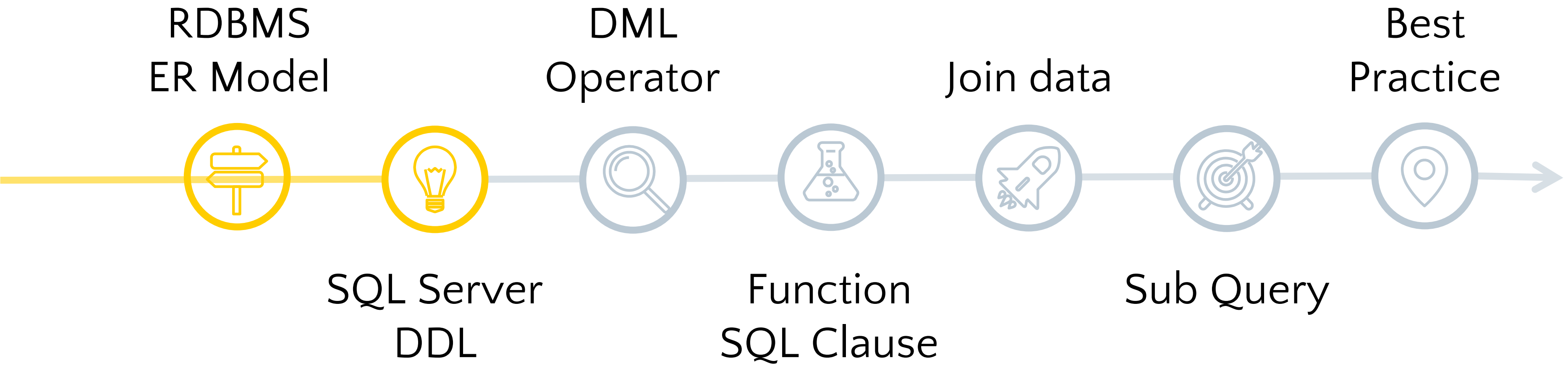
welcome back

SQL Essentials

Roadmap


RDBMS
ER Model
SQL Server
DDL
DML
Operator
Function
SQL Clause
Join data
Sub Query
Best
Practice

Previous lecture

- Database
- Relational database
- DBMS vs RDBMS
- Schema
- ER Model
- Entity
- Cardinality
- Relationships
- Convert ER model to schema

What we will explore today?

SQL Server

- Structure query language
- take a look on UI
- First command
- SQL Components

Data Definition Language

- Fun with database
- SQL data type
- Table In database
- Meaningful data with constraints
- SQL Process

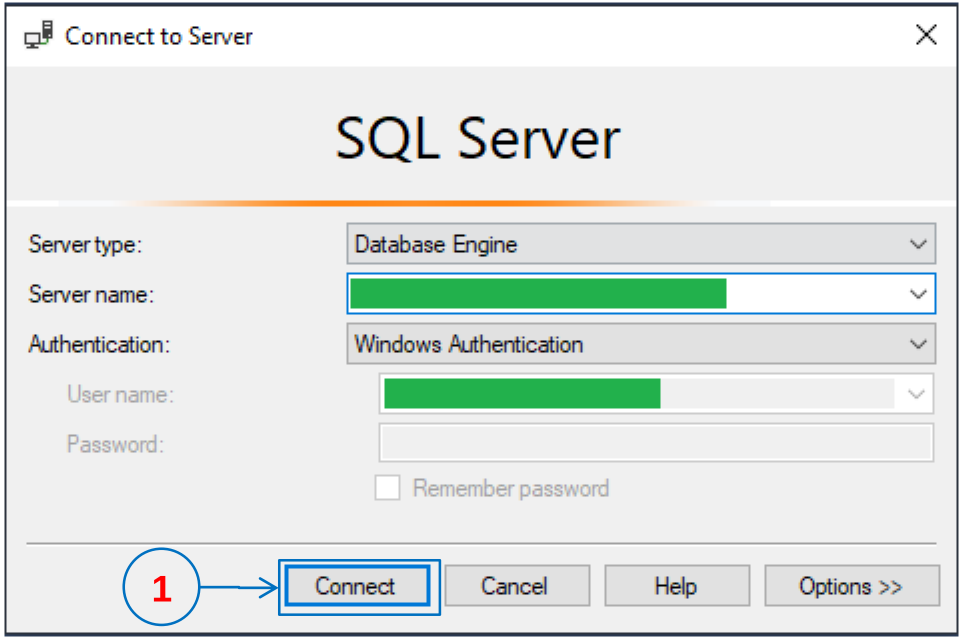
SQL Server

- Microsoft SQL Server is a relational database management system developed by Microsoft

What is SQL

- SQL stands for Structured Query Language. It's use to **store, manipulate, retrive data**

Hello SQL Server


Connect to Server

SQL Server

Server type: Database Engine

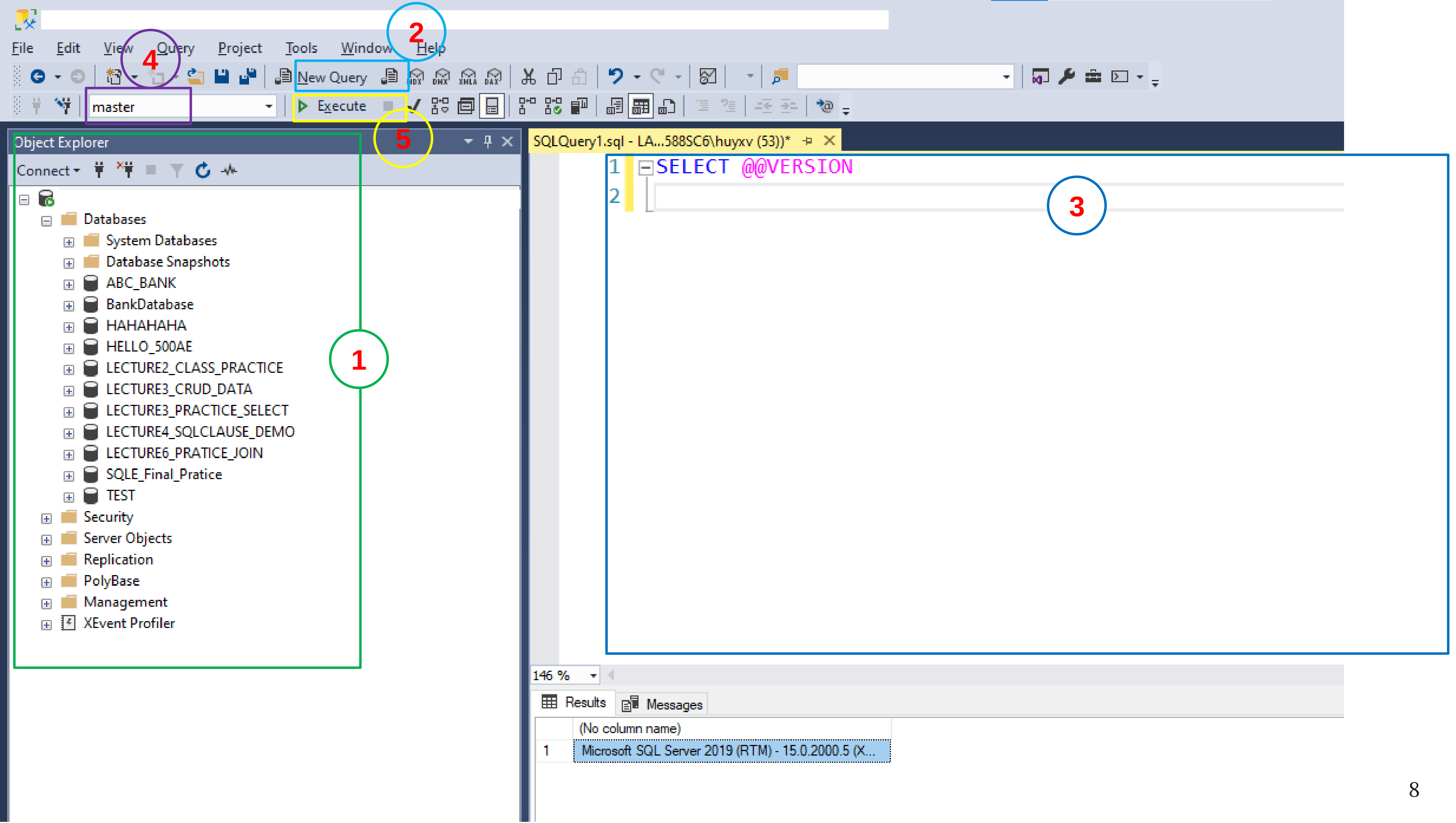
Server name: [redacted]

