Activity No. 12.1 - User-Defined Functions

Name: Efa, Christian Guevarra, Hans Angelo Mendoza, John Renzo Nicolas, Sean Julian Vinluan, Armando	**Date:** 25/11/2022
Section: CPE21S3	**Instructor:** Dr. Jonathan Vidal Taylar

Objectives:

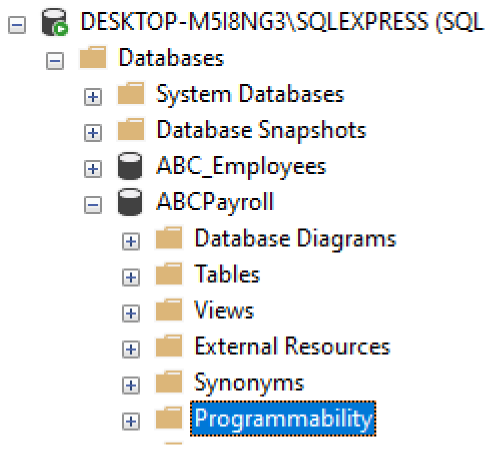
This activity aims to create and implement user-defined functions in databases

Intended Learning Outcomes (ILOs):

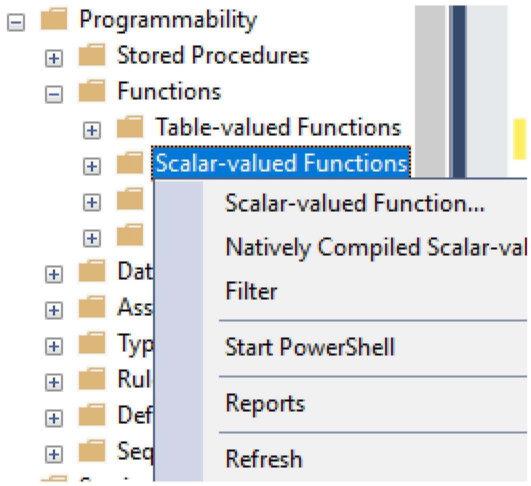
The students should be able to:
2.1 Create different types of user-defined functions
2.2Implement and execute user-defined functions in a database.

Output

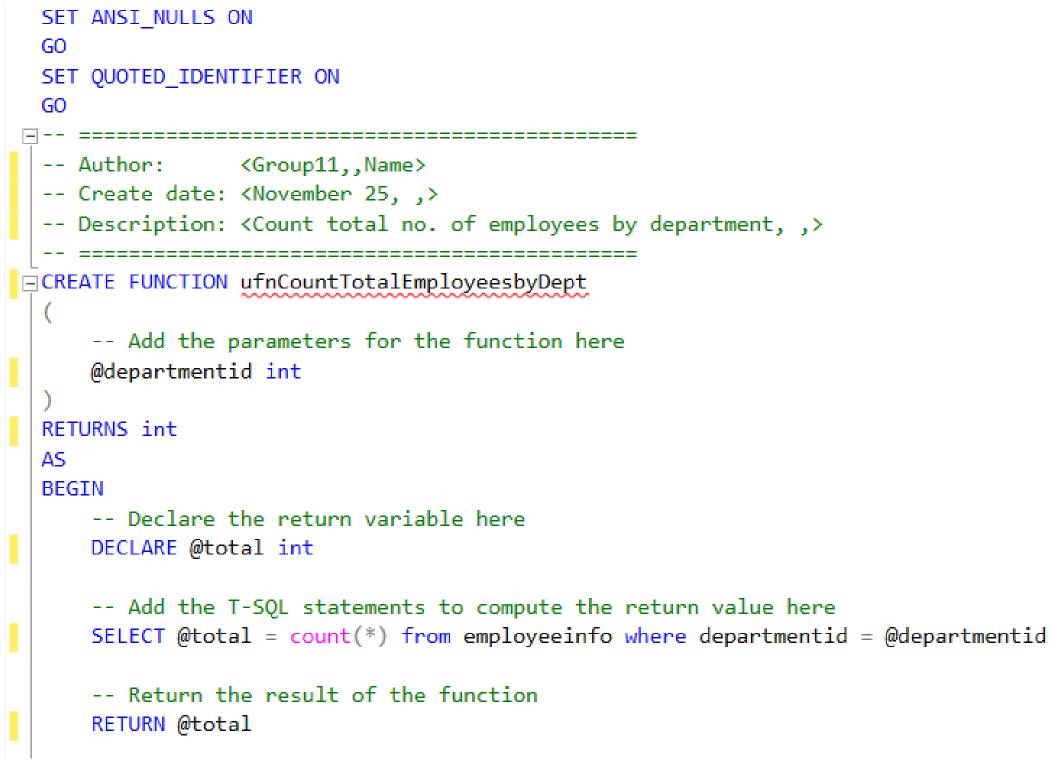
Scalar Functions
Step 1: In Object Explorer, connect to an instance of Database Engine and then expand that instance.
Step 2: Expand Databases, expand the ABCPayroll database, and then expand Programmability.



