- relation = table

VD : Sinh Viên

- schema (lược đồ ) = tên bảng (ds các thuộc tính )

VD: SinhVien(MaSV, HoTen, Email, MaLop ,Khoa)

-Tuples ( record) : Là các dòng chứa dữ liệu ở trong bảng, mỗi một dòng là 1 tuple.

- Domain: kiểu dữ liệu của mỗi thuộc tính

**\*Các loại kiểu dữ liệu:**

1.Char(n): xâu kí tự có độ dài cố định là n

2.Varchar(n): xâu kí tự có độ dài tối đa là n

3. Nchar , nvarchar : giống vs char và nvarchar nhưng dùng 2 kiểu này sẽ dc mã hóa thành dạng tiếng việt có dấu

**- Primary key (khóa chính ):** là tập các thuộc tính giúp xác định duy nhất 1 dòng ở trong bảng

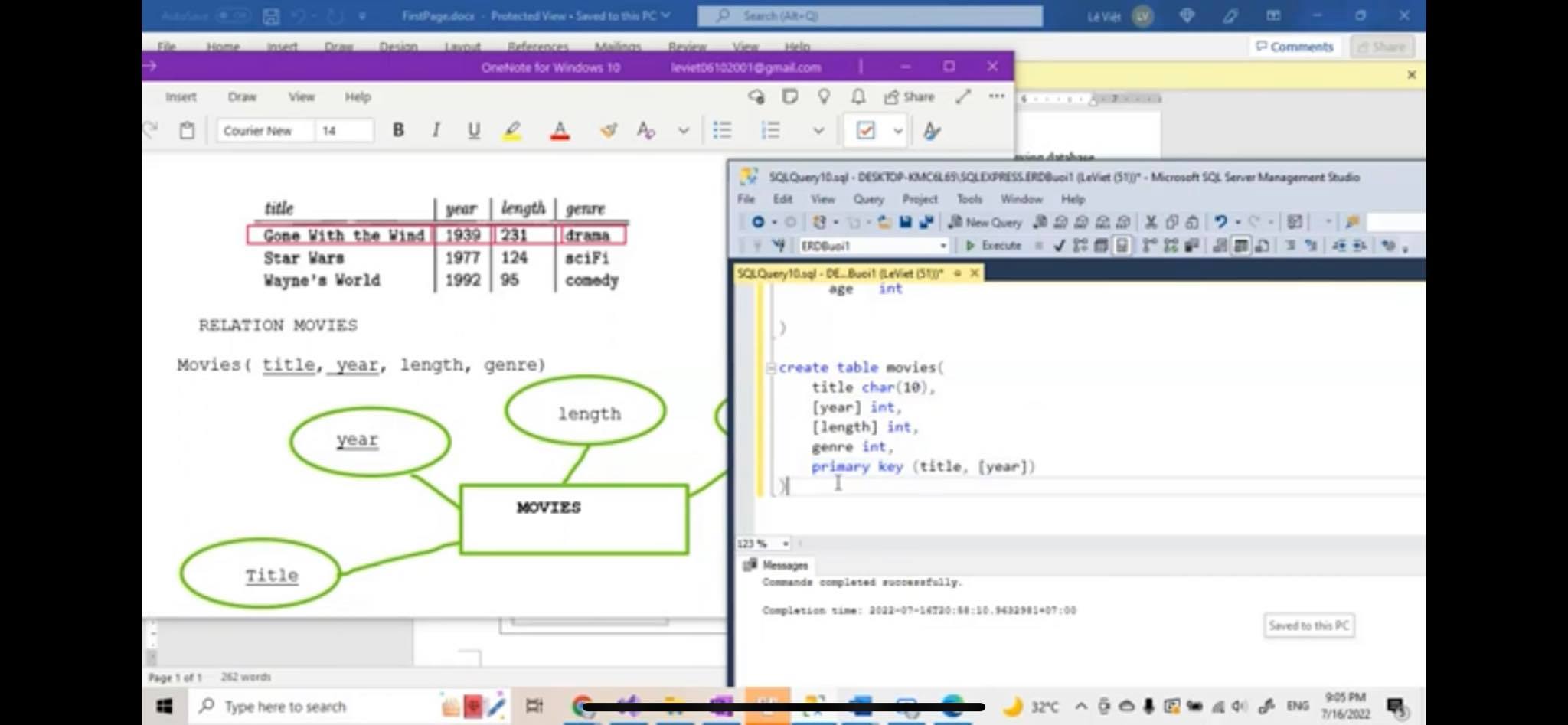
Khóa chính -🡪 biết được giá trị của tất cả các thuộc tính còn lại ở trong bảng .

\* Điều kiện để được làm khóa chính :

- Nó phải giúp xác định duy nhất 1 dòng trong bảng

- Key phải tối thiểu: không chứa các thuộc tính thừa nằm trong key.

- Mỗi bảng chỉ có duy nhất 1 key, một key có thể bao gồm nhiều thuộc tính.



- **Foreign key (Khóa ngoại)** : là một tập các thuộc tính tham chiếu (references) đến khóa chính các bảng khác.

-Tạo ra một mối quan hệ giữa các bảng với nhau.

**Entity Relationship**

**(Mô hình mối quan hệ thực thể )**

ERD = Entity Relationship Diagram.

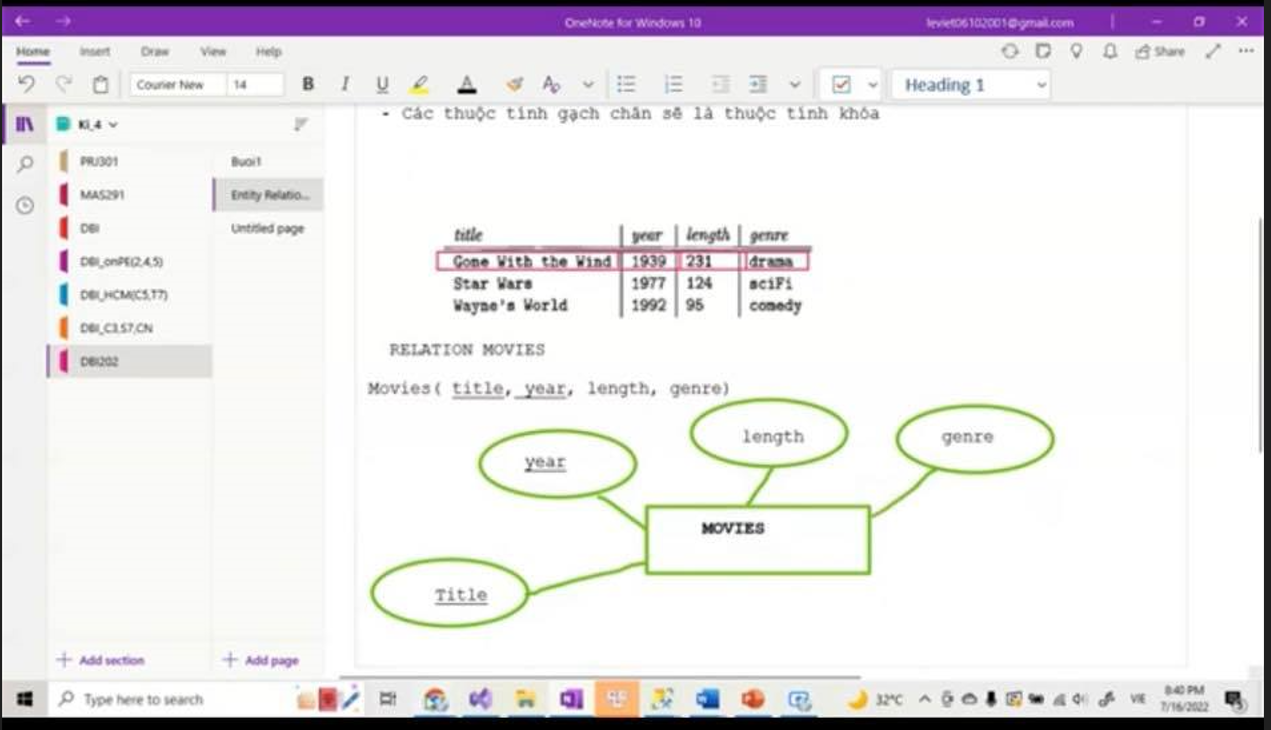
\* Trong mô hình ERD :

- thì mỗi một thực thể sẽ được biểu diễn bằng hình chữ nhật .

- Mỗi một thuộc tính sẽ được biểu diễn bằng hình oval nối với thực thể tương ứng.

- Các thuộc tính gạch chân sẽ là thuộc tính khóa .

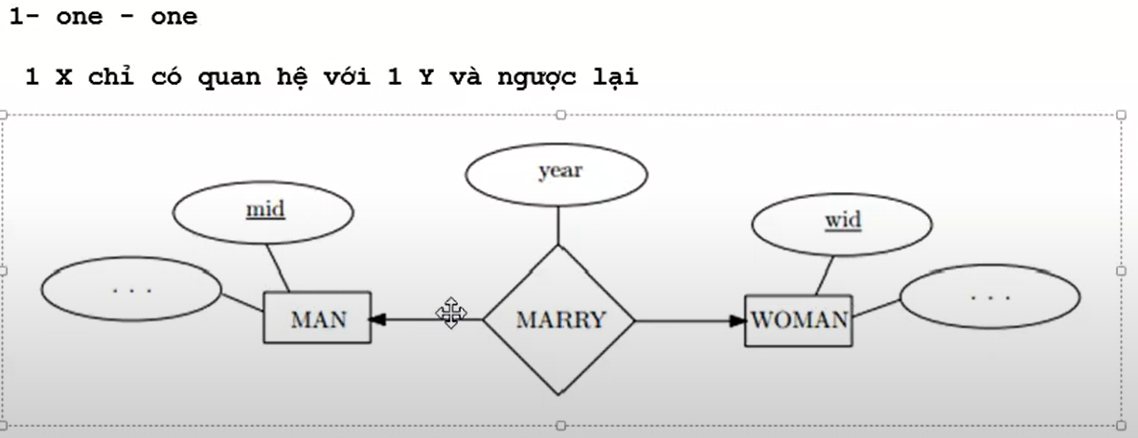
VD : Movies( title, year, length, genre) 🡪 biễu diễn theo lược đồ

ERD: 

\* Có 3 loại mối quan hệ : one – one , one –many, many-many.Mối quan hệ sẽ được biểu diễn bằng hình thoi.

**1) One –one:**

+) 1 **X** chỉ có quan hệ với 1 **Y** và ngược lại .