Authentication: Windows Authentication

User name: [redacted]

Password:

Remember password



Databases
- System Databases
- Database Snapshots
- ABC_BANK
- BankDatabase
- HAHAHAHA
- HELLO_500AE
- LECTURE2_CLASS_PRACTICE
- LECTURE3_CRUD_DATA
- LECTURE3_PRACTICE_SELECT
- LECTURE4_SQLCLAUSE_DEMO
- LECTURE6_PRATICE_JOIN
- SQLE_Final_Pratice
- TEST

Security
Server Objects
Replication
PolyBase
Management
XEvent Profiler

```sql
SELECT @@VERSION
```

	(No column name)
1	Microsoft SQL Server 2019 (RTM) - 15.0.2000.5 (X...

2 way to execute the SQL

- Press F5
- Execute button

Noted: If you do not select the code to run then SQL will run all the code in the editor by default.

SQL Main Component

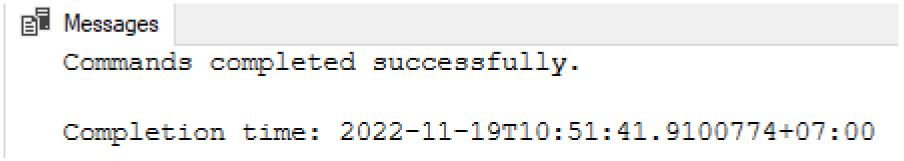
DDL	DESCRIPTION
DDL(Data Definition Language)	used to define **data structures**: database, table, column, relationships, contraints etc..
DML(Data Manipulation Language)	used for **insert, delete, update** data in a database
DCL(Data Control Language)	used to **control access** to data stored in a database.

Data Definition Language (DDL)

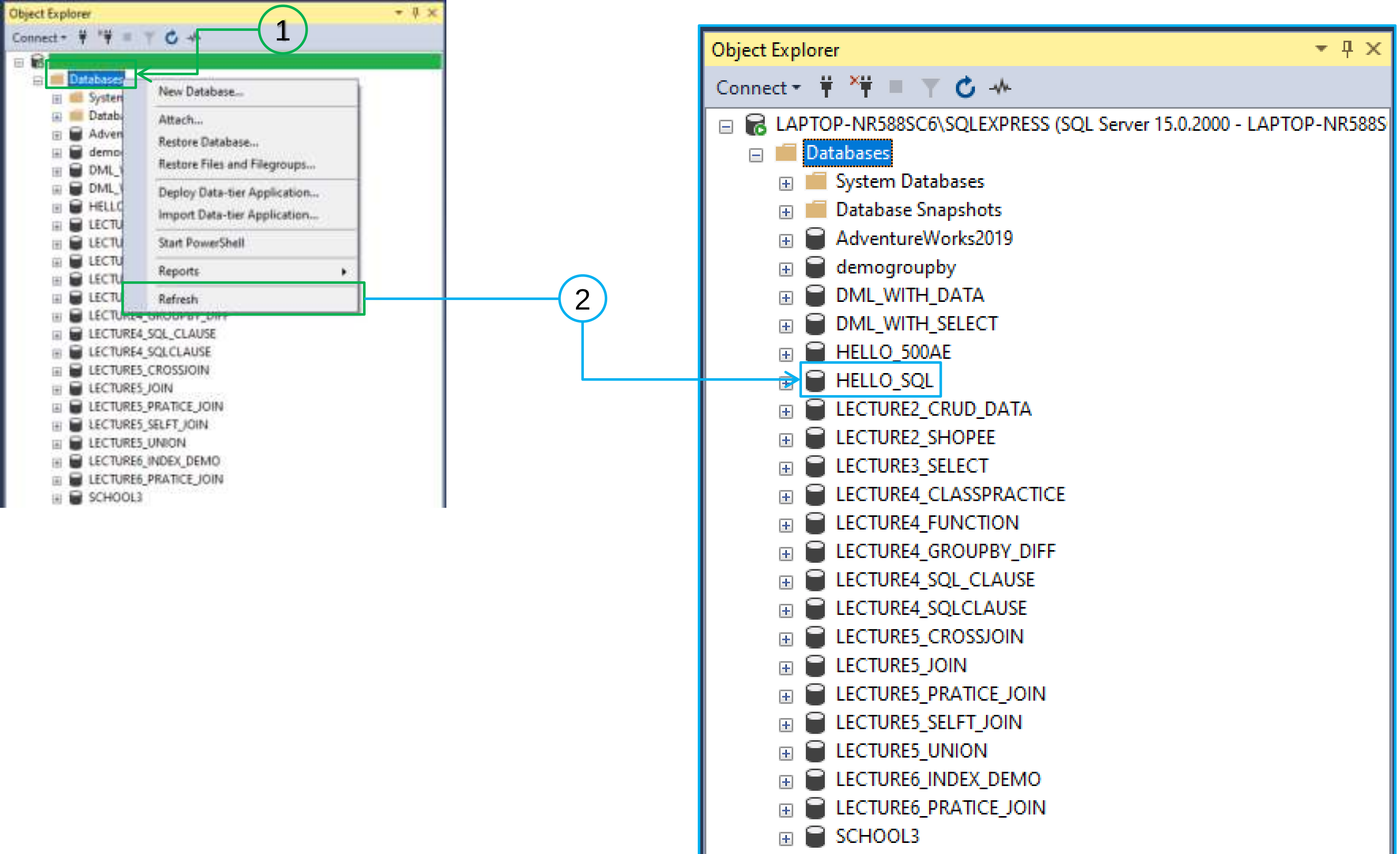
Create database syntax

```
CREATE DATABASE data_base_name
```

```
EX1: CREATE DATABASE HELLO_SQL
```



```
Commands completed successfully.

Completion time: 2022-11-19T10:51:41.9100774+07:00
```



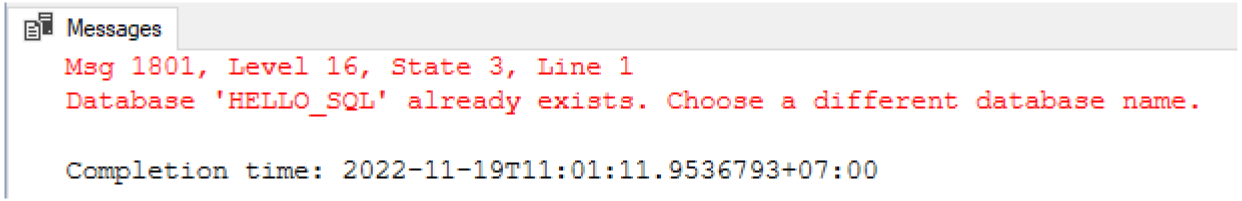
Object Explorer

Connect

LAPTOP-NR588SC6\SQLEXPRESS (SQL Server 15.0.2000 - LAPTOP-NR588S

- Databases
  - System Databases
  - Database Snapshots
  - AdventureWorks2019
  - demogroupby
  - DML_WITH_DATA
  - DML_WITH_SELECT
  - HELLO_500AE
  - HELLO_SQL
  - LECTURE2_CRUD_DATA
  - LECTURE2_SHOPEE
  - LECTURE3_SELECT
  - LECTURE4_CLASSPRACTICE
  - LECTURE4_FUNCTION
  - LECTURE4_GROUPBY_DIFF
  - LECTURE4_SQL_CLAUSE
  - LECTURE4_SQLCLAUSE
  - LECTURE5_CROSSJOIN
  - LECTURE5_JOIN
  - LECTURE5_PRATICE_JOIN
  - LECTURE5_SELFT_JOIN
  - LECTURE5_UNION
  - LECTURE6_INDEX_DEMO
  - LECTURE6_PRATICE_JOIN
  - SCHOOL3

