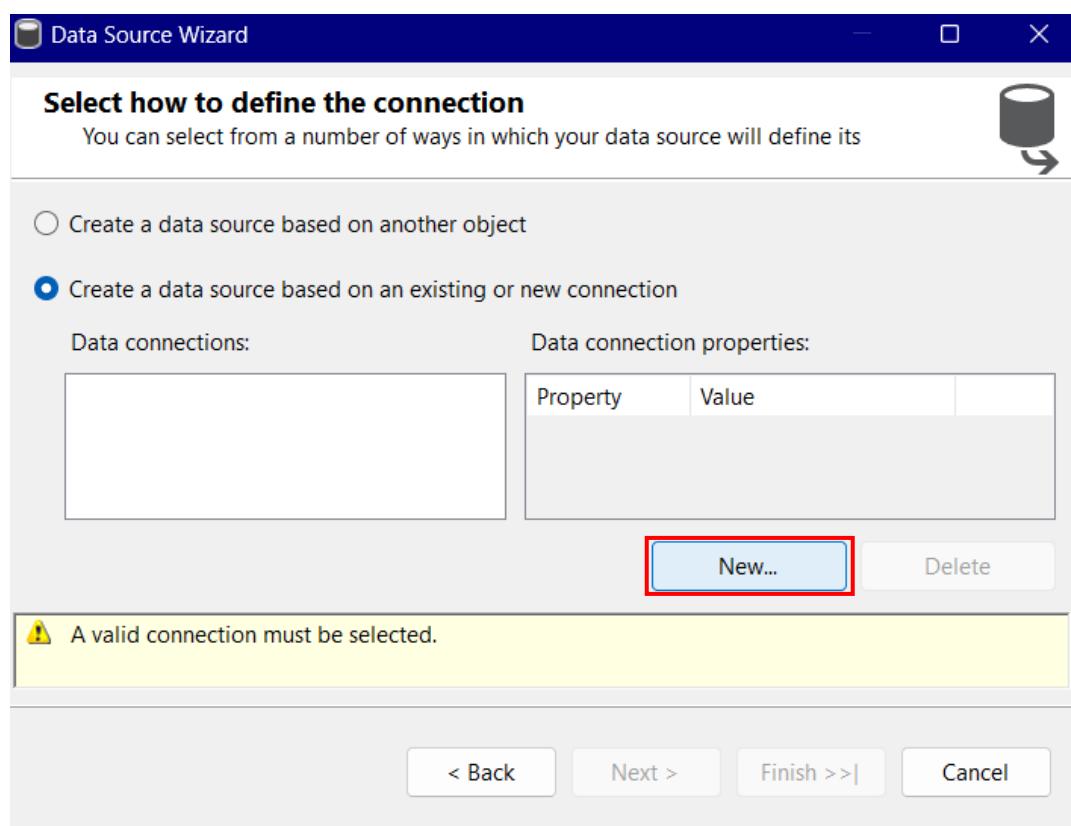


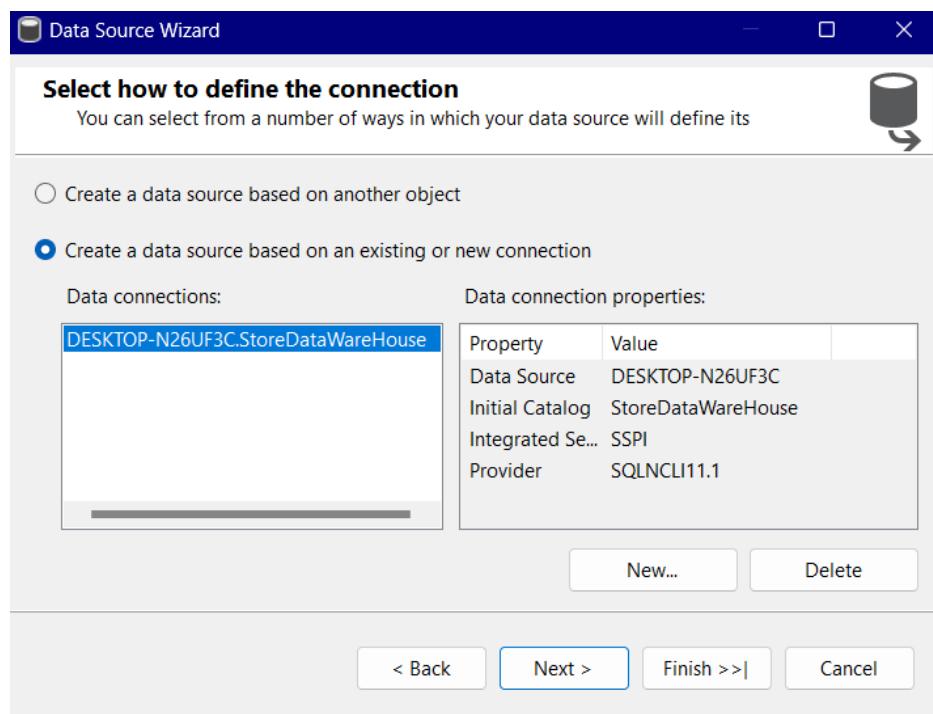
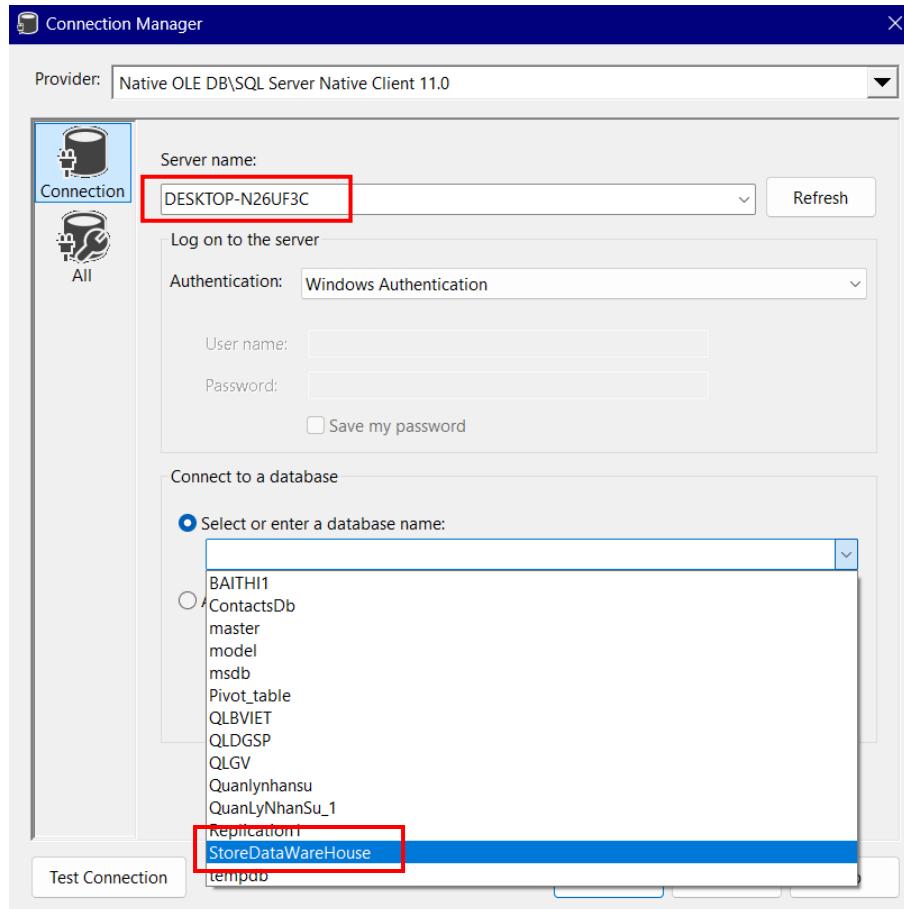
### 1.1 Tạo Data Source

- Chuột phải vào Data Source → Chọn New Data Source

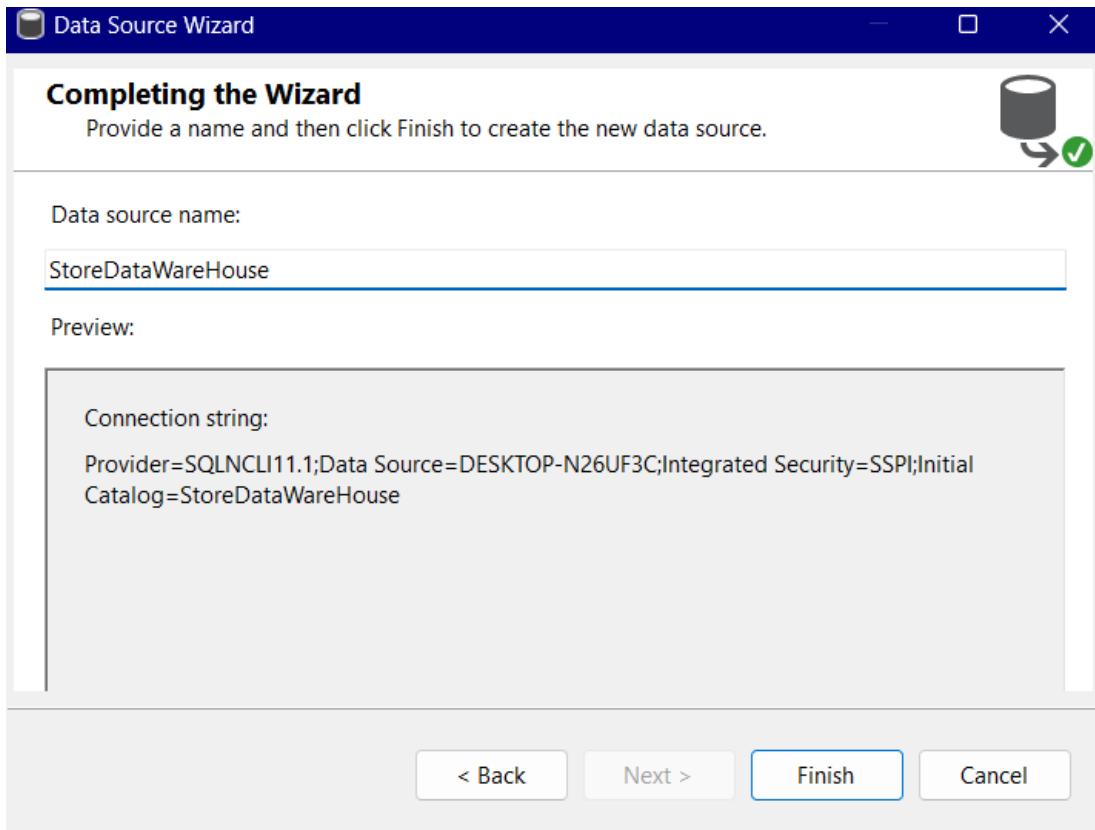
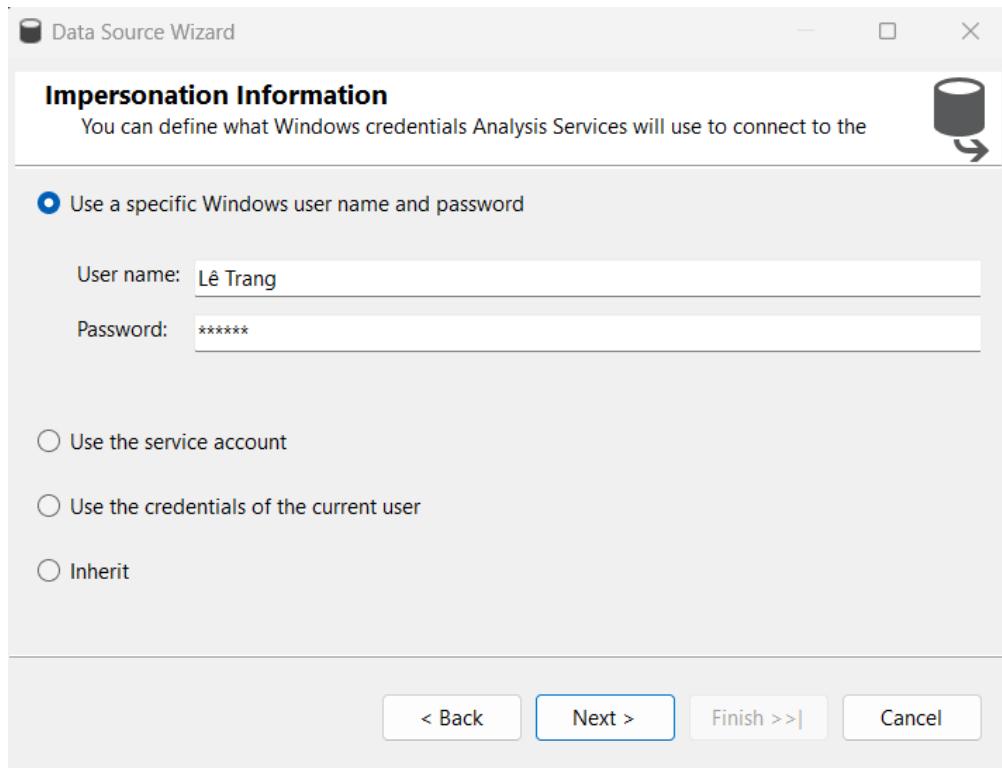


- Chọn New → Nhập Server Name → Chọn StoreDataWareHouse → Next



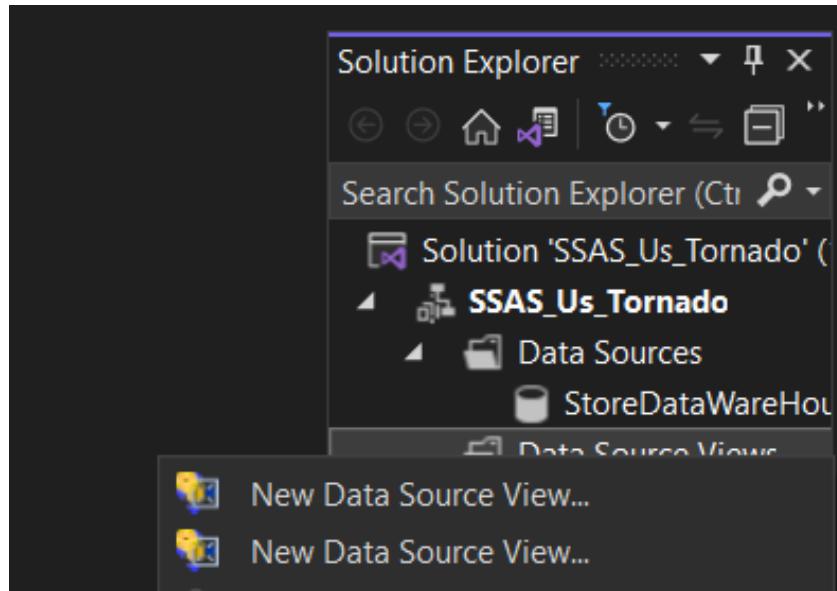


- Điền tên và mật khẩu user Window → Chọn Finish

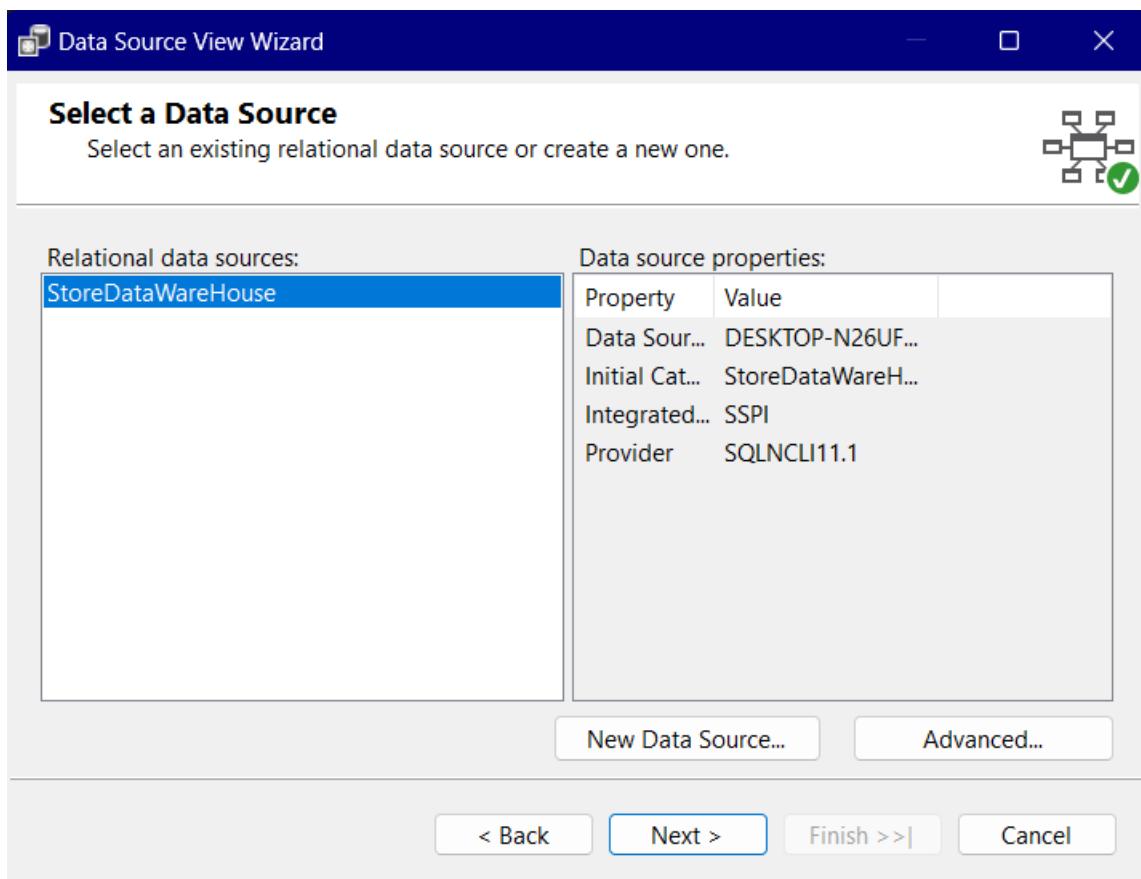


### 1.2 Tạo Data Source Views

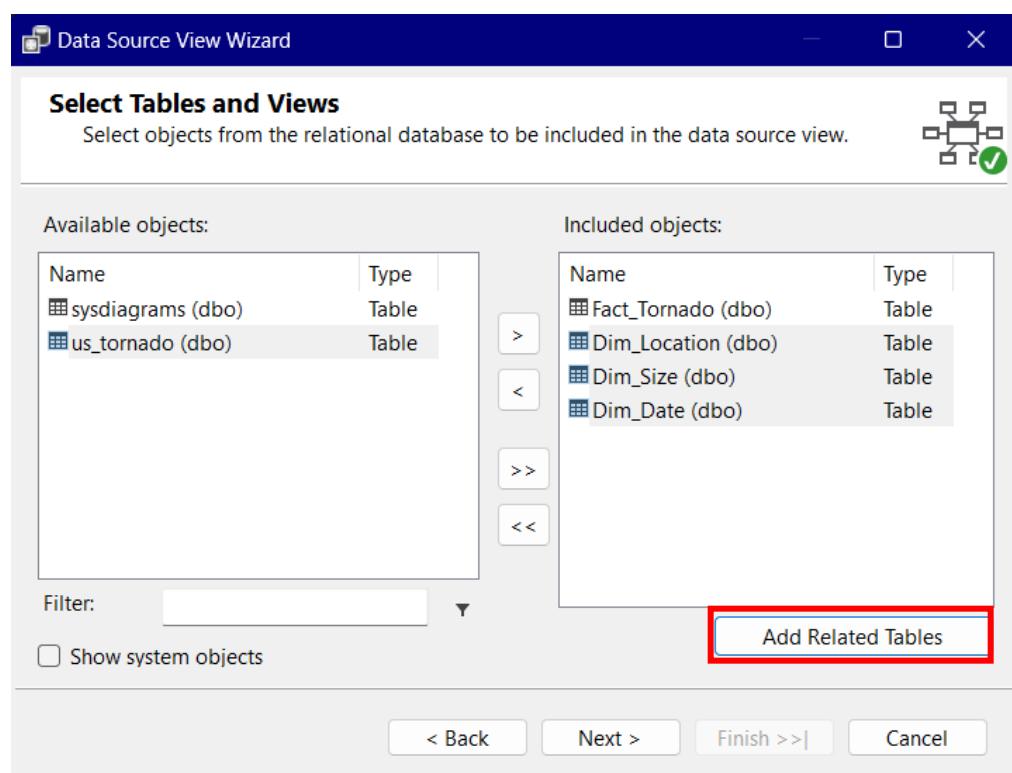
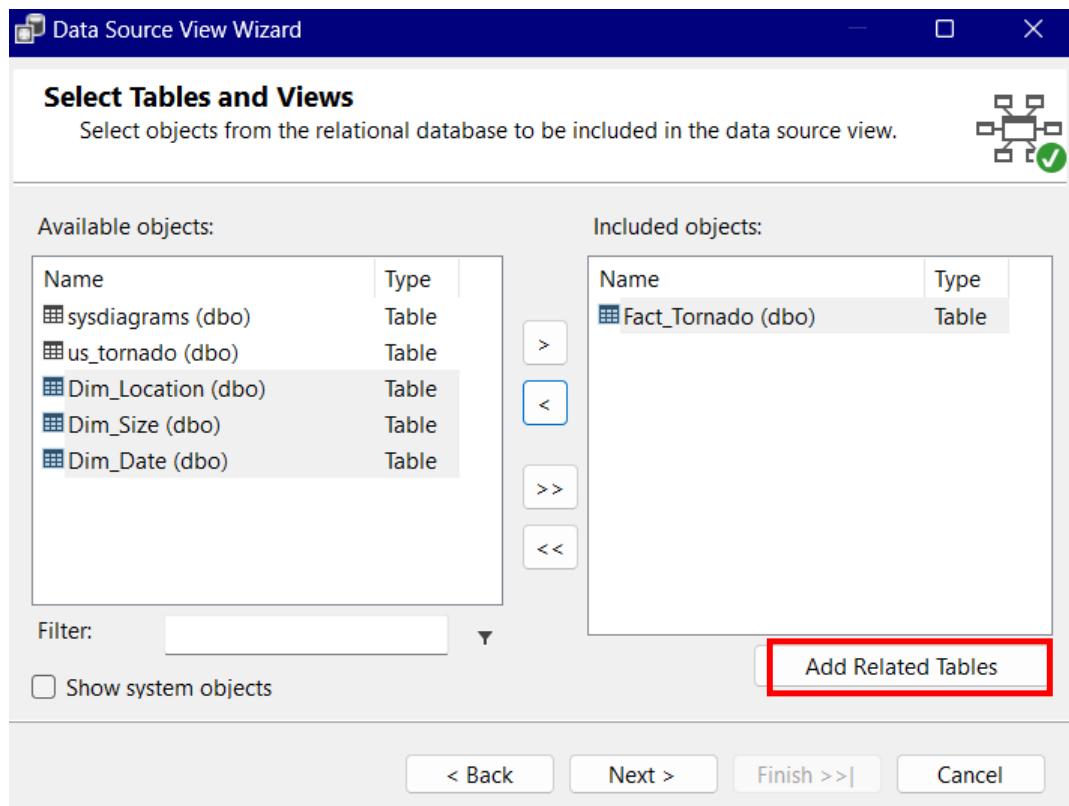
- Chuột phải Data Source Views → Chọn New Data Source View

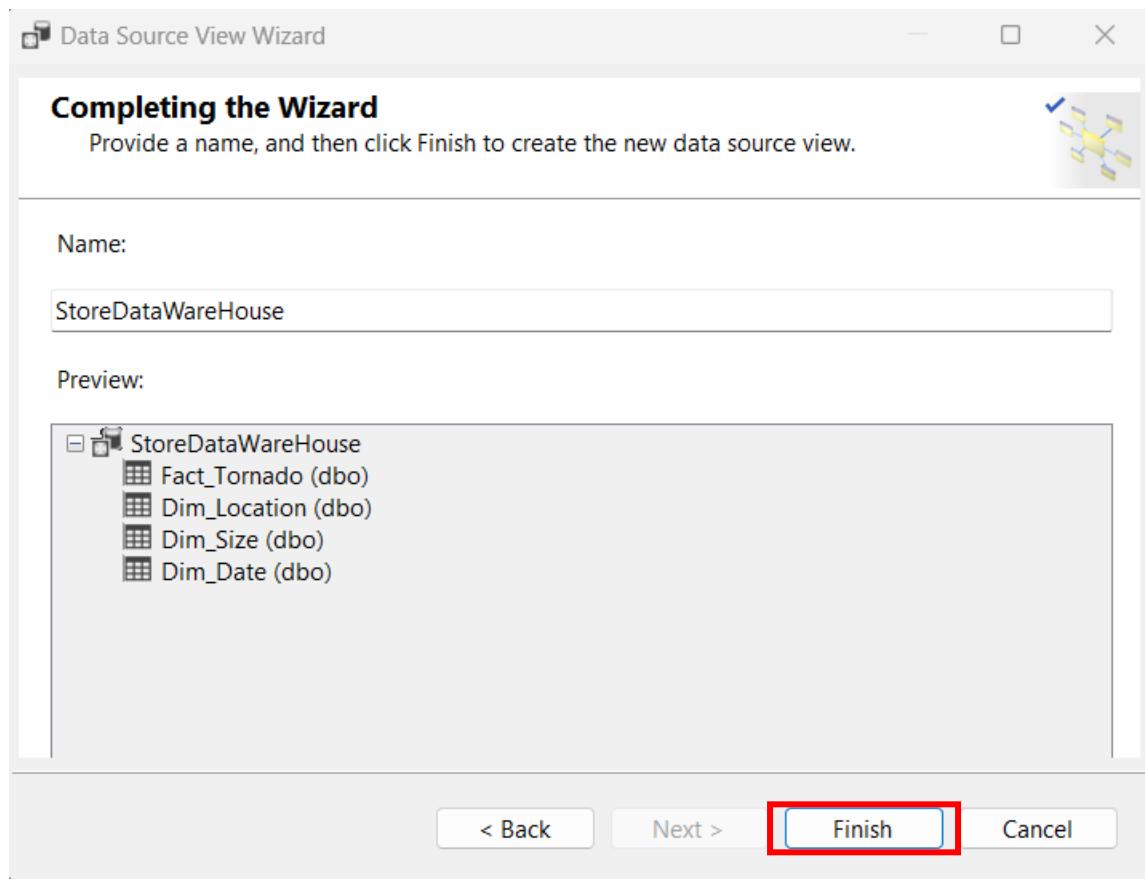


- Chọn Data Source vừa tạo → Chọn Next



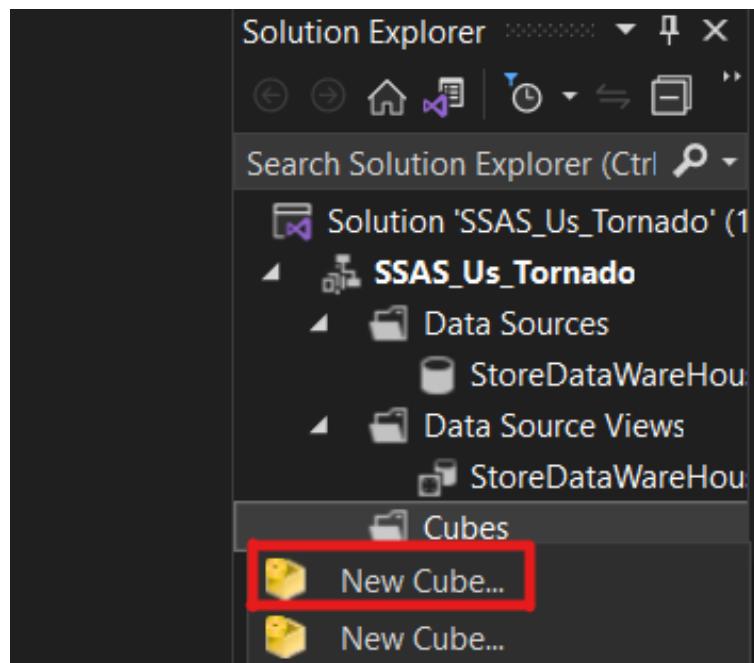
- Chọn bảng Fact\_Tornado → Chọn Add Related Tables → Chọn Finish