Step 3: Choose Functions, and then right-click Scalar-valued Functions. Choose Scalar-valued Function.

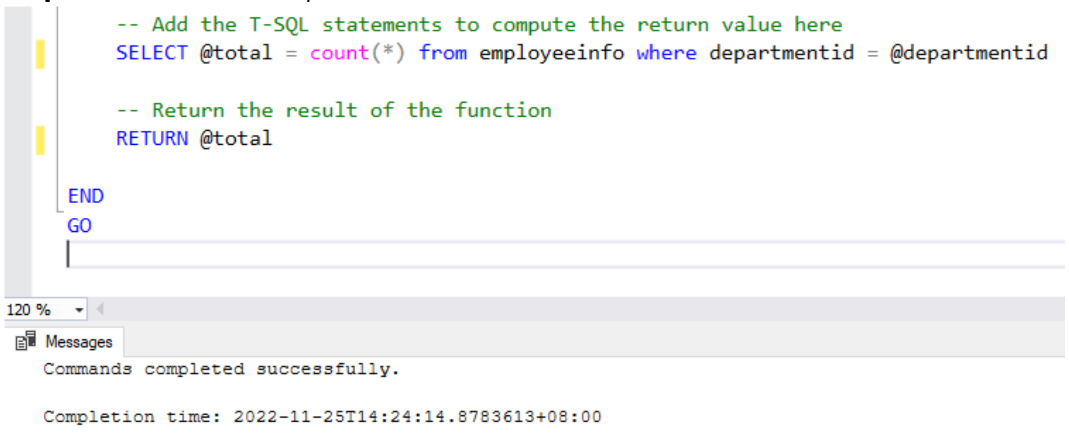


Step 4: Modify the function using the given screenshot. Type your name as author and the creation date.



```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
-- =============================================
-- Author:      <Group11,,Name>
-- Create date: <November 25, ,>
-- Description: <Count total no. of employees by department, ,>
-- =============================================
CREATE FUNCTION ufnCountTotalEmployeesbyDept
(
    -- Add the parameters for the function here
    @departmentid int
)
RETURNS int
AS
BEGIN
    -- Declare the return variable here
    DECLARE @total int

    -- Add the T-SQL statements to compute the return value here
    SELECT @total = count(*) from employeeinfo where departmentid = @departmentid

    -- Return the result of the function
    RETURN @total
```

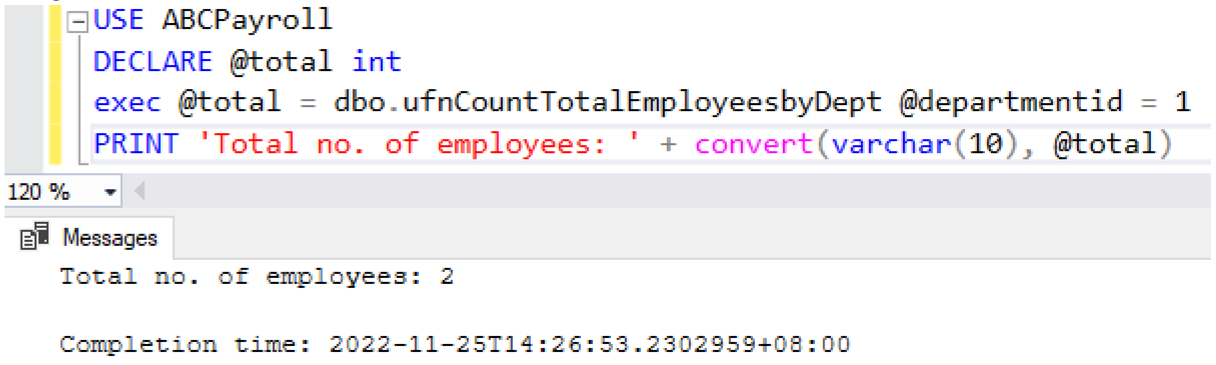
Step 5: Click execute or press F5 to save the stored function.



```
    -- Add the T-SQL statements to compute the return value here
    SELECT @total = count(*) from employeeinfo where departmentid = @departmentid

    -- Return the result of the function
    RETURN @total

END
GO
```

```
Commands completed successfully.

Completion time: 2022-11-25T14:24:14.8783613+08:00
```

Step 6: Test the ufnCountTotalEmployeesbyDept stored function. Create a new query and type the given statement. Click Execute.



```
USE ABCPayroll
DECLARE @total int
exec @total = dbo.ufnCountTotalEmployeesbyDept @departmentid = 1
PRINT 'Total no. of employees: ' + convert(varchar(10), @total)
```