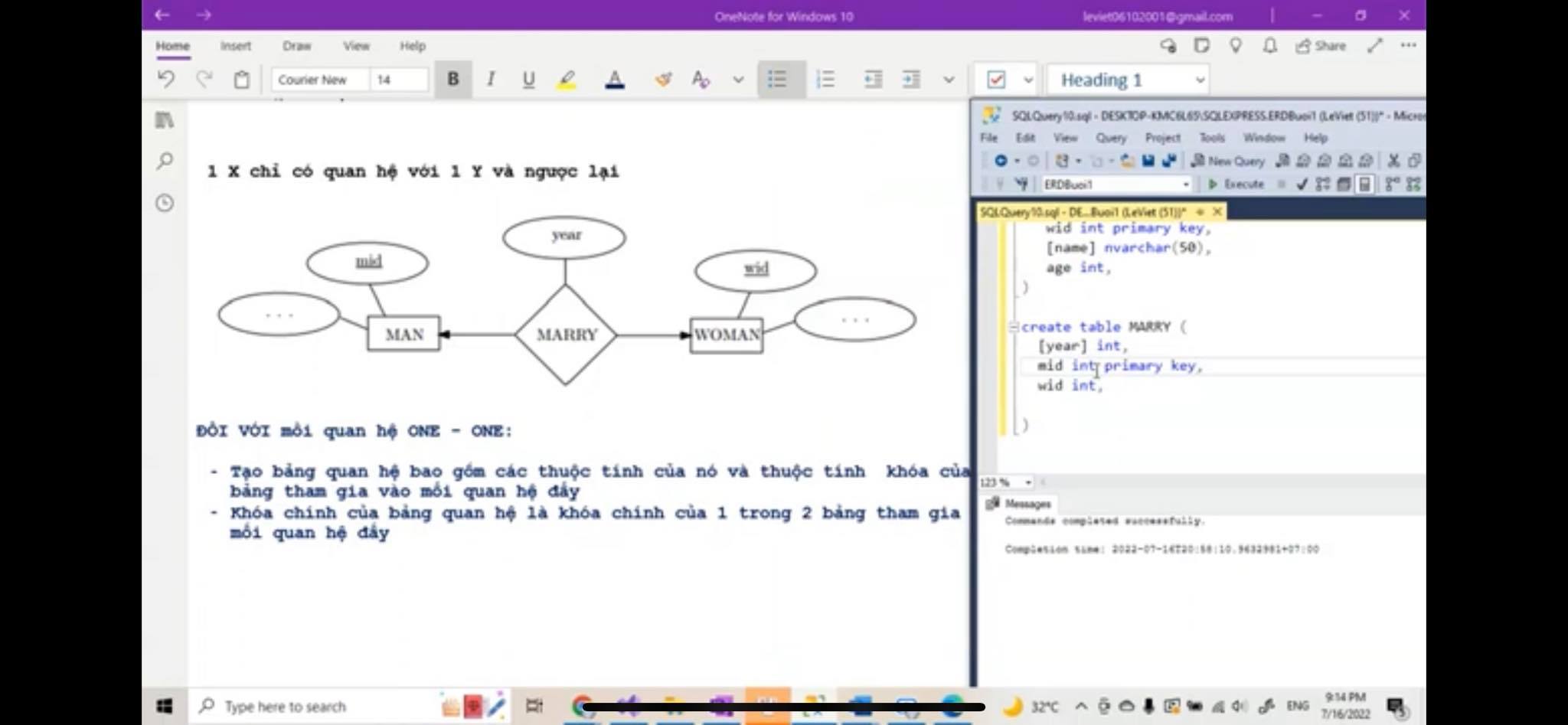
-Thuộc tính year là năm kết hôn của 2 người chứ không phải chứ không phải year của riêng man và woman nên dc để ở marry.

**\* Đối với mối quan hệ ONE – ONE:**

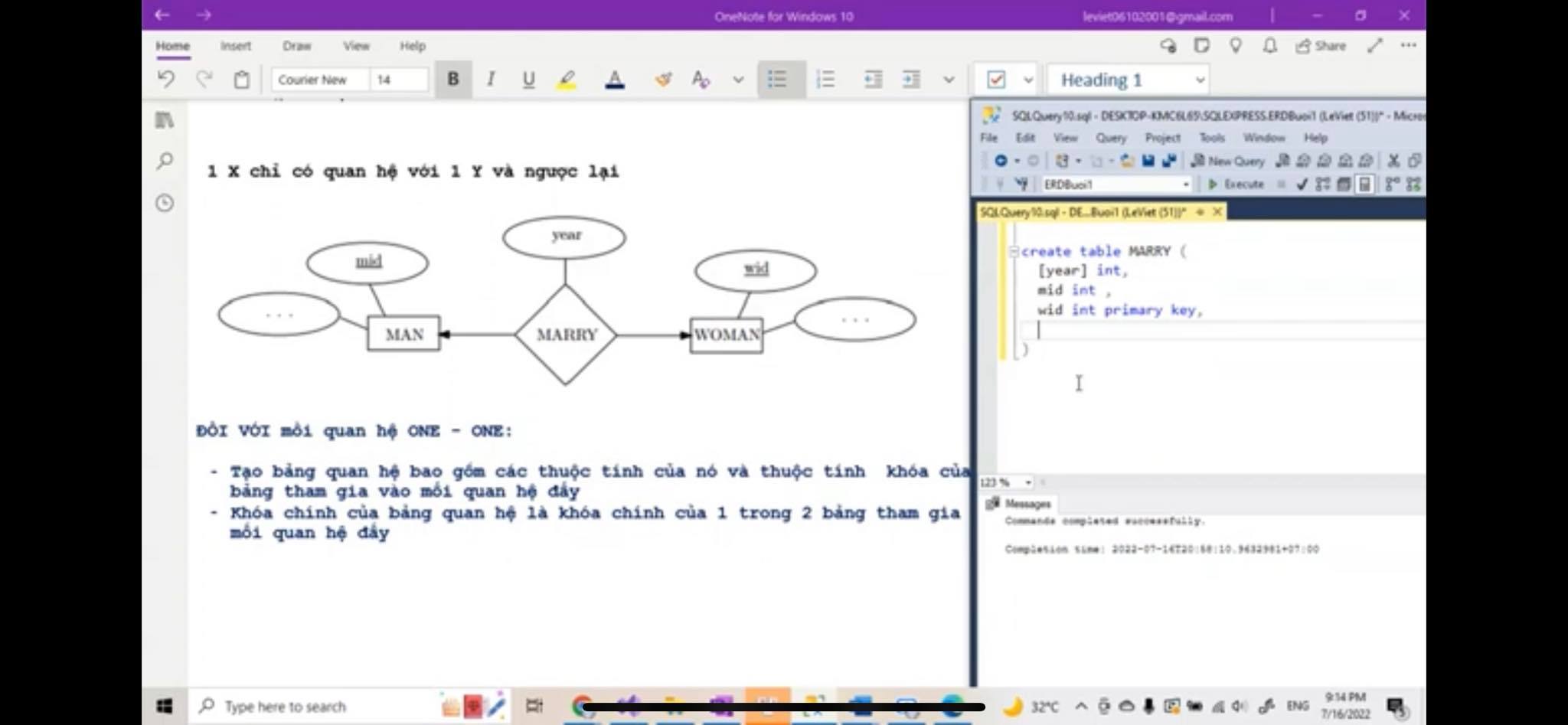
- Tạo bảng quan hệ bao gồm các thuộc tính của nó và thuộc tính khóa của 2 bảng tham gia vào mối quan hệ đấy .

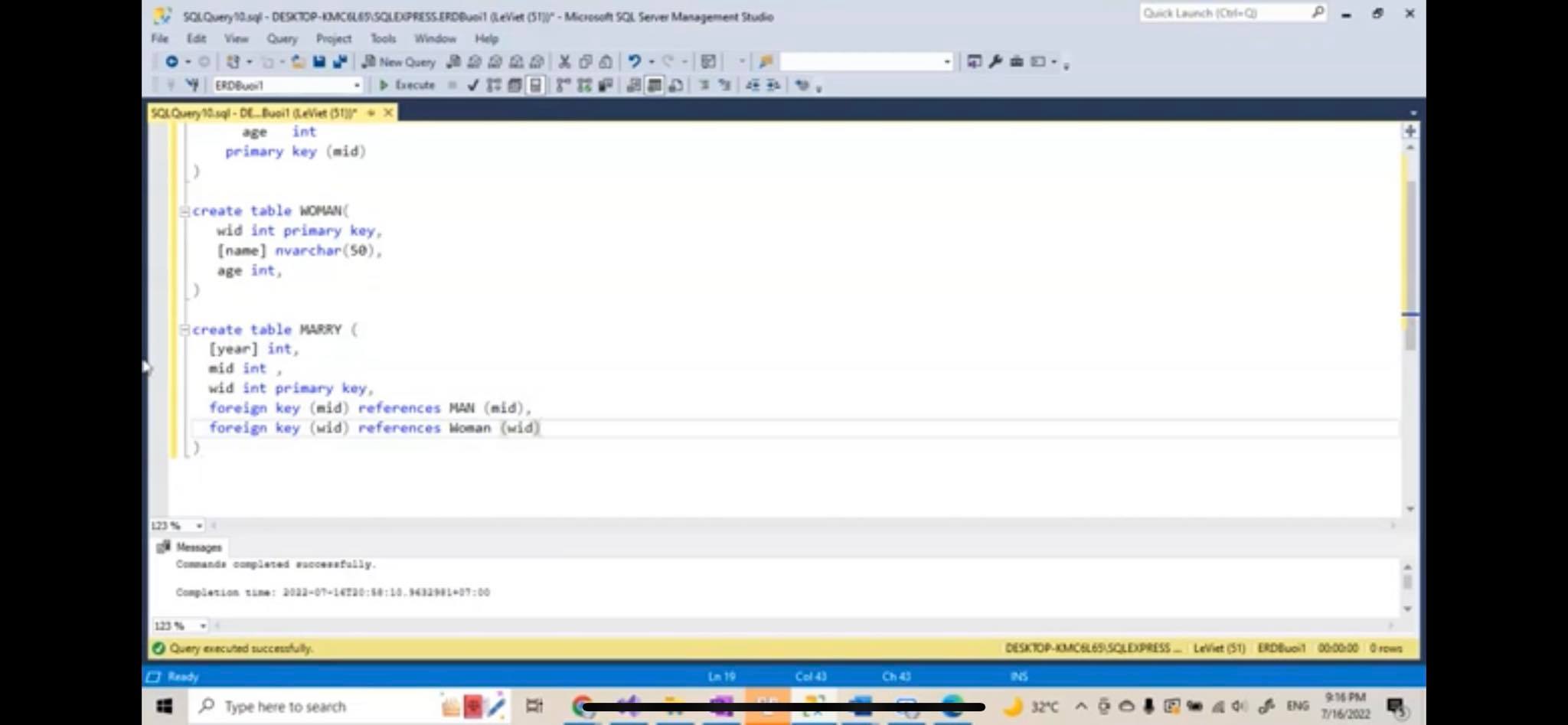
- Khóa chính của bảng quan hệ là khóa chính của 1 trong 2 bảng tham gia vào mối quan hệ đấy.