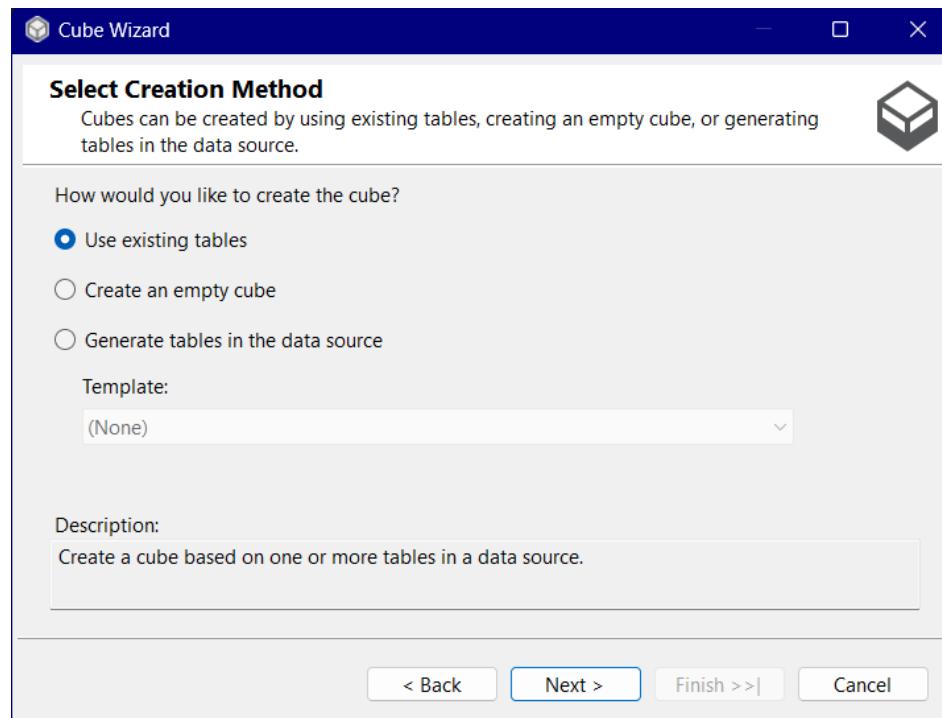




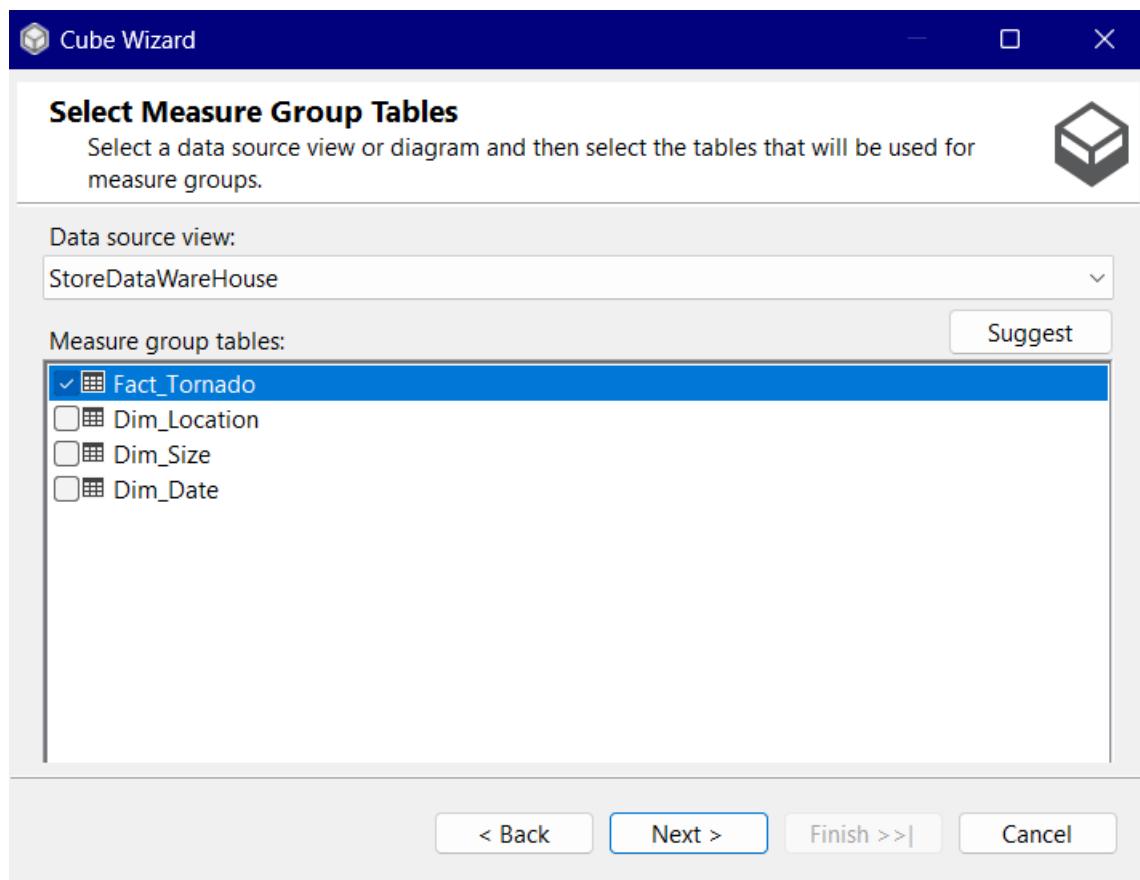
### 1.3 Tạo Cubes

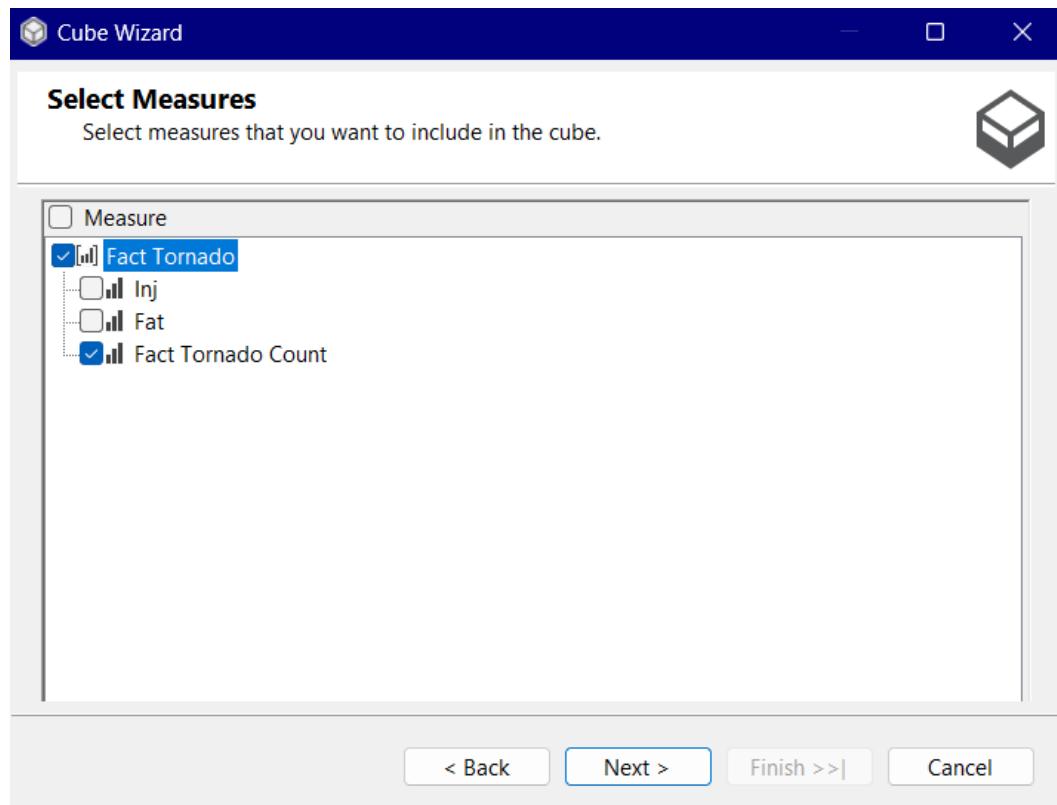
- Chuột phải Cubes → Chọn New Cube → Chọn Use existing tables



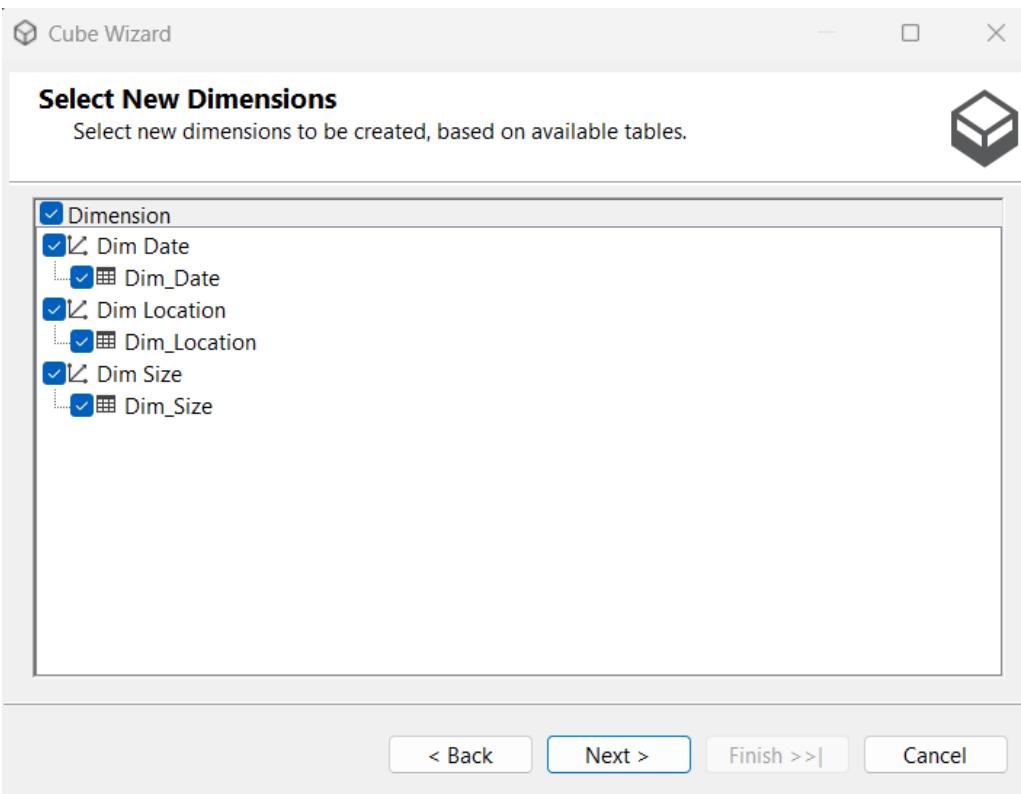


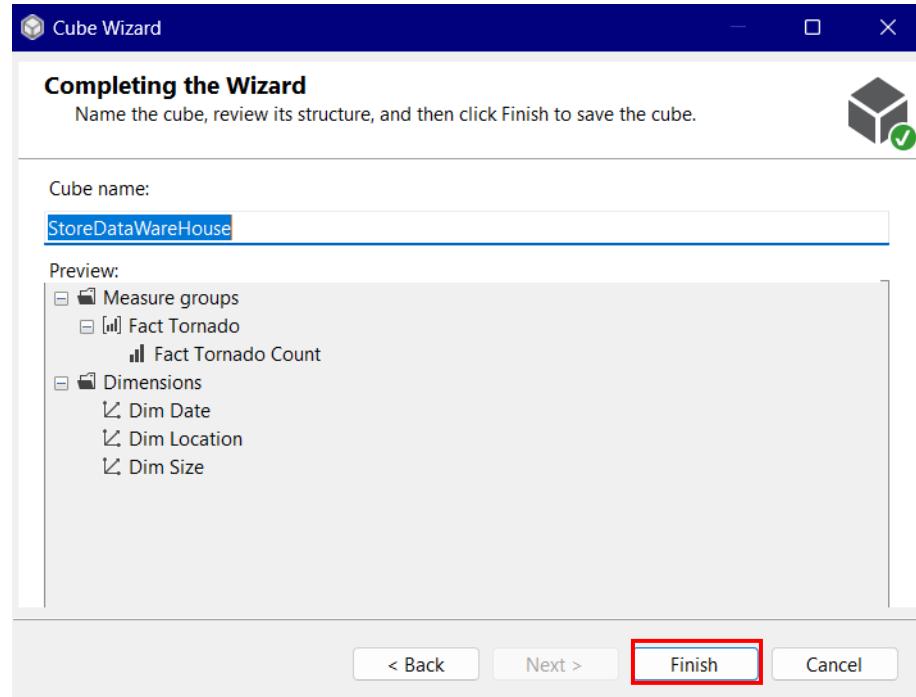
- Tick bảng Fact\_Tornado → Phân Measure chọn Fact\_Tornado Count



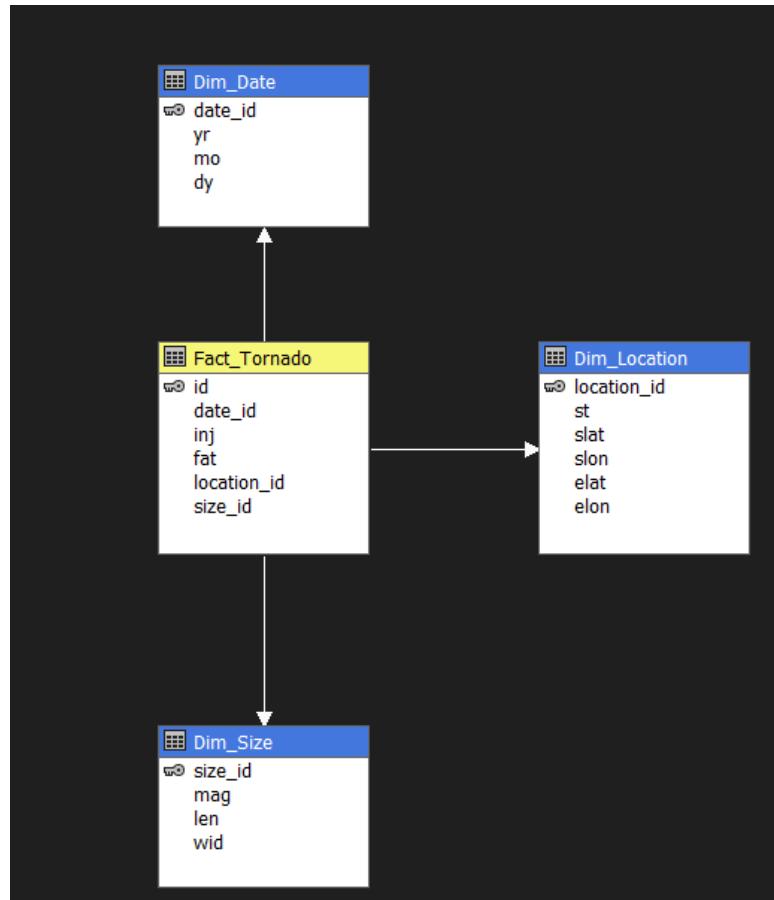


- Tick tất cả các bảng Dim → Chọn Finish

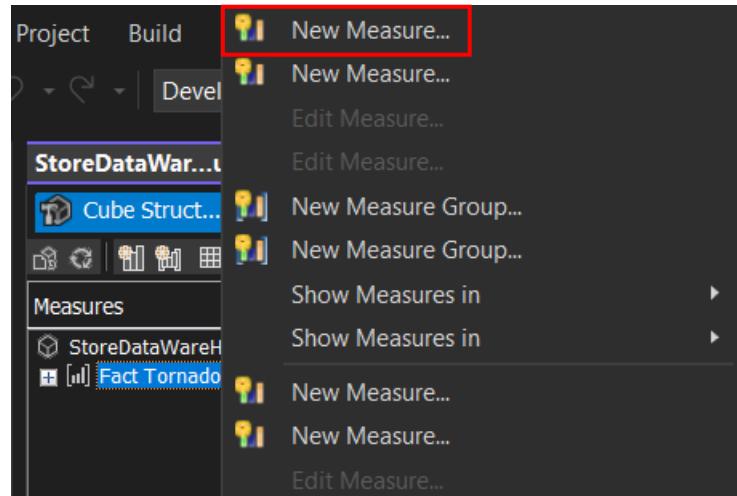




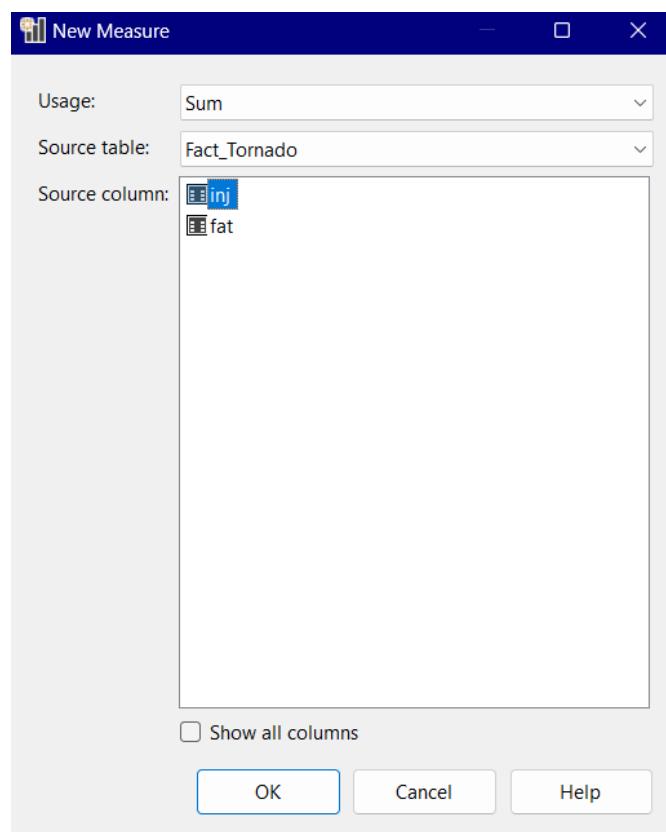
- Kết quả sau khi tạo thành công

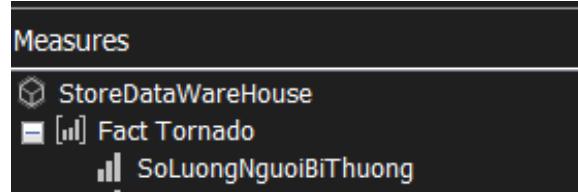


- Click chuột phải vào bảng Fact Tornado ở khung Measure → Chọn New Measure

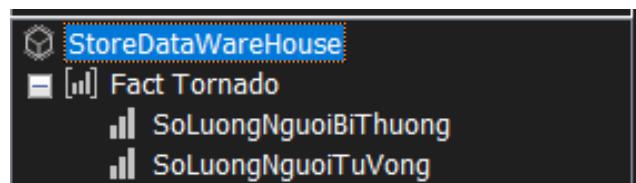
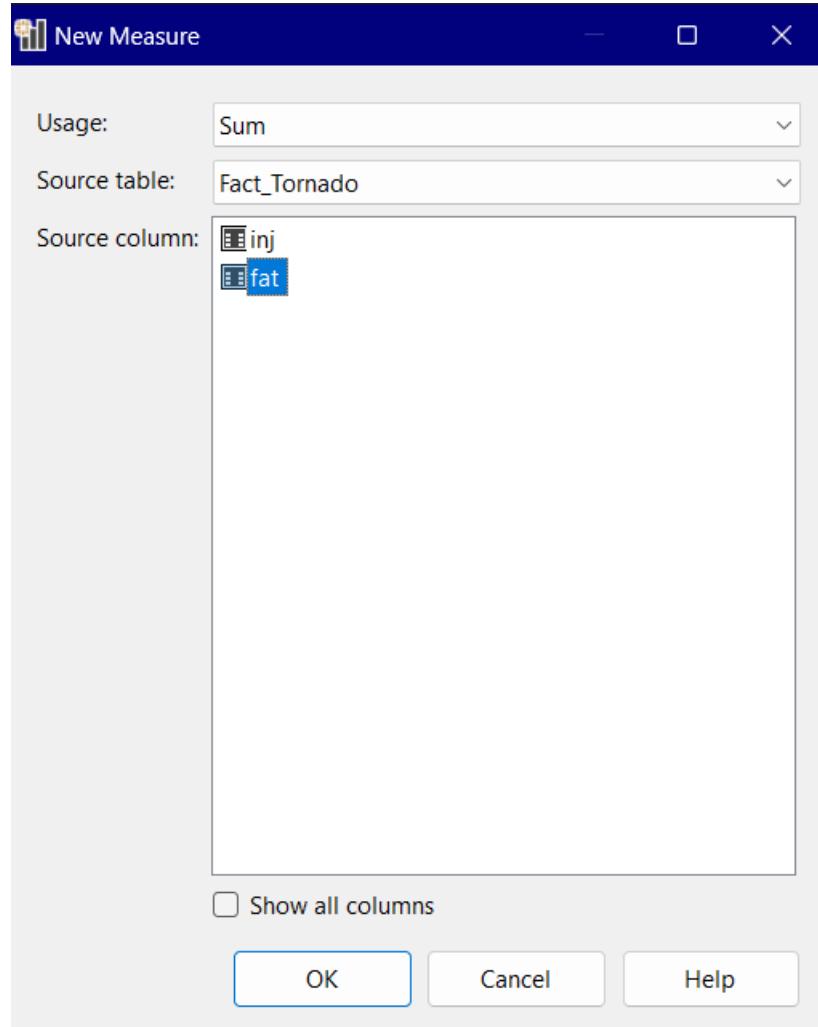


- Số lượng người bị thương
  - Chọn Usage là Sum → Chọn bảng Fact\_Tornado → Chọn inj





- Số lượng người tử vong
  - Chọn Usage là Sum → Chọn bảng Fact\_Tornado → Chọn fat

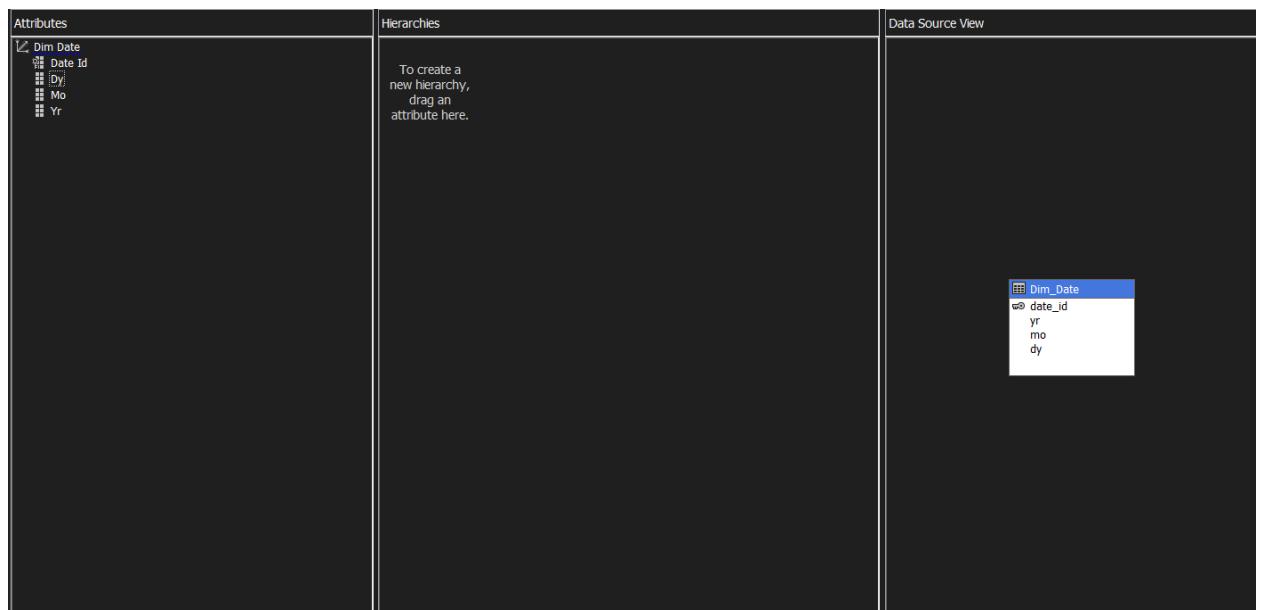
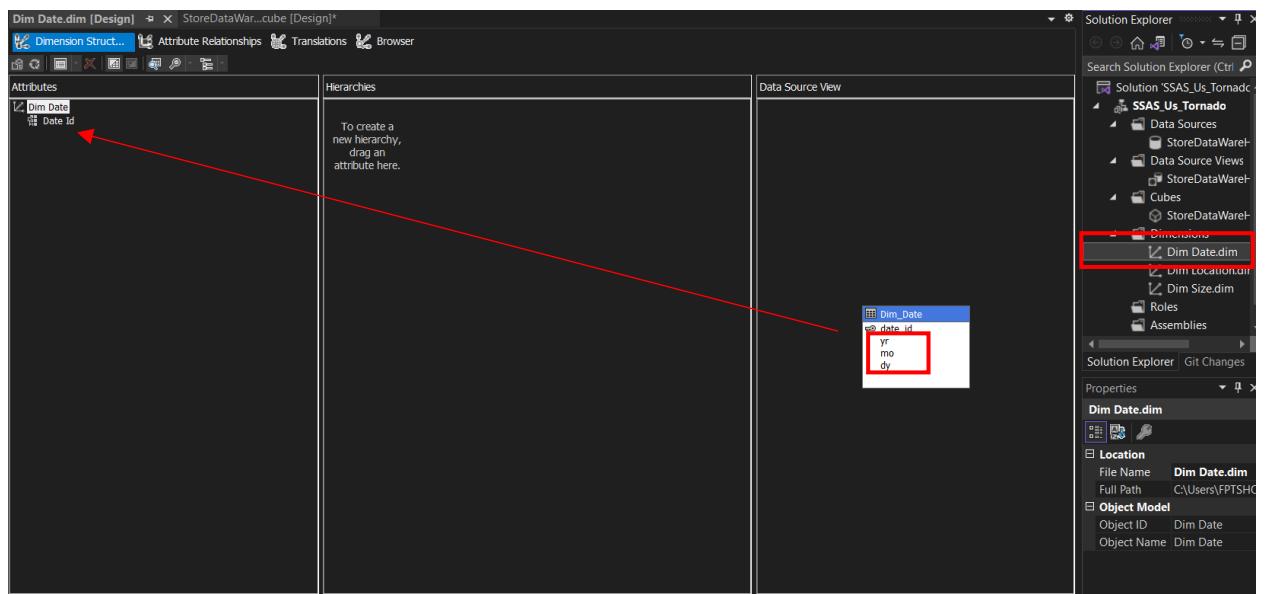


- Làm tương tự với các measure khác

### 1.4 Chỉnh sửa Dimension

- Dim\_Date

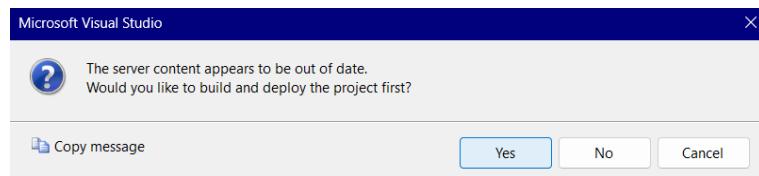
- Click vào Dim\_Date ở phần Dimension → Kéo thả các thuộc tính chưa có trong Attributes vào khung Attributes



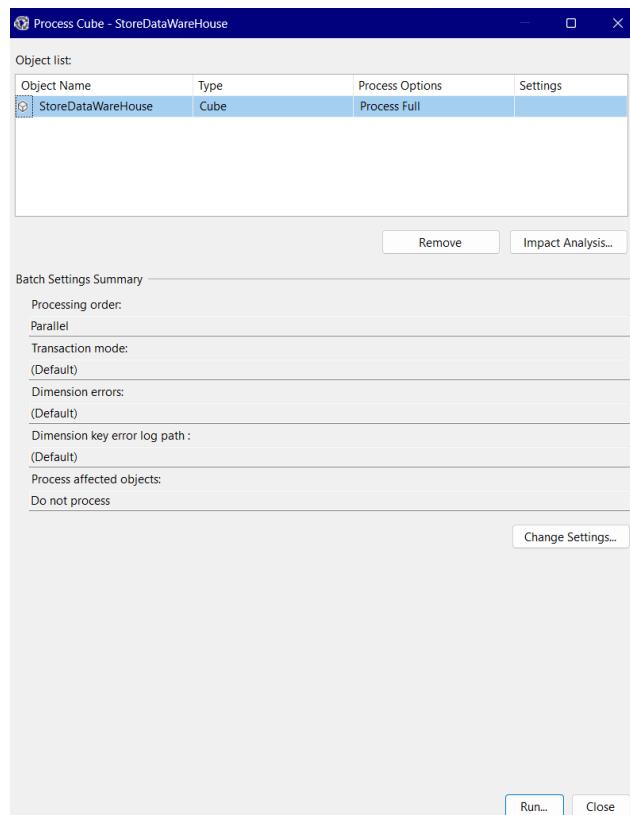
- Kéo lần lượt các thuộc tính Yr, Mo, Dy từ mục Attributes vào mục Hierarchies  
→ Chọn để phân cấp dữ liệu theo Drill down

### 1.5 Process

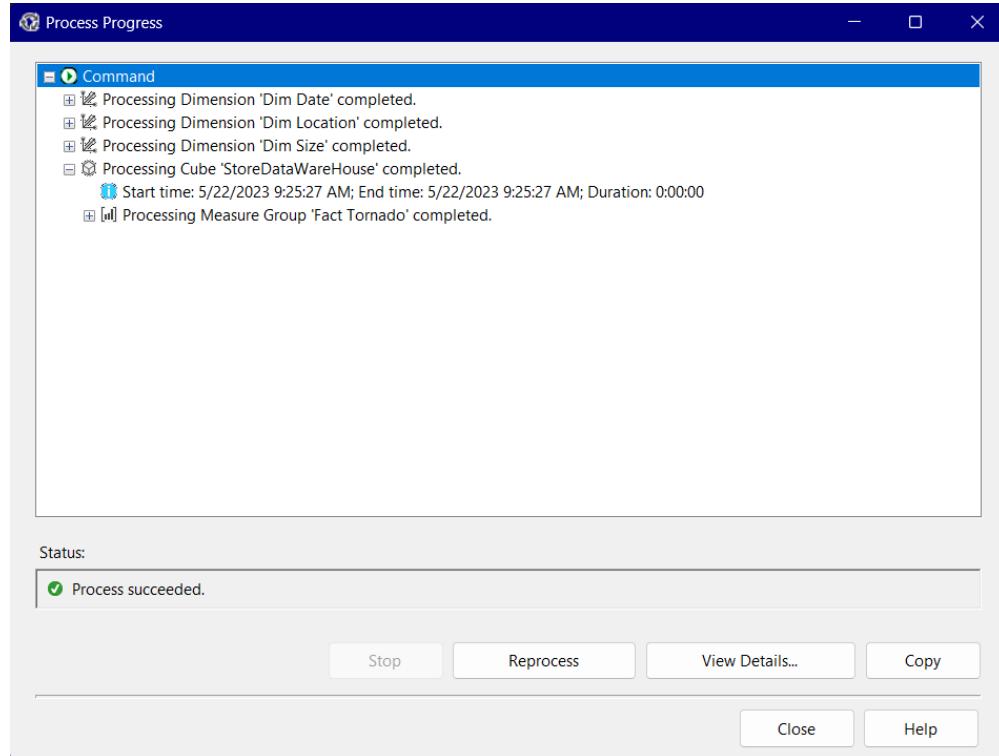
- Chuột phải vào StoreDataWareHouse.cube → Chọn Process → Hiện thông báo chọn Yes



- Chọn Run



- Hiển thị Process thành công



## 2. Thực thi truy vấn

### 2.1 Roll up

#### 2.1.1 Tổng số lượng lốc xoáy tính theo từng tháng của mỗi năm

The screenshot displays the SSAS Solution Explorer and the Data Editor. The Data Editor shows a table with the following data:

Yr	Mo	SolanXayRaLocXoayTrongNgay
1950	1	7
1950	10	2
1950	11	4
1950	12	4
1950	2	20
1950	3	21
1950	4	15
1950	5	61
1950	6	28
1950	7	23
1950	8	13
1950	9	3
1951	1	2
1951	10	2
1951	11	12
1951	12	10
1951	2	10
1951	3	6
1951	4	26
1951	5	57
1951	6	76
1951	7	23
1951	8	27
1951	9	9