Run the command again

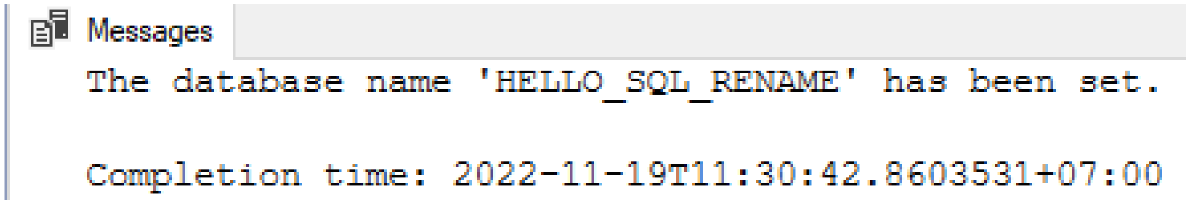


```
Msg 1801, Level 16, State 3, Line 1
Database 'HELLO_SQL' already exists. Choose a different database name.

Completion time: 2022-11-19T11:01:11.9536793+07:00
```

Modify database

```
ALTER DATABASE old_name MODIFY NAME = new_name
```

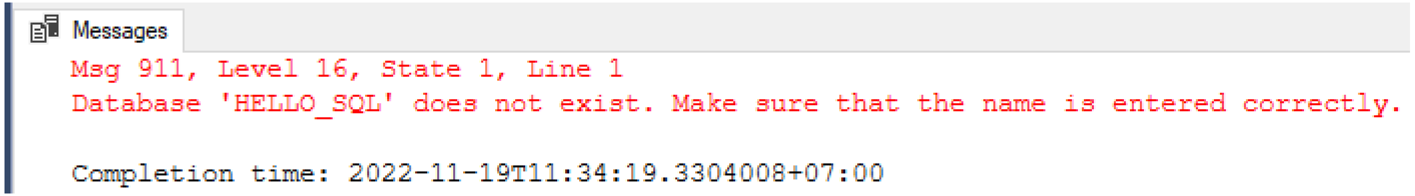
```
EX: ALTER DATABASE HELLO_SQL MODIFY NAME = HELLO_SQL_RENAME
```



```
The database name 'HELLO_SQL_RENAME' has been set.

Completion time: 2022-11-19T11:30:42.8603531+07:00
```

Refresh database to see the result

- Can we just run the modify command again?

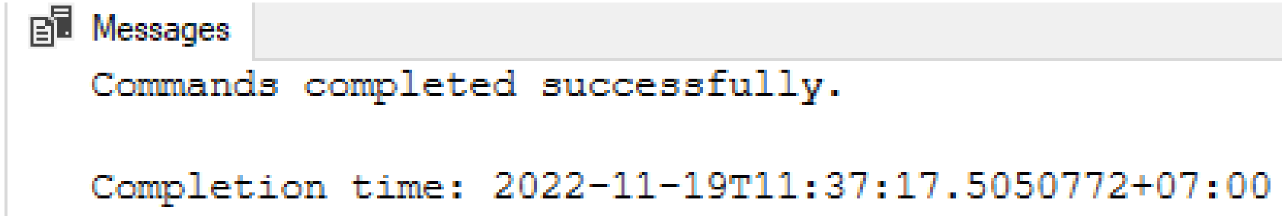


```
Msg 911, Level 16, State 1, Line 1
Database 'HELLO_SQL' does not exist. Make sure that the name is entered correctly.

Completion time: 2022-11-19T11:34:19.3304008+07:00
```

Drop database

```
DROP DATABASE data_base_name
```

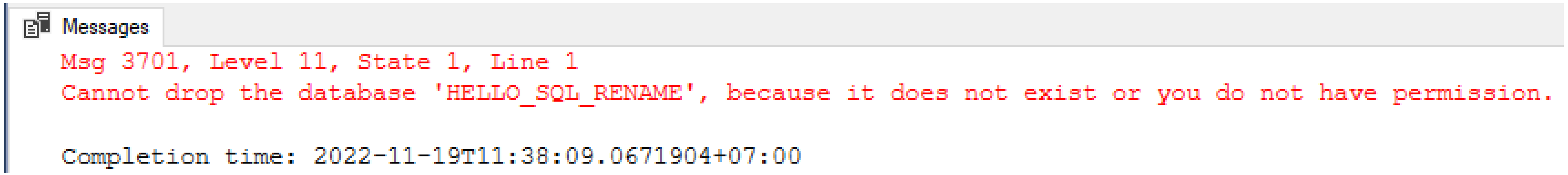
```
EX: DROP DATABASE HELLO_SQL_RENAME
```



```
Commands completed successfully.

Completion time: 2022-11-19T11:37:17.5050772+07:00
```

Refresh database to see the result

- Can we just run the drop command again?



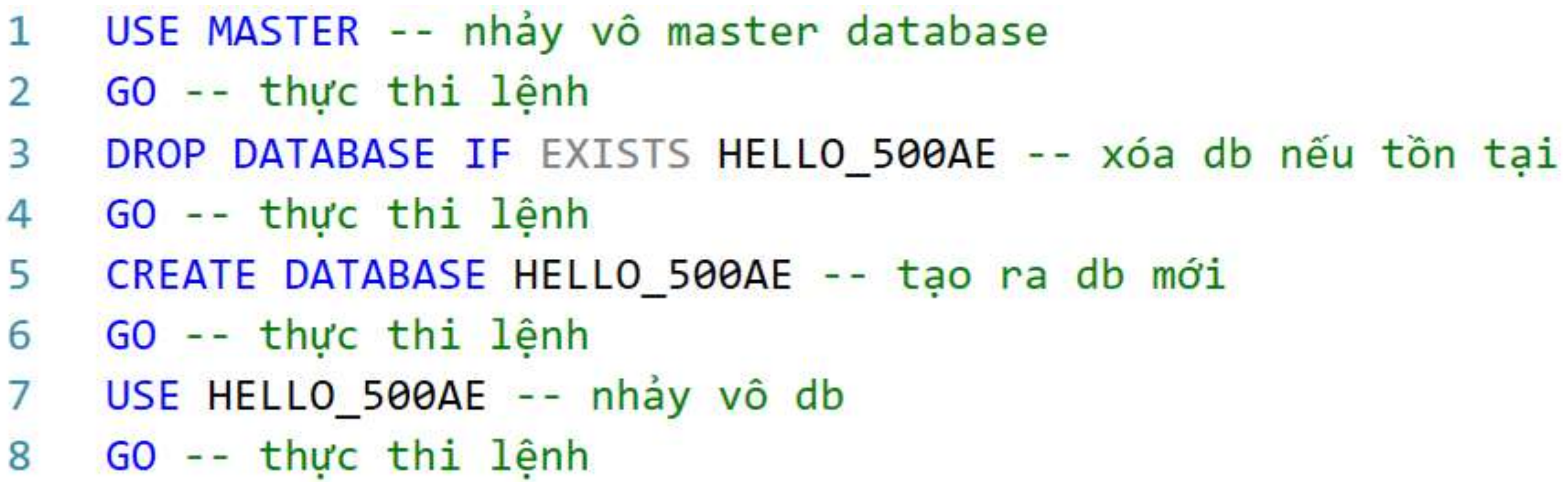
```
Msg 3701, Level 11, State 1, Line 1
Cannot drop the database 'HELLO_SQL_RENAME', because it does not exist or you do not have permission.

Completion time: 2022-11-19T11:38:09.0671904+07:00
```

Before delete

- Stand on other database
- Close all the connection to the target delete database

DROP then CREATE

open file “lecture2-drop-then-create.sql”



```sql
USE MASTER -- nhảy vô master database
GO -- thực thi lệnh
DROP DATABASE IF EXISTS HELLO_500AE -- xóa db nếu tồn tại
GO -- thực thi lệnh
CREATE DATABASE HELLO_500AE -- tạo ra db mới
GO -- thực thi lệnh
USE HELLO_500AE -- nhảy vô db
GO -- thực thi lệnh
```

SQL data type

SQL NULL Values

- A field with a NULL value is a field with no value.
- If a field in a table is optional, it is possible to insert a new record or update a record without adding a value to this field. Then, the field will be saved with a NULL value.