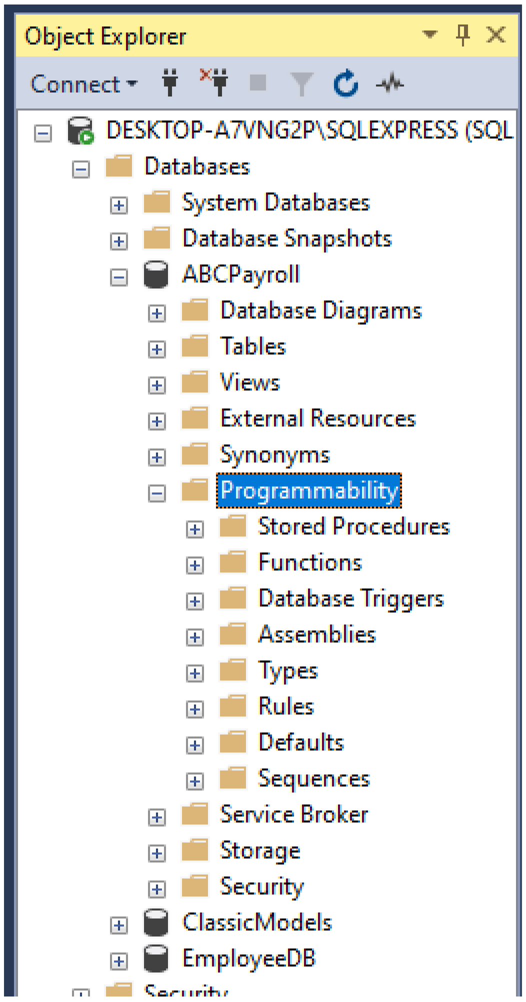
```
Total no. of employees: 2

Completion time: 2022-11-25T14:26:53.2302959+08:00
```

Table-valued Functions

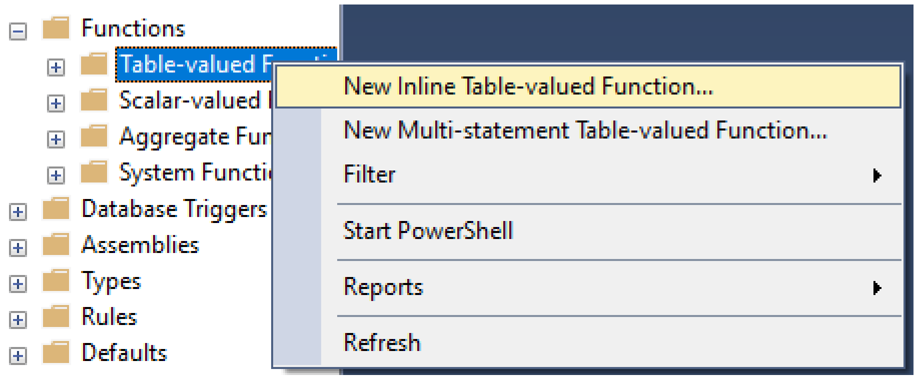
A. Inline Table-valued Function

Step 1: In Object Explorer, connect to an instance of Database Engine and then expand that instance.

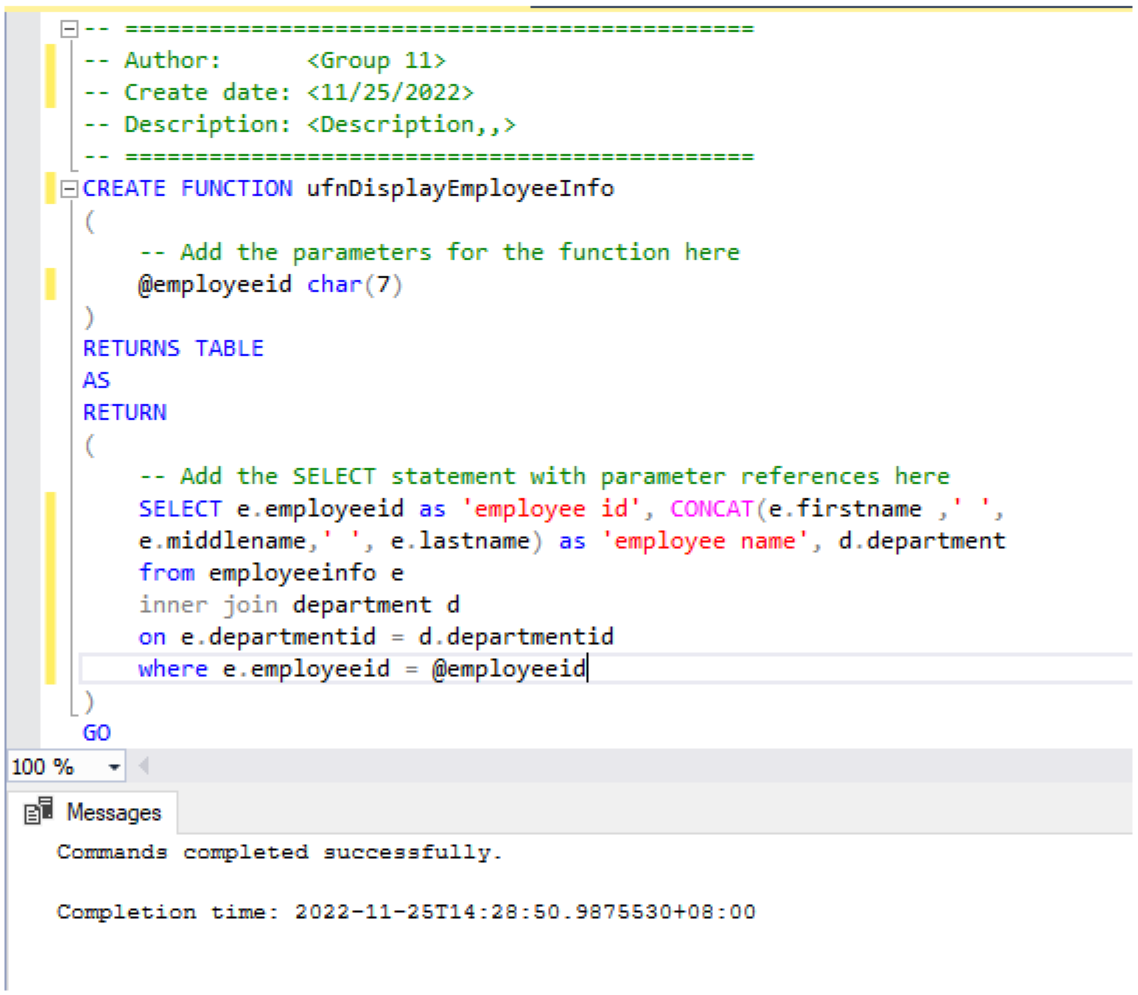
Step 2: Expand Databases, expand the ABCPayroll database, and then expand Programmability