TH1: mid là khóa chính:



TH2: wid là khóa chính:



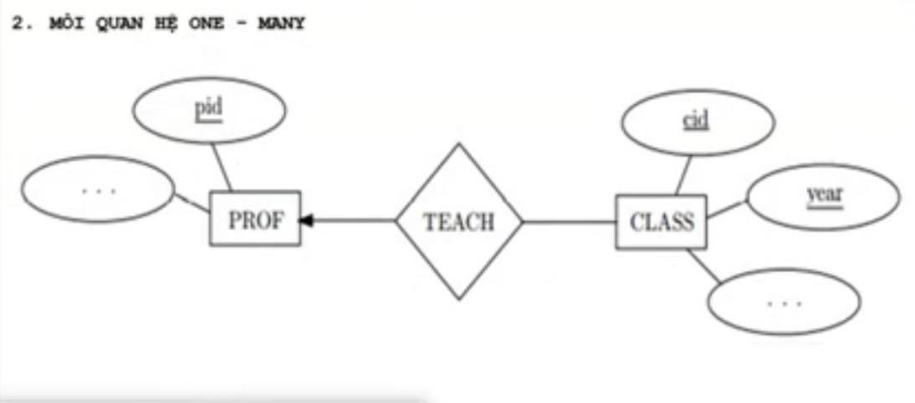


**\* Đối với mối quan hệ ONE-MANY:**

- Gỉa sử X có quan hệ ONE – MANY với Y thì :

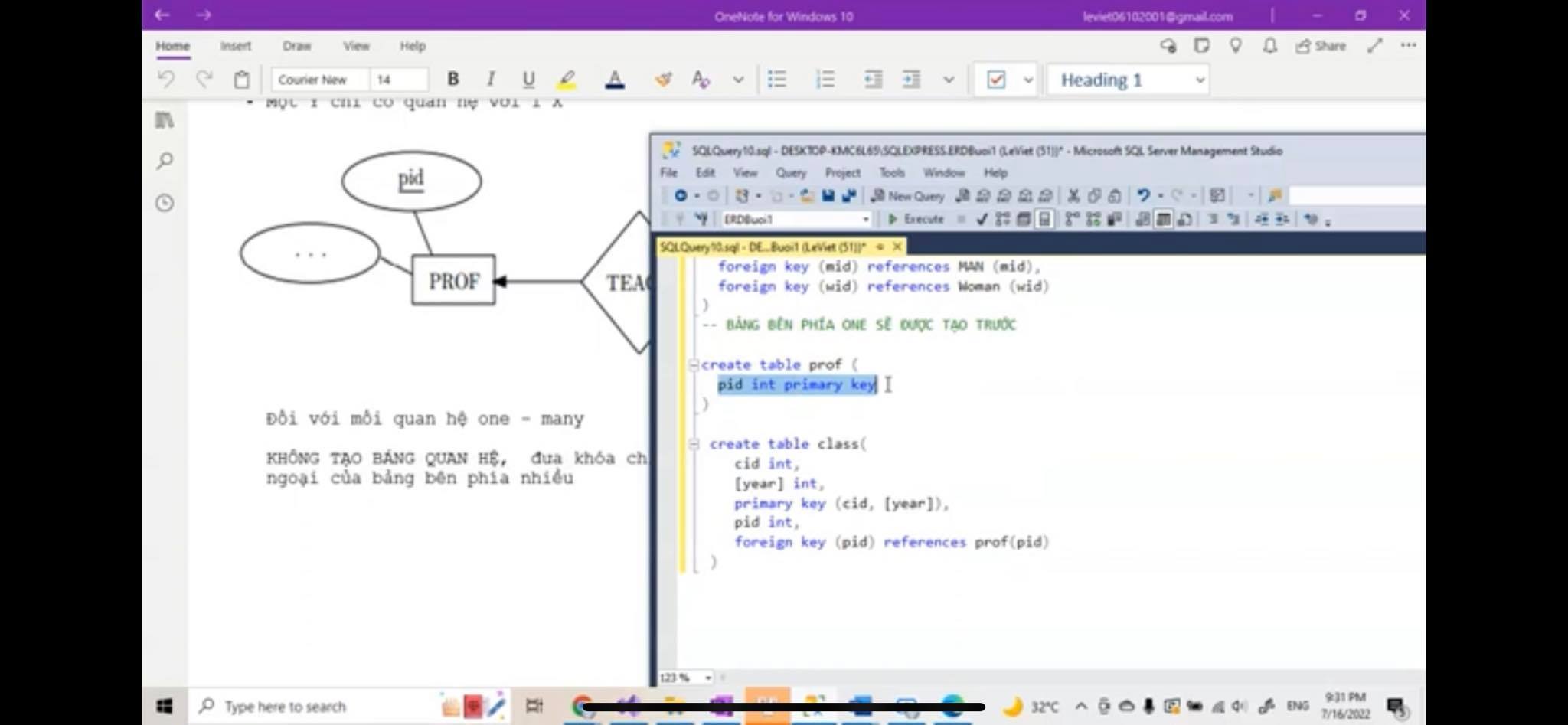
+ **Một** X sẽ có quan hệ với **Nhiều** Y

+ **Một** Y chỉ có quan hệ với **Một** X



**\* Đối với mối quan hệ ONE – MANY:**

-**KHÔNG TẠO BẢNG QUAN HỆ**,đưa **khóa chính** của bảng bên phía **ONE** sang làm **khóa ngoại** của bảng bên phía **MANY**.



- Bên **ONE** chứa mũi tên , bên **MANY** không chứa mũi tên.

- Bảng bên phía **ONE** sẽ được tạo trước bảng **MANY**.

VD: bảng **PROF** sẽ tạo trước bảng **CLASS**.

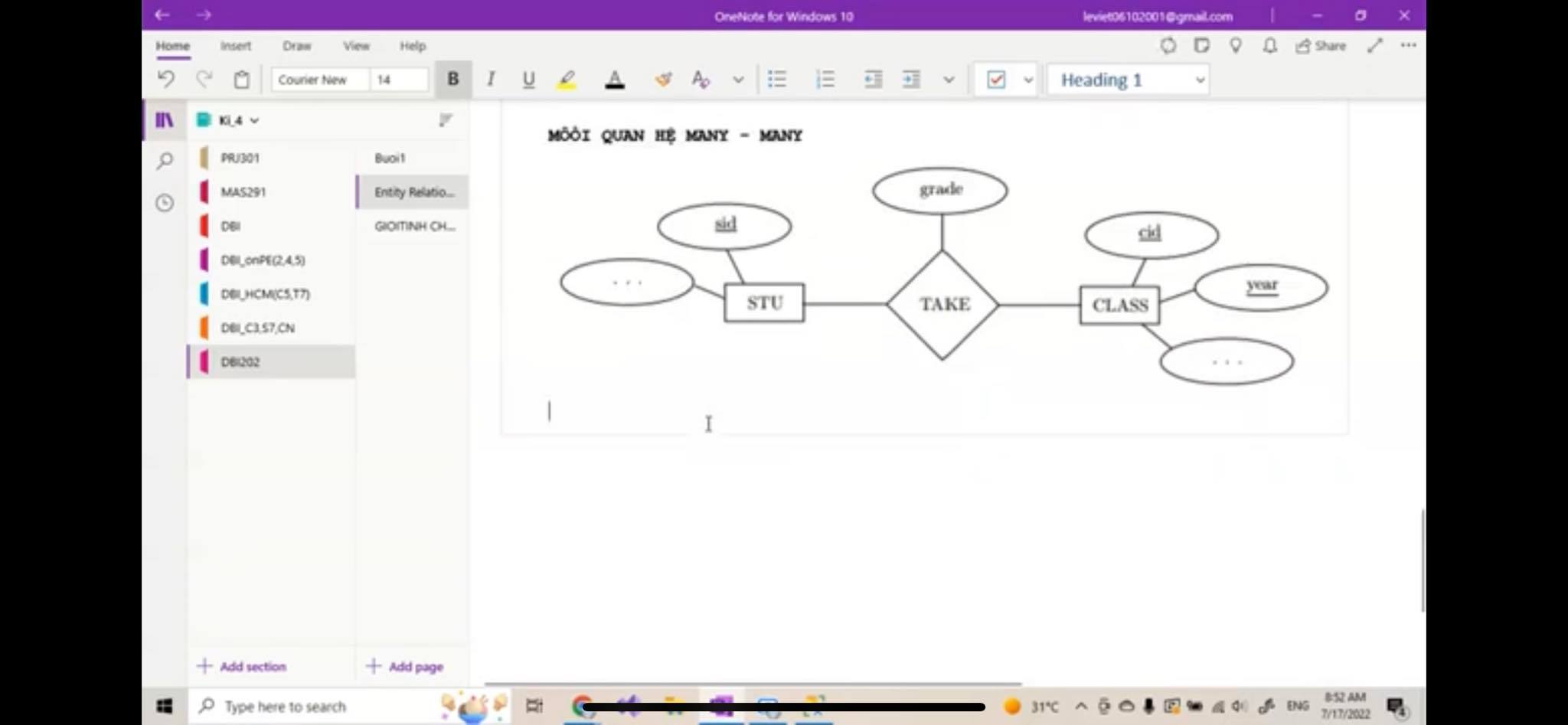
VD: + PROF : là **ONE**

+ CLASS: là bên **MANY**.

**\* Đối với mối quan hệ MANY – MANY:**

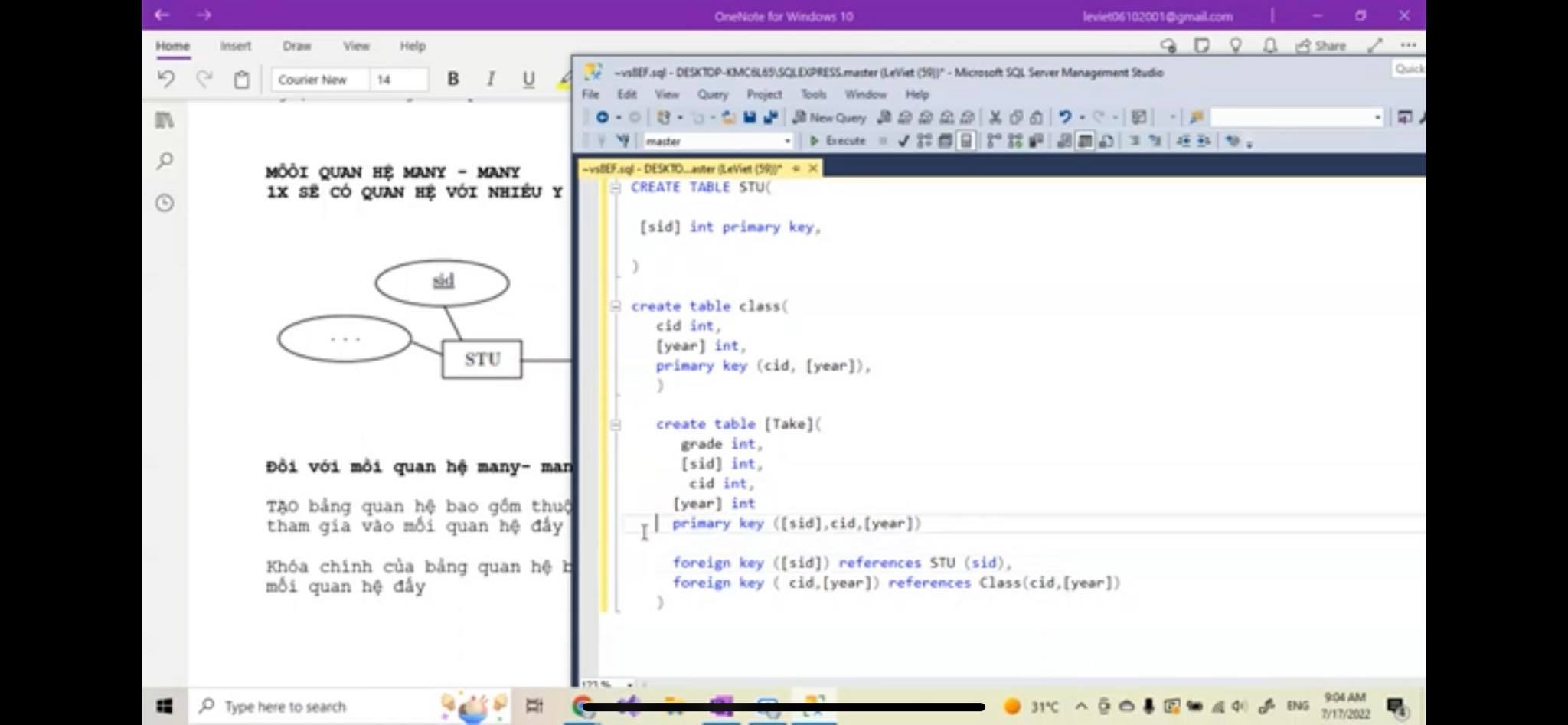
- **Một X** sẽ có quan hệ với **nhiều Y**

- **Một Y** sẽ có quan hệ với **nhiều X**



- Tạo bảng quan hệ bao gồm thuộc tính của nó và thuộc tính khóa của 2 bảng tham gia vào mối quan hệ đấy .