Check NULL

- It is not possible to test for NULL values with comparison operators, such as =, <, or <>.
- We will have to use the IS NULL and IS NOT NULL operators instead.

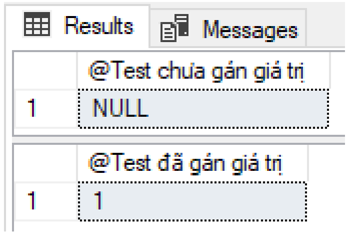
Exact numbers

DATA TYPE	SIZE	From	To
BIT	1 Bit	0	1
TINYINT	1 Byte	0	255
SMALLINT	2 Byte	-32768(2^15)	32767(2^15 -1)
INT	4 Bytes	- 2,147,483,648(-2^31)	+2,147,483,648(2^31 -1)
BIGINT	8 Bytes	-2^63	2^63 -1

-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807

Learn to THINK not just remember

```sql
DECLARE @Test int;
SELECT @Test AS '@Test chưa gán giá trị';
SET @Test = 1;
SELECT @Test AS '@Test đã gán giá trị';
```



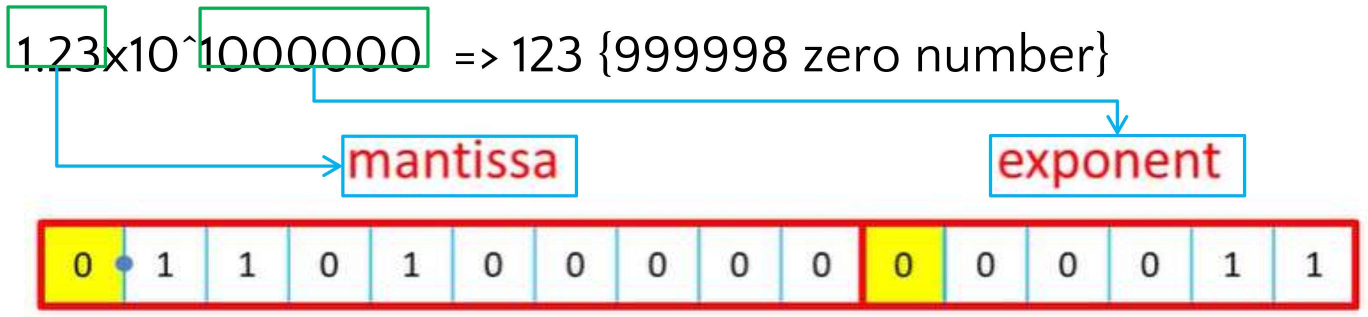
```
   @Test chưa gán giá trị
1  NULL

   @Test đã gán giá trị
1  1
```

Float & Real

DATA TYPE	SIZE	RANGE OF VALUE
Float[(n)]	Depend on n (default 53)	- 1.79E+308 to -2.23E-308, 0 and 2.23E-308 to 1.79E+308
Real	4 bytes	- 3.40E + 38 to -1.18E - 38, 0 and 1.18E - 38 to 3.40E + 38

n	Precision	Storage
1-24	7 digits	4 Bytes
25-53	15 digits	8 Bytes

- 1.23x10^1000000 => 123 {999998 zero number}



- n is the number of bits that store the **mantissa**

Decimal & numeric

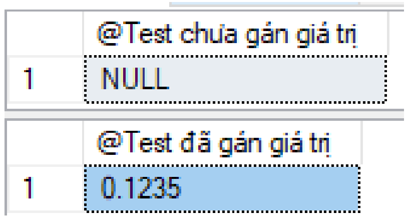
DATA TYPE	SIZE	RANGE OF VALUE
Decimal(p [,s])	5 -> 17 bytes	from - 10^38 +1 through 10^38 – 1
Numeric(p [,s])		

Precision	Bytes
1 - 9	5
10-19	9
20-28	13
29-38	17

- p (precision) total count number of number left & right decimal-point
- s (scale) the count of number after decimal-point
- Ex: 123.003 => p = 6 & s = 3

Try the code with decimal

```
DECLARE @Test decimal(4, 4);
SELECT @Test AS '@Test chưa gán giá trị';
SET @Test = 0.12345;
SELECT @Test AS '@Test đã gán giá trị';
```



```
   @Test chưa gán giá trị
1  NULL

   @Test đã gán giá trị
1  0.1235
```

Money & smallmoney

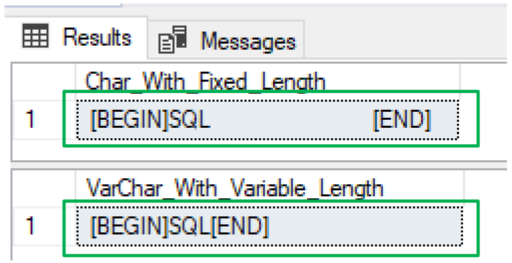
DATA TYPE	SIZE	Range Value
Smallmoney	4 bytes	- 214,748.3648 To 214,748.3647
Money	8 bytes	- 922,337,203,685,477.5808 To 922,337,203,685,477.5807

Character strings

DATA TYPE	LENGTH	DESCRIPTION
CHAR(N)	(1≤ n ≤ 8000) characters	Fixed-length
VARCHAR(N)	(1≤ n ≤ 8000) characters	Variable-length
VARCHAR(MAX)	2,147,483,647 characters	Variable-length
TEXT	2,147,483,647 characters	Variable-length

Fixed vs Variable length

```sql
DECLARE @myChar char(30) --create variable name @myChar
DECLARE @myVarchar varchar(30) --variable length @myVarchar
SET @myChar = 'SQL'
SET @myVarchar = 'SQL'
SELECT '[BEGIN]' + @myChar + '[END]' AS Char_With_Fixed_Length
SELECT '[BEGIN]' + @myVarchar + '[END]' AS VarChar_With_Variable_Length
```



	Char_With_Fixed_Length
1	[BEGIN]SQL [END]

	VarChar_With_Variable_Length
1	[BEGIN]SQL[END]

VARCHAR(MAX) vs TEXT

- TEXT alway store in blob(Binary large object)
- VARCHAR(max) will attempt to store the data directly in the row unless it exceeds the 8k limitation and at that point it stores it in a blob

Unicode Character Strings

DATA TYPE	LENGTH	DESCRIPTION
NCHAR(N)	(1≤ n ≤ 4000) characters	Fixed-length
NVARCHAR(N)	(1≤ n ≤ 4000) characters	Variable-length
NVARCHAR(MAX)	1,073,741,823 characters	Variable-length
NTEXT	1,073,741,823 characters	Variable-length

Why unicode character?

- 1 byte only can representation 256 differrent value.

EX: we can not fit all the japanese symbal to the 256 difference value.