Step 3: Choose Functions, and then right-click Table-valued Functions. Choose New Inline Table-valued Function



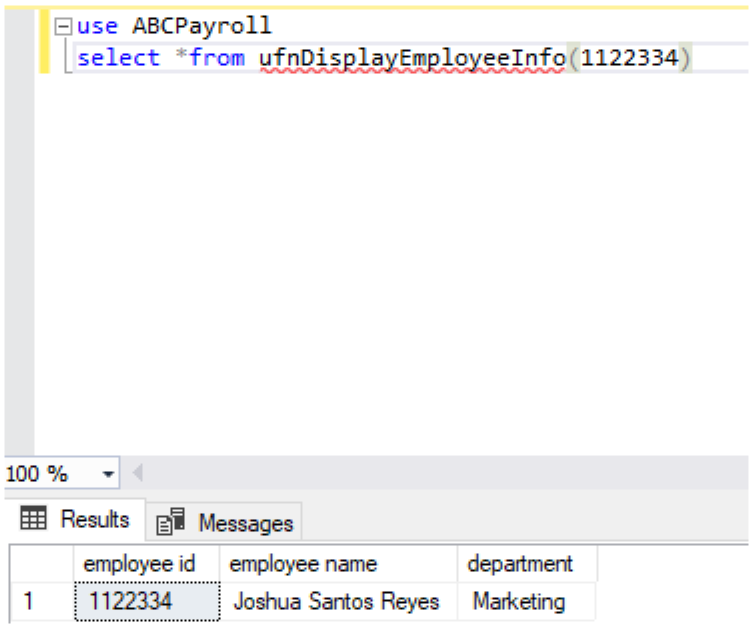
Step 4: Modify the function using the given screenshot. Type your name as author and the creation date.
Step 5: Click execute or press F5 to save the stored function.



```
-- =============================================
-- Author:      <Group 11>
-- Create date: <11/25/2022>
-- Description: <Description,,>
-- =============================================
CREATE FUNCTION ufnDisplayEmployeeInfo
(
    -- Add the parameters for the function here
    @employeeid char(7)
)
RETURNS TABLE
AS
RETURN
(
    -- Add the SELECT statement with parameter references here
    SELECT e.employeeid as 'employee id', CONCAT(e.firstname ,' ',
    e.middlename,' ', e.lastname) as 'employee name', d.department
    from employeeinfo e
    inner join department d
    on e.departmentid = d.departmentid
    where e.employeeid = @employeeid
)
GO
```

```
Commands completed successfully.

Completion time: 2022-11-25T14:28:50.9875530+08:00
```

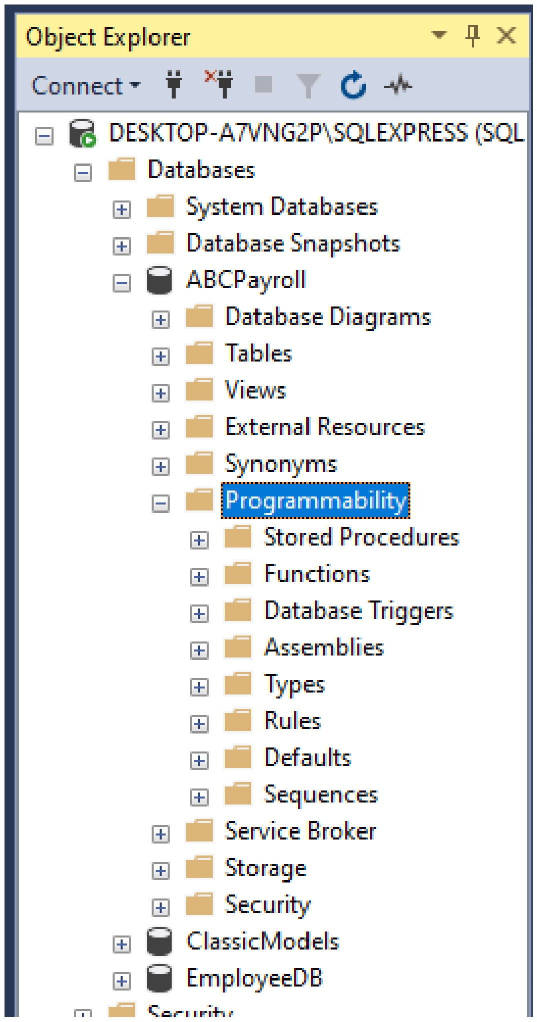
Step 6: Test the ufnDisplayEmployeeInfo stored function. Create a new query and type the given statement. Click Execute.