### 2.1.2 Tổng số lượng lốc xoáy theo từng năm

Yr	SoLanXayRaLocXoayTrongNgay
1950	201
1951	260
1952	240
1953	421
1954	550
1955	591
1956	504
1957	858
1958	564
1959	604
1960	616
1961	697
1962	657
1963	463
1964	704
1965	897
1966	585
1967	927
1968	657
1969	608
1970	653
1971	889
1972	741

## 2.2 Drill down

### 2.2.1 Số lần xảy ra lốc xoáy ở từng bang trong năm 2020

St	Yr	SoLanXayRaLocXoayTrongNgay
AL	2020	70
AR	2020	42
AZ	2020	2
CA	2020	8
CO	2020	35
CT	2020	7
DE	2020	4
FL	2020	58
GA	2020	60
IA	2020	28
IL	2020	63
IN	2020	15
KS	2020	17
KY	2020	22
LA	2020	42
MA	2020	2
MD	2020	20
ME	2020	1
MI	2020	2
MN	2020	59
MO	2020	19
MS	2020	82
MT	2020	3
NC	2020	46

### 2.2.2 Số lần xảy ra lốc xoáy ở từng bang theo từng tháng trong năm 2020

St	Yr	Mo	SoLanXayRaLocXoayTrongNgay
AL	2020	1	5
AL	2020	10	2
AL	2020	12	1
AL	2020	2	5
AL	2020	3	7
AL	2020	4	42
AL	2020	6	2
AL	2020	7	1
AL	2020	8	5
AR	2020	1	10
AR	2020	11	1
AR	2020	3	8
AR	2020	4	3
AR	2020	5	8
AR	2020	7	1
AR	2020	8	8
AR	2020	9	3
AZ	2020	7	2
CA	2020	11	2
CA	2020	5	1
CA	2020	8	3
CA	2020	9	2
CO	2020	5	9
CO	2020	6	5

 The Solution Explorer on the right shows the project structure for 'SSAS\_Us\_Tornado'."/>

## 2.3 Slide and Dice

### 2.3.1 Tìm top 3 bang có tổng số người tử vong cao nhất năm 1953

Yr	St	SoLuongNguoiTuVong
1...	MA	94
1...	MI	127
1...	TX	150

 The Solution Explorer on the right shows the project structure for 'SSAS\_Us\_Tornado'."/>

### 2.3.2 Tìm các bang với số lượng lốc xoáy tại bang đó có mức độ sút tàn phá từ “Nghiêm trọng” trở lên

The screenshot shows the SSAS Designer interface with the title bar "SSAS\_Us\_Tornado". The main area displays a MDX query results grid. The query is:

```
SELECT Dim Size Mag, Id, Distinct Count
FROM <Select dimension>
```

The results grid contains the following data:

St	Mag	Id Distinct Count
AL	3	137
AL	4	35
AL	5	7
AR	3	144
AR	4	29
AZ	3	3
CA	3	2
CO	3	23
CT	3	4
CT	4	2
DE	3	1
FL	3	40
FL	4	2
GA	3	70
GA	4	10
IA	3	112
IA	4	40
IA	5	6
IL	3	106
IL	4	29
IL	5	2
IN	3	91
IN	4	26

### 2.3.3 Tìm các năm có các cơn lốc xoáy có độ đo F/EF cao nhất và gây tử vong

The screenshot shows the SSAS Designer interface with the title bar "SSAS\_Us\_Tornado". The main area displays a MDX query results grid. The query is:

```
SELECT Dim Date Yr, Mag, SoLuongNguoiTuVong
FROM <Select dimension>
```

The results grid contains the following data:

Yr	Mag	SoLuongNguoiTuVong
1953	5	271
1955	5	100
1956	5	17
1957	5	55
1958	5	21
1960	5	5
1964	5	11
1966	5	80
1968	5	34
1970	5	26
1971	5	47
1974	5	148
1976	5	2
1977	5	22
1984	5	9
1985	5	18
1990	5	31
1991	5	17
1992	5	1
1997	5	27
1998	5	35
1999	5	36
2007	5	11

### 2.3.4 Số người bị thương do các cơn lốc xoáy có độ đo F/EF ở mức độ nhẹ đến trung bình

## IS217.N22.HTCL – Kho dữ liệu và OLAP

The screenshot shows the SSAS Designer interface with the 'StoreDataWareHouse.ube [Design]' tab selected. In the center, a 'Dimension Usage' dialog is open, displaying a table with one row:

Dimension	Hierarchy	Operator	Filter Expression	Parameters
Dim Size	Mag	Equal	{ 0, 1 }	

The left pane shows the 'Metadata' tree with various dimensions like Dim Location, Dim Date, etc. The right pane shows the 'Solution Explorer' with the 'StoreDataWarehouse' cube selected.

2.3.5 Trung bình một ngày có bao nhiêu người chết và bị thương ở bang MA

The screenshot shows the SSAS Designer interface with the 'StoreDataWareHouse.ube [Design]' tab selected. A MDX query is displayed in the central pane:

```
WITH
MEMBER [Measures].[SoLuongNguoiTuVong_Ave] AS
[Measures].[SoLuongNguoiTuVong]/12300,
FORMAT_STRING = "#0.00"
MEMBER [Measures].[SoLuongNguoiBiThuong_Ave] AS
[Measures].[SoLuongNguoiBiThuong]/12300,
FORMAT_STRING = "#0.00"
SELECT
NON EMPTY {{[Measures].[SoLuongNguoiTuVong_Ave]}, [Measures].[SoLuongNguoiBiThuong_Ave]} ON COLUMNS,
NON EMPTY {[Dim Location].[St].[MA]} ON ROWS
FROM [StoreDataWarehouse]
```

The results table shows data for the state 'MA':

St	SoLuongNguoiTuVong_Ave	SoLuongNguoiBiThuong_Ave
MA	0.008356536585366	0.124065040650407

The left pane shows the 'Metadata' tree with various measures like Fact Tornado, Fact Tornado 1, etc. The right pane shows the 'Solution Explorer' with the 'StoreDataWarehouse' cube selected.

2.3.6 Tìm các bang có các cơn lốc xoáy gây thương tích nhưng không gây tử vong trong năm 2001

Yr	St	SoLuongNguoiBiThuong
2...	CO	13
2...	CT	1
2...	GA	23
2...	IL	3
2...	KY	14
2...	ME	1
2...	MI	8
2...	MN	13
2...	NC	3
2...	ND	1
2...	NE	9
2...	OH	1
2...	SC	39
2...	TX	49
2...	VA	4
2...	WY	2

### 2.3.7 Tên tiêu bang có số lượng người bị thương > 5500 người

St	SoLuongNguoiBiThuong
AL	8672
MS	6452
OK	5997
TX	9412

## 2.4 Pivot

### 2.4.1 Thống kê số lượng lốc xoáy theo từng tháng và từng loại độ đo F/EF

Mo	Mag	SoLanXayRaLocXoayTrongNgay
1	0	611
1	1	690
1	2	321
1	3	79
1	4	11
10	0	1247
10	1	1019
10	2	413
10	3	87
10	4	6
10	5	1
11	0	817
11	1	1103
11	2	544
11	3	156
11	4	22
12	0	581
12	1	722
12	2	373
12	3	116
12	4	16
12	5	2
2	0	621
2	1	812

#### 2.4.2 Thống kê top 5 bang xảy ra lốc xoáy ít nhất

St	AK	SoLanXayRaLocXoayTrongNgay
AK	4	
DC	3	
PR	28	
RI	13	
VI	1	

#### 2.4.3 Thống kê tổng số người bị thương theo từng năm và từng bang

Yr	St	SoluongNguoIBiThuong
1950	AL	15
1950	AR	49
1950	CO	1
1950	CT	3
1950	FL	0
1950	GA	1
1950	IA	5
1950	IL	31
1950	IN	0
1950	KS	26
1950	KY	0
1950	LA	144
1950	MD	0
1950	MN	4
1950	MO	8
1950	MS	23
1950	NC	8
1950	ND	0
1950	NE	135
1950	NM	0
1950	OH	31
1950	OK	45
1950	PA	3

### 3. Ngôn ngữ MDX

#### 3.1 Tổng số lượng lốc xoáy theo từng tháng của mỗi năm

--Câu 1. Tổng số lượng lốc xoáy theo từng tháng của mỗi năm

```
SELECT NON EMPTY {[Measures].[Id Distinct Count]} ON COLUMNS, NON EMPTY {[Dim Date].[Yr].[Yr] * [Dim Date].[Mo].[Mo]} ON ROWS
FROM [Store DW];
```

Id	District Count
1950	7
1950	2
1950	4
1950	4
1950	20
1950	21
1950	15
1950	61
1950	28
1950	23
1950	13
1950	3
1951	2
1951	2
1951	12
1951	10
1951	10
1951	6
1951	26
1951	57
1951	76
1951	23
1951	27
1951	9
1952	12
1952	6
1952	3

Query executed successfully.

### 3.2 Tổng số lượng lốc xoáy theo từng năm

```
--Câu 2. Tổng số lượng lốc xoáy theo từng năm
SELECT NON EMPTY {[Measures].[Id Distinct Count]} ON COLUMNS, NON EMPTY
{([Dim Date].[Yr].[Yr])} ON ROWS
FROM [Store DW];
```

Year	Id Distinct Count
1950	201
1951	260
1952	240
1953	421
1954	550
1955	591
1956	504
1957	858
1958	564
1959	604
1960	616
1961	697
1962	657
1963	463
1964	704
1965	897
1966	585
1967	927
1968	657
1969	608
1970	653
1971	889
1972	741
1973	1102
1974	945
1975	919
1976	834

Query executed successfully.

### 3.3 Số lần xảy ra lốc xoáy ở từng bang trong năm 2020

```
--Câu 3. Số lần xảy ra lốc xoáy ở từng bang trong năm 2020
SELECT NON EMPTY {[Measures].[Id Distinct Count]} ON COLUMNS, NON EMPTY
{([Dim Date].[Yr].[Yr] * [Dim Location].[St].[St])} ON ROWS
FROM (SELECT {[Dim Date].[Yr].&[2020]}) ON COLUMNS
FROM [Store DW];
```

State	Id Distinct Count
AL	70
AR	42
AZ	2
CA	8
CO	35
CT	7
DE	4
FL	58
GA	60
IA	28
IL	63
IN	15
KS	17
KY	22
LA	42
MA	2
MD	20
ME	1
MI	2
MN	59
MO	19
MS	82
MT	3
NC	46
ND	22
NE	22
NH	2

Query executed successfully.

### 3.4 Số lần xảy ra lốc xoáy ở từng bang theo từng tháng trong năm 2020

## IS217.N22.HTCL – Kho dữ liệu và OLAP

MDXQuery1.mdx -...-5C9Q3ON\Tra My\* ×

Cube: Store DW

Measure Group: <All>

```
--Câu 4. Số lần xảy ra lốc xoáy ở từng bang theo từng tháng trong năm 2020
SELECT NON EMPTY {[Measures].[Id Distinct Count]} ON COLUMNS, NON EMPTY
{([Dim Date].[Yr].[Yr] * [Dim Date].[Mo].[Mo]*[Dim Location].[St].[St] )} ON ROWS
FROM (SELECT {[Dim Date].[Yr].&[2020]}) ON COLUMNS
FROM [Store DW]);
```

Messages Results

			Id Distinct Count
2020	1	AL	5
2020	1	AR	10
2020	1	FL	3
2020	1	GA	5
2020	1	IL	1
2020	1	KY	9
2020	1	LA	3
2020	1	MO	5
2020	1	MS	16
2020	1	OH	2
2020	1	OK	3
2020	1	SC	7
2020	1	TN	4
2020	1	TX	14
2020	10	AL	2
2020	10	GA	7
2020	10	KY	1
2020	10	MA	1
2020	10	MS	2
2020	10	NC	1
2020	10	NY	1
2020	10	SC	3
2020	10	WA	1
2020	11	AR	1
2020	11	CA	2
2020	11	FL	3
2020	11	IL	8

Query executed successfully. DESKTOP-5C9Q3ON DESKTOP-5C9Q3ON\Tra My SSAS\_Us\_Tornado 00:00:01

### 3.5 Tìm top 3 bang có tổng số người tử vong cao nhất năm 1953

MDXQuery1.mdx -...-5C9Q3ON\Tra My\* ×

Cube: Store DW

Measure Group: <All>

```
--Câu 5. Tìm top 3 bang có tổng số người tử vong cao nhất năm 1953
SELECT NON EMPTY {[Measures].[SoLuongNguoiTuVong]} ON COLUMNS, NON EMPTY
{ TOPCOUNT ([Dim Location].[St].[St], 3, [Measures].[SoLuongNguoiTuVong])} ON ROWS
FROM (SELECT {[Dim Date].[Yr].&[1953]}) ON COLUMNS
FROM [Store DW]);
```

Messages Results

			SoLuongNguoiTuVong
		TX	150
		MI	127
		MA	94

Query executed successfully. DESKTOP-5C9Q3ON DESKTOP-5C9Q3ON\Tra My SSAS\_Us\_Tornado 00:00:01

### 3.6 Tìm các bang với số lượng lốc xoáy tại bang đó có mức độ sút tàn phá từ "Nghiêm trọng" trở lên

## IS217.N22.HTCL – Kho dữ liệu và OLAP

MDXQuery1.mdx -...-5C9Q3ON(Tra My)\*

```
-- Câu 6. Tìm các bang với số lượng lốc xoáy tại bang đó có mức độ sức tàn phá từ "Nghiêm trọng" trở lên
SELECT NON EMPTY {[Measures].[Id Distinct Count]} ON COLUMNS,
NON EMPTY {[Dim Location].[St].[St] * [Dim Size].[Mag].[Mag]} ON ROWS
FROM (SELECT {[Dim Size].[Mag].[Mag].&[3],[Dim Size].[Mag].[Mag].&[4],[Dim Size].[Mag].[Mag].&[5]}) ON COLUMNS
FROM [Store DW];
```

	Id Distinct Count
AL	3
AL	4
AL	5
AR	3
AR	4
AZ	3
CA	3
CO	3
CT	3
CT	4
DE	3
FL	3
FL	4
GA	3
GA	4
IA	3
IA	4
IA	5
IL	3
IL	4
IL	5
IN	3
IN	4
IN	5
KS	3
KS	4
KS	5
KY	3
KY	4
KY	5

Query executed successfully.

3.7 Tìm các năm có các cơn lốc xoáy có độ đo F/EF cao nhất và gây tử vong

MDXQuery1.mdx -...-5C9Q3ON(Tra My)\*

```
-- Câu 7. Tìm các năm có các cơn lốc xoáy có độ đo F/EF cao nhất và gây tử vong
SELECT NON EMPTY {[Measures].[SoLuongNguoiTuVong]} ON COLUMNS,
NON EMPTY {[Dim Date].[Yr].[Yr]} * [Dim Size].[Mag].[Mag] ON ROWS
FROM (SELECT (FILTER([Dim Date].[Yr].[Yr], [Measures].[SoLuongNguoiTuVong] >0)) ON COLUMNS
FROM (SELECT {[Dim Size].[Mag].[Mag].&[5]}) ON COLUMNS
FROM [Store DW]);
```

	SoLuongNguoiTuVong
1953	271
1955	100
1956	5
1957	55
1958	21
1960	5
1964	11
1966	80
1968	34
1970	5
1971	47
1974	148
1976	5
1977	22
1984	9
1985	18
1990	31
1991	17
1992	1
1997	27
1998	35
1999	36
2007	11
2008	9
2011	290
2013	24

Query executed successfully.

3.8 Số người bị thương do các cơn lốc xoáy có độ đo F/EF ở mức độ nhẹ đến trung bình

## IS217.N22.HTCL – Kho dữ liệu và OLAP

MDXQuery1.mdx -> 5C9Q3ON\Tra My\* X

Cube: Store DW

Measure Group: <All>

```
-- Câu 8. Số người bị thương do các cơn lốc xoáy có độ F/EF ở mức độ nhẹ đến trung bình
SELECT NON EMPTY {[Measures].[SoLuongNguoiBiThuong]} ON COLUMNS,
NON EMPTY {[Dim Size].[Mag].[Mag]} ON ROWS
FROM (SELECT {[Dim Size].[Mag].[Mag].&[0], [Dim Size].[Mag].[Mag].&[1]}) ON COLUMNS
FROM [Store DW];
```

	SoLuongNguoiBiThuong
0	856
1	7182

Query executed successfully.

DESKTOP-5C9Q3ON | DESKTOP-5C9Q3ON\Tra My | SSAS\_Us\_Tornado | 00:00:01

### 3.9 Trung bình một ngày có bao nhiêu người chết và bị thương ở bang MA

MDXQuery1.mdx -> 5C9Q3ON\Tra My\* X

Cube: Store DW

Measure Group: <All>

```
-- Câu 9. Trung bình 1 ngày có bao nhiêu người chết và bị thương ở bang MA
SELECT
NON EMPTY {[Measures].[AvgNguoiTuVongTrong1Ngay]}, [Measures].[AvgNguoiBiThuongTrong1Ngay] ON COLUMNS,
NON EMPTY {[Dim Location].[St].[MA]} ON ROWS
FROM [Store DW];
```

	AvgNguoiTuVongTrong1Ngay	AvgNguoiBiThuongTrong1Ngay
MA	8.53658536585360E-03	0.124065040650407

Query executed successfully.

DESKTOP-5C9Q3ON | DESKTOP-5C9Q3ON\Tra My | SSAS\_Us\_Tornado | 00:00:01

### 3.10 Tìm các bang có các cơn lốc xoáy gây thương tích nhưng không gây tử vong trong năm 2001

```
--Câu 10. Tìm các bang có các con lộc xoáy gây thương tích nhưng không gây tử vong trong năm 2001
SELECT NON EMPTY { [Measures].[SoLuongNguoiBiThuong] } ON COLUMNS, NON EMPTY
{ ([Dim Date].[Yr].>[Dim Location].[St].[St]) } ON ROWS
FROM ( SELECT ( FILTER( [Dim Location].[St].[St], [Measures].[SoLuongNguoiBiThuong] > 0 and [Measures].[SoLuongNguoiTuVong]=0) )
ON COLUMNS FROM ( SELECT ( { [Dim Date].[Yr].>[2001] } ) ON COLUMNS
FROM [Store DW]);
```

	SoLuongNguoiBiThuong
2001 CO	13
2001 CT	1
2001 GA	23
2001 IL	3
2001 KY	14
2001 ME	1
2001 MI	8
2001 MN	13
2001 NC	3
2001 ND	1
2001 NE	9
2001 OH	1
2001 SC	39
2001 TX	49
2001 VA	4
2001 WY	2

Query executed successfully.

### 3.11 Tên các tiêu bang có số lượng người bị thương >5500 người

```
--Câu 11. Tên các tiêu bang có số lượng người bị thương > 5500 người
SELECT NON EMPTY ( [Measures].[SoLuongNguoiBiThuong] ) ON COLUMNS,
FILTER( [Dim Location].[St].[St], [Measures].[SoLuongNguoiBiThuong] > 5500 ) ON ROWS
FROM [Store DW];
```

	SoLuongNguoiBiThuong
AL	8672
MS	6452
OK	5997
TX	9412

Query executed successfully.

### 3.12 Thống kê số lượng lôc xoáy theo từng tháng và từng loại độ đo F/EF

```
--Câu 12. Thống kê số lượng lốc xoáy theo từng tháng và từng loại độ đo F/EF
SELECT NON EMPTY {[Measures].[Id Distinct Count]} ON COLUMNS,
NON EMPTY {[Dim Date].[Mo].[Mo], [Dim Size].[Mag].[Mag]} ON ROWS
FROM [Store DW];
```

		Id Distinct Count
12	5	2
12	9	8
2	0	621
2	1	812
2	2	382
2	3	108
2	4	21
2	5	1
3	0	1606
3	1	1648
3	2	866
3	3	288
3	4	67
3	5	4
3	-9	35
4	0	3626
4	1	3510
4	2	1669
4	3	527
4	4	170
4	5	20
4	-9	51
5	0	7318
5	1	4598

Query executed successfully.

### 3.13 Thống kê top 5 bang xảy ra lốc xoáy ít nhất

```
--Câu 13. Thống kê top 5 bang xảy ra lốc xoáy ít nhất
SELECT NON EMPTY {[Measures].[Id Distinct Count]} ON COLUMNS, NON EMPTY
BOTTOMCOUNT ([Dim Location].[St].[St], 5,[Measures].[Id Distinct Count])ON ROWS
FROM [Store DW];
```

		Id Distinct Count
VI		1
DC		3
AK		4
RI		13

Query executed successfully.

### 3.14 Thống kê tổng số người bị thương theo từng năm và từng bang

--Câu 14. Thống kê tổng số người tử vong theo từng năm và từng bang

```
SELECT NON EMPTY {[Measures].[SoLuongNguoiTuVong]} ON COLUMNS,
NON EMPTY {[Dim Date].[Yr].[Yr], [Dim Location].[St].[St]} ON ROWS
FROM [Store DW];
```

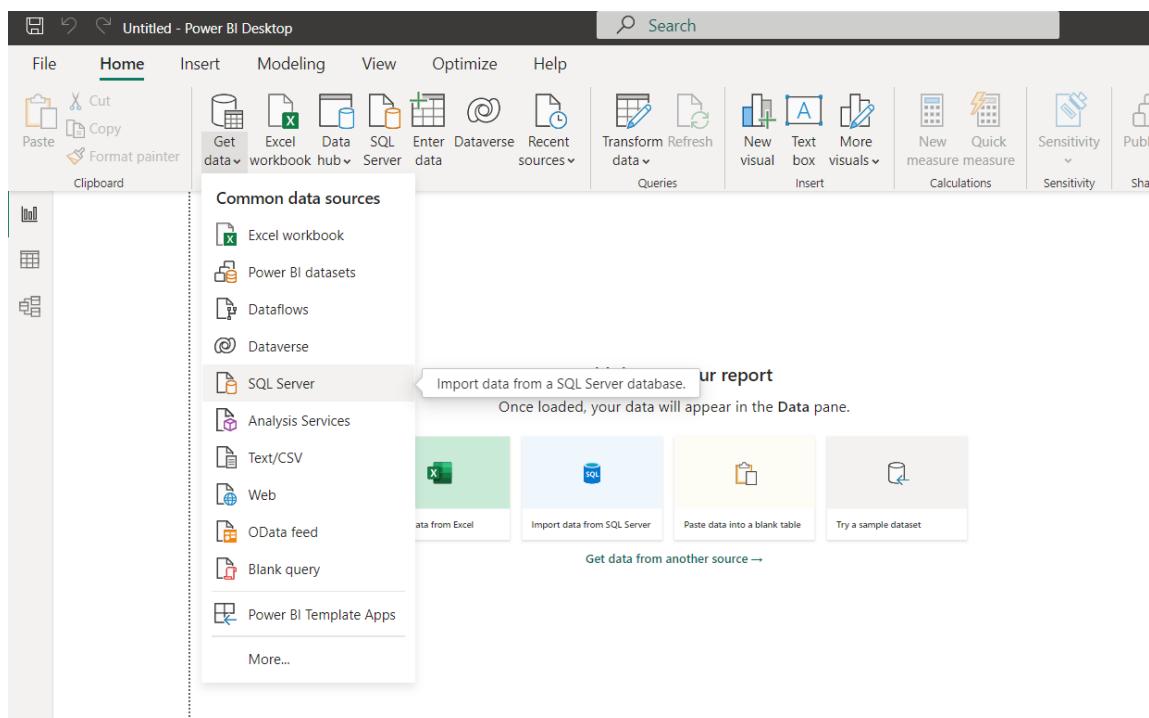
	Messaaes	Results
	SoLuongNguoiTuVong	
1950	AL	0
1950	AR	2
1950	CO	0
1950	CT	0
1950	FL	0
1950	GA	0
1950	IA	0
1950	IL	3
1950	IN	0
1950	KS	1
1950	KY	0
1950	LA	29
1950	MD	0
1950	MN	0
1950	MO	0
1950	MS	6
1950	NC	0
1950	ND	0
1950	NE	0
1950	NM	0
1950	OH	0
1950	OK	6
1950	PA	0
1950	SC	0
1950	SD	0
1950	TN	9
1950	TX	11
1950	WI	3

Query executed successfully.

## 4. Power BI – Excel

### 4.1 Tạo project Power BI

- Chọn Get Data → SQL Server



- Nhập Server name và tên Database



- Chọn bảng Dim và Fact → Chọn Load

Navigator

Display Options ▾

DESKTOP-N26UF3C: StoreDataWareHouse [7]

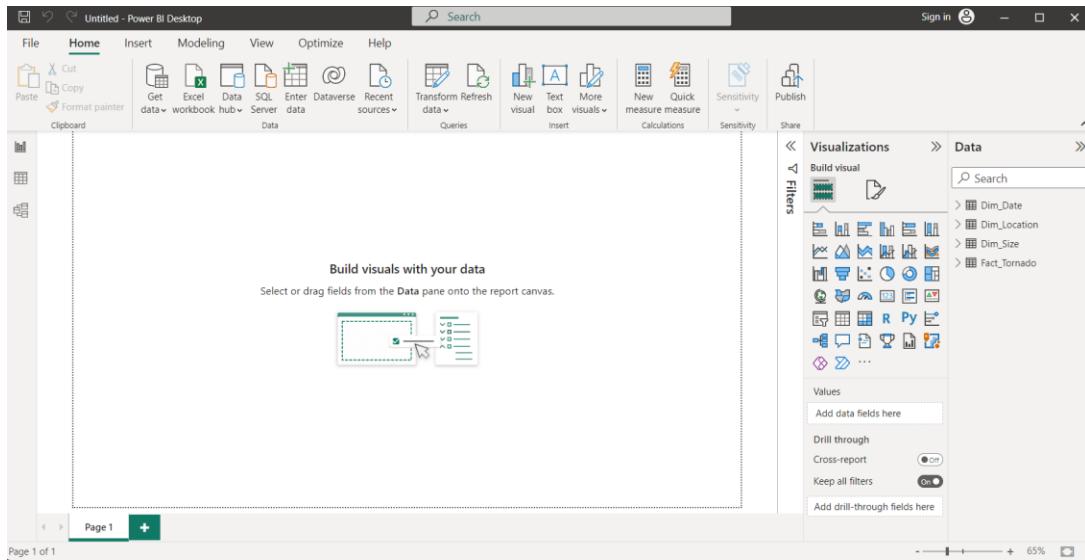
- Dim\_Date
- Dim\_Location
- Dim\_Size
- Fact\_Tornado
- sysdiagrams
- us\_tornado
- fn\_diagramobjects

Fact\_Tornado

id	date_id	inj	fat	location_id	size_id
1	1/3/1950	3	0	83393	132530
2	1/3/1950	3	0	99077	132067
3	1/3/1950	1	0	111894	99560
4	1/13/1950	1	1	69921	132019
5	1/25/1950	0	0	83393	122425
6	1/25/1950	5	0	99077	123621
7	1/26/1950	2	0	122554	123576
8	2/11/1950	0	0	122554	123628
9	2/11/1950	5	0	122554	122425
10	2/11/1950	6	0	122554	123501
11	2/11/1950	12	1	122554	132380
12	2/12/1950	0	0	69921	122425
13	2/12/1950	0	0	69921	122425
14	2/12/1950	0	0	93060	102825
15	2/12/1950	10	5	93060	122425
16	2/12/1950	25	5	93060	132040
17	2/12/1950	77	18	93060	134478
18	2/12/1950	0	0	101504	99560
19	2/12/1950	2	3	101504	122428
20	2/12/1950	0	0	122554	110666
21	2/12/1950	32	0	122554	99550
22	2/12/1950	0	0	122554	122425
23	2/12/1950	8	1	122554	127048
24	2/12/1950	15	3	122554	132035

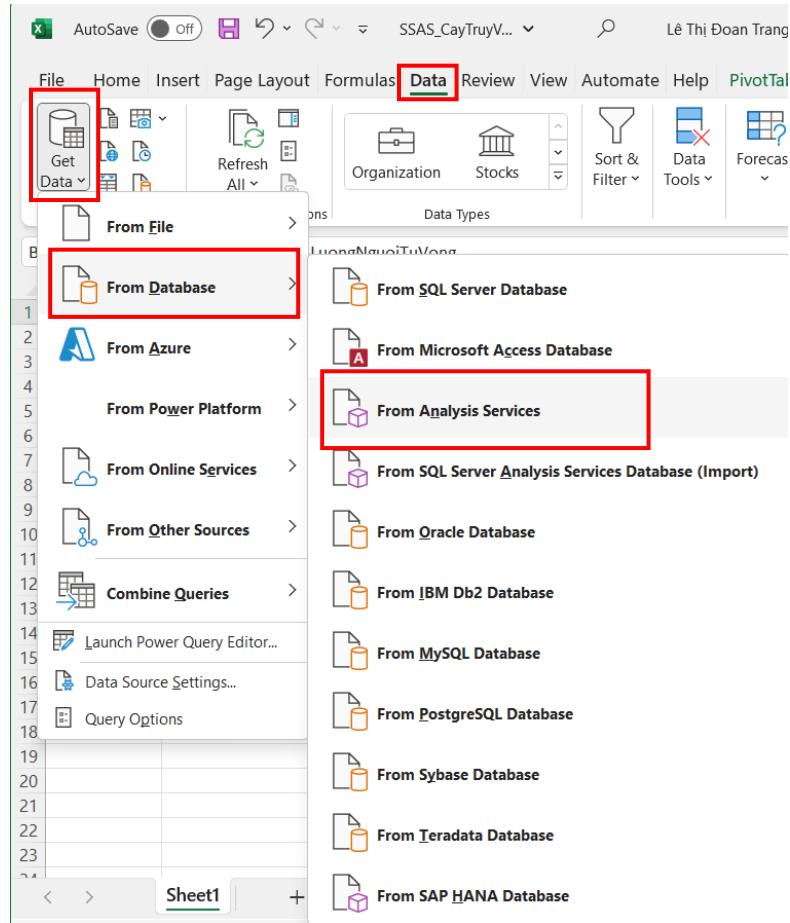
Select Related Tables Load Transform Data Cancel