=> result: We have unicode which every chacracter take 2 bytes (65536 differrent value)

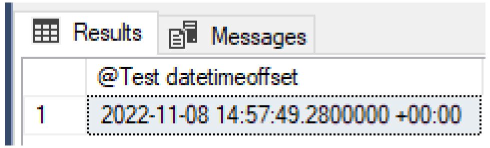
Binary strings

DATA TYPE	LENGTH	DESCRIPTION
Binary	(1≤ n ≤ 8000) BYTES	Fixed-length binary data
Varbinary	(1≤ n ≤ 8000) BYTES	Variable length binary data
Image	2,147,483,647 bytes	Variable length binary data

DATA TYPE	DESCRIPTION	Example
time	Store a time only to an accuracy of 100 nanoseconds	09:42:16.1420221
date	Store a date only. From January 1, 0001 to December 31, 9999	2008-01-15
smalldatetime	From January 1, 1900 to June 6, 2079 with an accuracy of 1 minute	2008-01-15 09:42:00
datetime	From January 1, 1753 to December 31, 9999 with an accuracy of 3.33 milliseconds	2008-01-15 09:42:16.142
datetime2	From January 1, 0001 to December 31, 9999 with an accuracy of 100 nanoseconds	2008-01-15 09:42:16.1420221
datetimeoffset	The same as datetime2 with the addition of a time zone offset	2008-01-15 09:42:16.1420221 +05:00

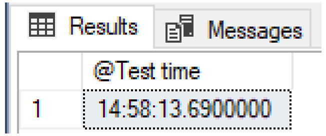
Try all the datetime datatype?

```
DECLARE @Test datetimeoffset;
SET @Test = GETDATE();
SELECT @Test AS '@Test datetimeoffset';
```



```
   @Test datetimeoffset
1  2022-11-08 14:57:49.2800000 +00:00
```

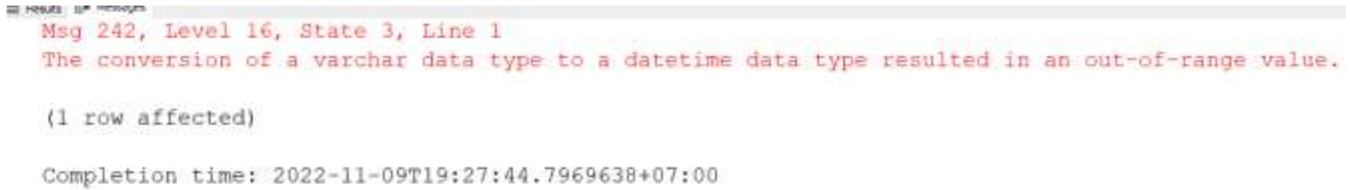
```
DECLARE @Test time;
SET @Test = GETDATE();
SELECT @Test AS '@Test time';
```



```
   @Test time
1  14:58:13.6900000
```

Datetime problem

```sql
DECLARE @Test datetime;
SET @Test = '25/12/2022';
SELECT @Test AS '@Test datetimeoffset';
```



```
Msg 242, Level 16, State 3, Line 1
The conversion of a varchar data type to a datetime data type resulted in an out-of-range value.

(1 row affected)

Completion time: 2022-11-09T19:27:44.7969638+07:00
```

The way to fix this issue

```
DECLARE @Test datetime;
SET @Test = '12/25/2022';
SELECT @Test AS '@Test datetimeoffset';
```

⇕

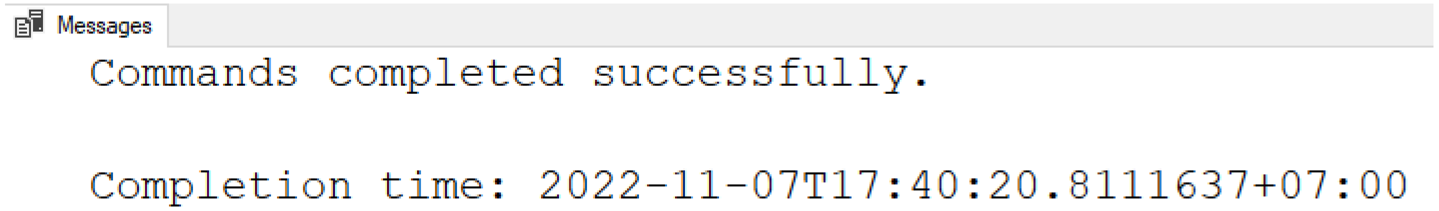
```
DECLARE @Test datetime;
SET @Test = CONVERT(datetime, '25/12/2022', 103);
SELECT @Test AS '@Test datetimeoffset';
```

Tables

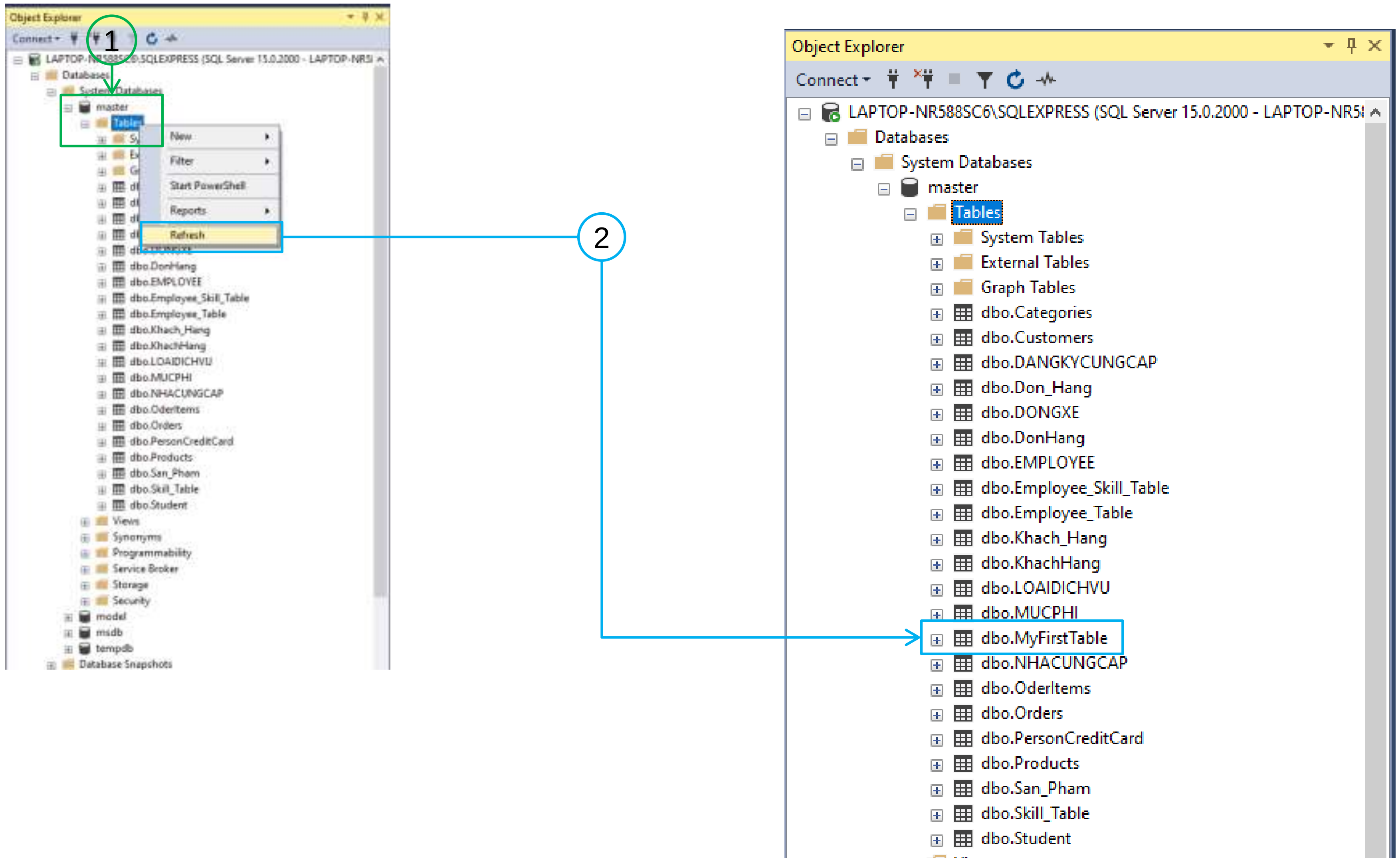
Create table

```
CREATE TABLE table_name (
    column1 datatype,
    column2 datatype,
    column3 datatype,
    ....
);
```

```
CREATE TABLE MyFirstTable(
    ID int,
    FullName nchar(50),
    Email varchar(20),
    PhoneNumber
varchar(10),
    DateOfBirth date,
    Wallet money
);
```



```
Commands completed successfully.

Completion time: 2022-11-07T17:40:20.8111637+07:00
```



```
Object Explorer
Connect ▾
LAPTOP-NR588SC6\SQLEXPRESS (SQL Server 15.0.2000 - LAPTOP-NR5
  Databases
    System Databases
      master
        Tables
          System Tables
          External Tables
          Graph Tables
          dbo.Categories
          dbo.Customers
          dbo.DANGKYCUNGCAP
          dbo.Don_Hang
          dbo.DONGXE
          dbo.DonHang
          dbo.EMPLOYEE
          dbo.Employee_Skill_Table
          dbo.Employee_Table
          dbo.Khach_Hang
          dbo.KhachHang
          dbo.LOAIDICHVU
          dbo.MUCPHI
          dbo.MyFirstTable
          dbo.NHACUNGCAP
          dbo.OderItems
          dbo.Orders
          dbo.PersonCreditCard
          dbo.Products
          dbo.San_Pham
          dbo.Skill_Table
          dbo.Student
```