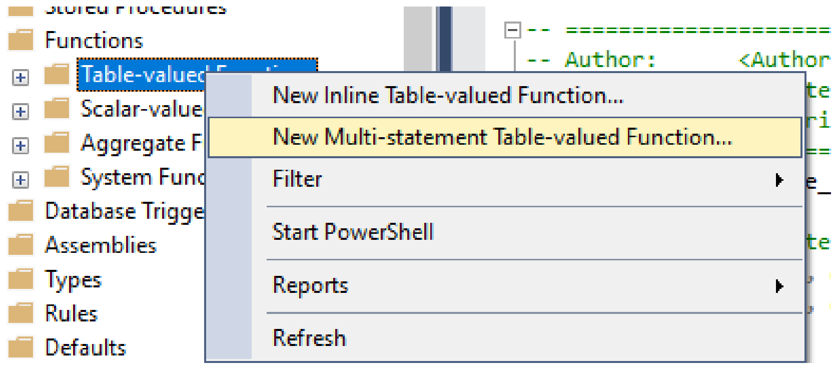
```
use ABCPayroll
select *from ufnDisplayEmployeeInfo(1122334)
```

	employee id	employee name	department
1	1122334	Joshua Santos Reyes	Marketing

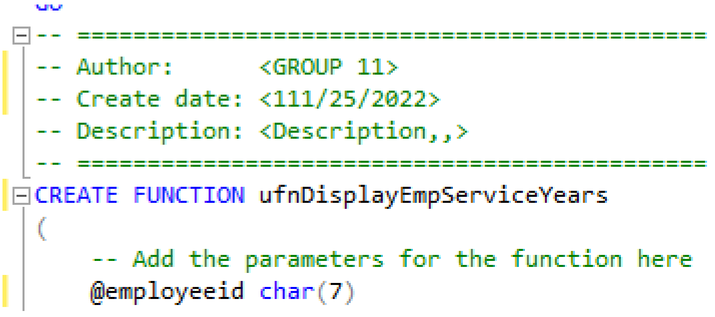
B. Multi-statement Table-valued Function
Step 1: In Object Explorer, connect to an instance of Database Engine and then expand that instance.
Step 2: Expand Databases, expand the ABCPayroll database, and then expand Programmability



Step 3: Choose Functions, and then right-click Table-valued Functions. Choose New Multi-statement Table-valued Function