- Load dữ liệu thành công

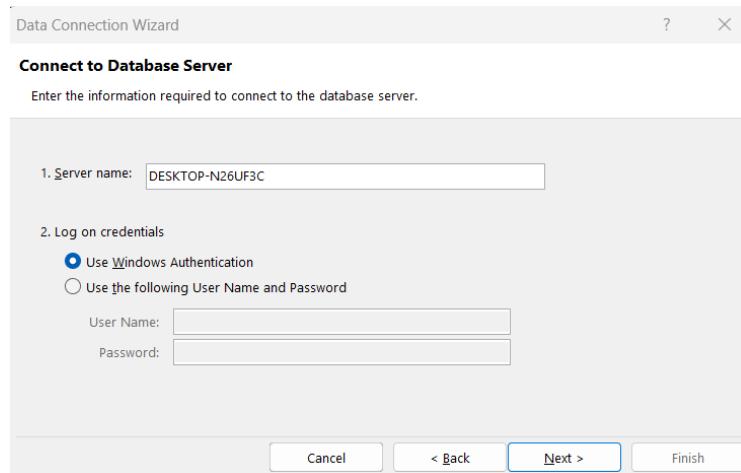


## 4.2 Tạo project Excel

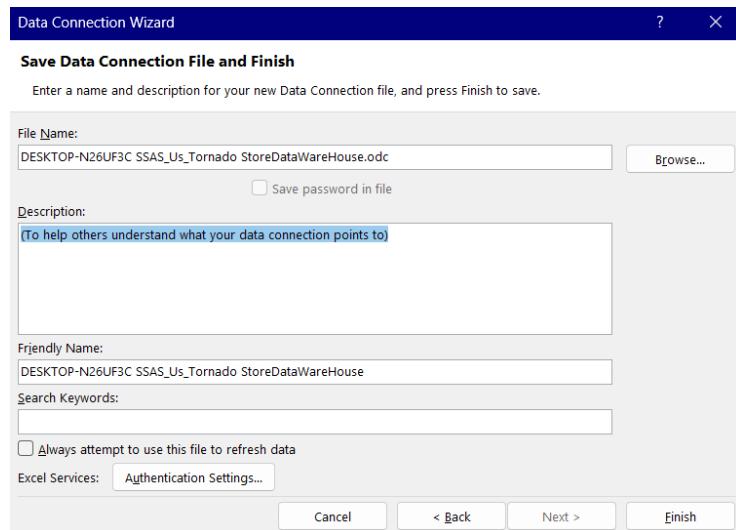
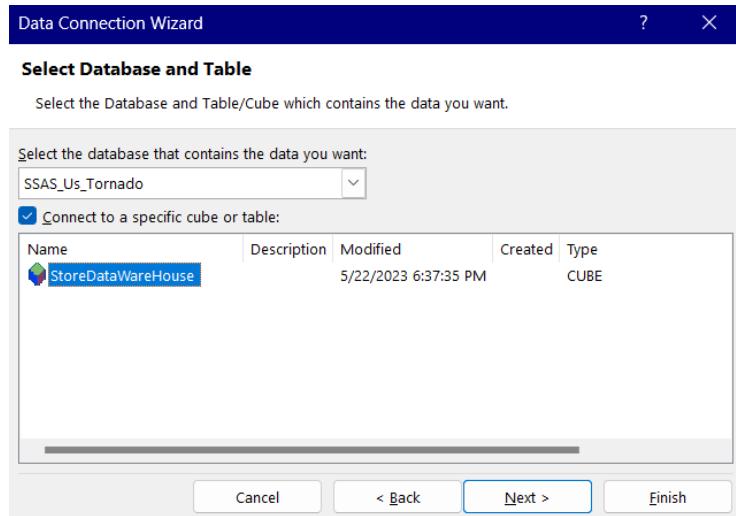
- Chọn Data → Chọn Get Data → Chọn From Database → Chọn From Analysis Service



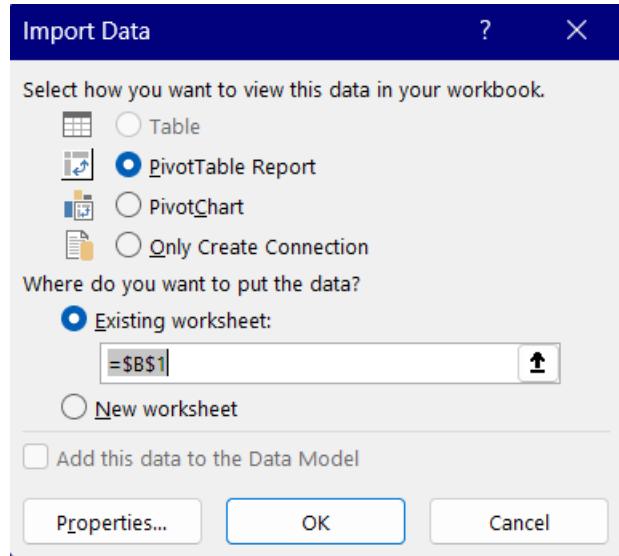
- Nhập Server name của SQL Server → Chọn Next



- Chọn Database → Chọn Next → Chọn Finish



- Chọn PivotTable Report → Chọn ô để bắt đầu → Chọn OK



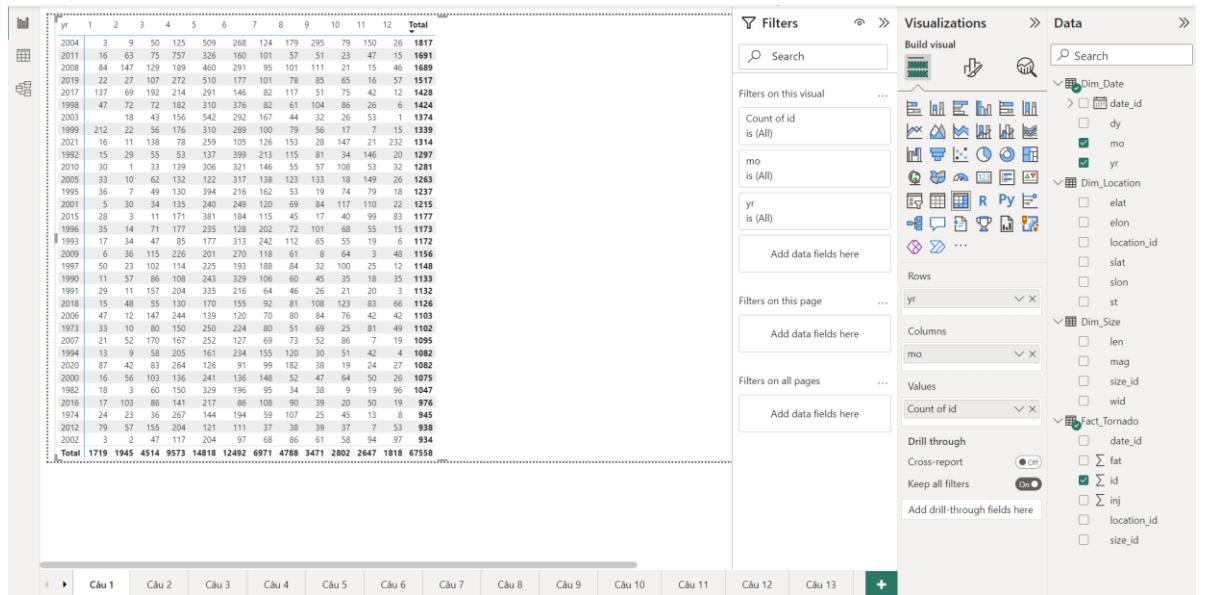
- Tạo thành công

The screenshot shows a Microsoft Excel spreadsheet with a PivotTable Fields pane open on the right side. The PivotTable Fields pane lists fields from the "Fact Tornado" data source. Under the "Values" category, the field "SoLanXayRaLocXoayTrongNgay" is checked. The main worksheet area shows a PivotTable with the label "PivotTable3" and a message "To build a report, choose fields from the PivotTable Field List". The ribbon at the top has the "PivotTable Analyze" tab selected. The status bar at the bottom indicates "Ready" and "Accessibility: Investigate".

### 4.3 Truy vấn

#### 4.2.1 Tổng số lượng lốc xoáy theo từng tháng của mỗi năm

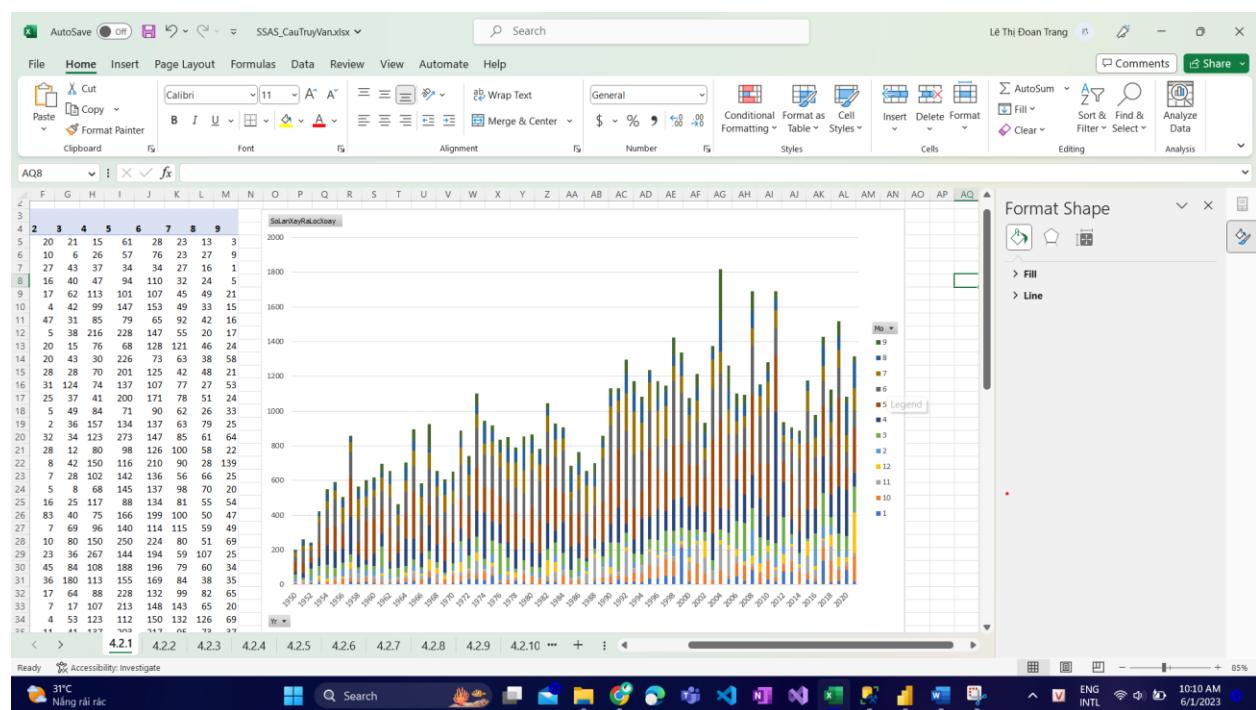
- Power BI



- Excel

## IS217.N22.HTCL – Kho dữ liệu và OLAP

The screenshot shows a Microsoft Excel spreadsheet titled "SSAS\_CayTruyVan.xlsx". The PivotTable Fields pane is open on the right side, showing fields for "Mo" (Month) and "Yr" (Year). The data grid below contains a table with columns for Year (1950-1968), Month (1-9), and various numerical values. The PivotTable Fields pane also includes sections for Filters, Columns, Rows, and Values.



→ Nhìn vào biểu đồ có thể thấy từ năm 2000 đến 2020 số trận lốc xoáy tăng đáng kể, đỉnh điểm là năm 2004 với số lượng lốc xoáy cao nhất là 1817 trận lốc.

#### 4.2.2 Tổng số lượng lốc xoáy theo từng năm

- Power BI

The screenshot shows two Power BI visualizations. On the left is a table titled "Count of id by yr" with a single column "yr" and a single row "Count of id". The data is as follows:

yr	Count of id
2004	1817
2011	1691
2009	1689
2019	1517
2017	1428
1998	1424
2003	1374
1999	1339
2021	1314
1992	1297
2010	1281
2005	1263
1995	1237
2001	1215
2015	1177
1996	1173
1993	1172
2000	1156
1997	1148
1990	1133
1991	1132
2018	1126
2006	1103
1973	1102
2007	1095
1994	1082
2020	1082
2000	1075
1982	1047
2016	976
Total	67558

On the right is a bar chart titled "Count of id by yr and yr" showing the count of IDs for each year from 1951 to 2021. The Y-axis represents "Count of id" ranging from 0 to 2,000. The X-axis represents "yr". The chart uses a color-coded legend where each color corresponds to a specific year, though many years share the same color.

The Power BI interface includes a "Filters" pane, a "Visualizations" pane, and a "Data" pane showing the schema of the Dim\_Date, Dim\_Location, and Fact\_Tornado tables.

This screenshot shows a bar chart titled "Count of id by yr and yr" with the same data as the previous screenshot. The Y-axis is "Count of id" (0 to 2,000) and the X-axis is "yr" (1951 to 2021). The chart displays the count of IDs for each year, with bars grouped by year. A legend on the right maps colors to specific years, such as 1951 (blue), 1952 (orange), 1953 (purple), etc.

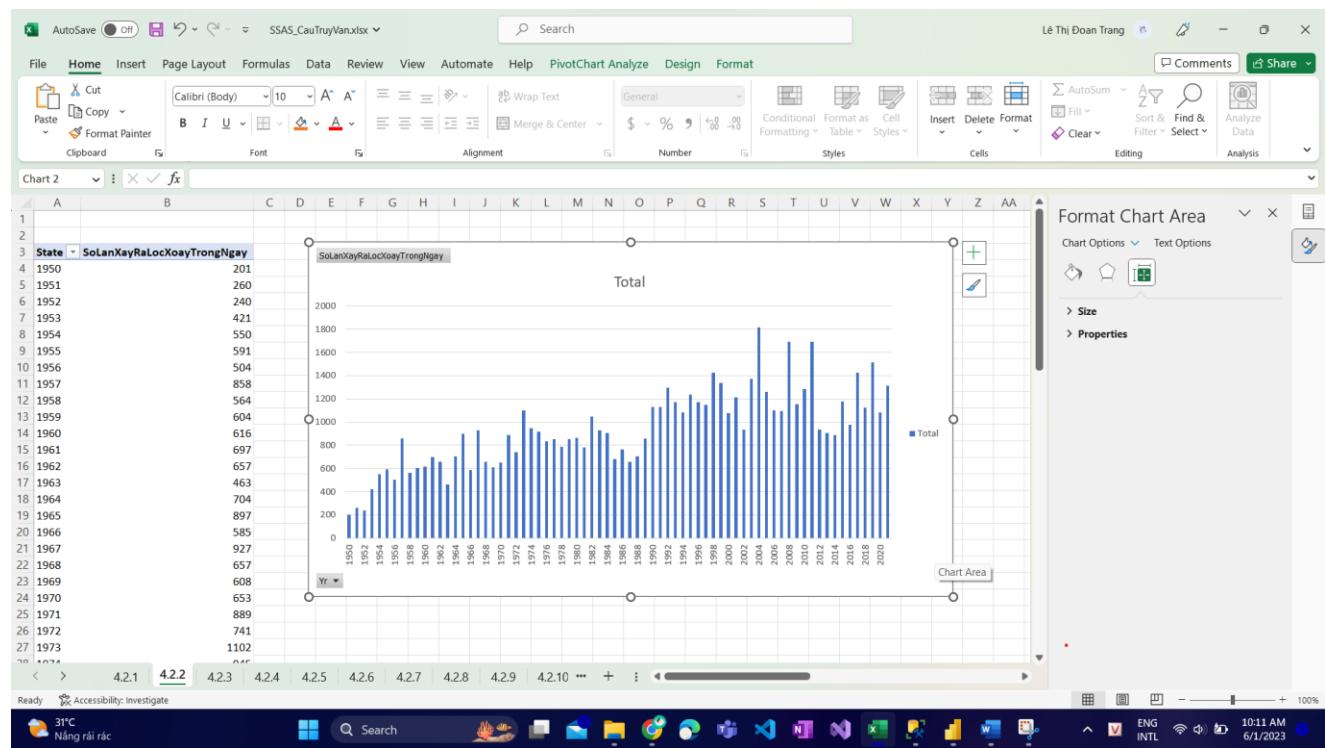
The Power BI interface includes a "Filters" pane, a "Visualizations" pane, and a "Data" pane showing the schema of the Dim\_Date, Dim\_Location, and Fact\_Tornado tables.

- Excel

## IS217.N22.HTCL – Kho dữ liệu và OLAP

SSAS\_CayTruyVan.xlsx

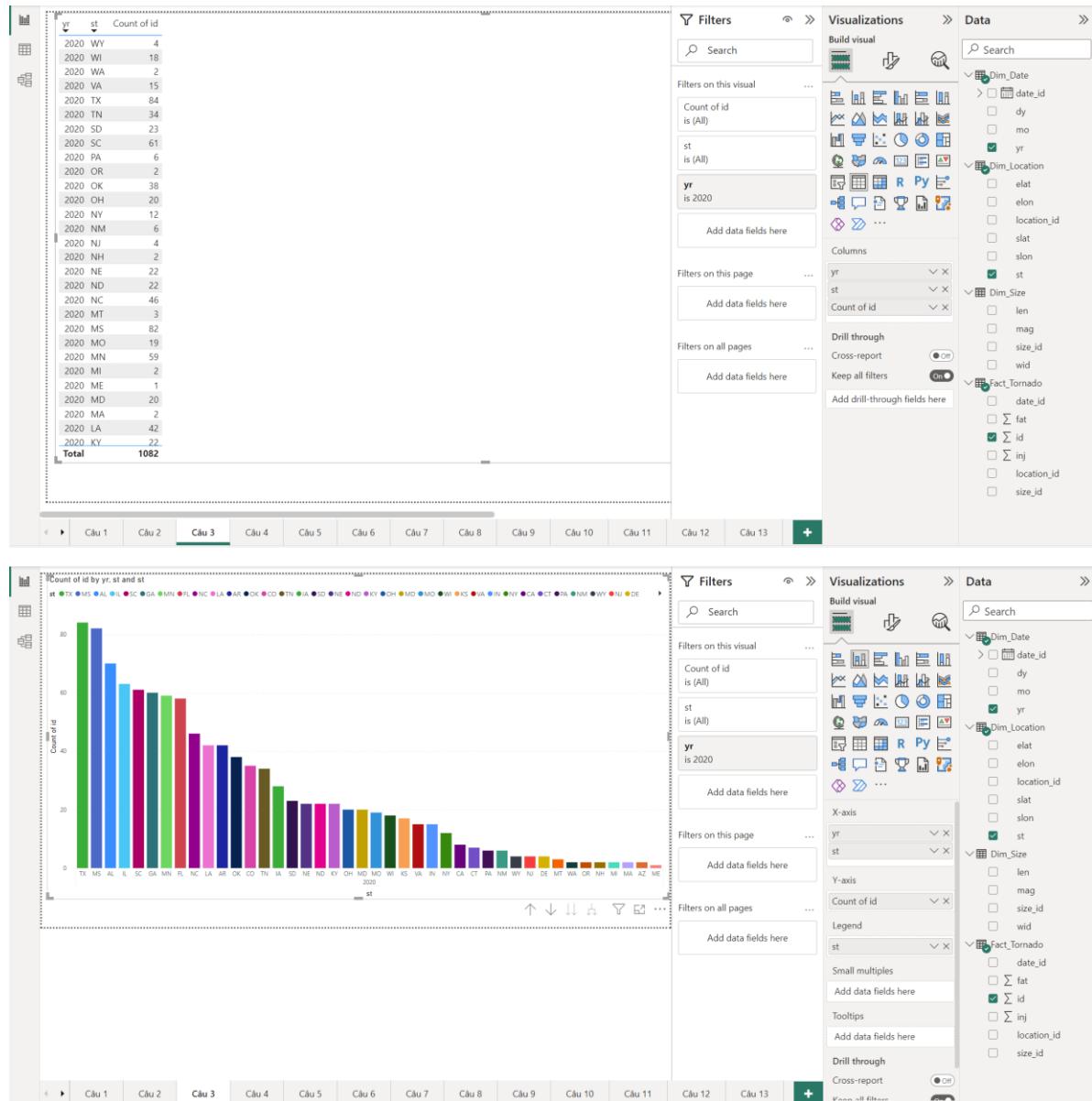
Year	SoLuongLocXoay
1950	201
1951	260
1952	240
1953	421
1954	550
1955	591
1956	504
1957	858
1958	564
1959	604
1960	616
1961	697
1962	657
1963	463
1964	704
1965	897
1966	585
1967	927
1968	657
1969	608
1970	653
1971	889



→ Từ biểu đồ có nhận xét số lượng lốc xoáy năm 2004 cao nhất với 1817 trận lốc và năm 1950 thấp nhất 201 trận lốc, trong suốt thập kỷ qua khí hậu Trái Đất đã thay đổi rất nhiều và theo chiều hướng xấu. Nhận xét trên cho thấy sự biến đổi khí hậu như vậy cũng ảnh hưởng tới việc xảy ra lốc xoáy thường xuyên hơn.

### 4.2.3 Số lần xảy ra lốc xoáy ở từng bang trong năm 2020

- Power BI

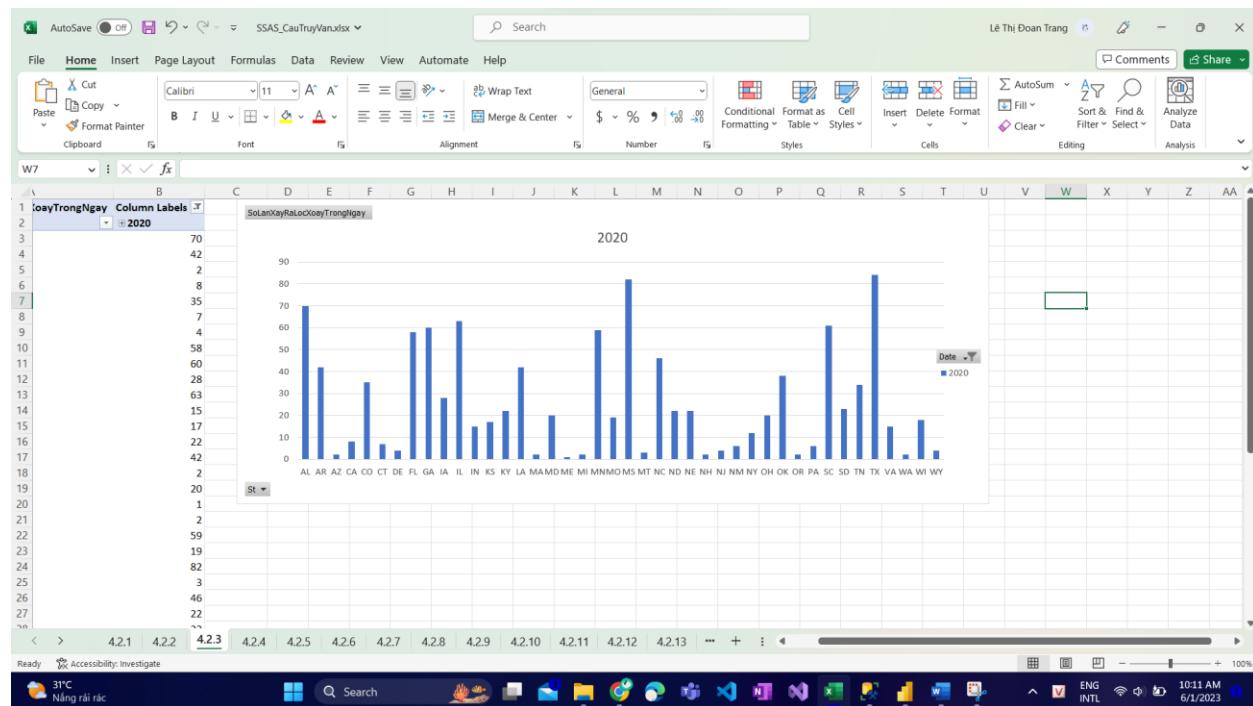


- Excel

## IS217.N22.HTCL – Kho dữ liệu và OLAP

The screenshot shows a Microsoft Excel window with the following details:

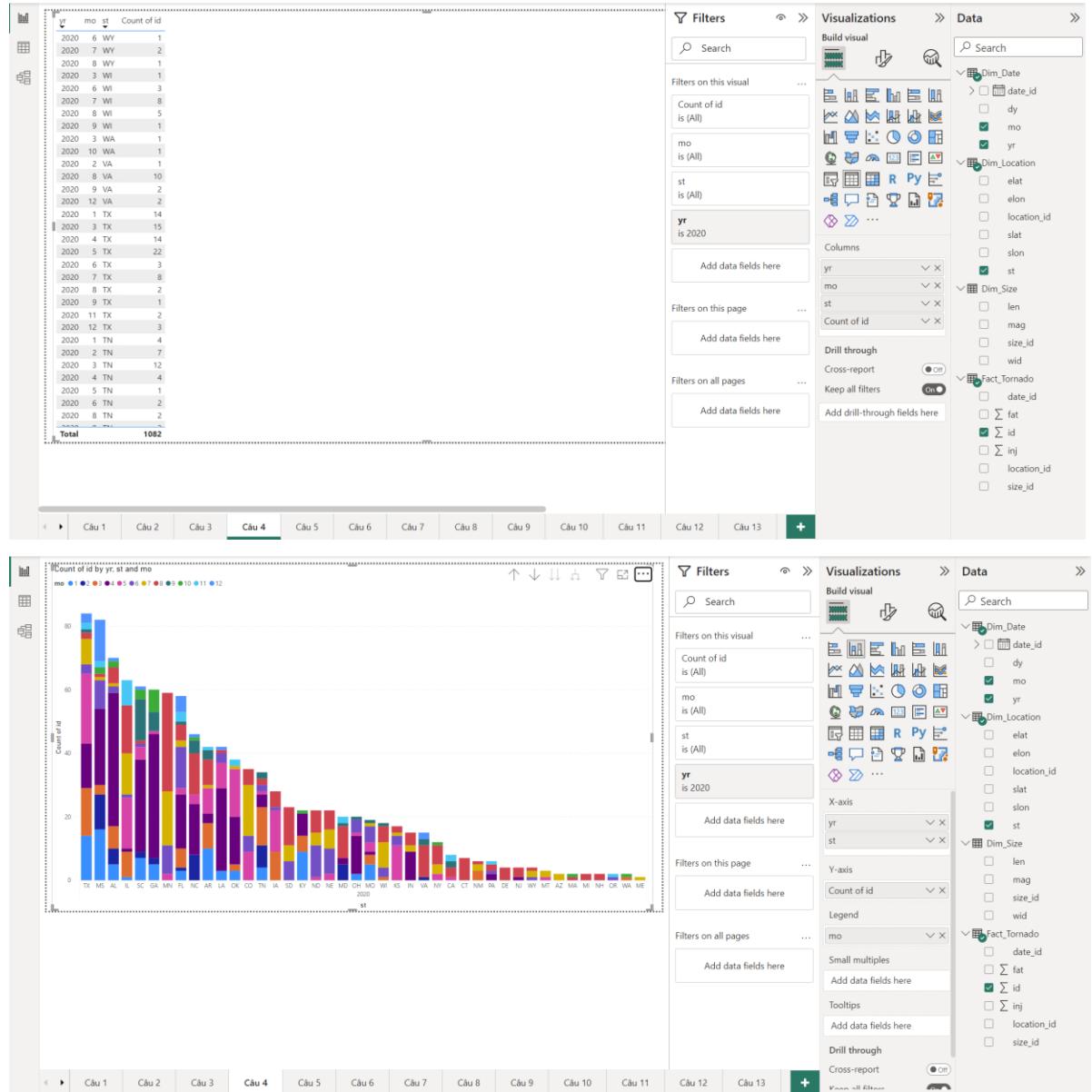
- PivotTable Fields pane:** Located on the right side of the interface. It displays the following fields:
  - Columns:** Yr (selected)
  - Filters:** None
  - Rows:** St (selected)
  - Values:** SoLanXayRaLocXoay
- Data Grid:** The main area shows a table with columns A through N. Row 3 contains the header "SoLanXayRaLocXoay" and "Year". Row 4 contains the header "State" and "2020". Rows 5 to 25 list various US states with their corresponding values.
- Toolbar:** Standard Excel ribbon tabs (File, Home, Insert, Page Layout, Formulas, Data, Review, View) and ribbon icons.
- Status Bar:** Shows the file name "SSAS\_CayTruyVan.xlsx" and the user "Lê Thị Đoàn Trang".



→ Từ biểu đồ thấy được ở những năm gần đây nhất là năm 2020, Texas là tiểu bang có số trận xoáy xảy ra cao nhất với 84 trận lốc xoáy trong một năm.