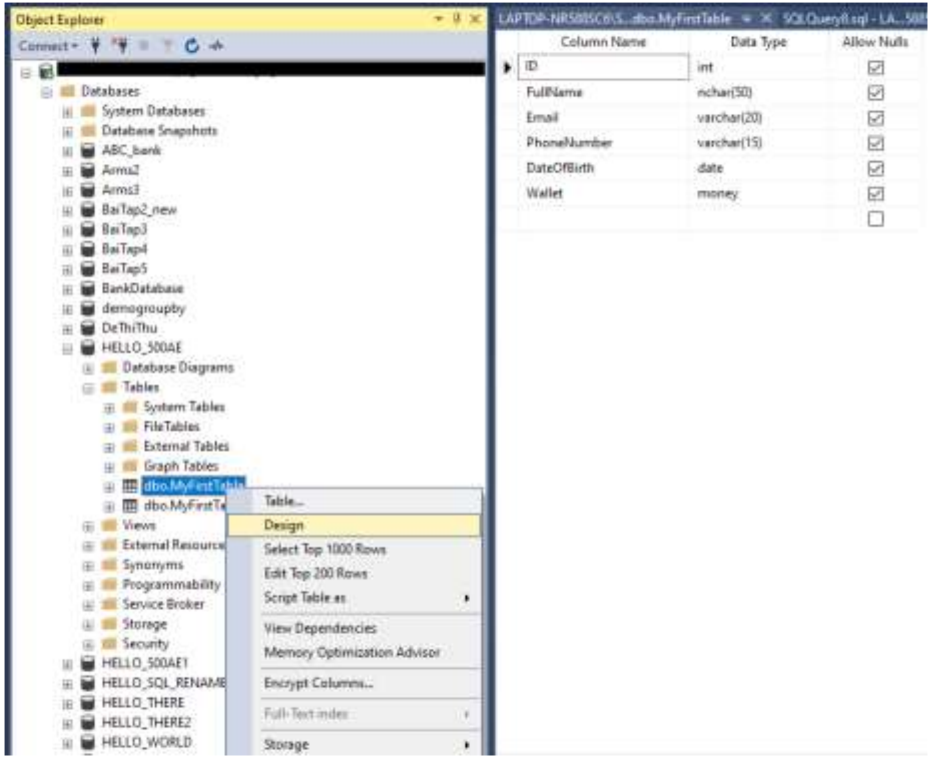
Target the correct database

```
USE HELLO_500AE;
GO;
CREATE TABLE MyFirstTable(
        ID int,
        FullName nchar(50),
        Email varchar(20),
        PhoneNumber varchar(10),
        DateOfBirth date,
        Wallet money
);
```

Modify column in table

```sql
-- add column in exists table
ALTER TABLE table_name
ADD column_name datatype;

-- drop column in exists table
ALTER TABLE table_name
DROP COLUMN column_name;

-- modified column in exists table
ALTER TABLE table_name
ALTER COLUMN column_name datatype;
```

View the design of table



Column Name	Data Type	Allow Nulls
ID	int	✓
FullName	nchar(50)	✓
Email	varchar(20)	✓
PhoneNumber	varchar(15)	✓
DateOfBirth	date	✓
Wallet	money	✓
		☐

Practice

- Add column City with nvarchar(100) into table MyFirstTable
- Modify column City datatype to nvarchar(500)
- Drop column City

Meaningful data

- SQL constraints are used to specify rules for the data in a table.
- This ensures the accuracy and reliability of the data in the table.
- If there is any violation between the constraint and the data action, the action is aborted.

Constraints

Constraint	Description
PRIMARY KEY	Uniquely identifies each row in a table
FOREIGN KEY	links between tables
UNIQUE	Ensures that all values in a column are different
DEFAULT	default value for a column if no value is specified
NOT NULL	Ensures that a column cannot have a NULL value
CHECK	Ensures that the values in a column satisfies a specific condition

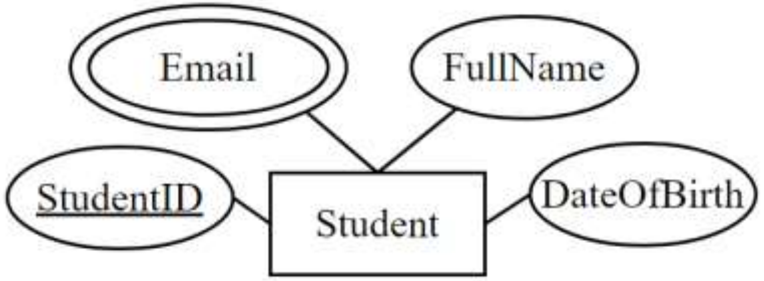
Primary Key

- A primary key is a feild or combination of fields which uniquely specify a row.
- Primary key values cannot be NULL.

Foreign key

- A FOREIGN KEY is a field (or collection of fields) in one table, that refers to the PRIMARY KEY in another table.
- The table with the foreign key is called the child table, and the table with the primary key is called the referenced or parent table

Foreign key



$$\begin{tikzcd} \text{Student}(\boxed{\underline{\text{StudentID}},}\ \text{FullName, DateOfBirth}) \\ \text{StudentEmail}(\boxed{\underline{\text{StudentID}}},\ \underline{\text{Email}}) \arrow[u] \end{tikzcd}$$

Example

StudentID	FullName	DateOfBirth
1	Snoop Dog	2/19/2000
2	The Rock	2/16/1999

StudentID	Email
1	snoop@high.com
1	snoop@low.com
2	power@man.com
2	supper@man.com

❌ NULL	wrongdata@man.com

Create table with constraints

```
CREATE TABLE MySecondTable(
        ID int PRIMARY KEY,
        FullName nchar(50) NOT NULL,
        Email varchar(20) UNIQUE,
        PhoneNumber varchar(10),
        DateOfBirth date DEFAULT GETDATE(),
        Wallet money CHECK (Wallet > 0)
);
```

Add constraints to table

```sql
CREATE TABLE MySecondTableWithAlter(
           ID int,
           FullName nchar(50),
           Email varchar(20),
           PhoneNumber varchar(10),
           DateOfBirth date,
           Wallet money
)
ALTER TABLE MySecondTableWithAlter ALTER COLUMN ID int NOT NULL;
ALTER TABLE MySecondTableWithAlter ADD PRIMARY KEY (ID);
ALTER TABLE MySecondTableWithAlter ALTER COLUMN FullName nchar(50) NOT NULL;
ALTER TABLE MySecondTableWithAlter ADD UNIQUE (Email);
ALTER TABLE MySecondTableWithAlter ADD CONSTRAINT df_DateOfBirth DEFAULT GETDATE() FOR DateOfBirth;
ALTER TABLE MySecondTableWithAlter ADD CHECK (Wallet> 0);
```

Add foregin key

```
ALTER TABLE StudentEmail
ADD FOREIGN KEY (StudentID)  REFERENCES Student(StudentID);
```