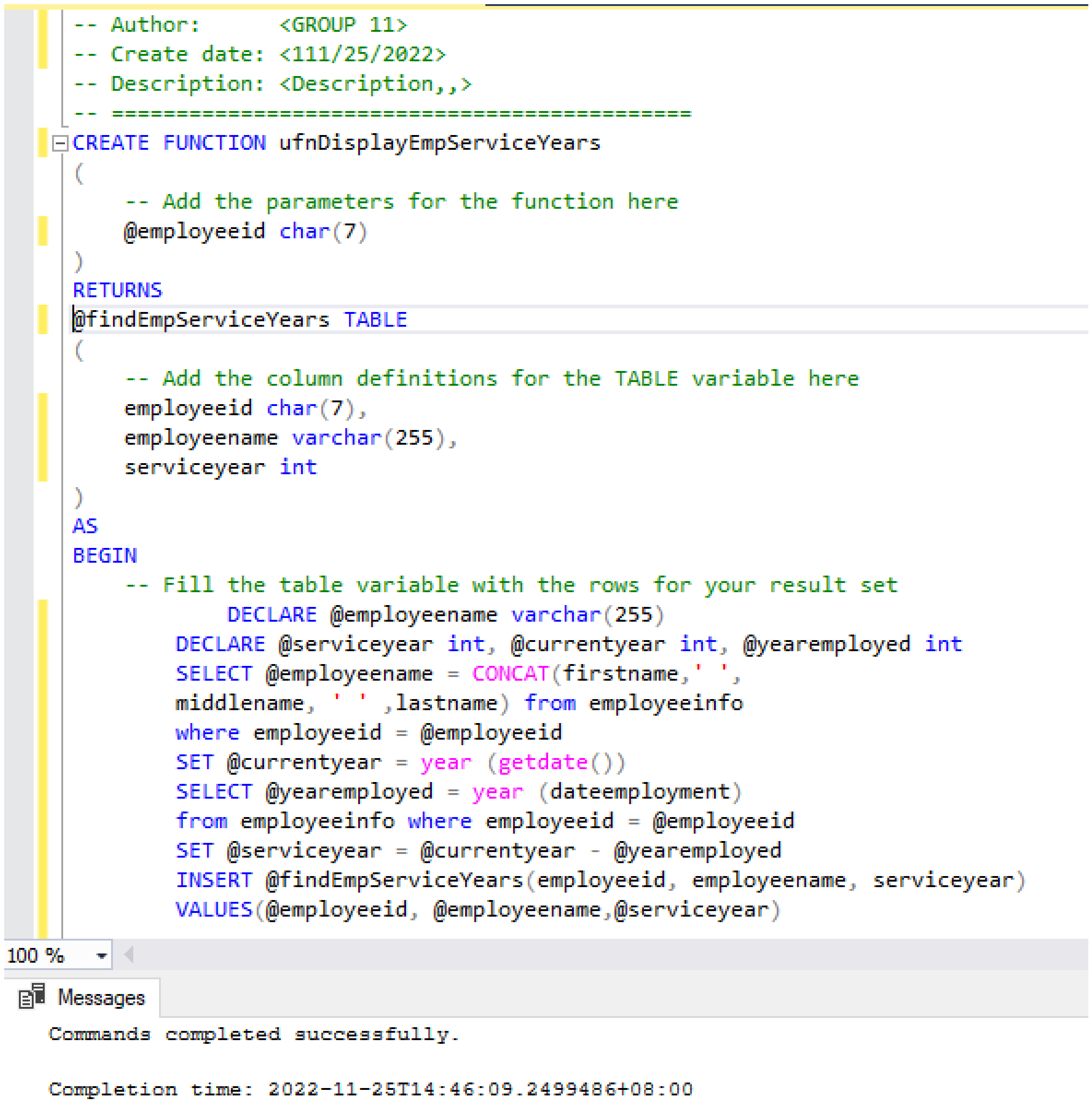


Step 4: Modify the function using the given screenshot. Type your name as author and the creation date.



```
-- ============================================
-- Author:      <GROUP 11>
-- Create date: <111/25/2022>
-- Description: <Description,,>
-- ============================================
CREATE FUNCTION ufnDisplayEmpServiceYears
(
    -- Add the parameters for the function here
    @employeeid char(7)
```

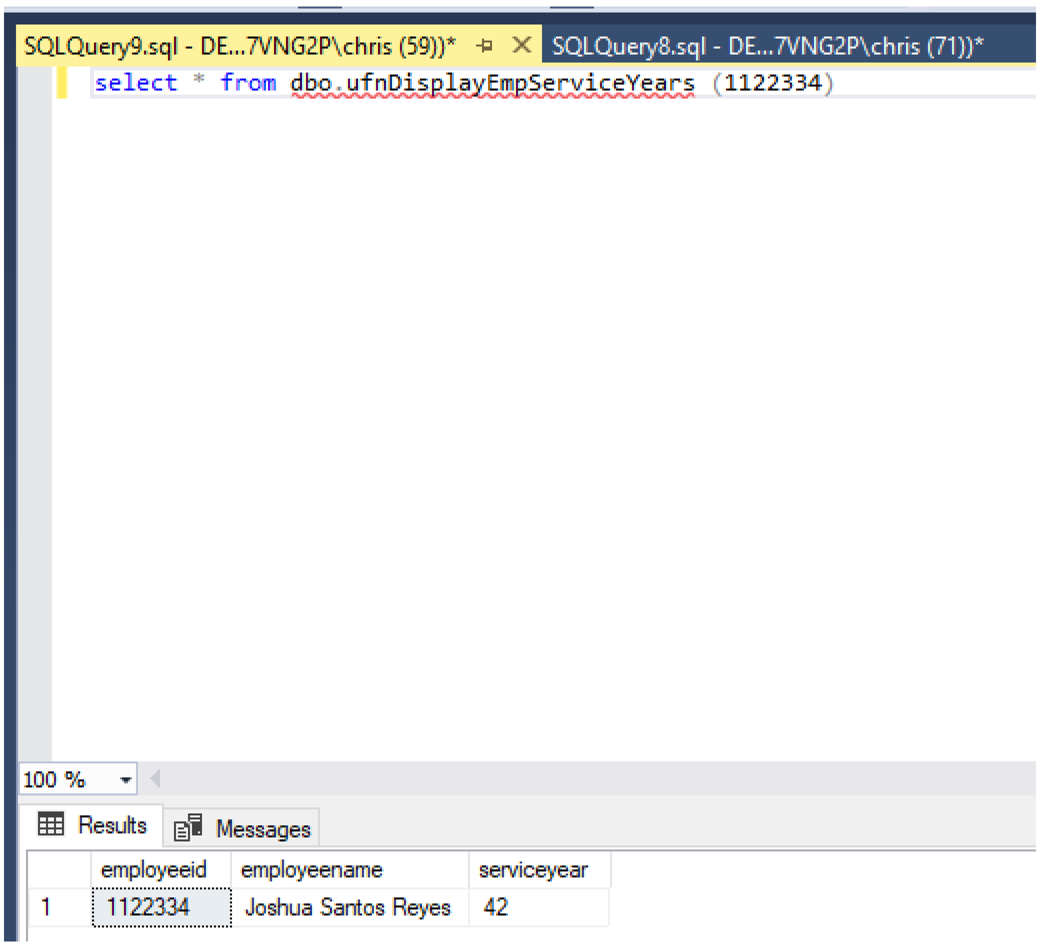
Step 5: Click execute or press F5 to save the stored function.



```
-- Author:      <GROUP 11>
-- Create date: <111/25/2022>
-- Description: <Description,,>
-- ============================================
CREATE FUNCTION ufnDisplayEmpServiceYears
(
    -- Add the parameters for the function here
    @employeeid char(7)
)
RETURNS
@findEmpServiceYears TABLE
(
    -- Add the column definitions for the TABLE variable here
    employeeid char(7),
    employeename varchar(255),
    serviceyear int
)
AS
BEGIN
    -- Fill the table variable with the rows for your result set
            DECLARE @employeename varchar(255)
        DECLARE @serviceyear int, @currentyear int, @yearemployed int
        SELECT @employeename = CONCAT(firstname,' ',
        middlename, ' ' ,lastname) from employeeinfo
        where employeeid = @employeeid
        SET @currentyear = year (getdate())
        SELECT @yearemployed = year (dateemployment)
        from employeeinfo where employeeid = @employeeid
        SET @serviceyear = @currentyear - @yearemployed
        INSERT @findEmpServiceYears(employeeid, employeename, serviceyear)
        VALUES(@employeeid, @employeename,@serviceyear)
```

```
Commands completed successfully.

Completion time: 2022-11-25T14:46:09.2499486+08:00
```