#### 4.2.4 Số lần xảy ra lốc xoáy ở từng bang theo từng tháng trong năm 2020

- Power BI



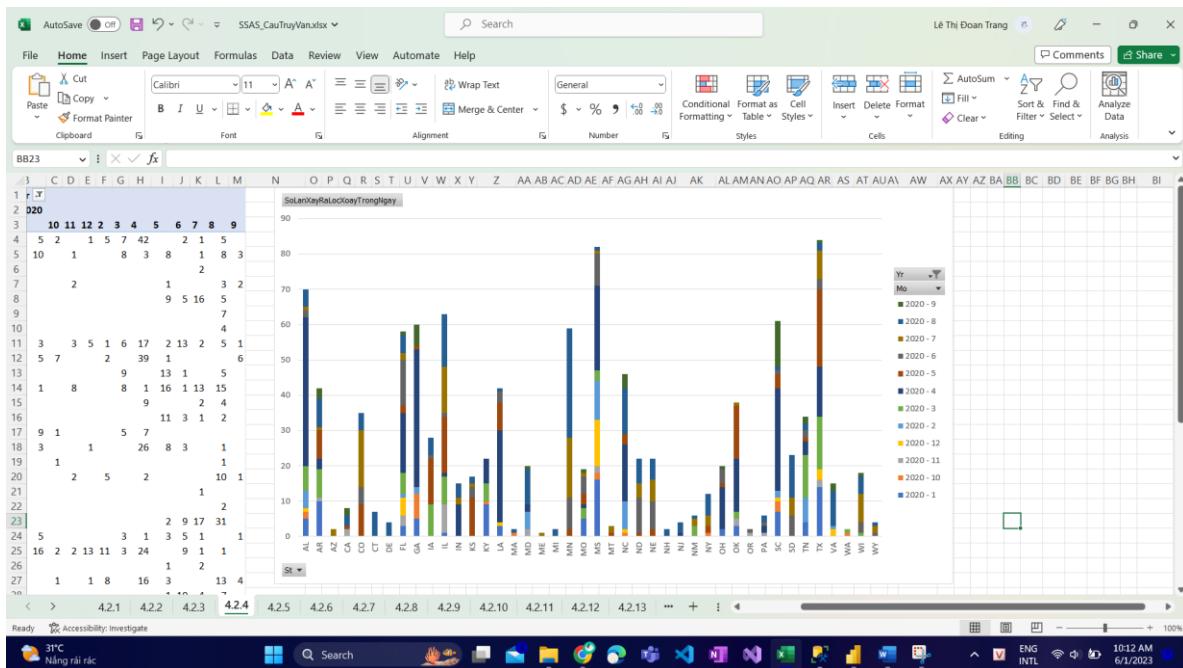
- Excel

## IS217.N22.HTCL – Kho dữ liệu và OLAP

The screenshot shows a Microsoft Excel window with the following details:

- File Tab:** AutoSave is off.
- PivotTable Fields pane:**
  - Show fields: (All)
  - Search bar.
  - Fields listed under "More fields" (Date Id, Dy, Mo, Yr) with Mo and Yr checked.
  - Drag fields between areas below.
  - Filters section: Columns (Yr, Mo), Rows (St), Values (SoLanXayRaLocXoay).
  - Defer Layout Update checkbox and Update button.
- PivotTable Grid:**

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
3	SoLanXayRaLocXoay	Year	2020															
4	State	1	10	11	12	2	3	4	5	6	7	8	9					
6	AL		5	2	1	5	7	42	2	1	5							
7	AR		10	1		8	3	8	1	8	3							
8	AZ										2							
9	CA										1	3	2					
10	CO									9	5	16	5					
11	CT												7					
12	DE												4					
13	FL		3	3	5	1	6	17	2	13	2	5	1					
14	GA		5	7		2	39	1				6						
15	IA						9		13	1	5							
16	IL		1	8		8	1	16	1	13	15							
17	IN						9		2	4								
18	KS								11	3	1	2						
19	KY		9	1			5	7										
20	LA		3		1		26	8	3		1							
21	MA				1						1							
22	MD		2	5		2				10	1							
23	ME									1								
24	MI									2								
25	MN								2	9	17	31						



→ Nhìn vào biểu đồ của powerBI các cột có phần màu tím đậm, màu hồng và ở excel là cột màu xanh navy, nâu đát là chiếm số lượng nhiều. Các màu đó lượt là tháng 4, 5 trong năm 2020, điều đó cho thấy lốc xoáy ở Hoa Kỳ thường xảy ra ở hai tháng này.

#### 4.2.5 Tìm top 3 bang có tổng số người tử vong cao nhất

- Power BI

The screenshot displays two Power BI visualizations and their corresponding filter panes.

**Top Visualization:**

- Type:** Table
- Columns:** st, yr, Sum of fat
- Data:**

st	yr	Sum of fat
TX	1953	150
MI	1953	127
MA	1953	94
<b>Total</b>		<b>371</b>

**Bottom Visualization:**

- Type:** Pie Chart
- Legend:** st (TX, MI, MA)
- Data:**

State	Percentage
TX	40.43%
MI	34.23%
MA	25.34%

**Filter Panes:**

- Common Filters (Both Visualizations):**
  - st: top 3 by Sum of fat
  - Sum of fat: is (All)
  - yr: is 1953
  - Add data fields here
- Filters on this page (Top Visualization):**
  - st: is TX, MI, MA
  - yr: is 1953
  - Sum of fat: is 150, 127, 94
  - Add data fields here
- Filters on all pages (Bottom Visualization):**
  - Sum of fat: is 150, 127, 94
  - Add data fields here
- Drill through (Bottom Visualization):**
  - Cross-report: Off
  - Keep all filters: On
  - Add drill-through fields here

**Data Source (Right Panel):**

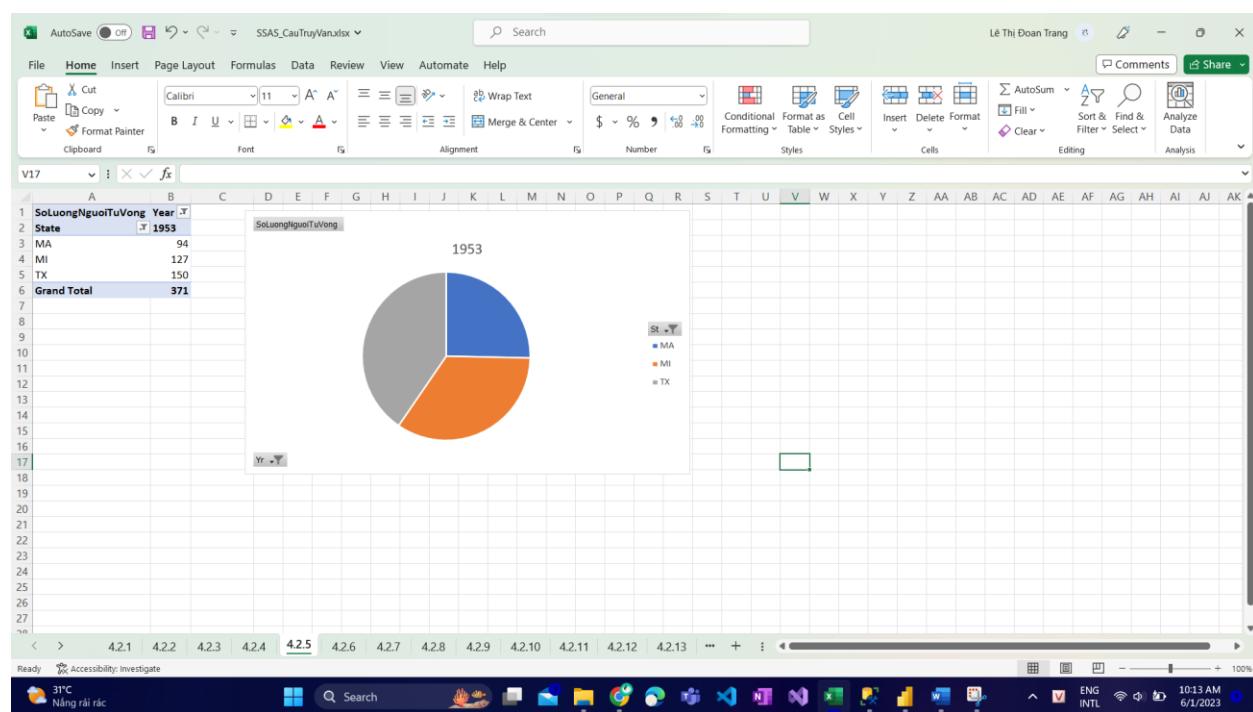
- Dim\_Date:** date\_id, dy, mo, yr
- Dim\_Location:** elat, elon, location\_id, slat, slon, st
- Dim\_Size:** len, mag, size\_id, wid
- Fact\_Tornado:** date\_id, fat, id, inj, location\_id, size\_id

- Excel

## IS217.N22.HTCL – Kho dữ liệu và OLAP

The screenshot shows a Microsoft Excel window with the following details:

- PivotTable Fields pane:** Located on the right side of the screen. It displays the "Dim Location" dimension under the "Yr" field, with "State-Latitude" and "State-Longitude" listed as children. A "More Fields" button is visible.
- Data Table:** In the main worksheet area, there is a table with columns "SoLuongNguoiTuVong" and "Year". The "Year" column contains the value "1953". The "SoLuongNguoiTuVong" column lists values for four states: TX (150), MA (94), MI (127), and another entry for TX (150).



→ Ở đây dùng biểu đồ tròn có thể dễ thấy được 3 tiểu bang này có số lượng lộc xoáy chiếm khá lớn đặc biệt là bang Texas có tận 150 trận lộc chiếm tới 40,43%.

#### 4.2.6 Tìm các bang với số lượng lốc xoáy tại bang đó có mức độ sức tàn phá từ "Nghiêm trọng" trở lên

- Power BI

The screenshot shows a Power BI interface with a table visualization on the left and the Power BI ribbon on the right.

**Table Visualization:**

st	3	4	5	Total
TX	324	49	6	<b>379</b>
OK	195	51	8	<b>254</b>
KS	188	38	6	<b>232</b>
MS	149	30	4	<b>183</b>
AL	137	35	7	<b>179</b>
AR	144	29		<b>173</b>
IA	112	40	6	<b>158</b>
IL	106	29	2	<b>137</b>
MO	102	34	1	<b>137</b>
NE	94	31	1	<b>126</b>
IN	91	26	2	<b>119</b>
TN	89	28	1	<b>118</b>
LA	97	9	1	<b>107</b>
KY	74	17	1	<b>92</b>
GA	70	10		<b>80</b>
MN	51	21	2	<b>74</b>
WI	54	17	3	<b>74</b>
SD	56	9	1	<b>66</b>
OH	45	15	3	<b>63</b>
MI	41	15	2	<b>58</b>
ND	36	11	2	<b>49</b>
SC	33	11		<b>44</b>
FL	40	2		<b>42</b>
NC	35	7		<b>42</b>
VA	34	2		<b>36</b>
PA	28	6		<b>34</b>
CO	23			<b>23</b>
<b>Total</b>	<b>2536</b>	<b>581</b>	<b>59</b>	<b>3176</b>

**Power BI Ribbon:**

- Filters
- Visualizations
- Data

The screenshot shows a Power BI interface with a bar chart visualization on the left and the Power BI ribbon on the right.

**Bar Chart Visualization:**

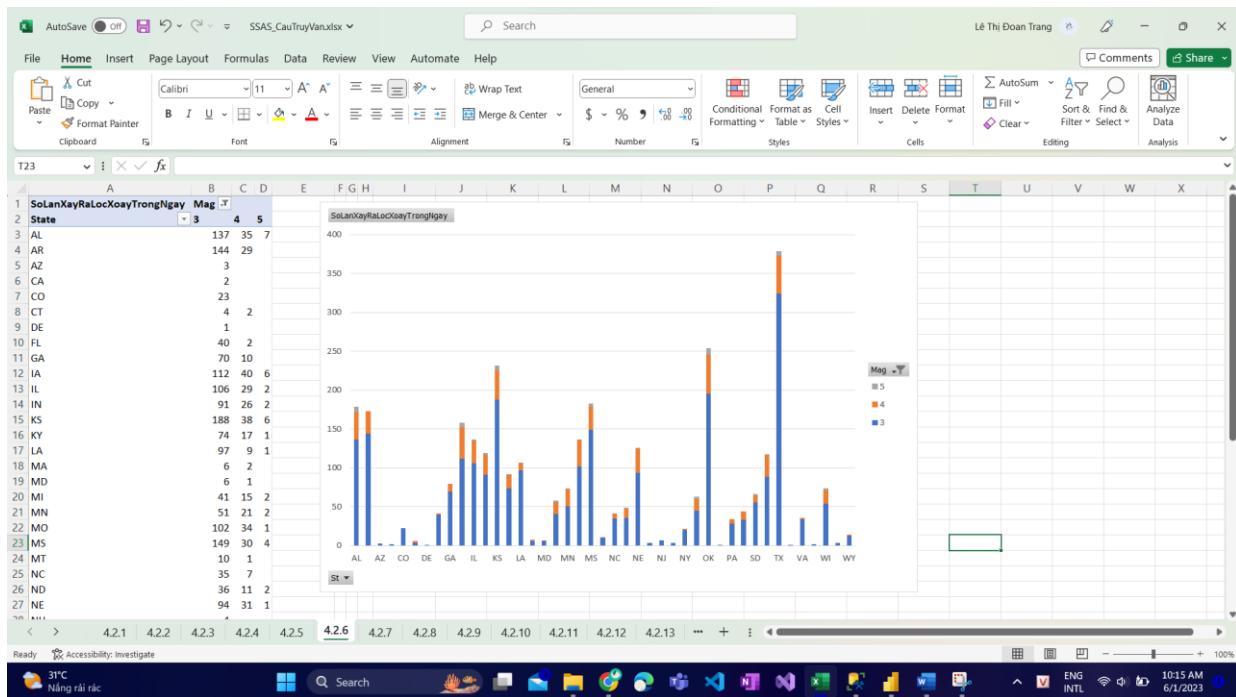
A bar chart titled "Count of id by st and mag" showing the count of IDs for each state (st) across different magnitude ranges (mag). The Y-axis is labeled "Count of id" and ranges from 0 to 400. The X-axis is labeled "st". The legend indicates three categories: mag 3 (blue), mag 4 (dark blue), and mag 5 (orange).

**Power BI Ribbon:**

- Filters
- Visualizations
- Data

- Excel

The screenshot shows a Microsoft Excel window with a PivotTable named "SoLanXayRaLocXoayTrongNgay" in the range A3:J23. The PivotTable Fields pane on the right shows fields for "Dim Size" (Size, Mag, Len, Wid), "Filters" (Columns, Size), "Rows" (St), and "Values" (SoLanXayRaLocXoayTrongNgay). The PivotTable grid displays data for various states (AL, AR, AZ, CA, CO, CT, DE, FL, GA, IA, IL, IN, KS, KY, LA, MA, MD, MI, MN, MO, MT, NC, ND, NE, NJ, NY, OK, PA, SD, TX, VA, WI, WY) across four categories (3, 4, 5, Grand Total).



→ Từ biểu đồ thấy được số lượng lốc xoáy ở mức 3 xảy ra rất nhiều với 2536 trận lốc đáy là mức có sức gió lên tới 332km/h rất nguy hiểm. Mức 4 tuy không nhiều bằng nhưng đa số cũng trên 10 cơn lốc 1 năm, mức này có sức gió vô vùng lớn có thể lên tới 418km/h mà trung bình 1 tháng xảy ra 1 lần thì cũng gây ra nhiều thương vong không kém.

#### 4.2.7 Tìm các năm có các cơn lốc xoáy có độ đo F/EF cao nhất và gây tử vong

- Power BI

The screenshot shows the Power BI interface with two main components:

- Table View:** On the left, a table titled "Sum of fat" is displayed with columns "yr" and "mag". The data shows various years from 2011 to 1992, with corresponding values for "mag" and "Sum of fat". A total row at the bottom shows a sum of 1347.
- Visualizations and Filters:** On the right, the Power BI ribbon is visible with tabs for "Filters", "Visualizations", and "Data". The "Filters" tab is active, showing filters for "mag" (is 5) and "Sum of fat" (is greater than 0). The "Visualizations" tab shows a bar chart titled "Sum of fat by yr, mag and yr". The "Data" tab displays the schema of the data sources, including Dim\_Date, Dim\_Location, Dim\_Size, and Fact\_Tornado tables.

The screenshot shows the Power BI interface with two main components:

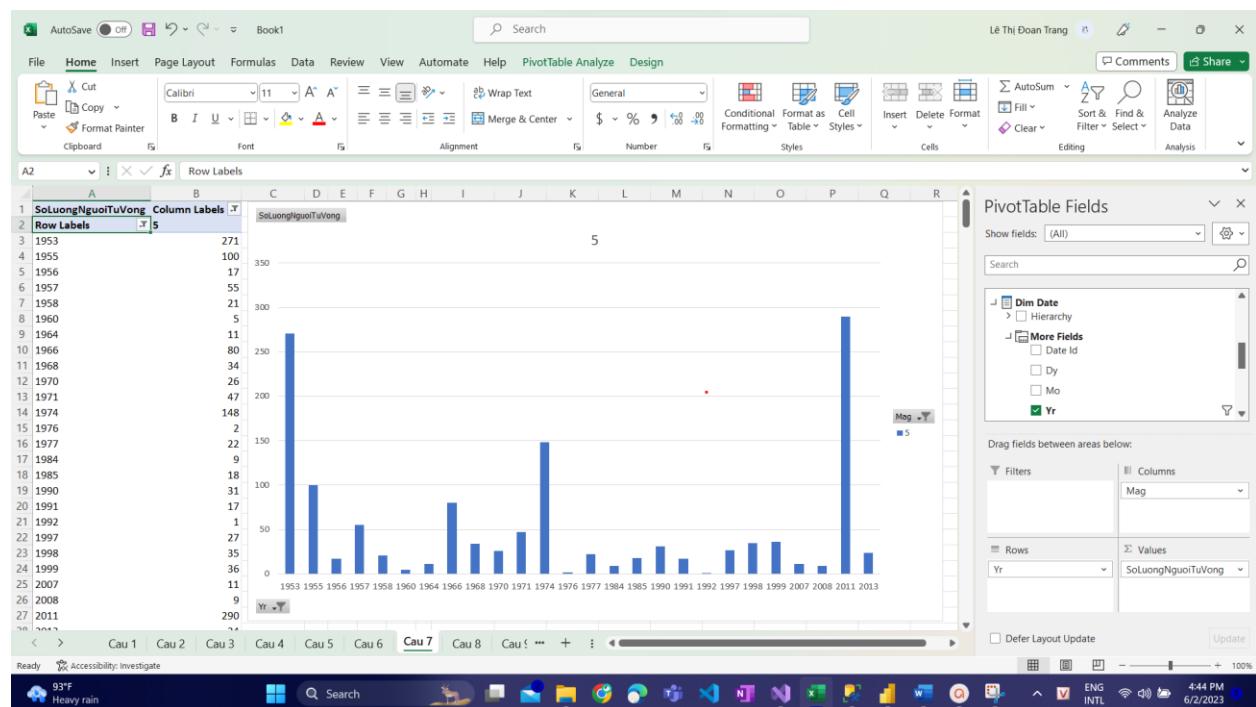
- Bar Chart View:** On the left, a bar chart titled "Sum of fat by yr, mag and yr" is displayed. The Y-axis is labeled "Sum of fat" and ranges from 0 to 300. The X-axis is labeled "mag" and shows categories for each year from 2011 to 1992. The bars are colored according to the "mag" value.
- Visualizations and Filters:** On the right, the Power BI ribbon is visible with tabs for "Filters", "Visualizations", and "Data". The "Filters" tab is active, showing filters for "mag" (is 5) and "Sum of fat" (is greater than 0). The "Visualizations" tab shows the same bar chart. The "Data" tab displays the schema of the data sources, including Dim\_Date, Dim\_Location, Dim\_Size, and Fact\_Tornado tables.

- Excel

## IS217.N22.HTCL – Kho dữ liệu và OLAP

The screenshot shows a Microsoft Excel window with the following details:

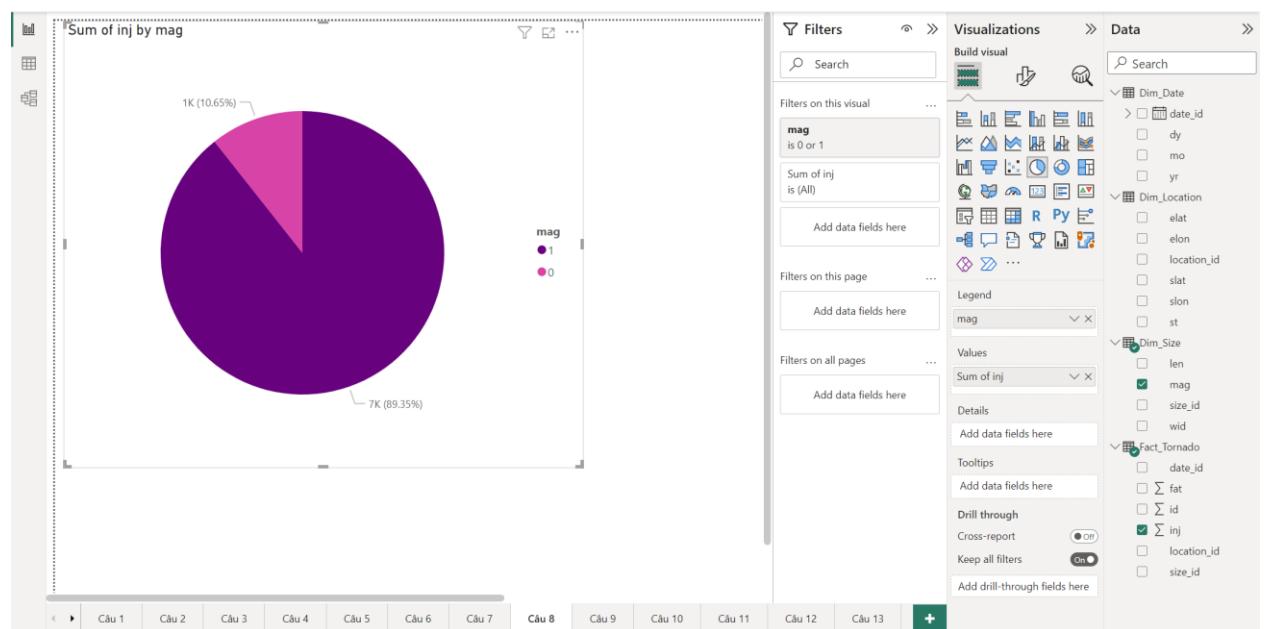
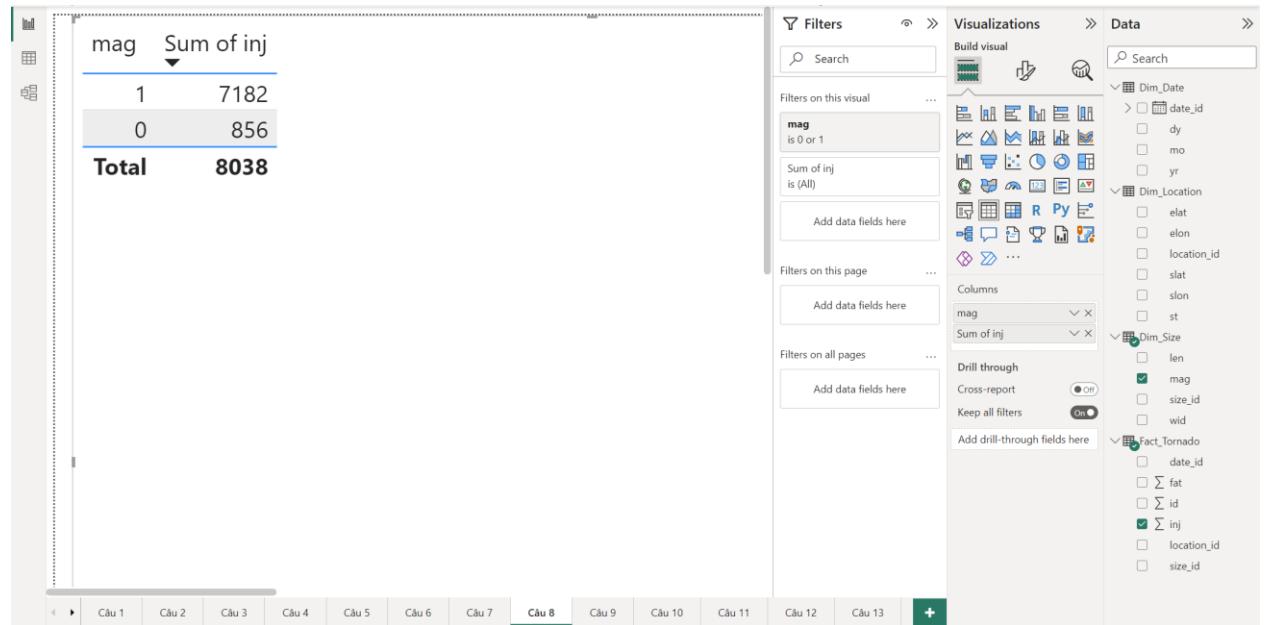
- PivotTable Fields pane:** Located on the right side of the screen. It displays the following settings:
  - Show fields:** (All)
  - Search:** (empty)
  - Dim Date:** Hierarchy (selected)
  - More Fields:**
    - Date Id
    - Dy
    - Mo
    - Yr (selected)
- Data Grid:** A table with columns A and B. Column A contains years from 1953 to 2013. Column B contains values corresponding to each year.



→ Nhìn vào biểu đồ kết luận năm 2011 là năm có số lượng lốc xoáy nhiều nhất ( 290 trận lốc ) với mức độ gió cao nhất trong thang đo.

#### 4.2.8 Số người bị thương do các cơn lốc xoáy có độ đo F/EF ở mức độ nhẹ đến trung bình

- Power BI

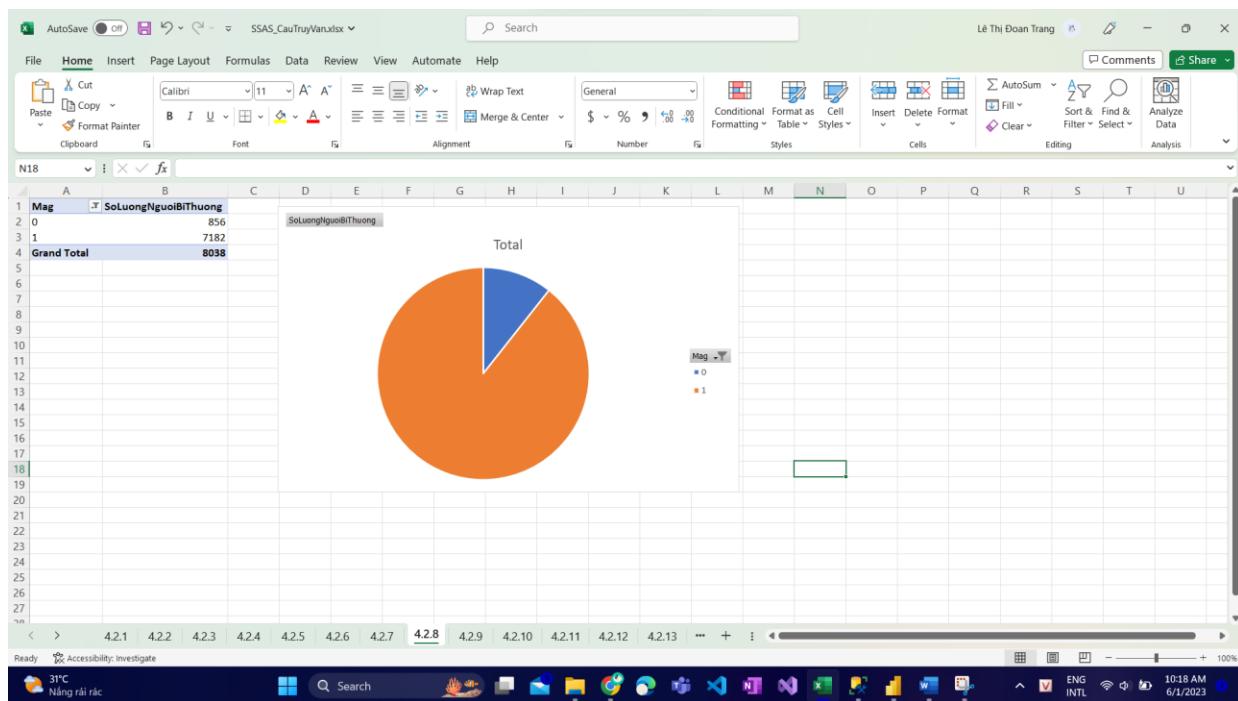


- Excel

## IS217.N22.HTCL – Kho dữ liệu và OLAP

The screenshot shows a Microsoft Excel spreadsheet titled "SSAS\_CayTruyVan.xlsx". The PivotTable Fields pane is open on the right side, showing fields like "More Fields" (Len, Mag, Size Id, Wid), "Filters" (empty), and "Columns" (empty). The main table on the left contains the following data:

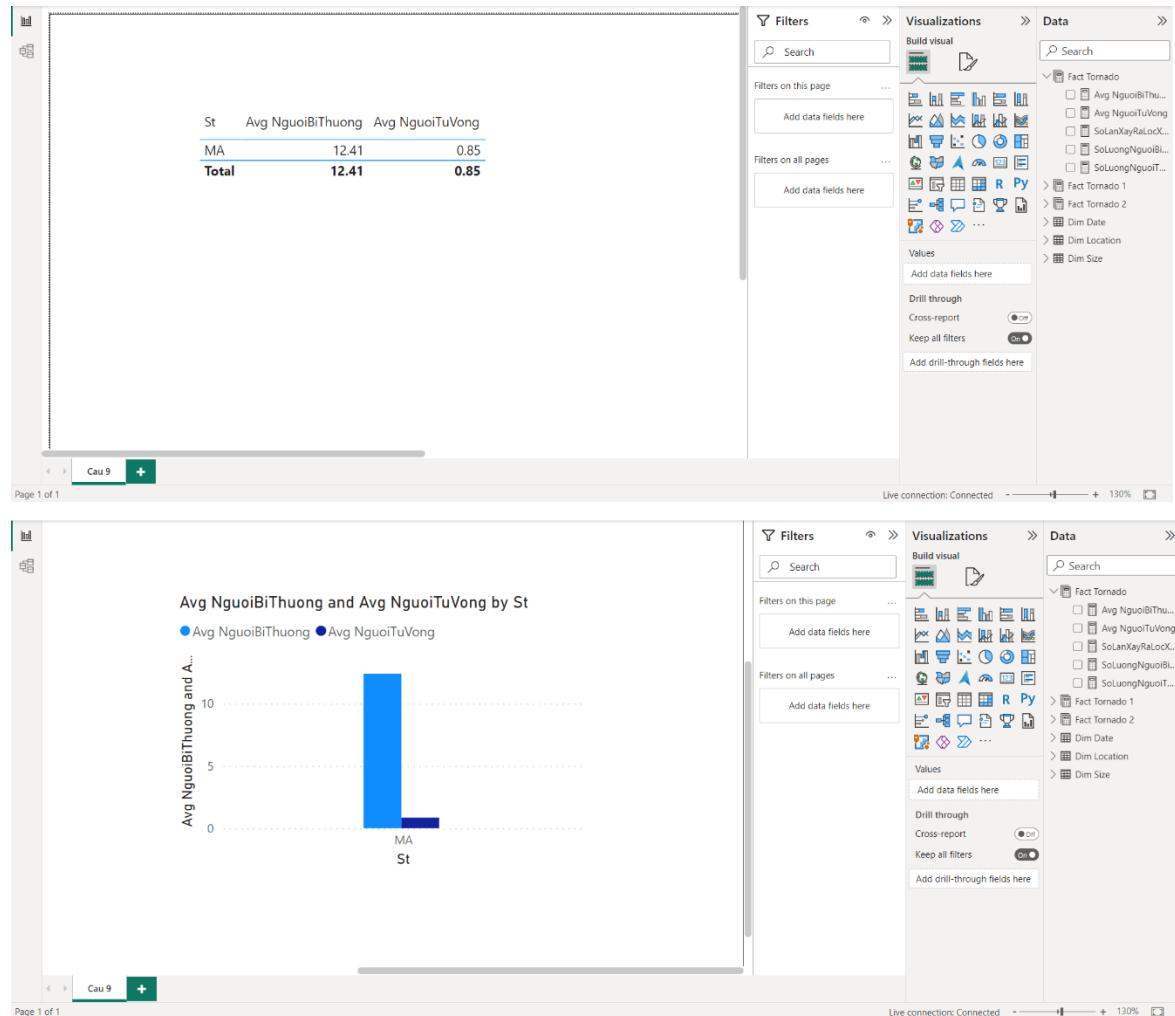
	State	SoLuongNguoiBiThuong
0		856
1		7182
<b>Grand Total</b>		<b>8038</b>



→ Từ biểu đồ thấy được lốc xoáy ở mức độ 1 (trung bình) đã gây ra nhiều thương tích đối với con người, tới hơn 7000 bị thương do lốc xoáy. Chiếm tận 89.35% trong tổng số trên.