Student

StudentID	FullName	DateOfBirth
1	Snoop Dog	2/19/2000
2	The Rock	2/16/1999

StudentEmail

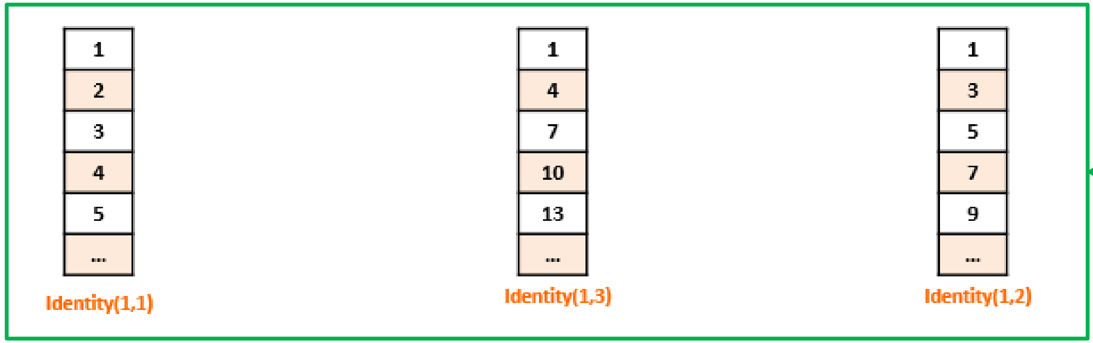
StudentID	Email
1	snoop@high.com
1	snoop@low.com
2	power@man.com
2	supper@man.com

Drop table

```
DROP TABLE table_name;

ex: DROP TABLE MySecondTableWithAlter;
```

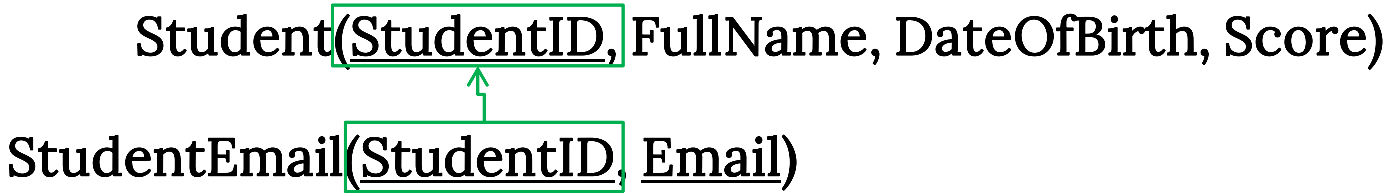
Auto generate Identity

```
CREATE TABLE StudentWithAutoIncreaseID(
        StudentID int PRIMARY KEY IDENTITY(1, 1),
        FullName nchar(50) NOT NULL,
        DateOfBirth date DEFAULT GETDATE()
)
```

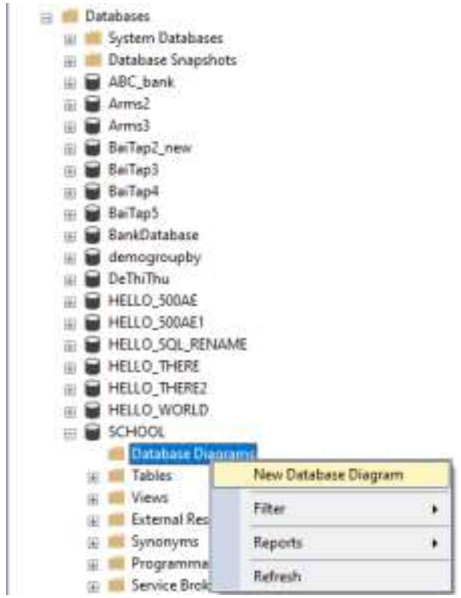


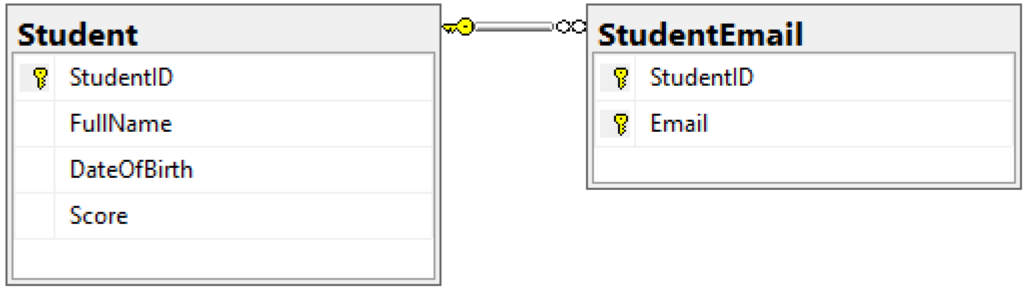
Practice

- StudentID start from 1 and increase one by one.
- FullName is not nullable
- DateOfBirth default is ‘10/22/2000’
- Score between 0 and 100
- Email is unique
- in ‘StudentEmail’ table Primary key is StudentID & Email

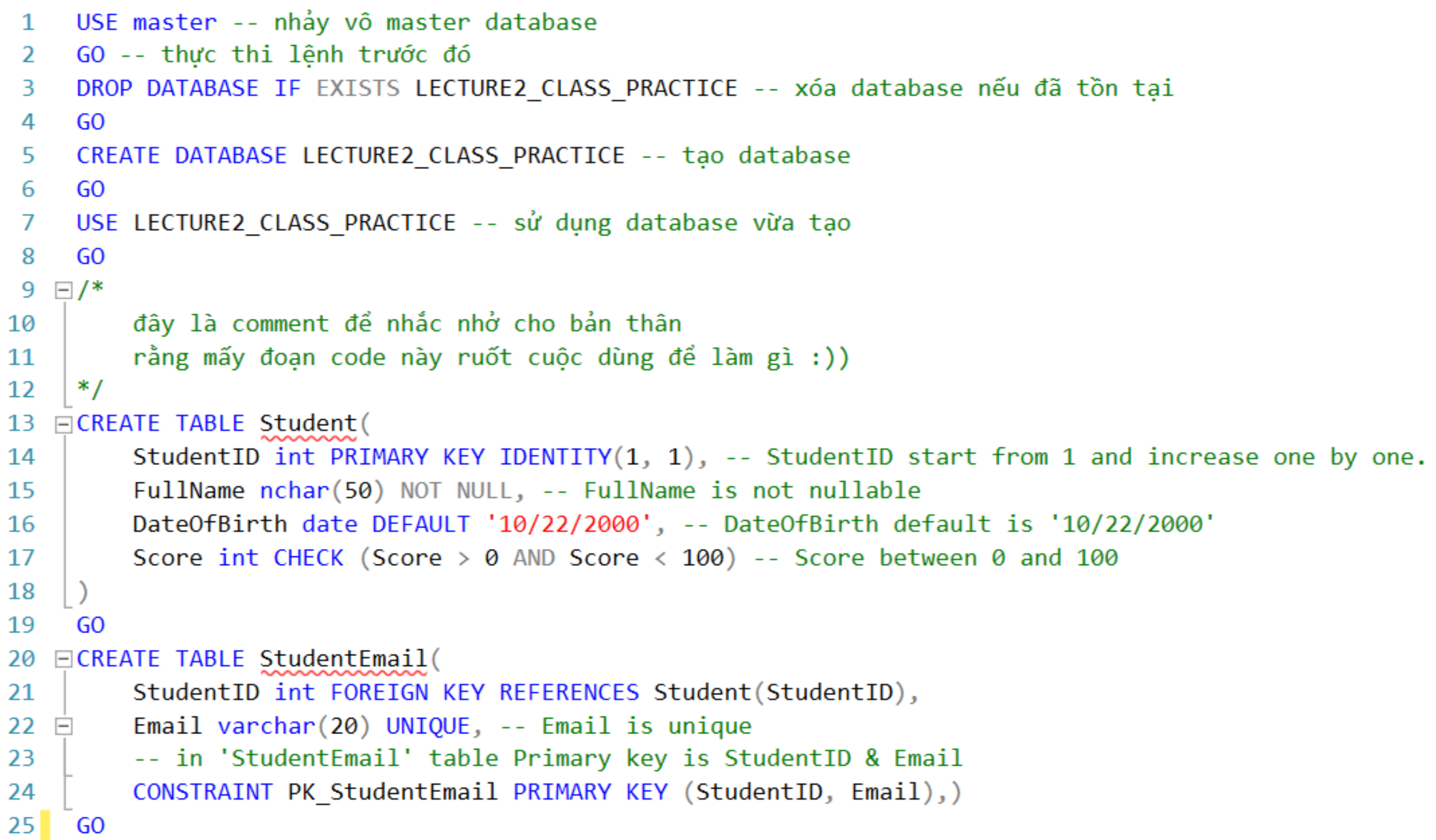

Student(StudentID, FullName, DateOfBirth, Score)
StudentEmail(StudentID, Email)