-Khóa chính của bảng quan hệ bao gồm khóa chính của cả 2 bảng tham gia vào mối quan hệ đấy

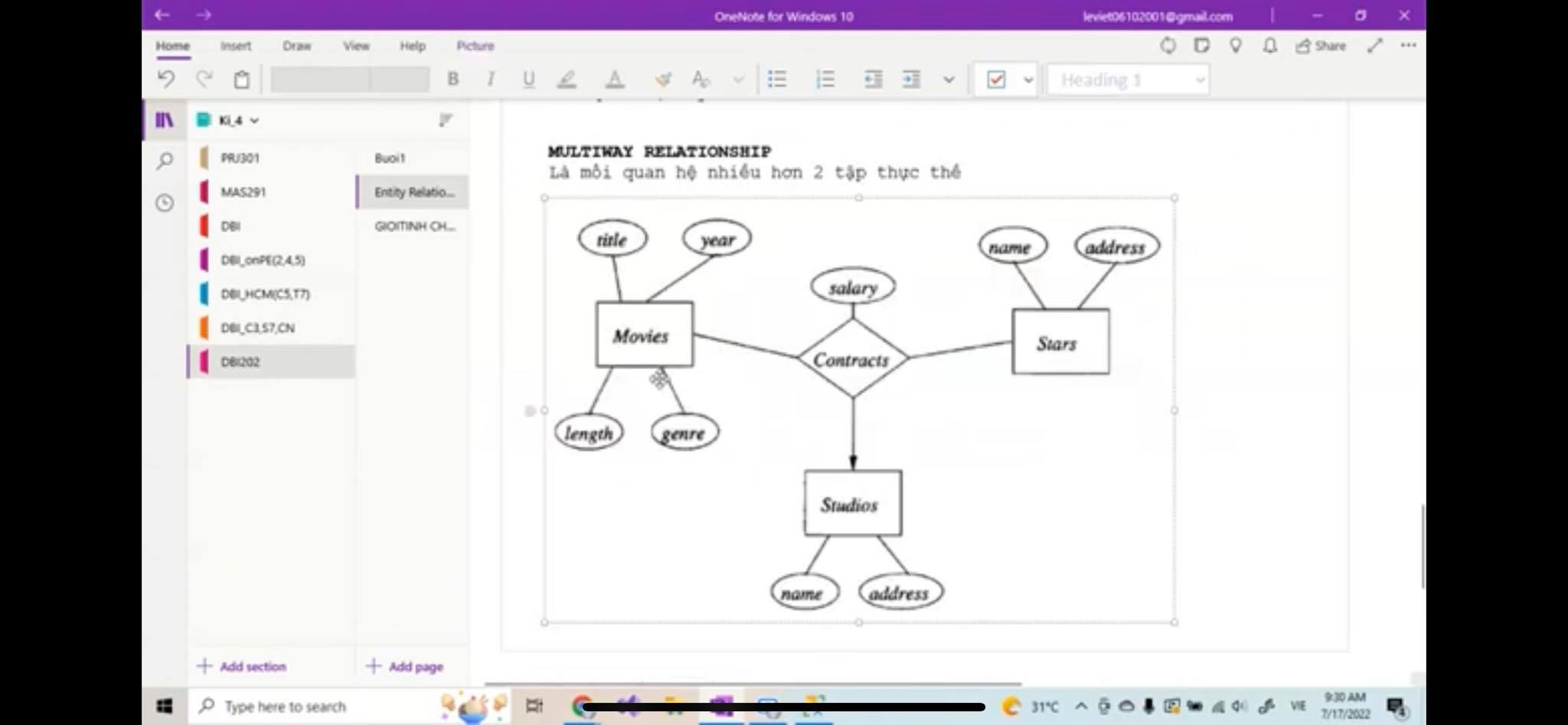


**MULTIWAY RELATIONSHIP:**

- Là mối quan hệ nhiều hơn 2 tập thực thể .

- Muốn xác định một thực thể là bên một hay bên nhiều thì phải dựa vào các thực thể còn lại.

VD:

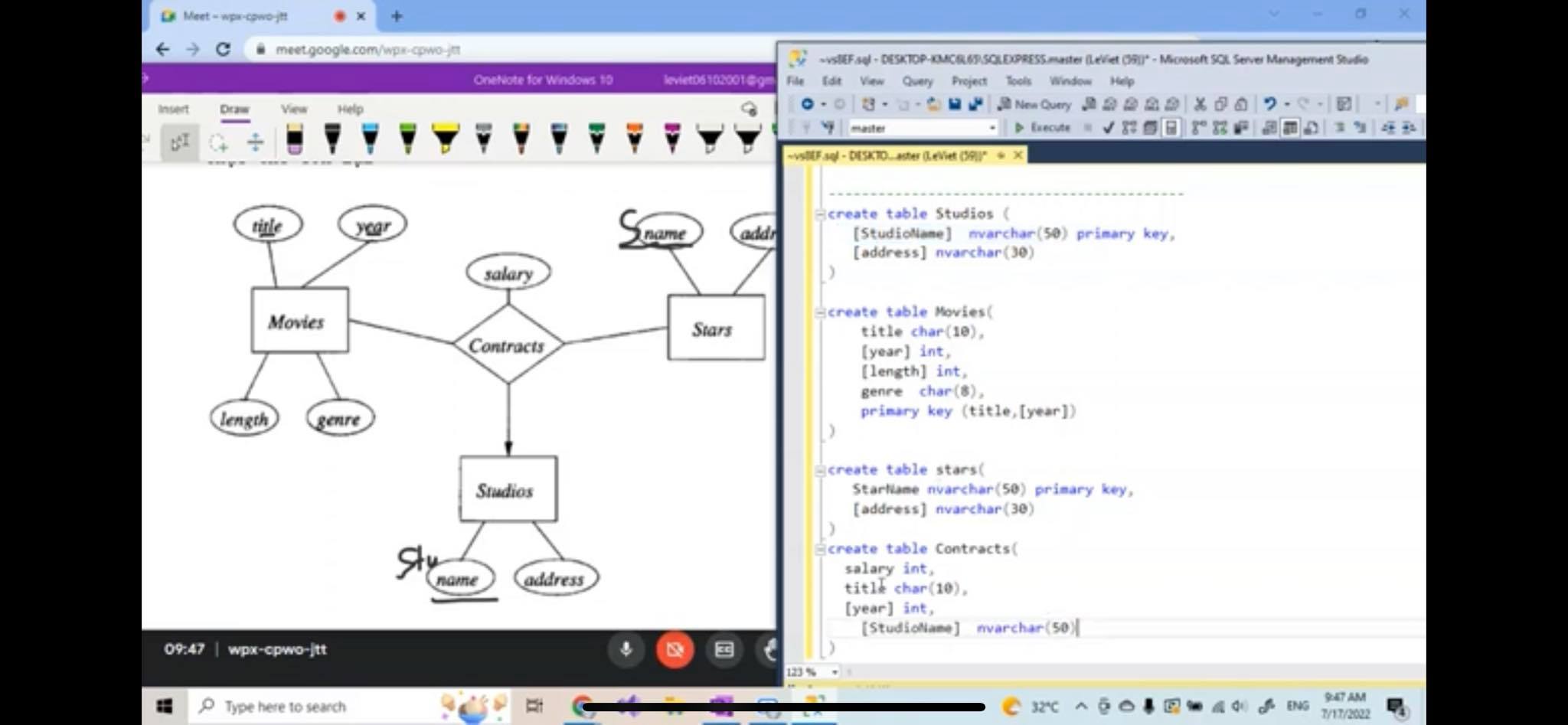


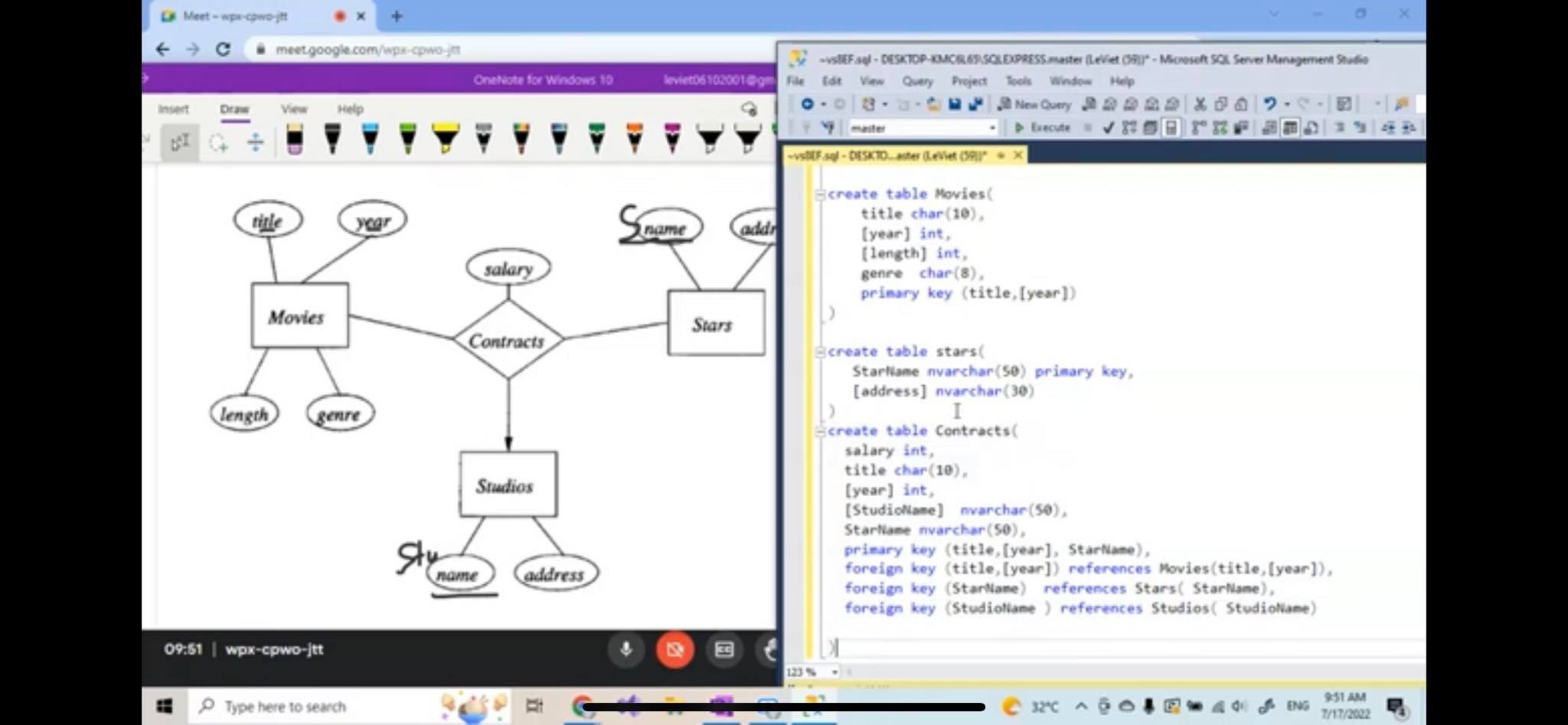
**\* Đối với MULTIWAY RELATIONSHIP:**

+) Tạo bảng quan hệ bao gồm :

- Thuộc tính của nó và thuộc tính khóa của các bảng tham gia vào mối quan hệ đấy.