#### 4.2.9 Trung bình 1 ngày có bao nhiêu người chết và bị thương ở bang MA

- Power BI



- Excel

## IS217.N22.HTCL – Kho dữ liệu và OLAP

The screenshot shows the Microsoft Excel ribbon with the 'PivotTable Analyze' tab selected. The main area displays a PivotTable with the following data:

	Avg NguoiTuVong	Avg NguoiBiThuong
MA	0.008536585	0.124065041
<b>Grand Total</b>	<b>0.008536585</b>	<b>0.124065041</b>

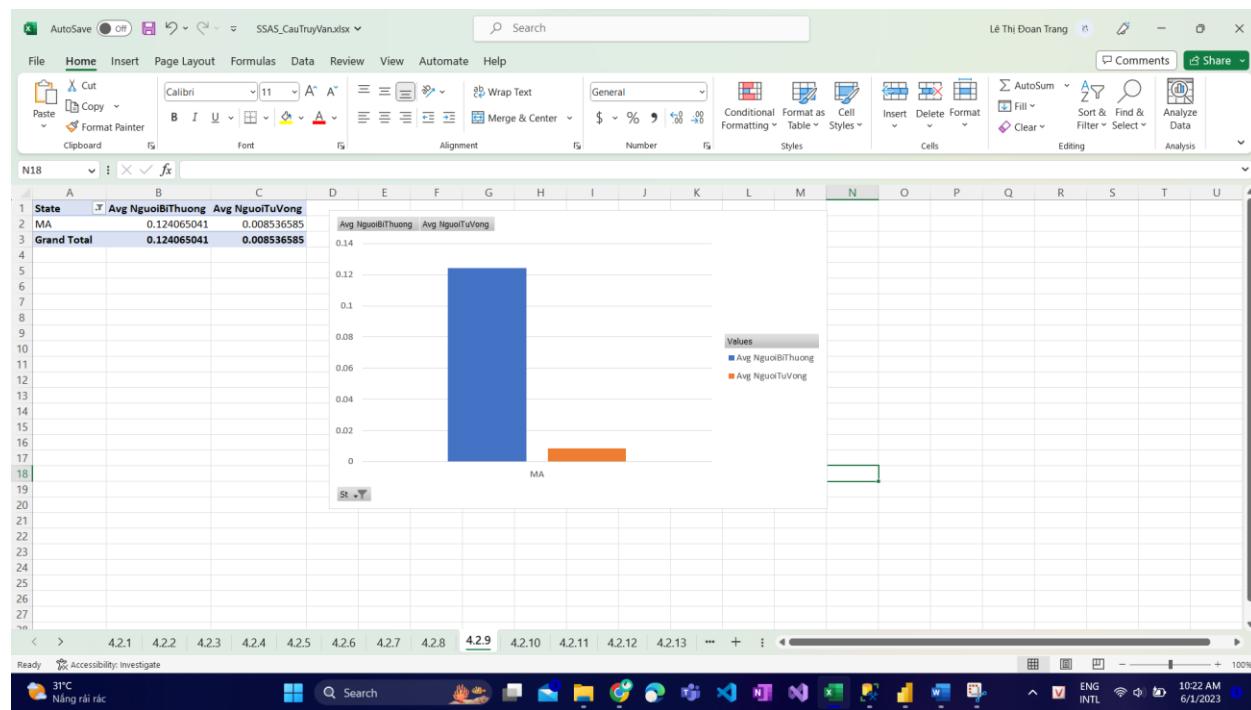
The 'PivotTable Fields' pane on the right shows the following fields:

- Date Id Distinct Count
- Fact Tornado
  - Avg NguoiBiThuong
  - Avg NguoiTuVong
  - SoLanXayRaLocXoayTrongNgay

Drag fields between areas below:

- Filters
- Columns
- Rows
- Values

Sheet tabs: Sheet1, Sheet3.



→ Từ biểu đồ cho thấy trung bình một ngày ở bang MA tỷ lệ bị thương khá ít chỉ có 0.12 lượng người và tỷ lệ tử vong cũng rất ít chỉ 0.0085 lượng người.

#### 4.2.10 Tìm các bang có các cơn lốc xoáy gây thương tích nhưng không gây tử vong trong năm 2001

- Power BI

The screenshot shows a Power BI interface with a table visualization and its corresponding filter context.

**Table Visualization:**

yr	st	Sum of inj	Sum of fat
2001	CO	13	0
2001	CT	1	0
2001	GA	23	0
2001	IL	3	0
2001	KY	14	0
2001	ME	1	0
2001	MI	8	0
2001	MN	13	0
2001	NC	3	0
2001	ND	1	0
2001	NE	9	0
2001	OH	1	0
2001	SC	39	0
2001	TX	49	0
2001	VA	4	0
2001	WY	2	0
<b>Total</b>		<b>184</b>	<b>0</b>

**Filter Context (Filters pane):**

- Filters on this visual:
  - st is (All)
  - Sum of fat is 0
  - Sum of inj is greater than 0
  - yr is 2001
- Filters on this page:
  - Add data fields here
- Filters on all pages:
  - Add data fields here

**Data pane:**

- Dim\_Date: date\_id, dy, mo, yr
- Dim\_Location: elat, elon, location\_id, slat, slon, st
- Dim\_Size: len, mag, size\_id, wid
- Fact\_Tornado: date\_id, fat, id, inj, location\_id, size\_id

The screenshot shows a Power BI interface with a bar chart visualization and its corresponding filter context.

**Bar Chart Visualization:**

Sum of inj and Sum of fat by yr, st and st

The chart displays the sum of injuries for each state (st) in 2001. The Y-axis represents the sum of injuries, ranging from 0 to 50. The X-axis lists the states: TX, SC, GA, KY, CO, MN, NE, MI, VA, IL, NC, WY, CT, ME, ND, OH.

st	Sum of inj
TX	49
SC	39
GA	23
KY	14
CO	13
MN	13
NE	10
MI	8
VA	4
IL	3
NC	3
WY	2
CT	1
ME	1
ND	1
OH	1

**Filter Context (Filters pane):**

- Filters on this visual:
  - st is (All)
  - Sum of fat is 0
  - Sum of inj is greater than 0
  - yr is 2001
- Filters on this page:
  - Add data fields here
- Filters on all pages:
  - Add data fields here

**Data pane:**

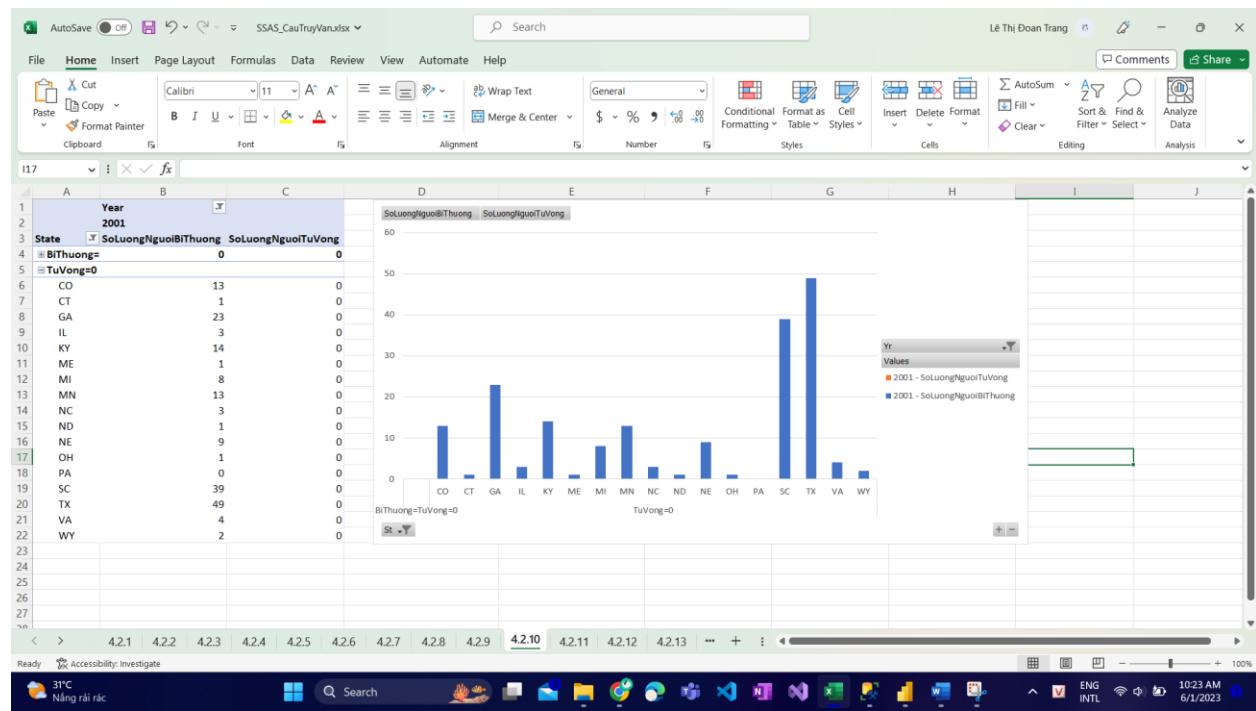
- Dim\_Date: date\_id, dy, mo, yr
- Dim\_Location: elat, elon, location\_id, slat, slon, st
- Dim\_Size: len, mag, size\_id, wid
- Fact\_Tornado: date\_id, fat, id, inj, location\_id, size\_id

- Excel

## IS217.N22.HTCL – Kho dữ liệu và OLAP

The screenshot shows a Microsoft Excel window with the file "SSAS\_CayTruyVan.xlsx" open. The PivotTable Fields pane is visible on the right side of the screen, showing fields from the "Fact Tornado" table, including "SoLuongNguoiBiThuong" which is selected. The main area displays a PivotTable with columns for State, Year, and two measures: SoLuongNguoiBiThuong and SoLuongNguoiTuVong. The data shows the number of people injured and killed in tornadoes by state in 2001.

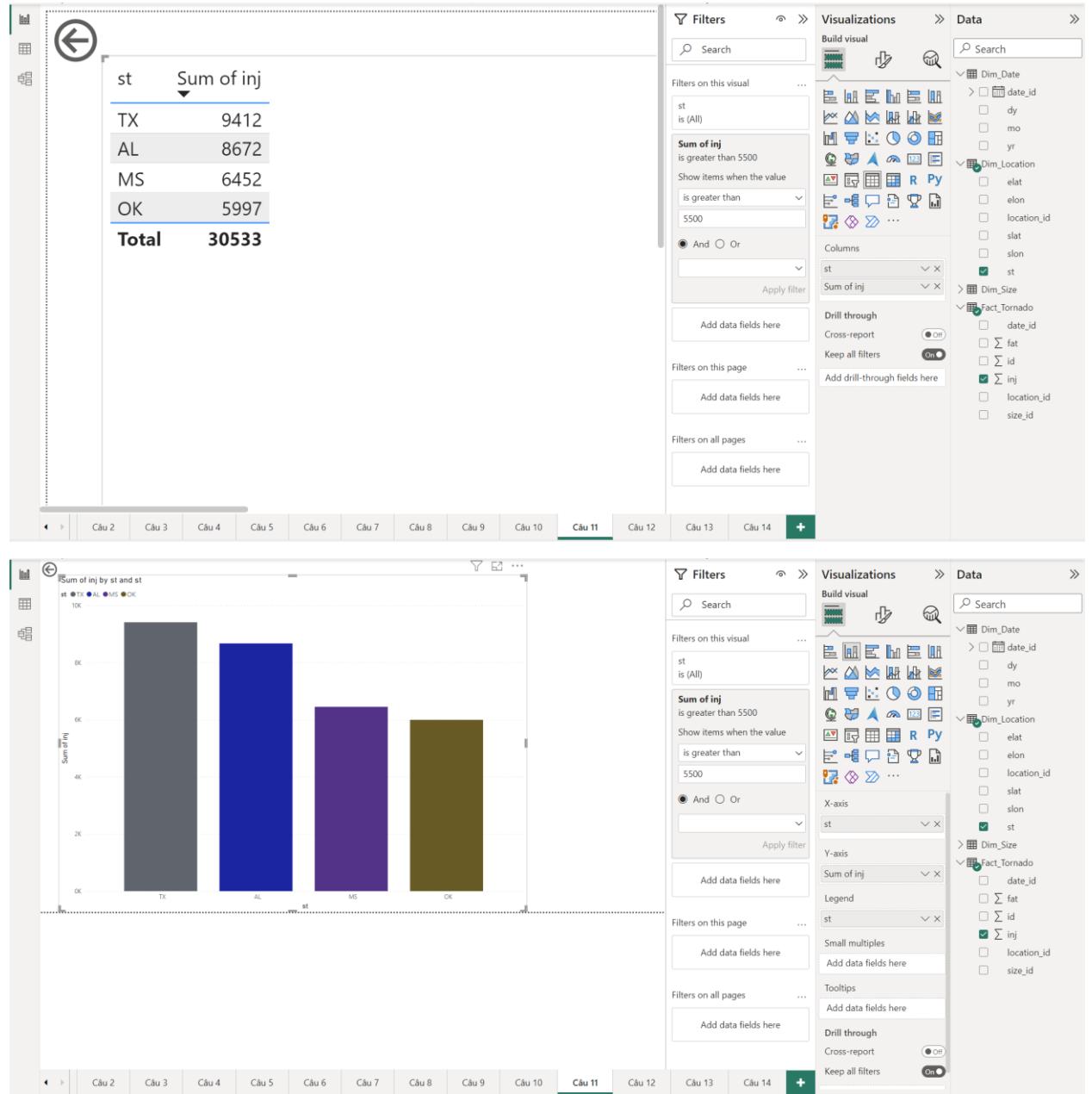
State	Year	SoLuongNguoiBiThuong	SoLuongNguoiTuVong
CO	2001	13	0
CT	2001	1	0
GA	2001	23	0
IL	2001	3	0
KY	2001	14	0
ME	2001	1	0
MI	2001	8	0
MN	2001	13	0
NC	2001	3	0
ND	2001	1	0
NE	2001	9	0
OH	2001	1	0
SC	2001	39	0
TX	2001	49	0
VA	2001	4	0
WY	2001	2	0



→ Từ biểu đồ có thể thấy chỉ có nửa số lượng tiểu bang là có người bị thương, còn lại là không bị thiệt hại về người quá nhiều. Đặc biệt trong biểu đồ Texas là bang có số lượng người bị thương nhiều nhất với 49 người trong năm 2001.

#### 4.2.11 Tên tiêu bang có số lượng người bị thương >5500 người

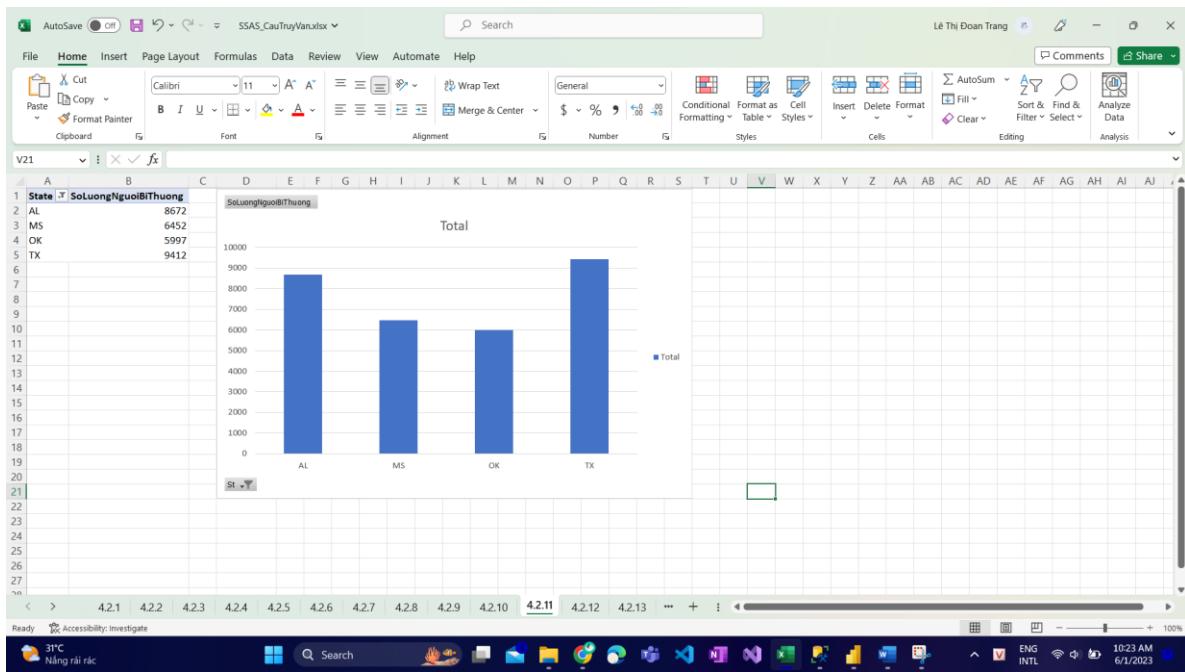
- Power BI



- Excel

## IS217.N22.HTCL – Kho dữ liệu và OLAP

The screenshot shows a Microsoft Excel window with the ribbon menu. The active tab is "Home". A PivotTable is displayed in the main area, with the first row containing "State" and "SoLuongNguoiBiThuong". The data rows show values for AL, MS, OK, and TX. To the right, the "PivotTable Fields" pane is open, showing fields like "Dim Date", "Fact Tornado", and "SoLuongNguoiBiThuong". The "Values" section is set to "SoLuongNguoiBiThuong". The status bar at the bottom indicates "4.2.11" and "4.2.10".

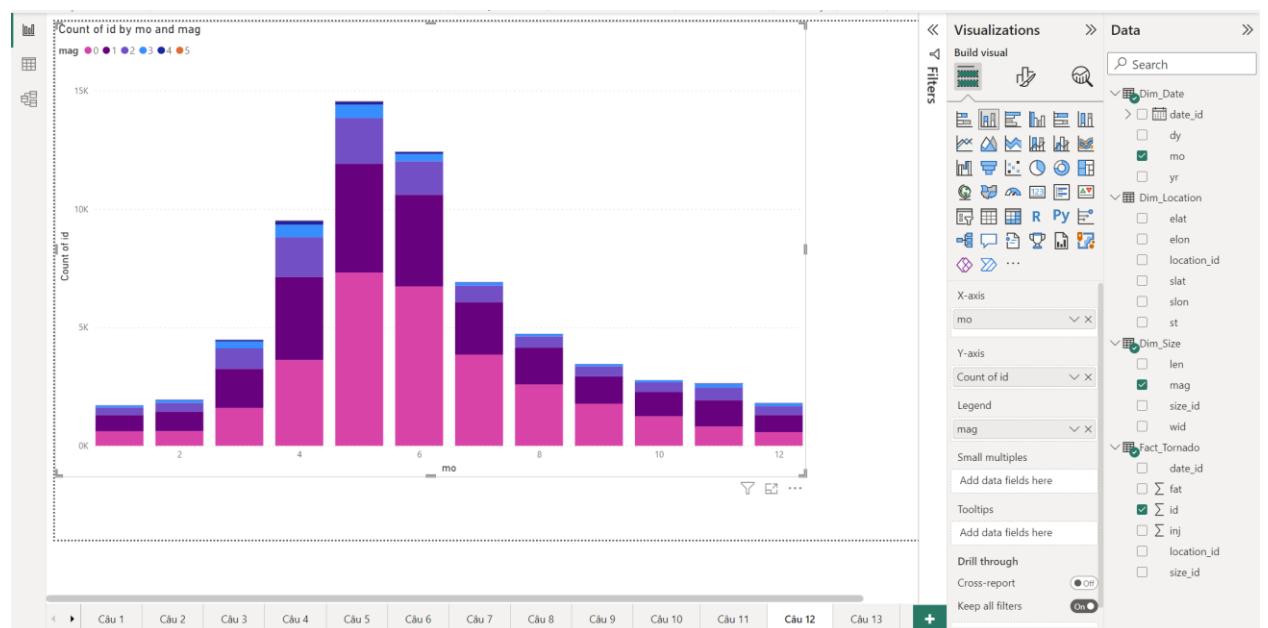
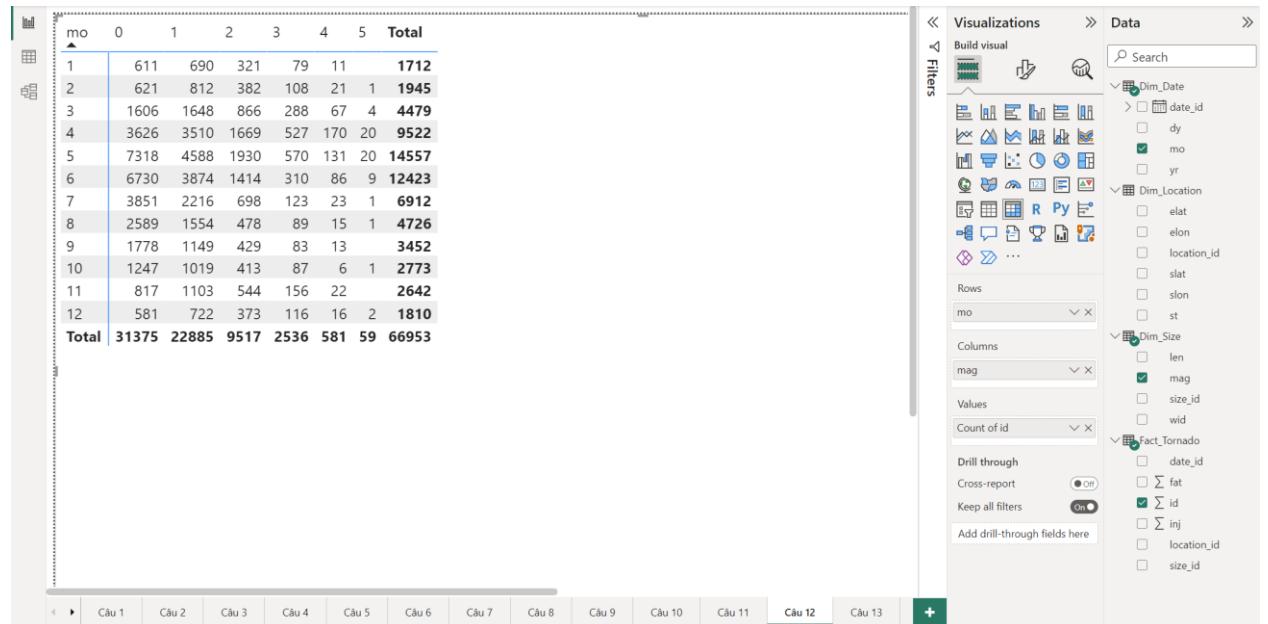


→ Từ biểu đồ rút ra được nhận xét là Texas là tiểu bang có số lượng người bị thương nhiều nhất ( 9412 ) trong tổng tất cả các tiểu bang ở Hoa Kỳ và theo sau với con số cũng không nhỏ là tiểu bang Alaska. Đây là 2 tiểu bang lớn nhất, nhì ở Hoa Kỳ, có diện tích khá rộng

và khí hậu thay đổi liên tục, do các nguyên nhân trên đã gây ra rất nhiều trận lốc xoáy ở 2 tiểu bang này và đã có rất nhiều thương vong.

#### 4.2.12 Thống kê số lượng lốc xoáy theo từng tháng và từng loại độ đo F/EF

- Power BI

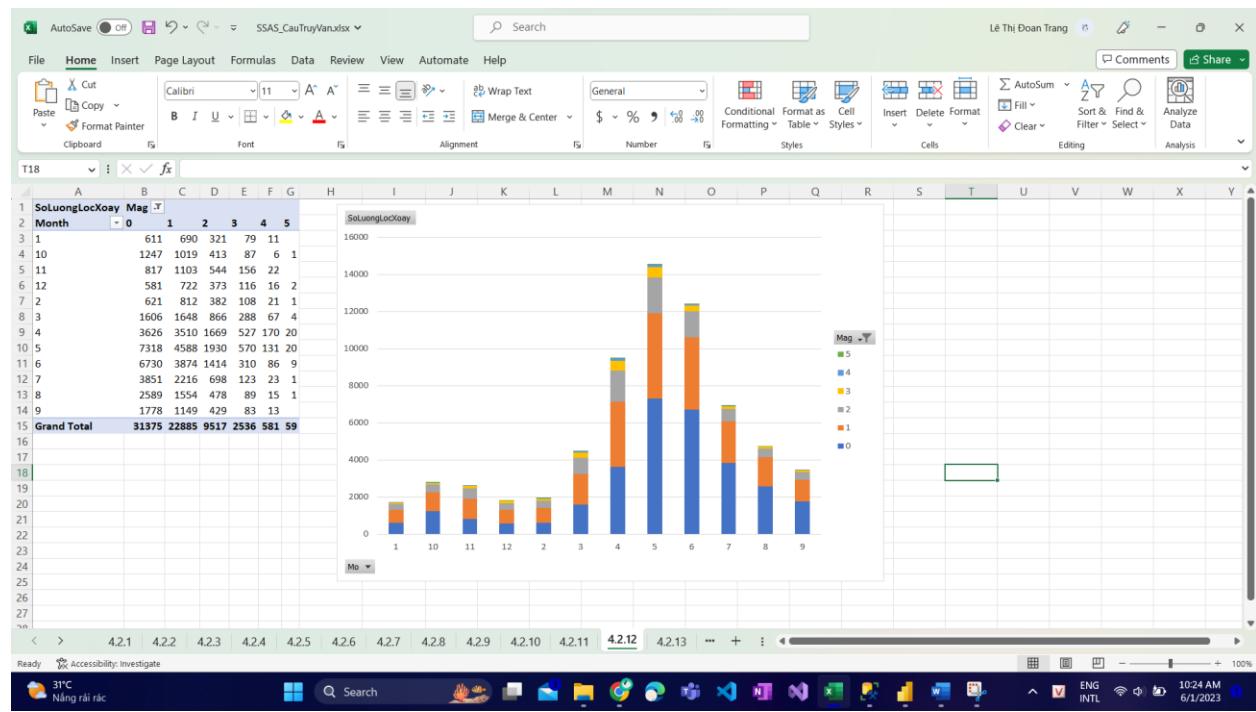


- Excel

## IS217.N22.HTCL – Kho dữ liệu và OLAP

The screenshot shows a Microsoft Excel window with the ribbon menu. A PivotTable is displayed in the main area, with the formula bar showing "SoLuongLocXoay". The PivotTable Fields pane is open on the right side, showing fields from the "Fact Tornado 1" and "Dim Date" tables, with "Mag" selected under Values and "Mo" selected under Rows. The data grid below shows monthly tornado counts from 1950 to 2021.

Month	0	1	2	3	4	5
1	611	690	321	79	11	
10	1247	1019	413	87	6	1
11	817	1103	544	156	22	
12	581	722	373	116	16	2
2	621	812	382	108	21	1
3	1606	1648	866	288	67	4
4	3626	3510	1669	527	170	20
5	7318	4588	1930	570	131	20
6	6730	3874	1414	310	86	9
7	3851	2216	698	123	23	1
8	2589	1554	478	89	15	1
9	1778	1149	429	83	13	

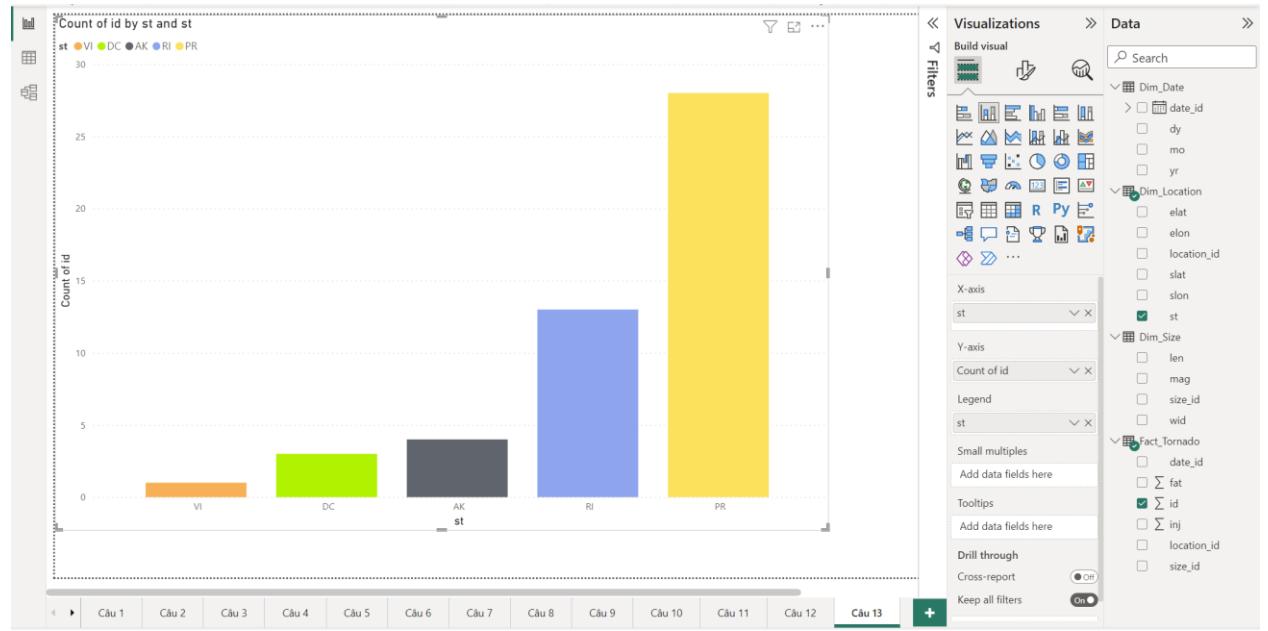


→ Biểu đồ trên đã tổng hợp số lượng lốc xoáy ở từng tháng từ năm 1950-2021, sau khi thống kê lại ta có kết luận tháng 5 là tháng xảy ra nhiều trận lốc xoáy nhất, từ mức độ nhẹ nhất cho đến mức độ siêu hủy diệt đều chiếm đa số so với các tháng khác.