Database diagram



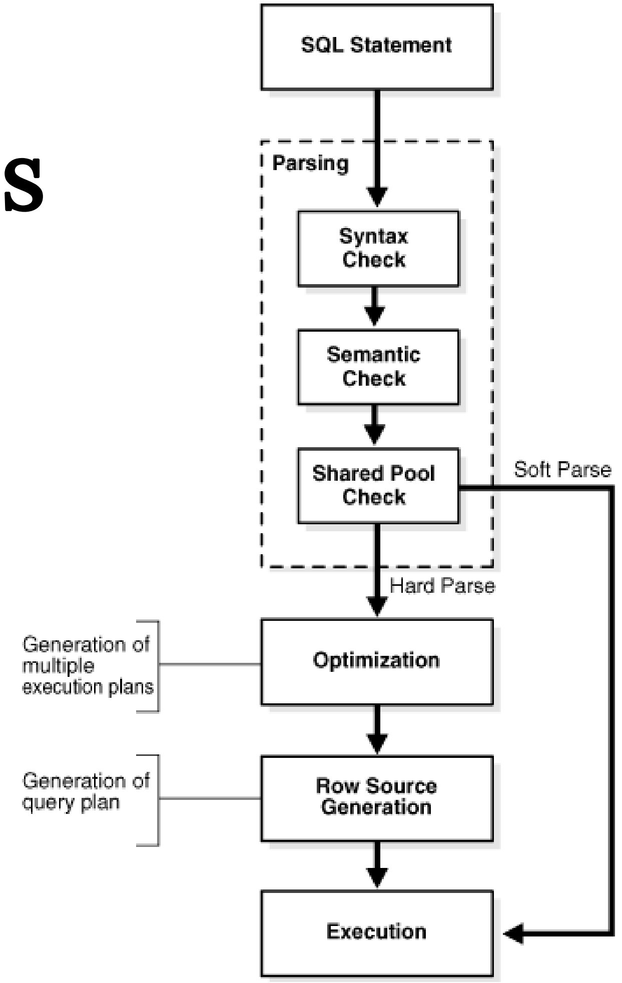


open file “lecture2-practice-create-table-result.sql”



```sql
USE master -- nhảy vô master database
GO -- thực thi lệnh trước đó
DROP DATABASE IF EXISTS LECTURE2_CLASS_PRACTICE -- xóa database nếu đã tồn tại
GO
CREATE DATABASE LECTURE2_CLASS_PRACTICE -- tạo database
GO
USE LECTURE2_CLASS_PRACTICE -- sử dụng database vừa tạo
GO
/*
    đây là comment để nhắc nhở cho bản thân
    rằng mấy đoạn code này ruốt cuộc dùng để làm gì :))
*/
CREATE TABLE Student(
    StudentID int PRIMARY KEY IDENTITY(1, 1), -- StudentID start from 1 and increase one by one.
    FullName nchar(50) NOT NULL, -- FullName is not nullable
    DateOfBirth date DEFAULT '10/22/2000', -- DateOfBirth default is '10/22/2000'
    Score int CHECK (Score > 0 AND Score < 100) -- Score between 0 and 100
)
GO
CREATE TABLE StudentEmail(
    StudentID int FOREIGN KEY REFERENCES Student(StudentID),
    Email varchar(20) UNIQUE, -- Email is unique
    -- in 'StudentEmail' table Primary key is StudentID & Email
    CONSTRAINT PK_StudentEmail PRIMARY KEY (StudentID, Email),)
GO
```

SQL process



Description of "Figure 3-1 Stages of SQL Processing"

SQL convention

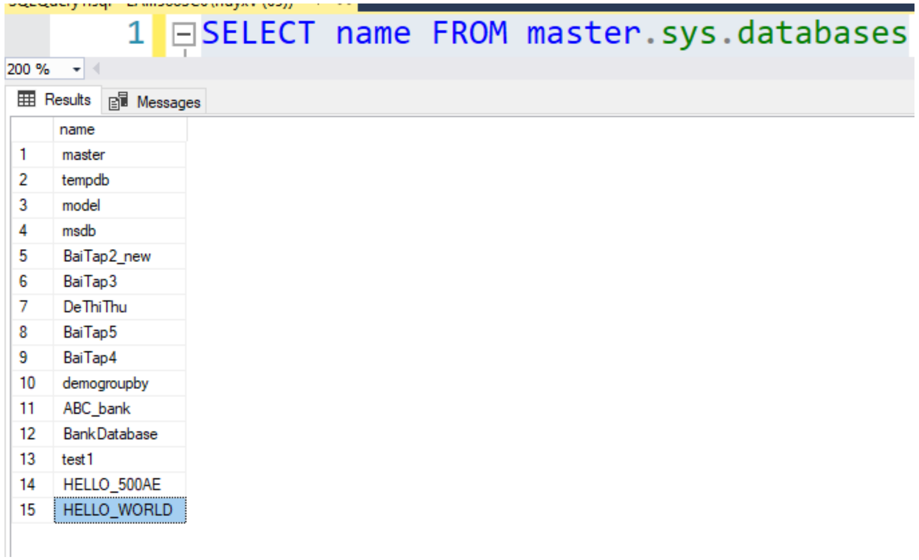
DO	TRY TO AVOID	Description
SELECT	select	SQL STATEMENT IS UPPER CASE
`int`	INT	Date type should be lowercase
EmployeeSalaryID	EmployeesalaryID	Follow Pascal case for variable, table, column
@studentCount	@@studentCount	Avoid @@ prefix
@studentCount	@sc	Clear meaning
EmployeeSalary	Employee Salary	Only use ([a-zA-Z][a-zA-Z0-9])

Name conventions

key word	prefix
PRIMARY KEY	PK_: Primary Key constraints
FOREIGN KEY	FK_: Foreign Key constraints
UNIQUE	UNI_: Unique constraints
DEFAULT	DF_: Default constraints
NOT NULL	Follow after column
CHECK	CHK_: Check constraints
View	view_: Views
Index	IX_: Indexes

See all databases

```
SELECT name
FROM master.sys.databases
```



```
SELECT name FROM master.sys.databases
```

	name
1	master
2	tempdb
3	model
4	msdb
5	BaiTap2_new
6	BaiTap3
7	DeThiThu
8	BaiTap5
9	BaiTap4
10	demogroupby
11	ABC_bank
12	BankDatabase
13	test1
14	HELLO_500AE
15	HELLO_WORLD

Break the limit

```
USE MASTER;
DROP DATABASE IF EXISTS HELLO_500AE;
CREATE DATABASE HELLO_500AE;
USE HELLO_500AE;

CREATE TABLE BreakTheLimit(
        Data char(8000) NOT NULL,
        MoreData char(54) NOT NULL
)
```

Extra Resources

Name	Link
floating point	https://www.youtube.com/watch?v=L8OYx1I8qNg
sql datatype	https://www.w3schools.com/sql/sql_datatypes.asp
alter table	https://www.w3schools.com/sql/sql_alter.asp
SQL process	https://docs.oracle.com/database/121/TGSQL/tgsql_sqlproc.htm#TGSQL176
break limit	https://www.linkedin.com/pulse/page-size-sql-server-mohammad-mehrabani/
became SQL god?	https://www.w3schools.com/sql/default.asp