- Khóa chính của bảng quan hệ sẽ bao gồm khóa chính của các bảng bên phía nhiều.

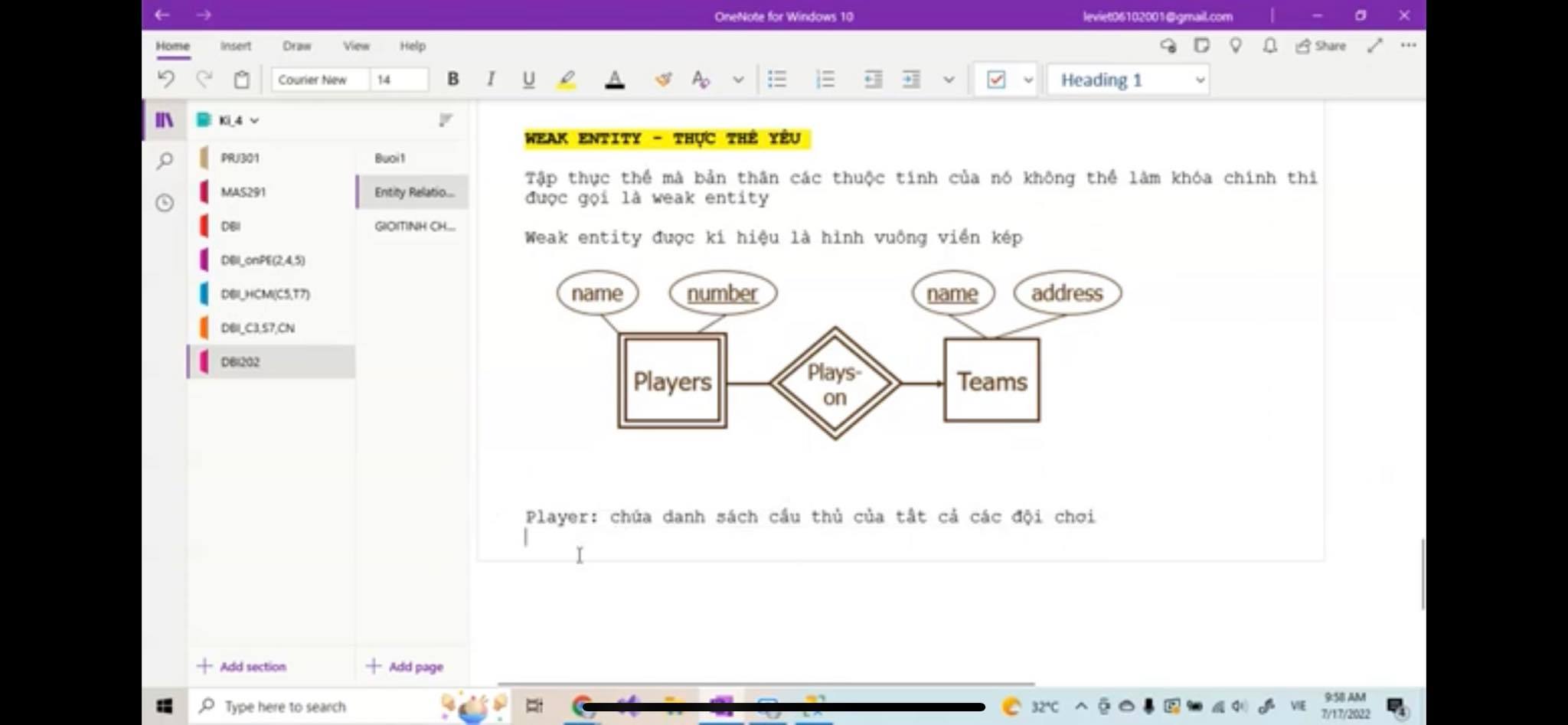




**WEAK ENTITY –THỰC THỂ YẾU :**

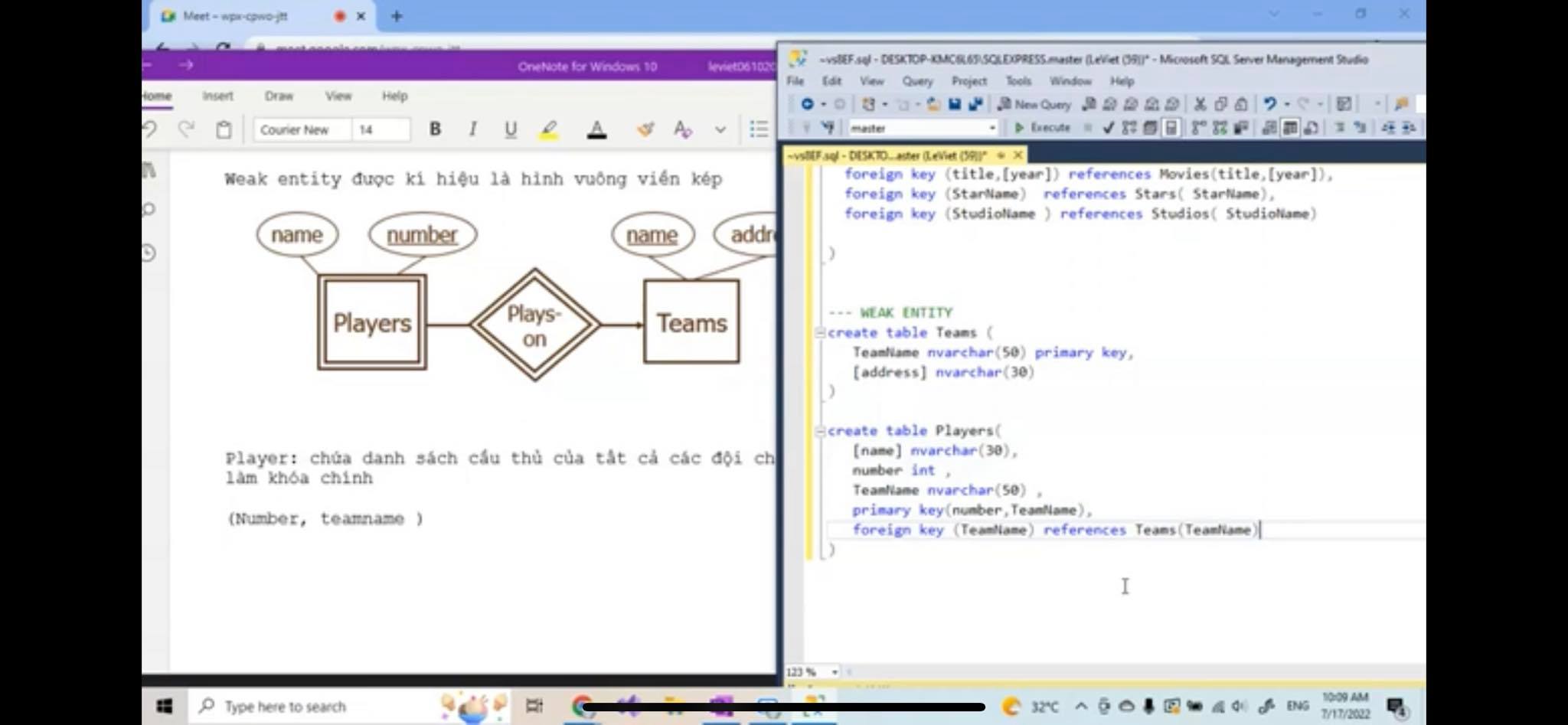
- Tập thực thể mà bản thân các thuộc tính của nó không thể làm khóa chính thì được gọi là **Weak Entity**.

- **Weak Entity** được kí hiệu là hình vuông viền kép.



- Player: chứa danh sách các cầu thủ của tất cả các đội chơi, lấy số áo (number) làm khóa chính.

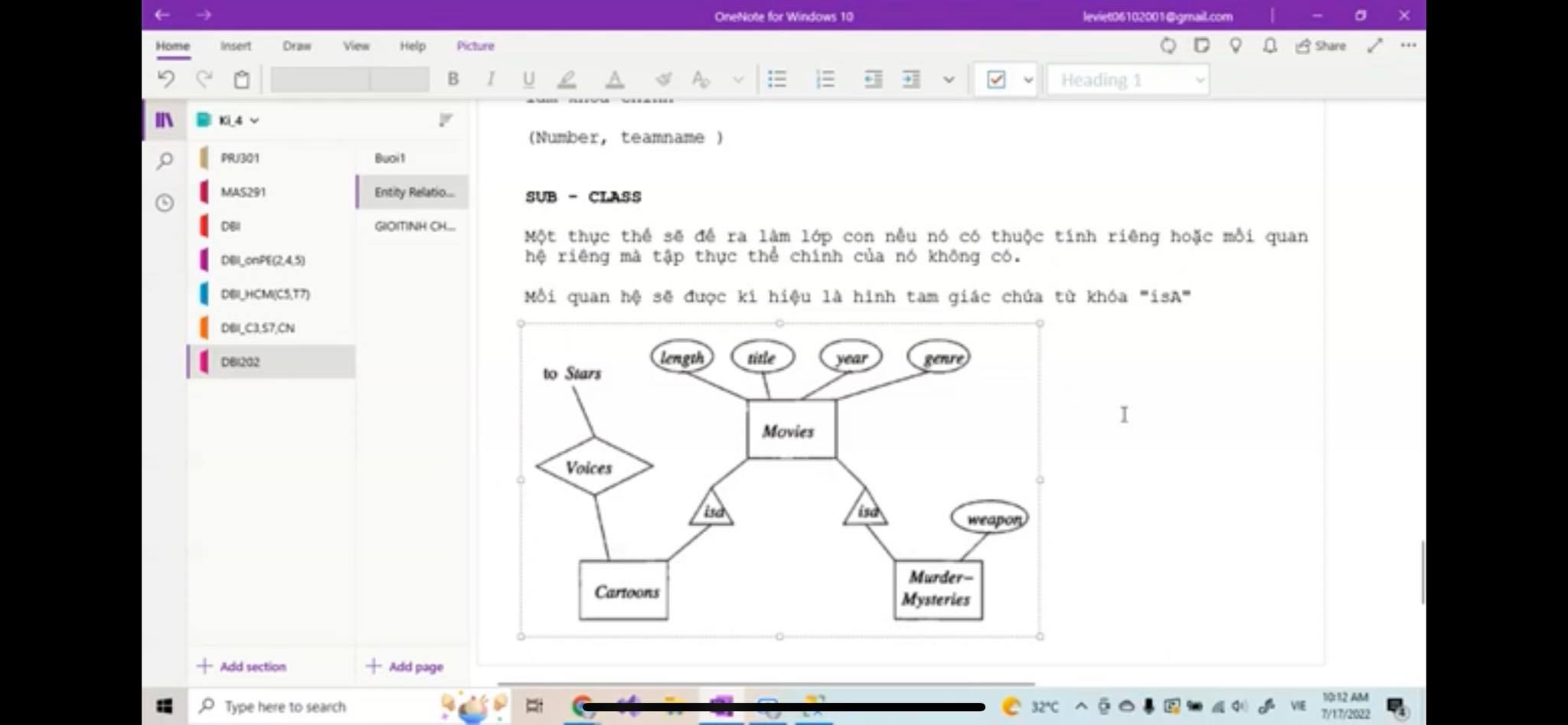
-Number, Teamname làm khóa chính.