#### 4.2.13 Thống kê top 5 bang xảy ra lốc xoáy ít nhất

- Power BI

st	Count of id
VI	1
DC	3
AK	4
RI	13
PR	28
<b>Total</b>	<b>49</b>

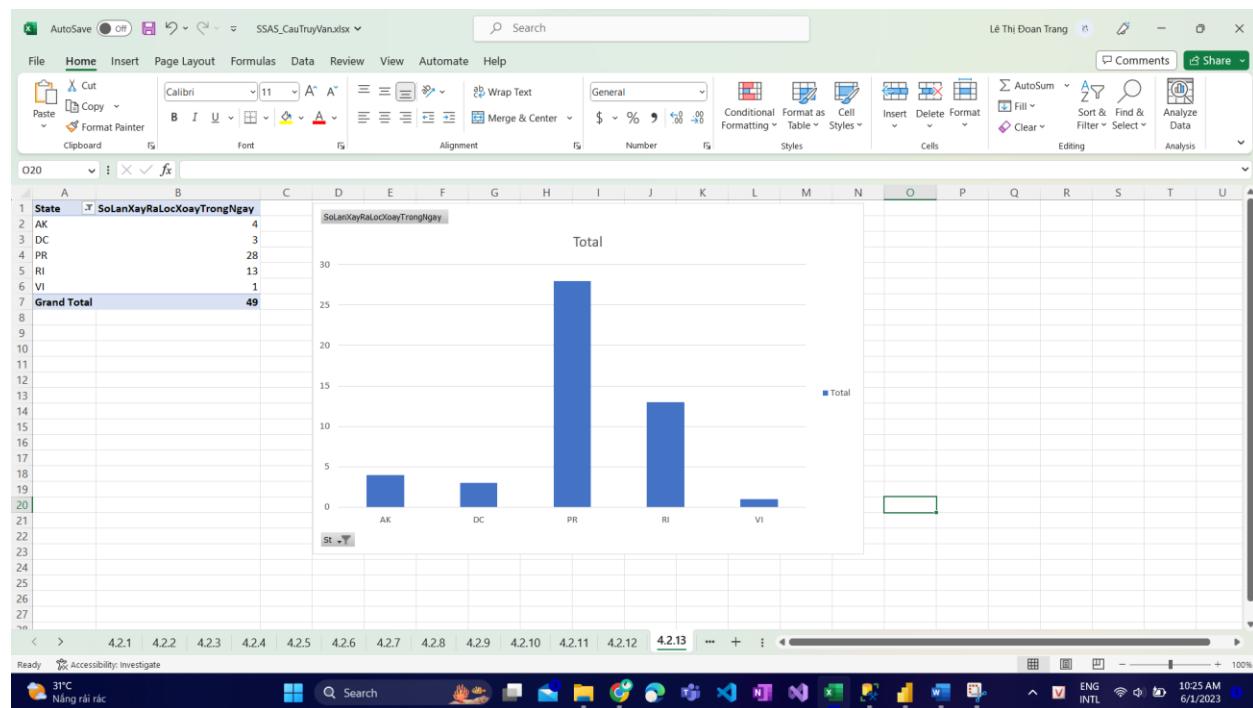


- Excel

## IS217.N22.HTCL – Kho dữ liệu và OLAP

The screenshot shows a Microsoft Excel spreadsheet titled "SSAS\_CayTruyVan.xlsx". The PivotTable Fields pane is open on the right side, showing fields like "SoLanXayRaLocXoay" (selected), "SoLuongNguoiBiThuong", "SoLuongNguoiTuVong", "SoThuongVongNhiieuNhatTrongNgay", and "SoTuVongNhiieuNhatTrongNgay". The main table displays the following data:

State	SoLanXayRaLocXoay
AK	4
DC	3
PR	28
RI	13
VI	1
<b>Grand Total</b>	<b>49</b>



→ Biểu đồ trên đã thống kê lại 4 tiểu bang an bình nhất trong tổng số các tiểu bang ở Hoa Kỳ, số lượng xảy ra lốc xoáy rất ít, tiêu biểu là bang VI một năm chỉ xảy ra lốc xoáy 1 lần.

#### 4.2.14 Thống kê tổng số người bị thương theo từng năm và từng bang

- Power BI

The screenshot shows a Power BI interface with a table visualization on the left and the Data pane on the right.

**Data pane:**

- Dim\_Date:** date\_id, dy, mo, yr (selected).
- Dim\_Location:** elat, elon, location\_id, slat, slon, st (selected).
- Dim\_Size:** len, mag, size\_id, wid.
- Fact\_Tornado:** date\_id, fat, id, inj, location\_id, size\_id.

**Table Visualization:**

yr	st	Sum of fat
1950	AL	0
1950	AR	2
1950	CO	0
1950	CT	0
1950	FL	0
1950	GA	0
1950	IA	0
1950	IL	3
1950	IN	0
1950	KS	1
1950	KY	0
1950	LA	29
1950	MD	0
1950	MN	0
1950	MO	0
1950	MS	6
1950	NC	0
1950	ND	0
1950	NE	0
1950	NM	0
1950	OH	0
1950	OK	6
1950	PA	0
1950	SC	0
1950	SD	0
1950	TN	9
1950	TX	11
1950	WI	3
1950	WV	0
1950	WY	0
<b>Total</b>		<b>6112</b>

The screenshot shows a Power BI interface with a stacked bar chart visualization on the left and the Data pane on the right.

**Data pane:**

- Dim\_Date:** date\_id, dy, mo, yr (selected).
- Dim\_Location:** elat, elon, location\_id, slat, slon, st (selected).
- Dim\_Size:** len, mag, size\_id, wid.
- Fact\_Tornado:** date\_id, fat, id, inj, location\_id, size\_id.

**Stacked Bar Chart:**

The chart displays the 'Sum of fat' by year ('yr') and state ('st'). The y-axis ranges from 0 to 600. The x-axis shows years from approximately 1950 to 2020. The bars are stacked by state, with colors corresponding to the legend: AK, AL, AR, AZ, CA, CO, CT, DC, DE, FL, GA, HI, ID, IL, IN, KS, KY, LA, MA, MD, ME, MI, MN, MO, MS, MT, ND, NE, NM, OH, OK, PA, SC, SD, TN, TX, WI, WV, WY.

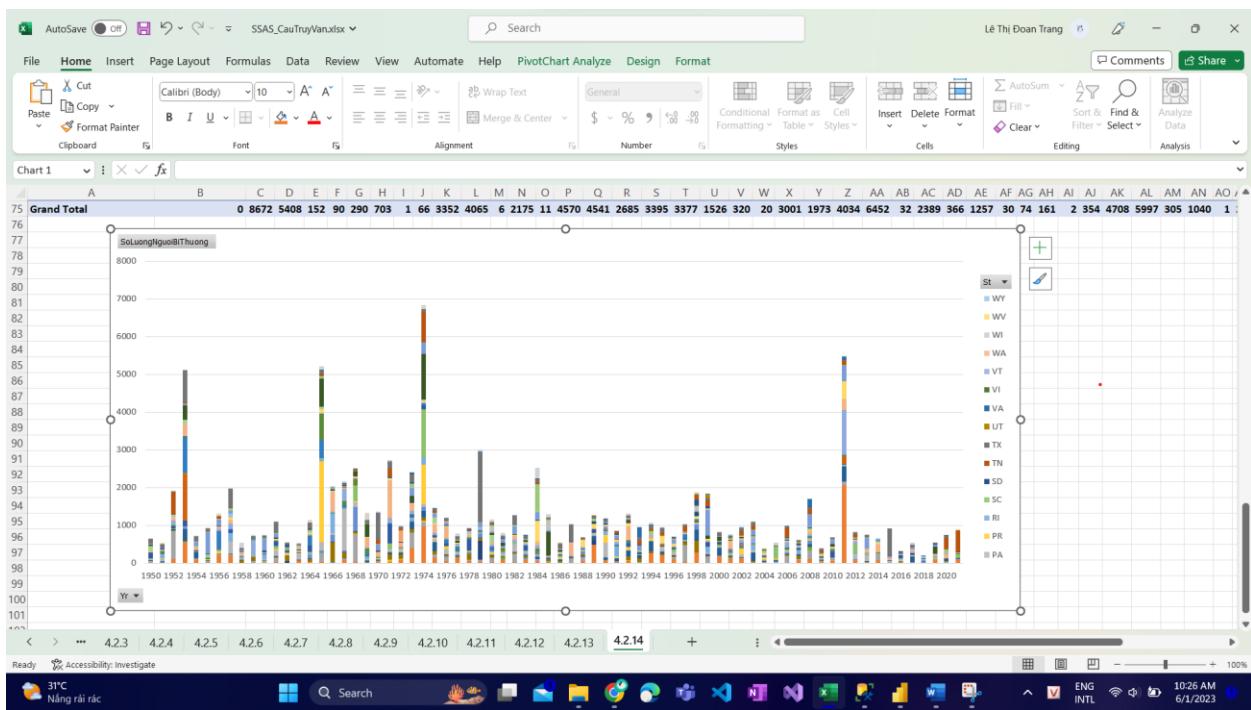
- Excel

## IS217.N22.HTCL – Kho dữ liệu và OLAP

The screenshot shows a Microsoft Excel spreadsheet titled "SSAS\_CayTruyVan.xlsx". The active sheet is "Sheet1". A PivotTable is being built in the range A3:F3. The PivotTable Fields pane on the right shows the following fields:

- Fact Tornado** (selected):
  - SoLuongNguoBiThuong (selected)
  - SoLuongNguoTuVong
  - SoThuongVongNheuNhieu
  - SoTuVongNhiuNhatTro
- Fact.Tornado.1

The PivotTable Fields pane also includes sections for Filters, Columns, Rows, and Values, with "SoLuongNguoBiThuong" selected under Values.



→ Biểu đồ trên đã thống kê lại số lượng người bị thương ở tất cả các tiểu bang Hoa Kỳ từ năm 1950 – 2021. Năm 1974 là năm có số lượng người bị thương cao nhất (6824 người) đến gần 7000 người bị thương và trong năm này tiểu bang KY là tiểu bang có số lượng người bị thương cao nhất (1275 người). Năm 2018 là năm có ít người bị thương nhất chỉ có 199 người trong 1 năm.

## CHƯƠNG IV: DATA MINING

### 1. Random Forest

Bước 1: Import thư viện

```
▶ import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
import numpy as np
```

Bước 2: Import dataset

```
▶ df = pd.read_csv('us_tornado_dataset_1950_2021.csv')
df.head()
```

⇨

	yr	mo	dy	date	st	mag	inj	fat	slat	slon	elat	elon	len	wid
0	1950	1	3	1950-01-03	IL	3	3	0	39.10	-89.30	39.12	-89.23	3.6	130
1	1950	1	3	1950-01-03	MO	3	3	0	38.77	-90.22	38.83	-90.03	9.5	150
2	1950	1	3	1950-01-03	OH	1	1	0	40.88	-84.58	0.00	0.00	0.1	10
3	1950	1	13	1950-01-13	AR	3	1	1	34.40	-94.37	0.00	0.00	0.6	17
4	1950	1	25	1950-01-25	IL	2	0	0	41.17	-87.33	0.00	0.00	0.1	100

Bước 3: Kiểm tra kích thước của tập dữ liệu

```
▶ #Print the shape
print('The shape of the dataset : ', df.shape)
```

⇨ The shape of the dataset : (67558, 14)

Bước 4: Xem tóm tắt tập dữ liệu

```
▶ # View summary of dataset
df.info()

↳ <class 'pandas.core.frame.DataFrame'>
RangeIndex: 67558 entries, 0 to 67557
Data columns (total 14 columns):
 #   Column  Non-Null Count  Dtype  
--- 
 0   yr      67558 non-null   int64  
 1   mo      67558 non-null   int64  
 2   dy      67558 non-null   int64  
 3   date    67558 non-null   object 
 4   st      67558 non-null   object 
 5   mag     67558 non-null   int64  
 6   inj     67558 non-null   int64  
 7   fat     67558 non-null   int64  
 8   slat    67558 non-null   float64 
 9   slon    67558 non-null   float64 
 10  elat    67558 non-null   float64 
 11  elon    67558 non-null   float64 
 12  len     67558 non-null   float64 
 13  wid     67558 non-null   int64  
dtypes: float64(5), int64(7), object(2)
memory usage: 7.2+ MB
```

## Bước 5: Kiểm tra kiểu dữ liệu của các cột

```
▶ # Check data types of columns
df.dtypes

↳ yr      int64
mo      int64
dy      int64
date    object
st      object
mag     int64
inj     int64
fat     int64
slat    float64
slon    float64
elat    float64
elon    float64
len     float64
wid     int64
dtype: object
```

## Bước 6: Xem thuộc tính thống kê của tập dữ liệu

# View statistical properties of dataset  
df.describe().T

	count	mean	std	min	25%	50%	75%	max
<b>yr</b>	67558.0	1991.341618	19.330015	1950.0000	1976.0000	1994.00	2008.00	2021.0000
<b>mo</b>	67558.0	5.976761	2.438192	1.0000	4.0000	6.00	7.00	12.0000
<b>dy</b>	67558.0	15.921016	8.736773	1.0000	8.0000	16.00	24.00	31.0000
<b>mag</b>	67558.0	0.691273	1.283375	-9.0000	0.0000	1.00	1.00	5.0000
<b>inj</b>	67558.0	1.437876	18.263956	0.0000	0.0000	0.00	0.00	1740.0000
<b>fat</b>	67558.0	0.090470	1.484106	0.0000	0.0000	0.00	0.00	158.0000
<b>slat</b>	67558.0	37.142412	5.093979	17.7212	33.2200	37.03	40.93	61.0200
<b>slon</b>	67558.0	-92.784618	8.689103	-163.5300	-98.4500	-93.60	-86.73	-64.7151
<b>elat</b>	67558.0	22.730695	18.588638	0.0000	0.0000	32.48	38.61	61.0200
<b>elon</b>	67558.0	-56.245590	45.489157	-163.5300	-94.7098	-84.42	0.00	0.0000
<b>len</b>	67558.0	3.478340	8.278775	0.0000	0.1000	0.80	3.13	234.7000
<b>wid</b>	67558.0	106.577030	205.802676	0.0000	20.0000	50.00	100.00	4576.0000

## Bước 7: Kiểm tra có bao nhiêu giá trị null ở các cột

# Check for missing values  
df.isnull().sum().sort\_values(ascending=False)

<b>yr</b>	0
<b>mo</b>	0
<b>dy</b>	0
<b>date</b>	0
<b>st</b>	0
<b>mag</b>	0
<b>inj</b>	0
<b>fat</b>	0
<b>slat</b>	0
<b>slon</b>	0
<b>elat</b>	0
<b>elon</b>	0
<b>len</b>	0
<b>wid</b>	0
<b>dtype:</b>	int64

## Bước 8: Chuẩn bị dữ liệu:

- Chuyển các giá trị của ‘mag’ (độ đo F/EF) : về giá trị là ‘<3’ và ‘>=3’
  - Với ‘<3’ : lốc xoáy không mạnh (-9: yếu, 0: nhẹ, 1: trung bình, 2: đáng kể)
  - Với ‘>=3’: lốc xoáy mạnh (3: nghiêm trọng, 4: hủy diệt, 5: siêu hủy diệt)
- Sau đó tách tập dữ liệu thành các tập X(đầu vào) và tập Y(đầu ra)

```
[9] df['mag'] = df['mag'].replace({-9:'<3', 0:'<3', 1:'<3', 2:'<3', 3:'>=3', 4:'>=3', 5:'>=3'})
    # <3: lốc xoáy không mạnh
    # >=3 lốc xoáy mạnh

[10] #Tách dữ liệu thành các tập X(đầu vào) và tập Y(đầu ra)
    X = df[['st','inj','fat','slat','slon','elat','elon','len','wid']]

    y = df['mag']
```

Bước 9: Chia dữ liệu thành tập Train và Test theo tỉ lệ 70:30. Và kiểm tra kích thước của tập X\_train, X\_test

```
[11] from sklearn.model_selection import train_test_split

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3, random_state = 0)

[12] # Check the shape of X_train and X_test
    X_train.shape, X_test.shape

((47290, 9), (20268, 9))
```

Bước 10: Mã hóa các biến phân loại

- Xem các biến phân loại và các biến số trong tập train

```
[13] # Hiển thị các biến phân loại trong tập train
    categorical = [col for col in X_train.columns if X_train[col].dtypes == 'O']

    categorical

    ['st']

[14] # Hiển thị các biến số trong tập train
    numerical = [col for col in X_train.columns if X_train[col].dtypes != 'O']

    numerical

    ['inj', 'fat', 'slat', 'slon', 'elat', 'elon', 'len', 'wid']
```

- Kiểm tra các giá trị null của các cột ở tập X\_train và X\_test

```
[15] X_train[categorical].isnull().sum()
```

```
st      0  
dtype: int64
```

```
[16] X_test[categorical].isnull().sum()
```

```
st      0  
dtype: int64
```

```
[17] X_train.isnull().sum()
```

```
st      0  
inj     0  
fat     0  
slat    0  
slon    0  
elat    0  
elon    0  
len     0  
wid     0  
dtype: int64
```

```
[18] X_test.isnull().sum()
```

```
st      0  
inj     0  
fat     0  
slat    0  
slon    0  
elat    0  
elon    0  
len     0  
wid     0  
dtype: int64
```

- Cài đặt và import thư viện category\_encoders

IS217.N22.HTCL – Kho dữ liệu và OLAP

```
[20] !pip install category_encoders

Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting category_encoders
  Downloading category_encoders-2.6.1-py2.py3-none-any.whl (81 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 81.9/81.9 kB 4.5 MB/s eta 0:00:00
Requirement already satisfied: numpy>=1.14.0 in /usr/local/lib/python3.10/dist-packages (from category_encoders) (1.22.4)
Requirement already satisfied: scikit-learn>=0.20.0 in /usr/local/lib/python3.10/dist-packages (from category_encoders) (1.2.2)
Requirement already satisfied: scipy>=1.0.0 in /usr/local/lib/python3.10/dist-packages (from category_encoders) (1.10.1)
Requirement already satisfied: statsmodels>=0.9.0 in /usr/local/lib/python3.10/dist-packages (from category_encoders) (0.13.5)
Requirement already satisfied: pandas>=1.0.5 in /usr/local/lib/python3.10/dist-packages (from category_encoders) (1.5.3)
Requirement already satisfied: patsy>=0.5.1 in /usr/local/lib/python3.10/dist-packages (from category_encoders) (0.5.3)
Requirement already satisfied: python-dateutil>=2.8.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.0.5>category_encoders) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.0.5>category_encoders) (2022.7.1)
Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packages (from patsy>=0.5.1>category_encoders) (1.16.0)
Requirement already satisfied: joblib>=1.1.1 in /usr/local/lib/python3.10/dist-packages (from scikit-learn>=0.20.0>category_encoders) (1.2.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn>=0.20.0>category_encoders) (3.1.0)
Requirement already satisfied: packaging>=21.3 in /usr/local/lib/python3.10/dist-packages (from statsmodels>=0.9.0>category_encoders) (23.1)
Installing collected packages: category_encoders
Successfully installed category_encoders-2.6.1
```

```
[21] # import category encoders  
import category_encoders as ce
```

- Xem lại dữ liệu của biến phân loại trong tập X\_train trước khi mã hóa

```
[19] # Xem dữ liệu của các biến phân loại trong tập X_train  
X_train[categorical].head()
```

st	
7360	KS
8210	SD
39146	FL
21602	OK
24363	NE

- Mã hóa biến phân loại ‘st’ bằng one-hot encoding

```
[22] # Mã hóa các biến phân loại bằng one-hot encoding
encoder = ce.OneHotEncoder(cols=['st'])

X_train = encoder.fit_transform(X_train)

X_test = encoder.transform(X_test)
```

- Kiểm tra dữ liệu và kích thước trong tập X\_train: Có thể thấy, ban đầu chỉ có 9 cột trong tập X\_train nhưng giờ số cột là 60 cột

```
[23] X_train.head()
```

	st_1	st_2	st_3	st_4	st_5	st_6	st_7	st_8	st_9	st_10	...	st_51	st_52	inj	fat	slat	slon	elat	elon	len	wid
7360	1	0	0	0	0	0	0	0	0	0	...	0	0	0	37.92	-97.42	0.00	0.00	0.1	10	
8210	0	1	0	0	0	0	0	0	0	0	...	0	0	0	45.20	-100.30	0.00	0.00	0.1	10	
39146	0	0	1	0	0	0	0	0	0	0	...	0	0	0	24.92	-80.63	24.92	-80.63	2.0	50	
21602	0	0	0	1	0	0	0	0	0	0	...	0	0	1	0	35.22	-97.43	0.00	0.00	0.3	10
24363	0	0	0	0	1	0	0	0	0	0	...	0	0	0	40.72	-98.80	0.00	0.00	0.5	30	

5 rows × 60 columns

```
[24] X_train.shape
```

(47290, 60)

- Tương tự với tập X\_test, ban đầu cũng chỉ có 9 cột và hiện tại là 60 cột

```
[25] X_test.head()
```

	st_1	st_2	st_3	st_4	st_5	st_6	st_7	st_8	st_9	st_10	...	st_51	st_52	inj	fat	slat	slon	elat	elon	len	wid
44086	0	0	0	0	0	0	0	0	0	0	...	0	0	0	35.8800	-86.1000	35.9200	-86.0800	3.00	100	
32153	0	0	0	0	0	0	0	0	0	0	...	0	0	0	40.0300	-109.5300	0.0000	0.0000	0.10	10	
56458	0	0	0	0	0	0	0	0	0	0	...	0	0	0	29.8816	-92.1028	29.8945	-92.1004	0.90	20	
30138	0	0	1	0	0	0	0	0	0	0	...	0	0	0	27.3000	-81.8500	0.0000	0.0000	0.40	50	
54955	0	0	0	0	0	0	0	0	0	0	...	0	0	0	38.5233	-89.0708	38.5278	-89.0675	0.36	75	

5 rows × 60 columns

```
[26] X_test.shape
```

(20268, 60)

## Bước 11: Chuẩn hóa dữ liệu

```
[27] cols = X_train.columns
```

```
[28] from sklearn.preprocessing import RobustScaler

scaler = RobustScaler()

X_train = scaler.fit_transform(X_train)

X_test = scaler.transform(X_test)
```

```
[29] X_train = pd.DataFrame(X_train, columns=[cols])
```

```
[30] X_test = pd.DataFrame(X_test, columns=[cols])
```

## Bước 12: Áp dụng mô hình Random Forest Classifier với tham số mặc định = 100

```
[32] # import Random Forest classifier  
  
from sklearn.ensemble import RandomForestClassifier  
  
# instantiate the classifier with n_estimators = 100  
  
rfc_100 = RandomForestClassifier(n_estimators=100, random_state=0)  
  
  
# fit the model to the training set  
  
rfc_100.fit(X_train, y_train)  
  
  
# Predict on the test set results  
  
y_pred_100 = rfc_100.predict(X_test)  
  
  
# Check accuracy score  
  
print('Model accuracy score with 100 decision-trees : {:.4f}'.format(accuracy_score(y_test, y_pred_100)))  
  
Model accuracy score with 100 decision-trees : 0.9614
```

→ Từ kết quả trên, có thể thấy độ chính xác của mô hình là khoảng 96%

Bước 13: Tìm các đặc trưng quan trọng với mô hình Random Forest

- Xây dựng lại mô hình với 100 cây quyết định

```
▶ # create the classifier with n_estimators = 100  
  
clf = RandomForestClassifier(n_estimators=100, random_state=0)  
  
  
# fit the model to the training set  
  
clf.fit(X_train, y_train)
```

⇨  RandomForestClassifier  
RandomForestClassifier(random\_state=0)

- Kết quả điểm các đặc trưng theo thứ tự giảm dần

```
[34] # view the feature scores

feature_scores = pd.Series(clf.feature_importances_, index=X_train.columns).sort_values(ascending=False)

feature_scores
```

Feature	Importance Score
inj	1.782818e-01
len	1.541995e-01
wid	1.261384e-01
slon	1.162108e-01
slat	1.116082e-01
fat	8.627922e-02
elat	8.423128e-02
elon	8.251341e-02
st_6	3.826116e-03
st_1	3.631415e-03
st_23	3.258808e-03
st_4	3.146409e-03
st_8	2.789243e-03
st_16	2.614027e-03
st_19	2.610260e-03
st_33	2.504629e-03
st_15	2.501791e-03
st_10	2.420064e-03
st_18	2.379475e-03
st_31	2.268296e-03
st_20	2.220179e-03
st_11	1.918494e-03
...	...

→ Có thể thấy, đặc trưng quan trọng nhất là 'inj' và ít quan trọng nhất là 'st\_52'

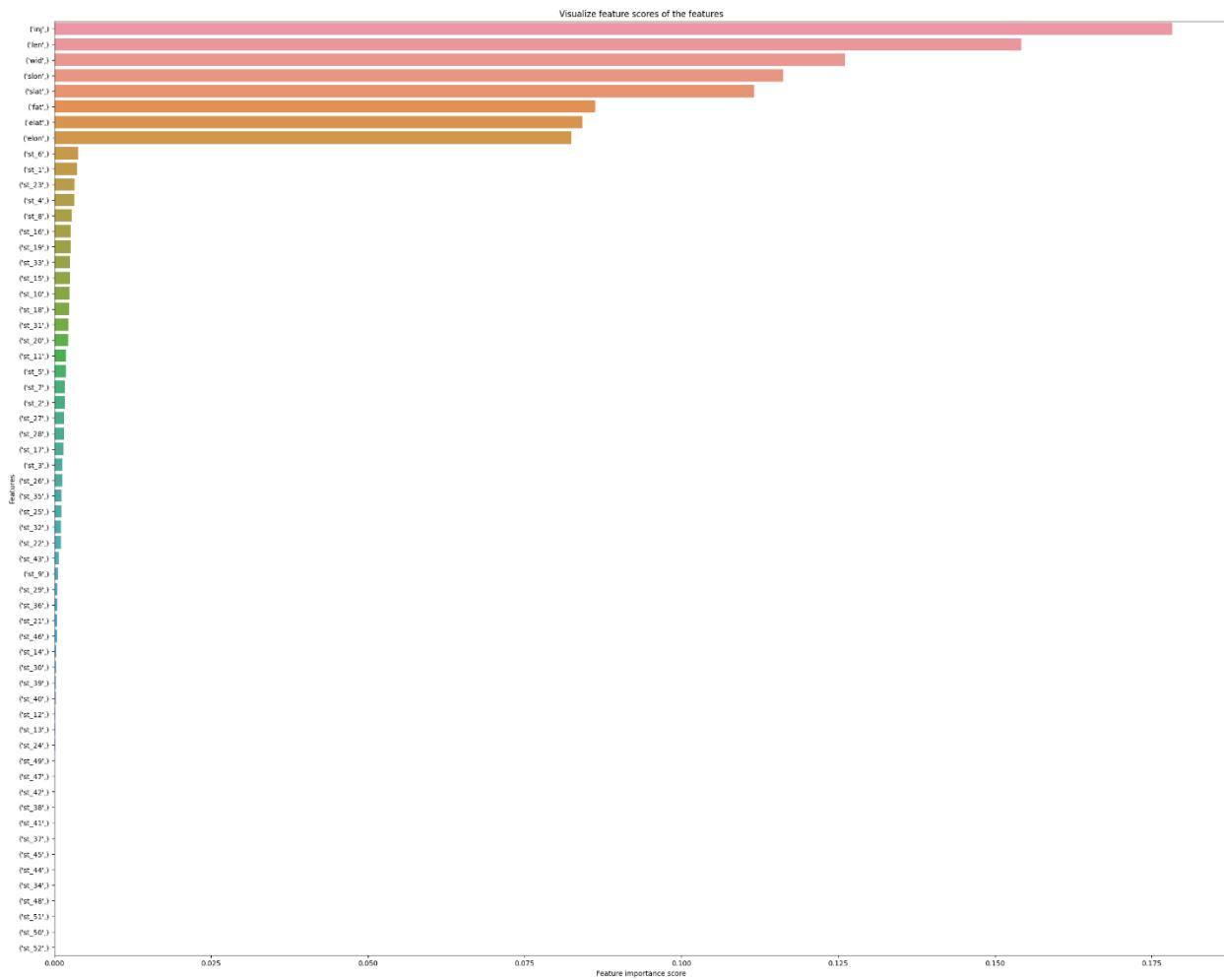
Bước 14: Trực quan hóa điểm đặc trưng với matplotlib và seaborn để thấy được các đặc trưng quan trọng và ít quan trọng rõ hơn

```
[35] index_strings = [str(idx) for idx in feature_scores.index]
```



```
# Creating a seaborn bar plot

f, ax = plt.subplots(figsize=(30, 24))
ax = sns.barplot(x=feature_scores, y=index_strings, data=df)
ax.set_title("Visualize feature scores of the features")
ax.set_yticklabels(feature_scores.index)
ax.set_xlabel("Feature importance score")
ax.set_ylabel("Features")
plt.show()
```



→ Dựa vào biểu đồ, thấy được đặc trưng quan trọng nhất là ‘inj’ và ít quan trọng nhất là ‘st 52’

Bước 15: Tao ma trận nhầm lẫn và chia thành 4 phần

Ma trận nhầm lẫn là một công cụ để tóm tắt hiệu suất của thuật toán phân loại. Một ma trận nhầm lẫn sẽ cho chúng ta một bức tranh rõ ràng về hiệu suất của mô hình phân loại và các loại lỗi do mô hình tạo ra. Nó cung cấp cho chúng tôi một bản tóm tắt các dự đoán chính xác và không chính xác được chia nhỏ theo từng danh mục. Tóm tắt được thể hiện dưới dạng bảng. Bốn loại kết quả có thể xảy ra khi đánh giá hiệu suất của mô hình phân loại:

- True Positives (TP): xảy ra khi số lượng dữ liệu được dự đoán đúng là positive
  - True Negatives (TN): xảy ra khi số lượng dữ liệu được dự đoán đúng là negative
  - False Positives (FP): xảy ra khi số lượng dữ liệu được dự đoán sai thành positive nhưng thực tế là negative
  - False Negatives (FN): xảy ra khi số lượng dữ liệu được dự đoán sai thành negative nhưng thực tế là positive

```
[76] # Tạo ma trận nhầm lẫn và chia thành 4 phần
from sklearn.metrics import confusion_matrix

cm = confusion_matrix(y_test, y_pred)

print('Confusion matrix\n\n', cm)
```

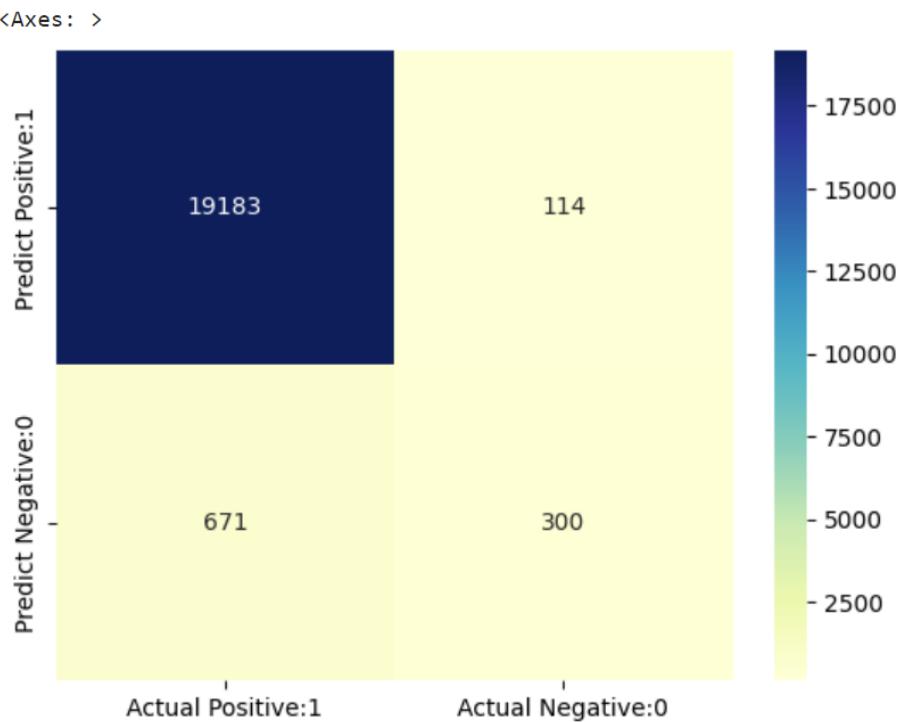
Confusion matrix

```
[[19183    114]
 [ 671    300]]
```

Bước 16: Trực quan hóa ma trận nhầm lẫn với seaborn heatmap

```
# Trực quan hóa ma trận nhầm lẫn với seaborn heatmap
cm_matrix = pd.DataFrame(data=cm, columns=['Actual Positive:1', 'Actual Negative:0'],
                           index=['Predict Positive:1', 'Predict Negative:0'])

sns.heatmap(cm_matrix, annot=True, fmt='d', cmap='YlGnBu')
```



+ Actual Positive:1 : là số lượng lôc xoáy thực tế có mức tàn phá < 3 (mức tàn phá -9, 0, 1, 2: chia như vậy vì dựa theo trung bình số lượng bị thương và tử vong ở từng mức từ rất ít đến nhiều).