**SUB – CLASS:**

- Một thực thể sẽ đề ra lớp con nếu nó có thuộc tính riêng hoặc mối quan hệ riêng mà tập thực thể chính của nó không có .

- Mối quan hệ sẽ được kí hiệu là hình tam giác chứa từ khóa **“isA”.**



Munder-Mysteries: có thuộc tính riêng là **weapon và có 5 thuộc tính** (4 cái kế thừa của lớp cha (length, title,year,genre) và 1 thuộc tính weapon riêng.)

Cartoons: có mối quan hệ riêng với **Voices** và **to Stars và có 4 thuộc tính** (kế thừa của lớp cha (length, title,year,genre)

**\* Đối với mối quan hệ sub – class:**

- Tạo bảng **cha** bao gồm các thuộc tính của nó, chứa thông tin của các bảng con.

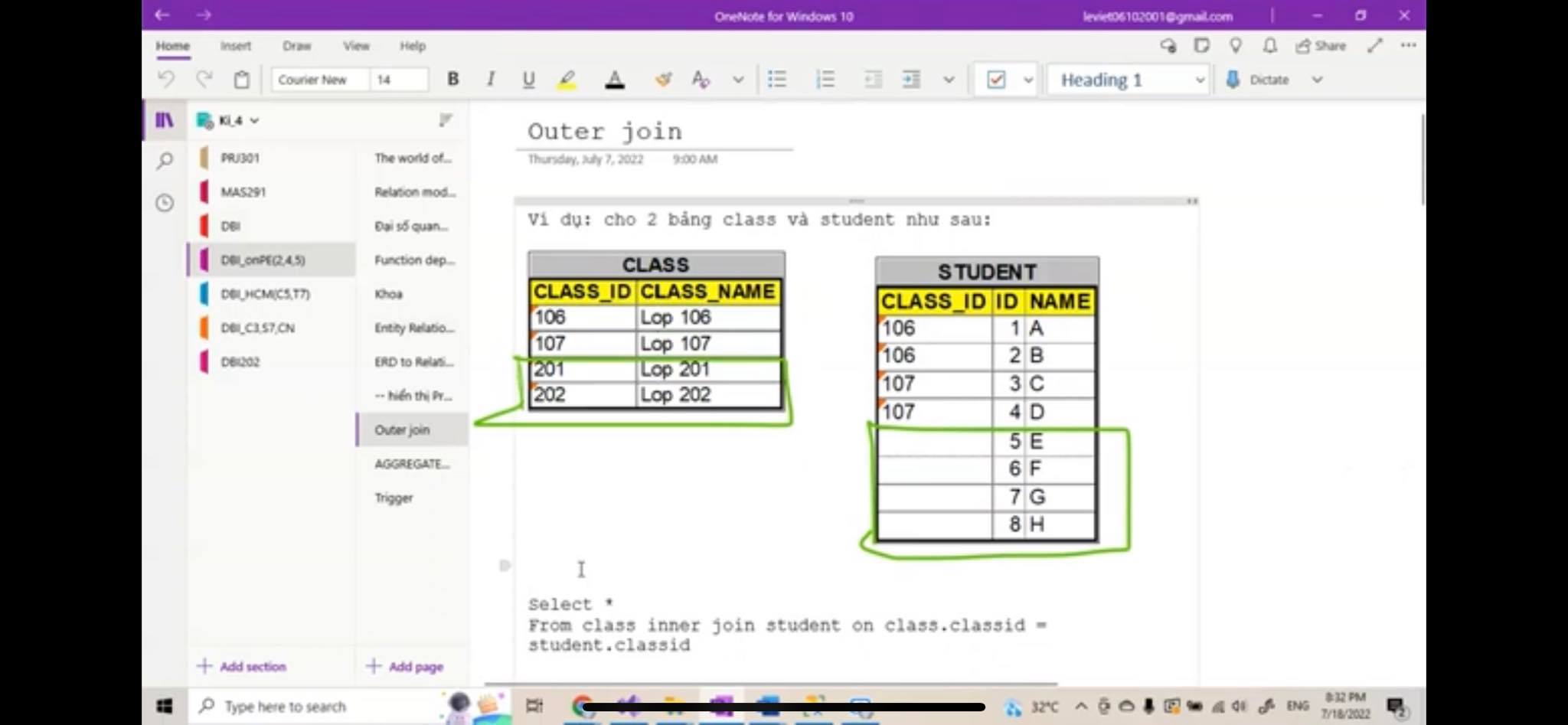
-Tạo bảng **con** bao gồm thuộc tính khóa của bảng cha và thuộc tính riêng nếu có .

Movies( length, title, year, genre) 🡪 chứa thông tin của tất cả các bộ phim bao gồm cartoons và munder-mysteries.

Munder-mysteries ( title, year, weapon) 🡪 Đánh dấu bộ phim nào là phim munder-mysteries

- Khóa ngoại (title, year) tham chiếu đến Movies(title, year) Cartoon (title, year).

**Outer Jion**:



B1: Nối từng dòng của bảng class và bảng student.

B2: Chỉ giữ lại những dòng thỏa mãn điều kiện join. (SinhVien.MaLop = Lop.MaLop)

- Outer join là phép kết cho phép giữ lại những dòng KHÔNG thỏa mãn điều kiện kết.

-Có 3 loại : left join , right join , full outer join.

- Cần chú ý đến bảng nào nằm bên trái và bên phải từ khóa JOIN.

|  |  |  |
| --- | --- | --- |
| Bảng nằm bên trái JOIN | JOIN | Bảng nằm bên phải JOIN |

**1.LEFT JOIN.**

Cú Pháp :

**Select** danh sách thuộc tính

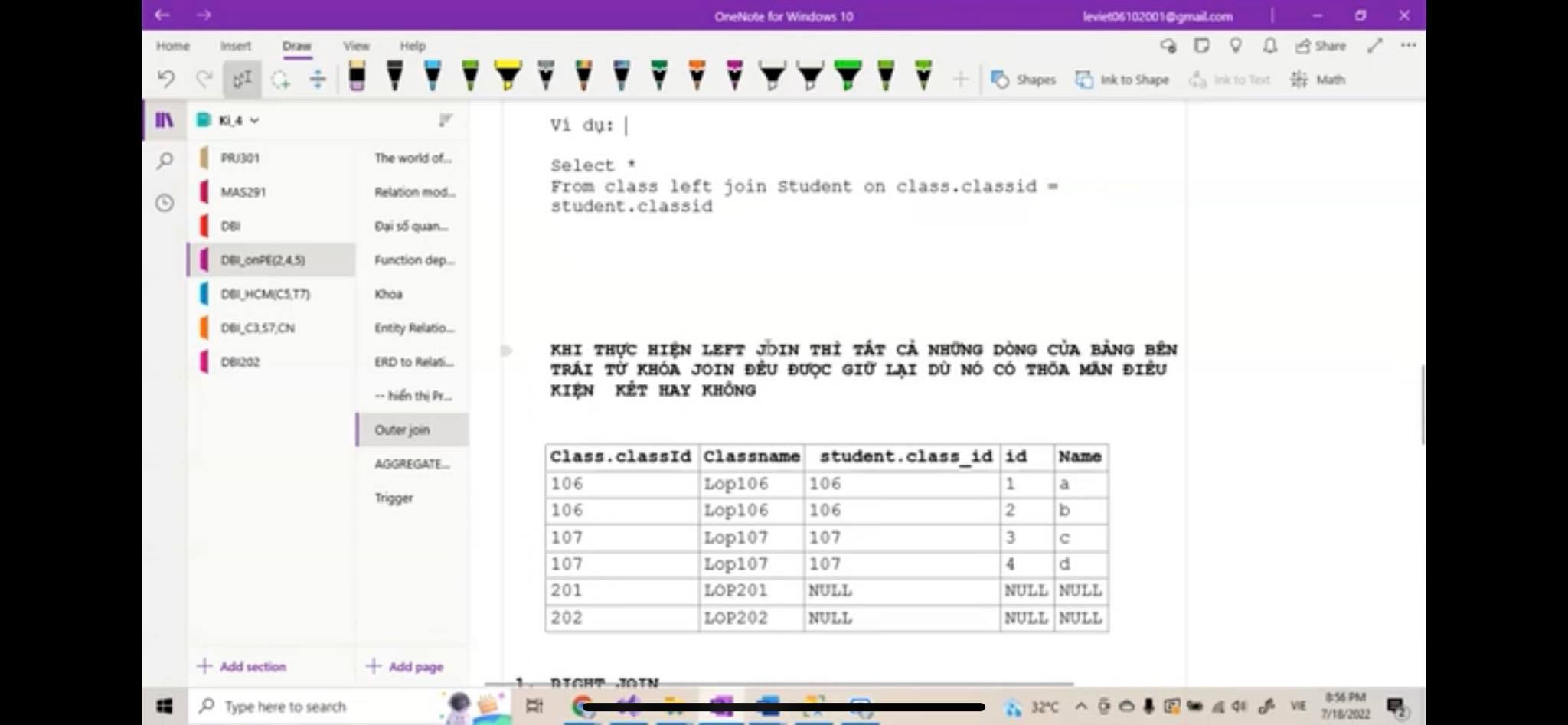
**From** A **left join** B **on**  điều kiện kết.

Ví Dụ:

Select \*

From class left join Student on class.classid = student.classid

**KHI THỰC HIỆN LEFT JOIN THÌ TẤT CẢ NHỮNG DÒNG CỦA BẢNG BÊN TRÁI TỪ KHÓA JOIN ĐỀU ĐƯỢC GIỮ LẠI DÙ NÓ CÓ THỎA MÃN ĐIỀU KIỆN KẾT HAY KHÔNG.**



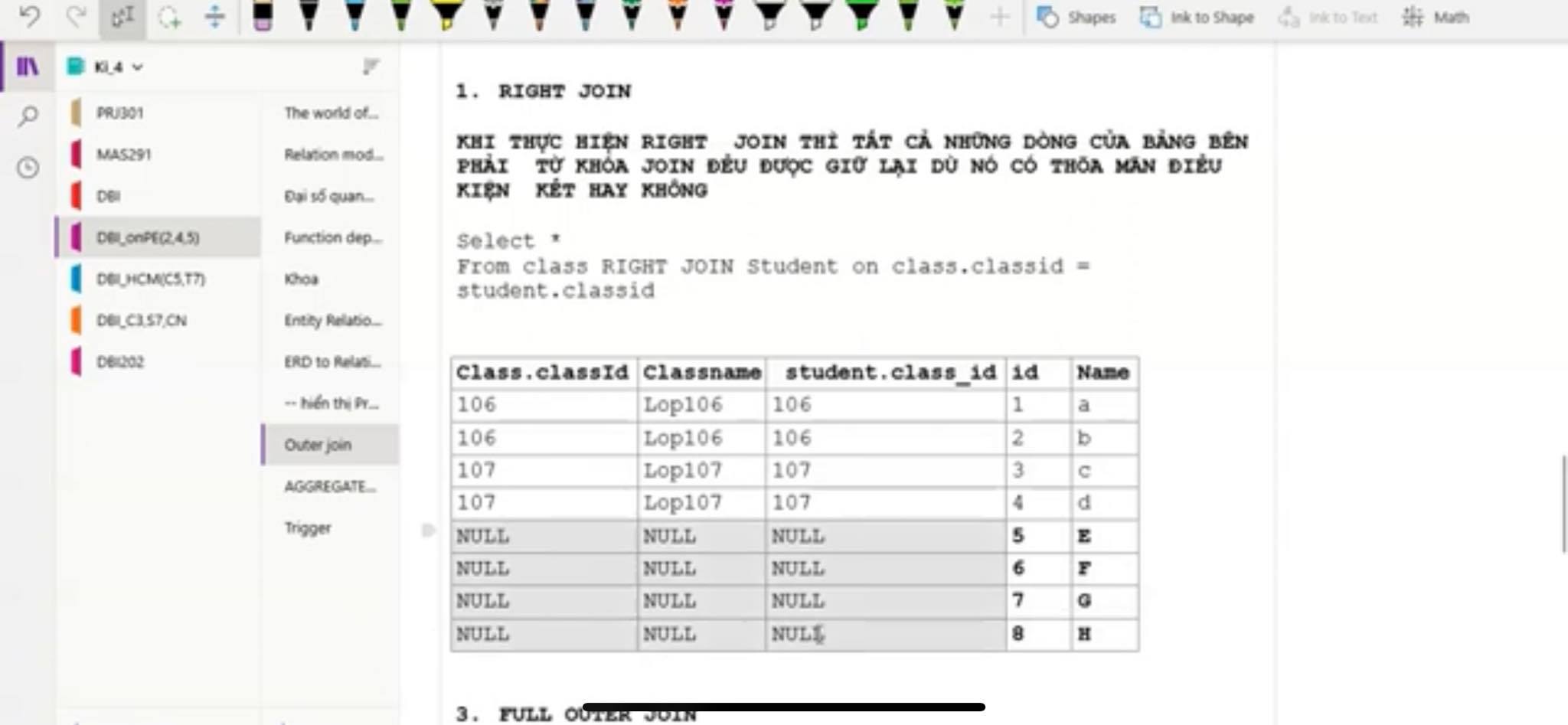
**2.RIGHT JOIN.**

Cú Pháp:

Select \*

From class RIGHT JOIN Student on class.classid = student.classid

**KHI THỰC HIỆN RIGHT JOIN THÌ TẤT CẢ NHỮNG DÒNG CỦA BẢNG BÊN PHẢI TỪ KHÓA JOIN ĐỀU ĐƯỢC GIỮ LẠI DÙ NÓ CÓ THỎA MÃN ĐIỀU KIỆN KẾT HAY KHÔNG.**



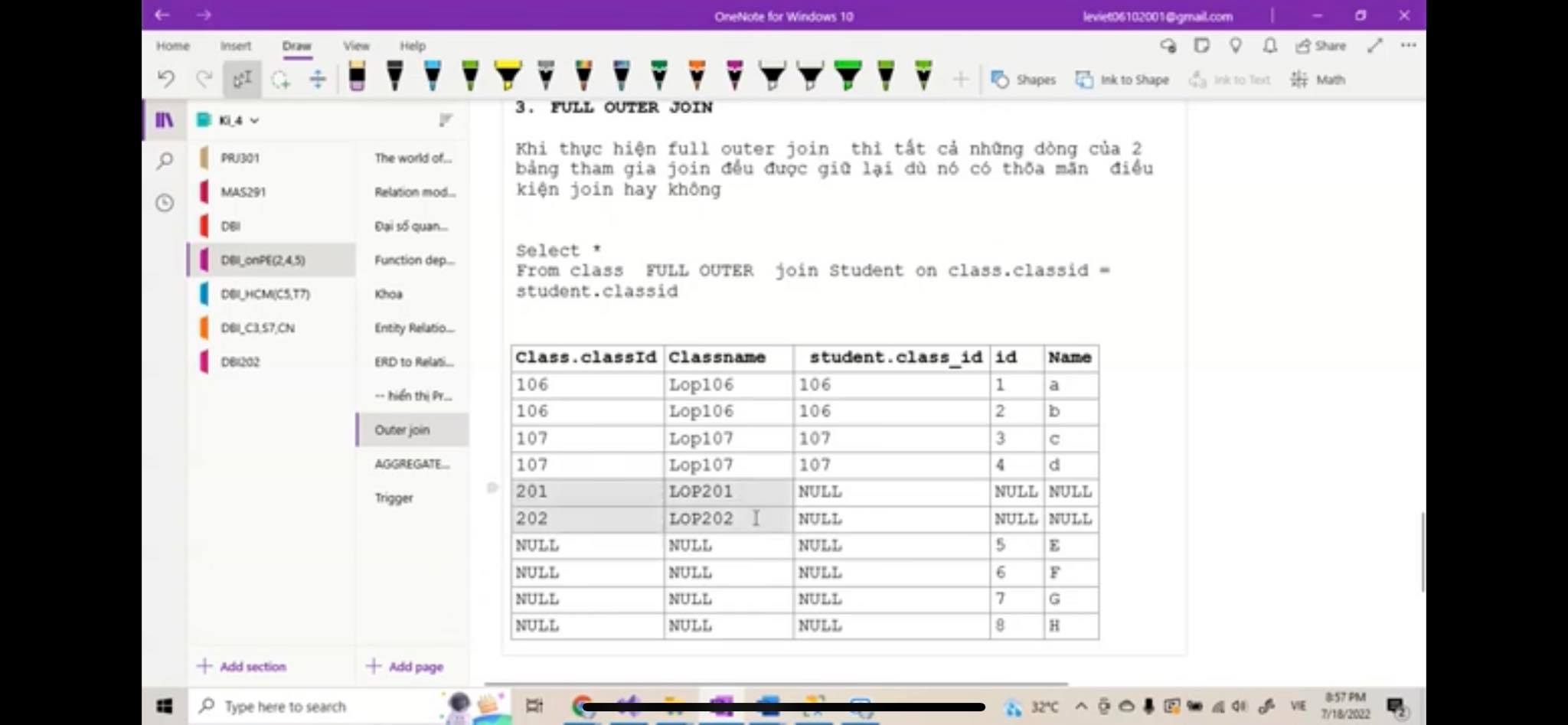
**3.FULL OUTER JOIN.**

Cú pháp:

Select \*

From class FULL OUTER join Student on class.classid = student.classid

- **Khi thực hiện full outer join thì tất cả những dòng của 2 bảng tham gia join đều được giữ lại dù nó có thỏa mãn điều kiện join hay không.**

****

**Triiger :**

- Triiger là các hàm, các lệnh đã được viết sẵn từ trước, được chạy khi sự kiện lien quan đến trigger đấy xảy ra .

- Các sự kiện bao gồm : insert, delete, update.

- Cú pháp:

**Create trigger** tên trigger

**On** tên bảng gắn vs sự kiện

**For** tên sự kiện ( insert, delete, update)

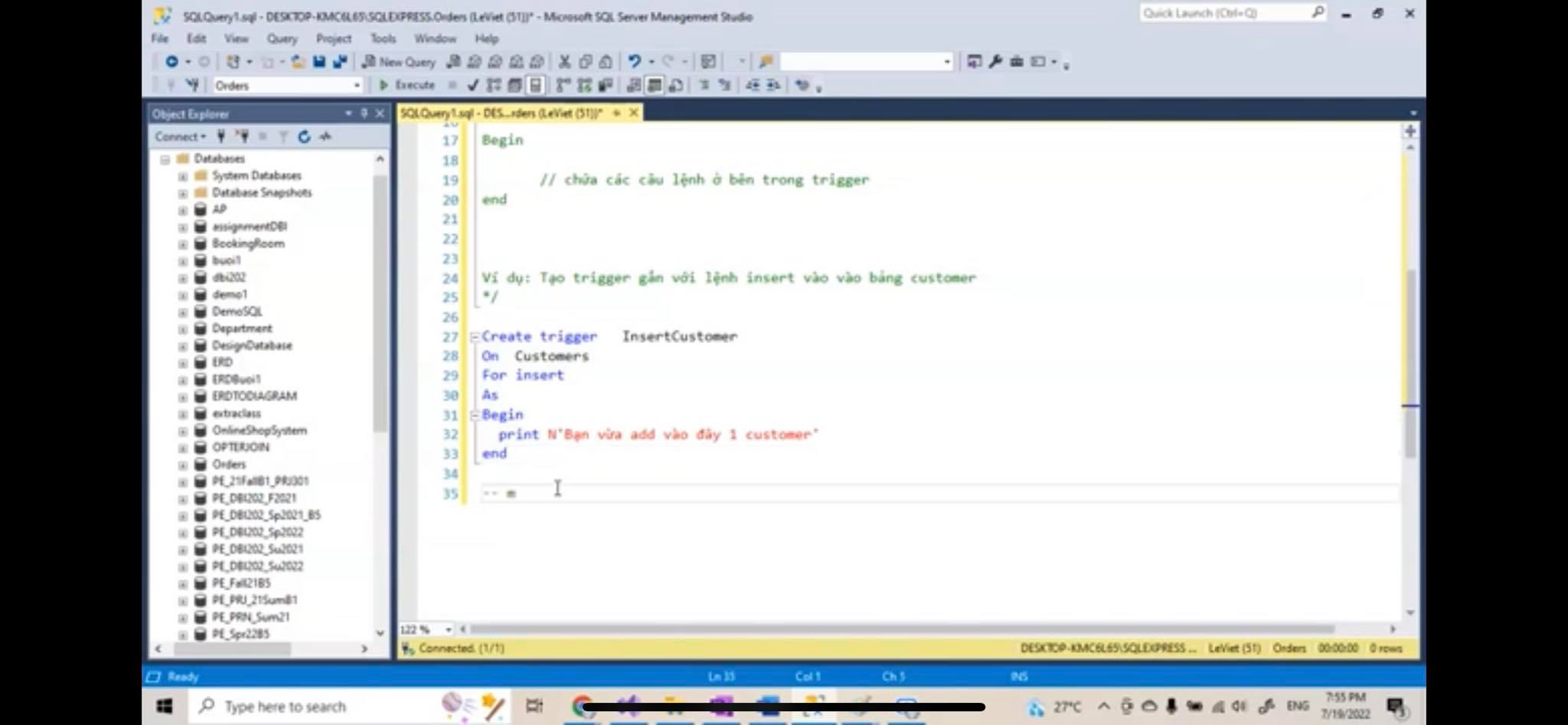
**As**

**Begin**

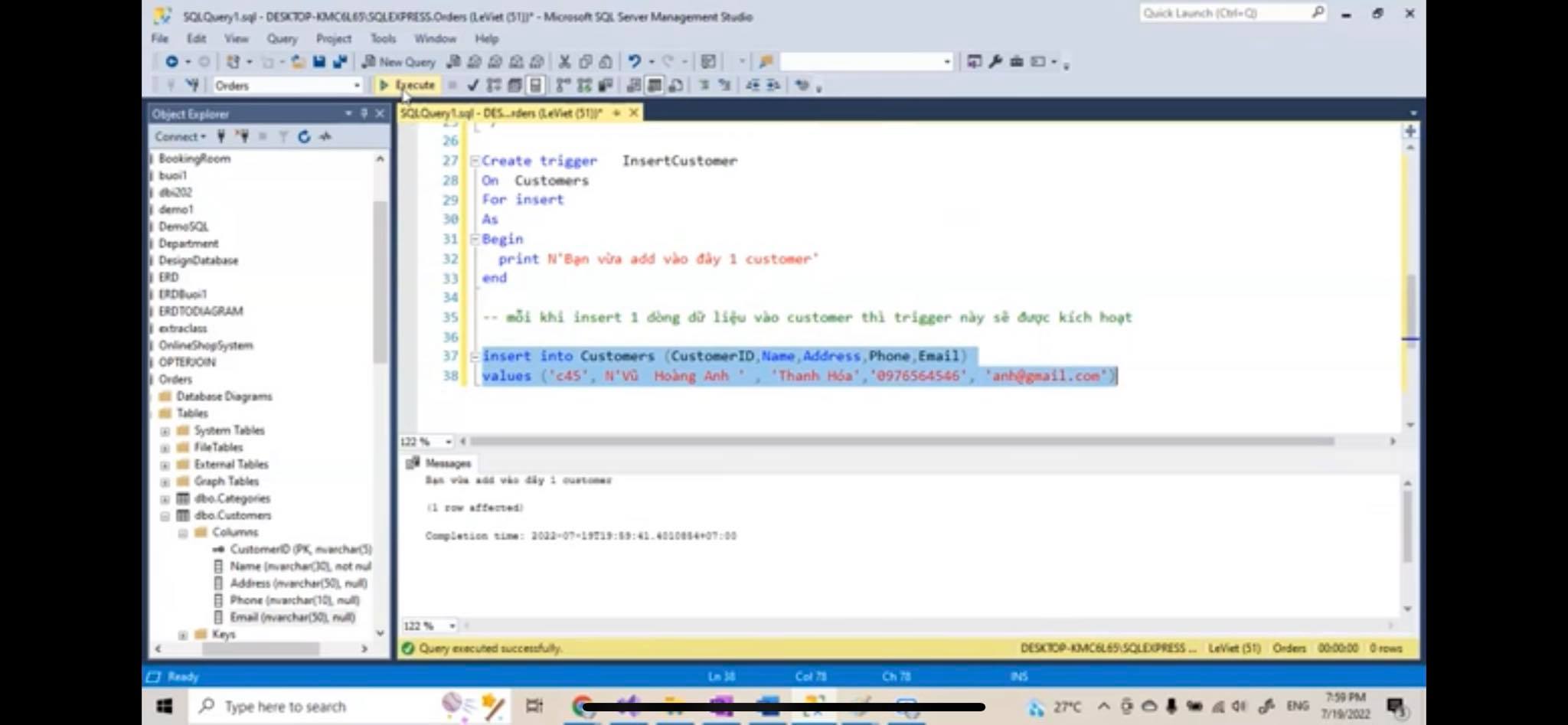
/// chứa các câu lệnh ở bên trong trigger

**End.**

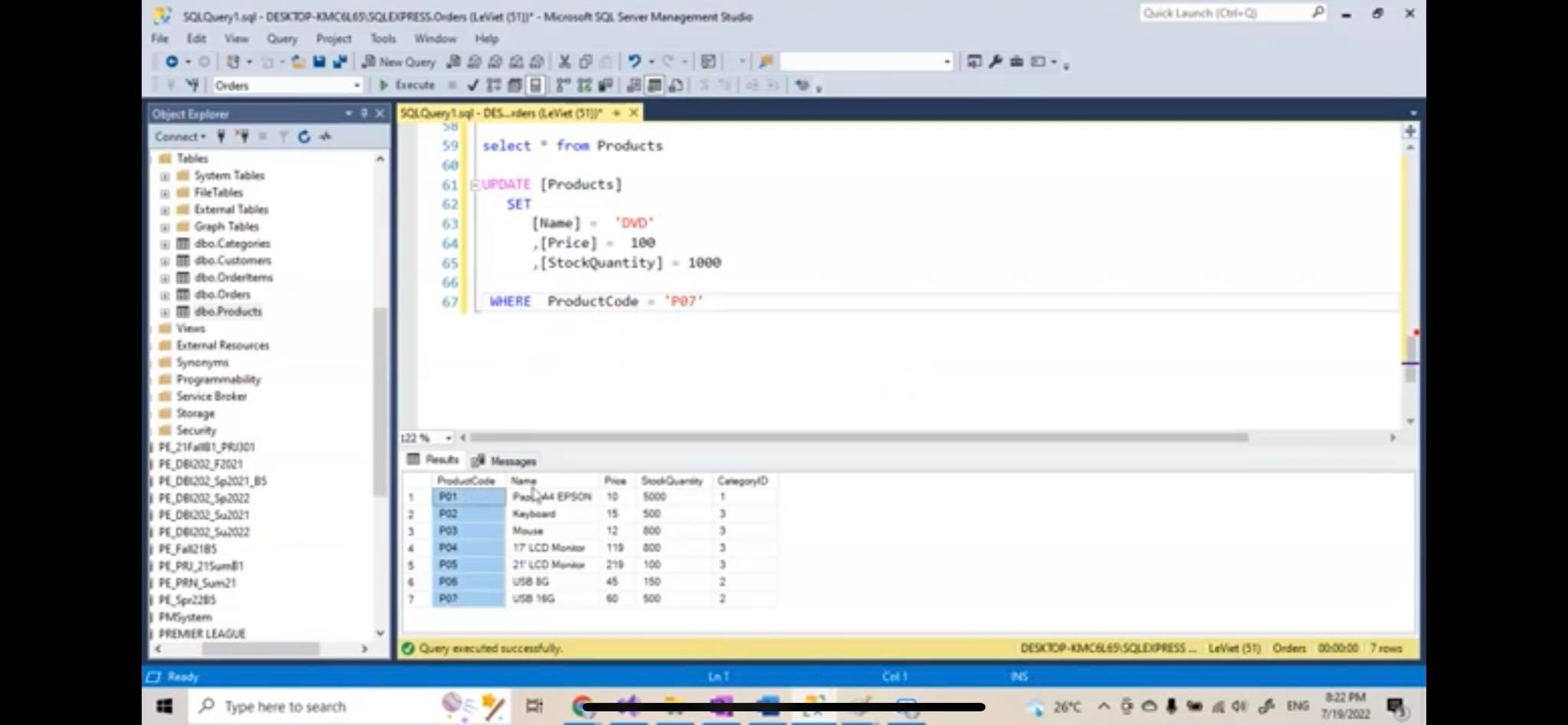
Ví dụ: Tạo trigger gắn với lệnh insert vào bảng Customer.

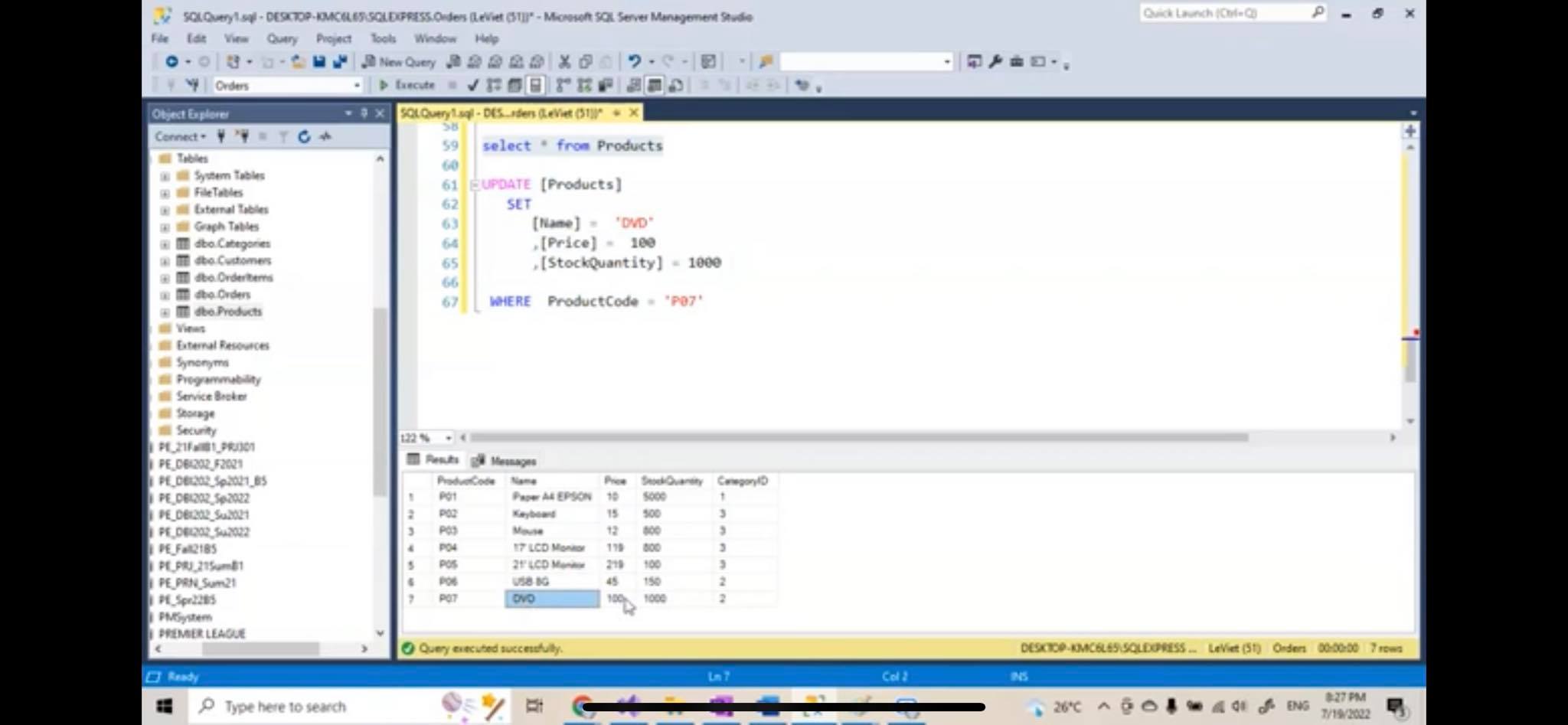


- Mỗi khi insert 1 dòng dữ liệu vào Customer thì trigger này sẽ được kích hoạt.



--UPDATE TRIGGER:





**NOTE:**

-Ứng với sự kiện là insert thì sẽ có bảng inserted được tự động tạo ra, bảng này chứa những dòng vừa được insert vào

- Ứng với lệnh delete thì sẽ có bảng deleted được tự động tạo ra, chứa những dòng mà mình vừa xóa đi .

- Ứng với lệnh update thì bảng inserted và deleted sẽ tự động tạo ra.

+ deleted: chứa dòng vừa bị xóa

+ inserted : chứa dòng vừa được thêm.

**Store procedure:**

- là các câu lệnh được viết sẵn , có thể gọi khi cần.

- Cú pháp:

Create procedure tên thủ tục VD: create procedure **proc\_Orders**

As

Begin

// chứa các câu lệnh

End.

Thực thi:

Execute … tên thủ tục …… VD: Execute **proc\_Orders**

Exec … tên thủ tục …… VD: Execute **proc\_Orders**

**\* Thủ tục tham số đầu ra:**

- Cần khai báo tham số đầu ra SAU KHI khai báo tham số đầu vào.

- Tạo thủ tục tính tổng của 2 số nguyên.