+ Predict Positive:1 : là số lượng lôc xoáy dự đoán được có mức tàn phá < 3.

+ Actual Negative:0 : là số lượng lốc xoáy thực tế có mức tàn phá  $\geq 3$  ( mức tàn phá 3, 4, 5: chia như vậy vì ở các mức này trung bình số lượng số bị thương và tử vong rất nhiều so với các mức còn lại ).

+ Predict Negative:0 : là số lượng lốc xoáy dự đoán được có mức tàn phá  $\geq 3$

→ Dựa vào biểu đồ trên có nhận xét:

- Ô màu xanh navy là ô chỉ số lượng lốc xoáy được dự đoán trùng khớp với số lượng lốc xoáy thực tế ở mức tàn phá  $< 3$  ( trùng khớp 19183 / 20268 ).

- Ô màu vàng nhạt gốc dưới phải là ô chỉ số lượng lốc xoáy được dự đoán trùng khớp với số lượng lốc xoáy thực tế ở mức tàn phá  $\geq 3$  ( trùng khớp 300 / 1085 ).

#### Bước 17: Đánh giá hiệu suất của mô hình

- Với các độ đo:

- Accuracy (độ chính xác toàn bộ): Tỷ lệ giữa số lượng dự đoán đúng và tổng số mẫu.
- Precision (độ chính xác): Tỷ lệ giữa số lượng dự đoán đúng của lớp positive và tổng số dự đoán positive (bao gồm cả dự đoán đúng và dự đoán sai)
- Recall : Tỷ lệ giữa số lượng dự đoán đúng của lớp positive và tổng số mẫu thực tế thuộc lớp positive
- F1-score: Một phép đo kết hợp của precision và recall. Giá trị F1-score càng cao thể hiện mô hình có khả năng cân bằng giữa precision và recall

```
[41] from sklearn.metrics import classification_report
```

```
print(classification_report(y_test, y_pred))
```

	precision	recall	f1-score	support
<3	0.97	0.99	0.98	19297
$\geq 3$	0.72	0.31	0.43	971
accuracy			0.96	20268
macro avg	0.85	0.65	0.71	20268
weighted avg	0.95	0.96	0.95	20268

➔ Ta thấy:

- Accuracy = 0.96 → độ chính xác của mô hình là 0.96, tức là mô hình dự đoán đúng khoảng 96%
  - Precision của lớp “<3” là 0.97 và lớp “>=3” là 0.72 → Mô hình có độ chính xác cao trong việc dự đoán lớp “<3” hơn là “>=3”
  - Recall của lớp “<3” là 0.99 và lớp “>=3” là 0.31
  - F1-score của lớp “<3” là 0.98 và lớp “>=3” là 0.43
- ➔ Dựa vào các kết quả trên cho thấy mô hình có khả năng dự đoán lớp “<3” tốt hơn lớp “>=3”.

## 2. Decision Tree

Bước 1: Import thư viện

```
[2] # Importing the required libraries
    import pandas as pd
    import numpy as np
    import matplotlib.pyplot as plt
    from sklearn import metrics
    import seaborn as sns
    from sklearn.model_selection import train_test_split
    from sklearn import tree
    from sklearn.tree import DecisionTreeClassifier, plot_tree
    from sklearn.preprocessing import RobustScaler
    import category_encoders as ce
    from sklearn import datasets
```

Bước 2: Import dataset

```
[3] df = pd.read_csv('us_tornado_dataset_1950_2021.csv')
```

```
[4] df.head()
```

	yr	mo	dy	date	st	mag	inj	fat	slat	slon	elat	elon	len	wid
0	1950	1	3	1950-01-03	IL	3	3	0	39.10	-89.30	39.12	-89.23	3.6	130
1	1950	1	3	1950-01-03	MO	3	3	0	38.77	-90.22	38.83	-90.03	9.5	150
2	1950	1	3	1950-01-03	OH	1	1	0	40.88	-84.58	0.00	0.00	0.1	10
3	1950	1	13	1950-01-13	AR	3	1	1	34.40	-94.37	0.00	0.00	0.6	17
4	1950	1	25	1950-01-25	IL	2	0	0	41.17	-87.33	0.00	0.00	0.1	100

Bước 3: Kiểm tra kích thước của tập dữ liệu

```
[5] df.shape
```

(67558, 14)

Bước 4: Xem tóm tắt tập dữ liệu

[6] df.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 67558 entries, 0 to 67557
Data columns (total 14 columns):
 #   Column  Non-Null Count  Dtype  
--- 
 0   yr      67558 non-null   int64  
 1   mo      67558 non-null   int64  
 2   dy      67558 non-null   int64  
 3   date    67558 non-null   object  
 4   st      67558 non-null   object  
 5   mag     67558 non-null   int64  
 6   inj     67558 non-null   int64  
 7   fat     67558 non-null   int64  
 8   slat    67558 non-null   float64 
 9   slon    67558 non-null   float64 
 10  elat    67558 non-null   float64 
 11  elon    67558 non-null   float64 
 12  len     67558 non-null   float64 
 13  wid     67558 non-null   int64  
dtypes: float64(5), int64(7), object(2)
memory usage: 7.2+ MB
```

Bước 5: Xem thuộc tính thống kê của tập dữ liệu

```
[7] df.describe().T
```

	count	mean	std	min	25%	50%	75%	max
<b>yr</b>	67558.0	1991.341618	19.330015	1950.0000	1976.0000	1994.00	2008.00	2021.0000
<b>mo</b>	67558.0	5.976761	2.438192	1.0000	4.0000	6.00	7.00	12.0000
<b>dy</b>	67558.0	15.921016	8.736773	1.0000	8.0000	16.00	24.00	31.0000
<b>mag</b>	67558.0	0.691273	1.283375	-9.0000	0.0000	1.00	1.00	5.0000
<b>inj</b>	67558.0	1.437876	18.263956	0.0000	0.0000	0.00	0.00	1740.0000
<b>fat</b>	67558.0	0.090470	1.484106	0.0000	0.0000	0.00	0.00	158.0000
<b>slat</b>	67558.0	37.142412	5.093979	17.7212	33.2200	37.03	40.93	61.0200
<b>slon</b>	67558.0	-92.784618	8.689103	-163.5300	-98.4500	-93.60	-86.73	-64.7151
<b>elat</b>	67558.0	22.730695	18.588638	0.0000	0.0000	32.48	38.61	61.0200
<b>elon</b>	67558.0	-56.245590	45.489157	-163.5300	-94.7098	-84.42	0.00	0.0000
<b>len</b>	67558.0	3.478340	8.278775	0.0000	0.1000	0.80	3.13	234.7000
<b>wid</b>	67558.0	106.577030	205.802676	0.0000	20.0000	50.00	100.00	4576.0000

Bước 6: Kiểm tra có bao nhiêu giá trị null ở các cột

```
[8] df.isna().sum()
```

```

yr      0
mo      0
dy      0
date    0
st      0
mag     0
inj     0
fat     0
slat    0
slon    0
elat    0
elon    0
len     0
wid     0
dtype: int64

```

Bước 7: Chuẩn bị dữ liệu

- Chuyển các giá trị của ‘mag’ (độ đo F/EF) : về giá trị là ‘<3’ và ‘>=3’

- Với ' $<3$ ' : lốc xoáy không mạnh (-9: yếu, 0: nhẹ, 1: trung bình, 2: đáng kể)
- Với ' $\geq 3$ ' : lốc xoáy mạnh (3: nghiêm trọng, 4: hủy diệt, 5: siêu hủy diệt)
- Sau đó tách tập dữ liệu thành các tập X(đầu vào) và tập Y(đầu ra)

```
[9] df['mag'] = df['mag'].replace({-9:'<3', 0:'<3', 1:'<3', 2:'<3', 3:'>=3', 4:'>=3', 5:'>=3'})
    # <3: lốc xoáy không mạnh
    # >=3 lốc xoáy mạnh

[10] #Tách dữ liệu thành các tập X(đầu vào) và tập Y(đầu ra)
    X = df[['st','inj','fat','slat','slon','elat','elon','len','wid']]

    y = df['mag']
```

Bước 8: Chia dữ liệu thành tập Train và Test theo tỉ lệ 70:30. Và kiểm tra kích thước của tập X\_train, X\_test

```
[11] X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3, random_state = 0)

[12] # Check the shape of X_train and X_test
    X_train.shape, X_test.shape

((47290, 9), (20268, 9))
```

Bước 9: Mã hóa các biến phân loại

- Xem các biến phân loại và các biến số trong tập train

```
[13] # Hiển thị các biến phân loại trong tập train
    categorical = [col for col in X_train.columns if X_train[col].dtypes == 'O']

    categorical

['st']

[14] # Hiển thị các biến số trong tập train
    numerical = [col for col in X_train.columns if X_train[col].dtypes != 'O']

    numerical

['inj', 'fat', 'slat', 'slon', 'elat', 'elon', 'len', 'wid']
```

- Kiểm tra các giá trị null của các cột ở tập X\_train và X\_test

```
[15] X_train[categorical].isnull().sum()
```

```
st      0  
dtype: int64
```

```
[16] X_test[categorical].isnull().sum()
```

```
st      0  
dtype: int64
```

```
[17] X_train.isnull().sum()
```

```
st      0  
inj     0  
fat     0  
slat    0  
slon    0  
elat    0  
elon    0  
len     0  
wid     0  
dtype: int64
```

```
[18] X_test.isnull().sum()
```

```
st      0  
inj     0  
fat     0  
slat    0  
slon    0  
elat    0  
elon    0  
len     0  
wid     0  
dtype: int64
```

- Xem lại dữ liệu của biến phân loại trong tập X\_train trước khi mã hóa

```
# Xem dữ liệu của các biến phân loại trong tập X_train  
X_train[categorical].head()
```

st	
7360	KS
8210	SD
39146	FL
21602	OK
24363	NE

- Mã hóa biến phân loại ‘st’ bằng one-hot encoding

```
# Mã hóa các biến phân loại bằng one-hot encoding
encoder = ce.OneHotEncoder(cols=['st'])

X_train = encoder.fit_transform(X_train)

X_test = encoder.transform(X_test)
```

- Kiểm tra dữ liệu và kích thước trong tập X\_train: Có thể thấy, ban đầu chỉ có 9 cột trong tập X\_train nhưng giờ số cột là 60 cột

```
X_train.head()

   st_1  st_2  st_3  st_4  st_5  st_6  st_7  st_8  st_9  st_10 ...  st_51  st_52  inj  fat  slat  slon  elat  elon  len  wid
7360    1     0     0     0     0     0     0     0     0     0 ...      0     0     0     0    37.92 -97.42  0.00  0.00  0.1   10
8210    0     1     0     0     0     0     0     0     0     0 ...      0     0     0     0    45.20 -100.30 0.00  0.00  0.1   10
39146   0     0     1     0     0     0     0     0     0     0 ...      0     0     0     0    24.92 -80.63 24.92 -80.63 2.0   50
21602   0     0     0     1     0     0     0     0     0     0 ...      0     0     1     0    35.22 -97.43  0.00  0.00  0.3   10
24363   0     0     0     0     1     0     0     0     0     0 ...      0     0     0     0    40.72 -98.80  0.00  0.00  0.5   30
5 rows × 60 columns
```

```
X_train.shape
(47290, 60)
```

- Tương tự với tập X\_test, ban đầu cũng chỉ có 9 cột và hiện tại là 60 cột

```
X_test.head()

   st_1  st_2  st_3  st_4  st_5  st_6  st_7  st_8  st_9  st_10 ...  st_51  st_52  inj  fat  slat  slon  elat  elon  len  wid
44086   0     0     0     0     0     0     0     0     0     0 ...      0     0     0     0    35.8800 -86.1000 35.9200 -86.0800 3.00 100
32153   0     0     0     0     0     0     0     0     0     0 ...      0     0     0     0    40.0300 -109.5300 0.0000 0.0000 0.10 10
56458   0     0     0     0     0     0     0     0     0     0 ...      0     0     0     0    29.8816 -92.1028 29.8945 -92.1004 0.90 20
30138   0     0     1     0     0     0     0     0     0     0 ...      0     0     0     0    27.3000 -81.8500 0.0000 0.0000 0.40 50
54955   0     0     0     0     0     0     0     0     0     0 ...      0     0     0     0    38.5233 -89.0708 38.5278 -89.0675 0.36 75
5 rows × 60 columns
```

```
X_test.shape
(20268, 60)
```

## Bước 10: Chuẩn hóa dữ liệu

```
cols = X_train.columns

from sklearn.preprocessing import RobustScaler
scaler = RobustScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

X_train = pd.DataFrame(X_train, columns=[cols])
X_test = pd.DataFrame(X_test, columns=[cols])
```

Bước 11: Xây dựng mô hình Decision Tree và đánh giá độ chính xác

```
▶ # Create Decision Tree classifier object
clf = DecisionTreeClassifier()

# Train Decision Tree Classifier
clf = clf.fit(X_train,y_train)

#Predict the response for test dataset
y_pred = clf.predict(X_test)

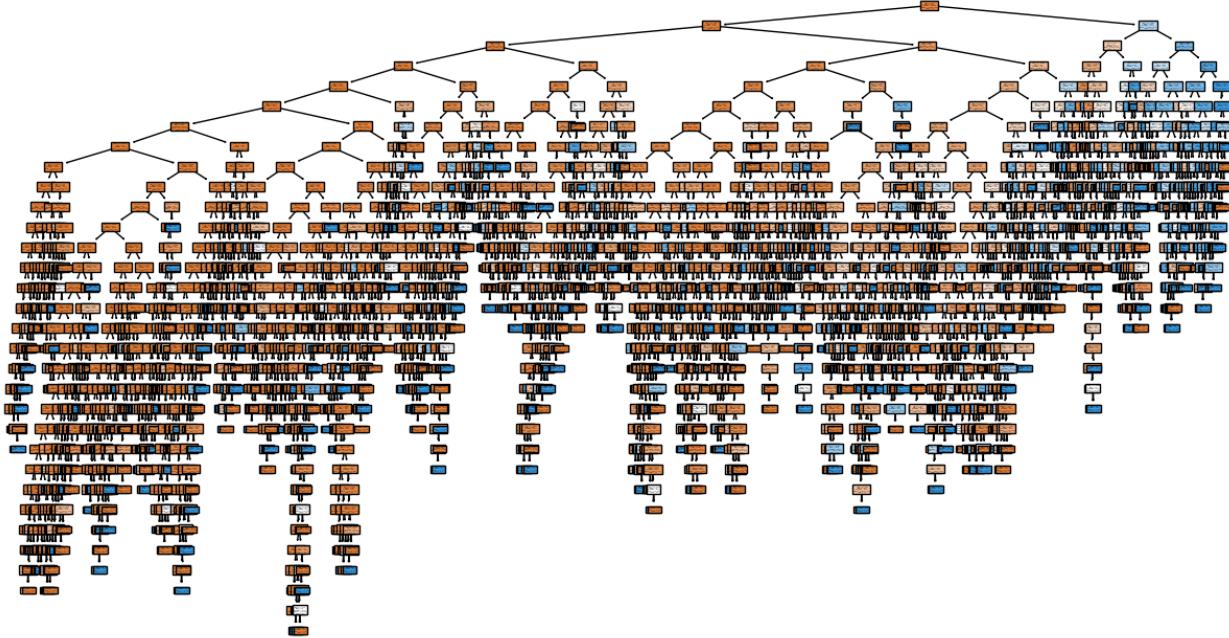
[22] # Model Accuracy
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))

Accuracy: 0.9403986579830275
```

➔ Ta thấy Accuracy = 0.94, tức là độ chính xác của mô hình khoảng 94%

Bước 12: Trực quan hóa cây quyết định

```
▶ plt.figure(figsize=(15, 8))
plot_tree(clf, filled=True, feature_names=X_train.columns, class_names=clf.classes_,proportion = True)
plt.show()
```



### Bước 13: Tối ưu hóa cây quyết định

- Xây dựng lại mô hình Decision Tree với gini và max\_depth = 3 và đánh giá mô hình

```
[45] # Create Decision Tree classifier object
     clf = DecisionTreeClassifier(criterion="gini", max_depth=3)

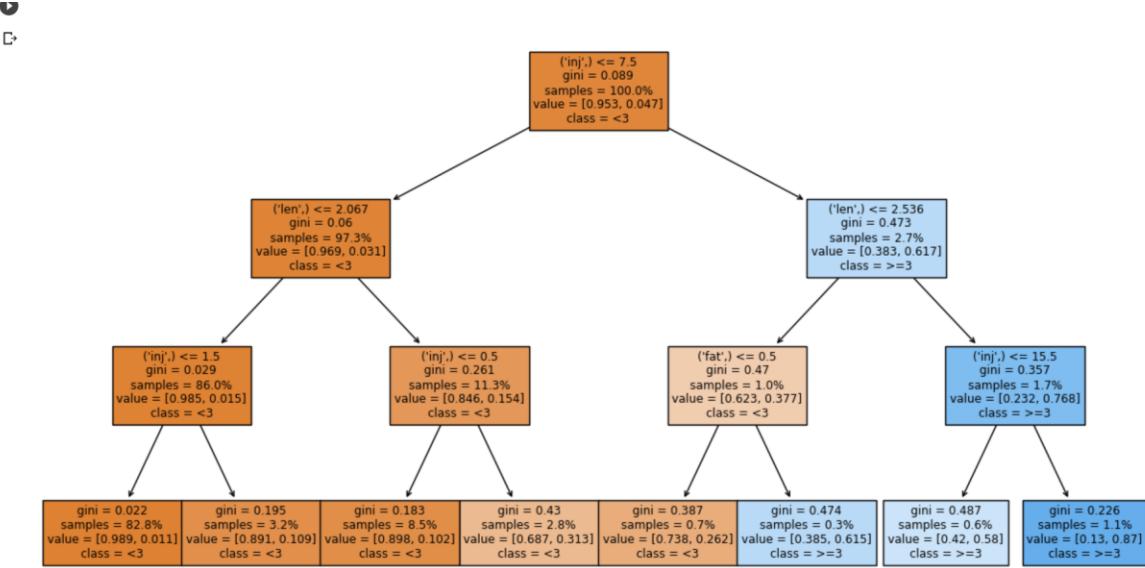
     # Train Decision Tree Classifier
     clf = clf.fit(X_train,y_train)

     #Predict the response for test dataset
     y_pred = clf.predict(X_test)

     # Model Accuracy
     print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

Accuracy: 0.9589500690744029

- ➔ Ta thấy, sau khi tối ưu hóa cây quyết định thì độ chính xác là 0.95, tức là độ chính xác mô hình khoảng 95% cao hơn so với độ chính xác khi chưa tối ưu hóa (94%)
- Trực quan hóa cây quyết định



→ Từ biểu đồ rút ra được quy luật là nào có cùng class = <3 sẽ được xếp về bên trái, được tô cùng màu cam (hoặc cam nhạt hơn) và nào có cùng class = >=3 sẽ được xếp về bên phải, được tô cùng màu xanh dương (hoặc xanh dương nhạt hơn).

Bước 14: Tìm các đặc trưng quan trọng

```

▶ feature_scores = pd.Series(clf.feature_importances_, index=X_train.columns).sort_values(ascending=False)

feature_scores
  
```

inj	0.767751
len	0.211539
fat	0.020710
st_1	0.000000
st_44	0.000000
st_33	0.000000
st_34	0.000000
st_35	0.000000
st_36	0.000000
st_37	0.000000
st_38	0.000000
st_39	0.000000
st_40	0.000000

Bước 15: Trực quan hóa điểm đặc trưng với matplotlib và seaborn để thấy được các đặc trưng quan trọng và ít quan trọng rõ hơn

```
[48] index_strings = [str(idx) for idx in feature_scores.index]
```



```
f, ax = plt.subplots(figsize=(30, 24))
ax = sns.barplot(x=feature_scores, y=index_strings, data=df)
ax.set_title("Visualize feature scores of the features")
ax.set_yticklabels(feature_scores.index)
ax.set_xlabel("Feature importance score")
ax.set_ylabel("Features")
plt.show()
```



- ➔ Dựa vào biểu đồ, thấy được đặc trưng quan trọng nhất là 'inj' và ít quan trọng nhất là 'wid'
- Bước 16: Tiến hành xóa trường ít quan trọng nhất trong X\_train và X\_test

```
▶ # Xóa trường ít quan trọng nhất 'wid' trong X_train và X_test  
X_train = X_train.drop(['wid'], axis=1)  
  
↳ <ipython-input-50-5634b426d3bc>:2: PerformanceWarning: dropping on a non-lexsorted multi-index without a level parameter may impact performance.  
    X_train = X_train.drop(['wid'], axis=1)  
<ipython-input-50-5634b426d3bc>:4: PerformanceWarning: dropping on a non-lexsorted multi-index without a level parameter may impact performance.  
    X_test = X_test.drop(['wid'], axis=1)
```

Bước 17: Xây dựng lại mô hình Decision Tree và đánh giá lại mô hình

```
▶ # Create Decision Tree classifier object  
clf = DecisionTreeClassifier(criterion="gini", max_depth=3)  
  
# Train Decision Tree Classifier  
clf = clf.fit(X_train,y_train)  
  
#Predict the response for test dataset  
y_pred = clf.predict(X_test)  
  
# Model Accuracy  
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))  
  
↳ Accuracy: 0.9589500690744029
```

➔ Kết quả trên cho thấy độ chính xác của mô hình sau khi loại bỏ trường ít quan trọng nhất vẫn không thay đổi (95%)

Bước 18: Tạo ma trận nhầm lẫn và chia thành 4 phần

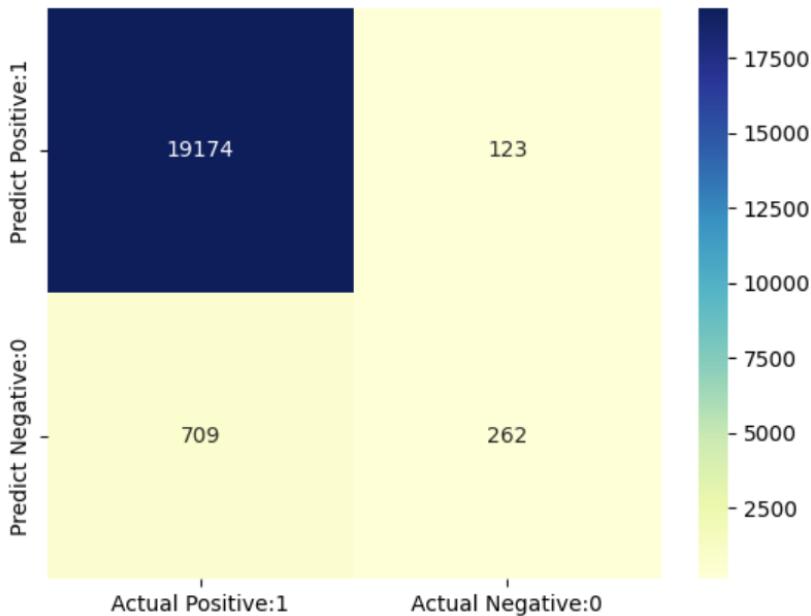
```
▶ # Tạo ma trận nhầm lẫn  
from sklearn.metrics import confusion_matrix  
cm = confusion_matrix(y_test, y_pred)  
  
print('Confusion matrix\n\n', cm)  
  
↳ Confusion matrix  
  
[[19174  123]  
 [ 709  262]]
```

Bước 19: Trực quan hóa ma trận nhầm lẫn với seaborn heatmap

```
# Trực quan hóa ma trận nhầm lẫn với seaborn heatmap
cm_matrix = pd.DataFrame(data=cm, columns=['Actual Positive:1', 'Actual Negative:0'],
                           index=['Predict Positive:1', 'Predict Negative:0'])

sns.heatmap(cm_matrix, annot=True, fmt='d', cmap='YlGnBu')
```

⇨ <Axes: >



+ Actual Positive:1 : là số lượng lốc xoáy thực tế có mức tàn phá  $< 3$  ( mức tàn phá -9, 0, 1, 2: chia như vậy vì dựa theo trung bình số lượng bị thương và tử vong ở từng mức từ rất ít đến nhiều ).

+ Predict Positive:1 : là số lượng lốc xoáy dự đoán được có mức tàn phá  $< 3$ .

+ Actual Negative:0 : là số lượng lốc xoáy thực tế có mức tàn phá  $\geq 3$  ( mức tàn phá 3, 4, 5: chia như vậy vì ở các mức này trung bình số lượng số bị thương và tử vong rất nhiều so với các mức còn lại ).

+ Predict Negative:0 : là số lượng lốc xoáy dự đoán được có mức tàn phá  $\geq 3$

→ Dựa vào biểu đồ trên có nhận xét:

- Ô màu xanh navy là ô chỉ số lượng lốc xoáy được dự đoán trùng khớp với số lượng lốc xoáy thực tế ở mức tàn phá  $< 3$  ( trùng khớp 19174 / 20268 ).

- Ô màu vàng nhạt gốc dưới phải là ô chỉ số lượng lốc xoáy được dự đoán trùng khớp với số lượng lốc xoáy thực tế ở mức tàn phá  $\geq 3$  ( trùng khớp 262 / 1085 ).

## Bước 20: Đánh giá hiệu suất của mô hình

```
[55] from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred))
```

	precision	recall	f1-score	support
<3	0.96	0.99	0.98	19297
>=3	0.68	0.27	0.39	971
accuracy			0.96	20268
macro avg	0.82	0.63	0.68	20268
weighted avg	0.95	0.96	0.95	20268

➔ Ta thấy:

- Accuracy = 0.96 → độ chính xác của mô hình là 0.96, tức là mô hình dự đoán đúng khoảng 96%
  - Precision của lớp “<3” là 0.96 và lớp “>=3” là 0.68 → Mô hình có độ chính xác cao trong việc dự đoán lớp “<3” hơn là “>=3”
  - Recall của lớp “<3” là 0.99 và lớp “>=3” là 0.27
  - F1-score của lớp “<3” là 0.98 và lớp “>=3” là 0.39
- ➔ Dựa vào các kết quả trên cho thấy mô hình có khả năng dự đoán lớp “<3” tốt hơn lớp “>=3”.

## TÀI LIỆU THAM KHẢO

- [1] A.Navlani, “Decision Tree Classification in Python Tutorial”, Feb 2023. Địa chỉ:  
<https://www.datacamp.com/tutorial/decision-tree-classification-python>
- [2] “Feature Importance”, Codecademy. Địa chỉ:  
<https://www.codecademy.com/article/fe-feature-importance-final>
- [3] D.Xuân, “Triển khai cây quyết định bằng Python”, 29/12/2020. Địa chỉ:  
<https://cafedev.vn/tu-hoc-ml-trien-khai-cay-quyet-dinh-bang-python/>
- [4] “1.10. Decision Trees”, scikit-learn. Địa chỉ: [https://scikit-learn.org/stable/modules/tree.html?fbclid=IwAR13wHlob8urPQYel3AaD\\_tvSuce2jfG0rRicyVNSmBUAMAyG1J0g0mK0v8#classification](https://scikit-learn.org/stable/modules/tree.html?fbclid=IwAR13wHlob8urPQYel3AaD_tvSuce2jfG0rRicyVNSmBUAMAyG1J0g0mK0v8#classification)