Step 6: Test the ufnDisplayEmpServiceYears stored function. Create a new query and type the givenstatement. Click Execute.


SQLQuery9.sql - DE...7VNG2P\chris (59))*
SQLQuery8.sql - DE...7VNG2P\chris (71))*
100 %
Results
Messages

```
select * from dbo.ufnDisplayEmpServiceYears (1122334)
```

	employeeid	employeename	serviceyear
1	1122334	Joshua Santos Reyes	42

Supplementary Activity

Do the following tasks and copy screenshot(s) of your output.

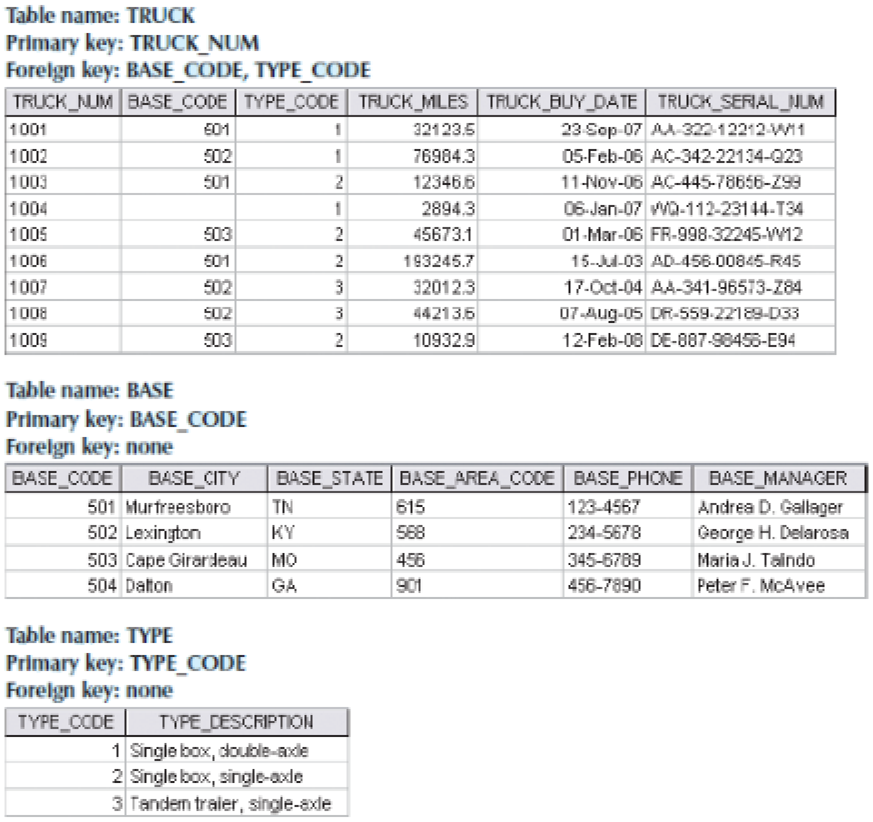


Table name: TRUCK
Primary key: TRUCK_NUM
Foreign key: BASE_CODE, TYPE_CODE

TRUCK_NUM	BASE_CODE	TYPE_CODE	TRUCK_MILES	TRUCK_BUY_DATE	TRUCK_SERIAL_NUM
1001	501	1	32123.5	23-Sep-07	AA-322-12212-W11
1002	502	1	76984.3	05-Feb-06	AC-342-22134-Q23
1003	501	2	12346.6	11-Nov-06	AC-445-78656-Z99
1004		1	2894.3	06-Jan-07	WQ-112-23144-T34
1005	503	2	45673.1	01-Mar-06	FR-998-32245-W12
1006	501	2	193245.7	15-Jul-03	AD-456-00845-R45
1007	502	3	32012.3	17-Oct-04	AA-341-96573-Z84
1008	502	3	44213.6	07-Aug-05	DR-559-22189-D33
1009	503	2	10932.9	12-Feb-08	DE-887-98456-E94

Table name: BASE
Primary key: BASE_CODE
Foreign key: none

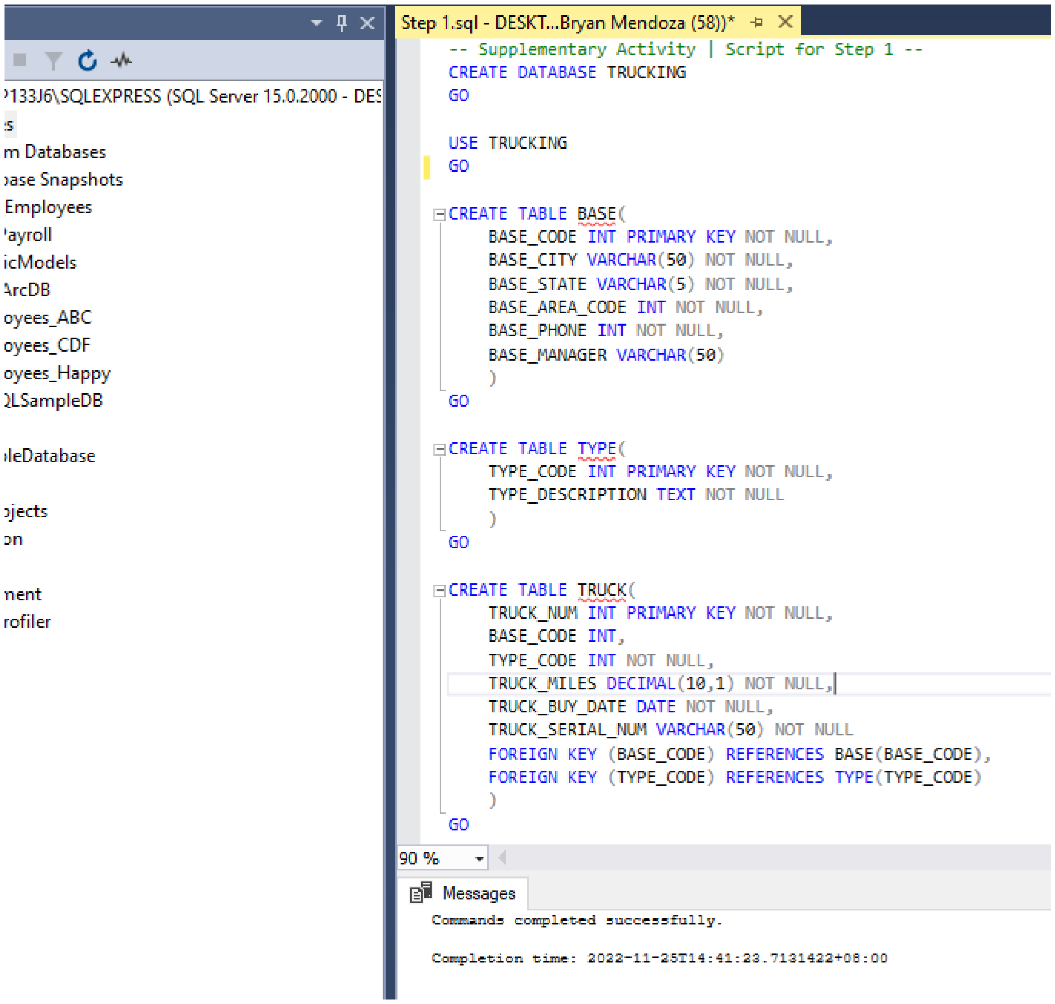
BASE_CODE	BASE_CITY	BASE_STATE	BASE_AREA_CODE	BASE_PHONE	BASE_MANAGER
501	Murfreesboro	TN	615	123-4567	Andrea D. Gallager
502	Lexington	KY	568	234-5678	George H. Delarosa
503	Cape Girardeau	MO	456	345-6789	Maria J. Talindo
504	Dalton	GA	901	456-7890	Peter F. McAvee

Table name: TYPE
Primary key: TYPE_CODE
Foreign key: none

TYPE_CODE	TYPE_DESCRIPTION
1	Single box, double-axle
2	Single box, single-axle
3	Tandem trailer, single-axle

1. Create a script to create the Trucking database and the following tables. Use the appropriate data types and assign the primary keys and foreign keys.

Creation of the Database



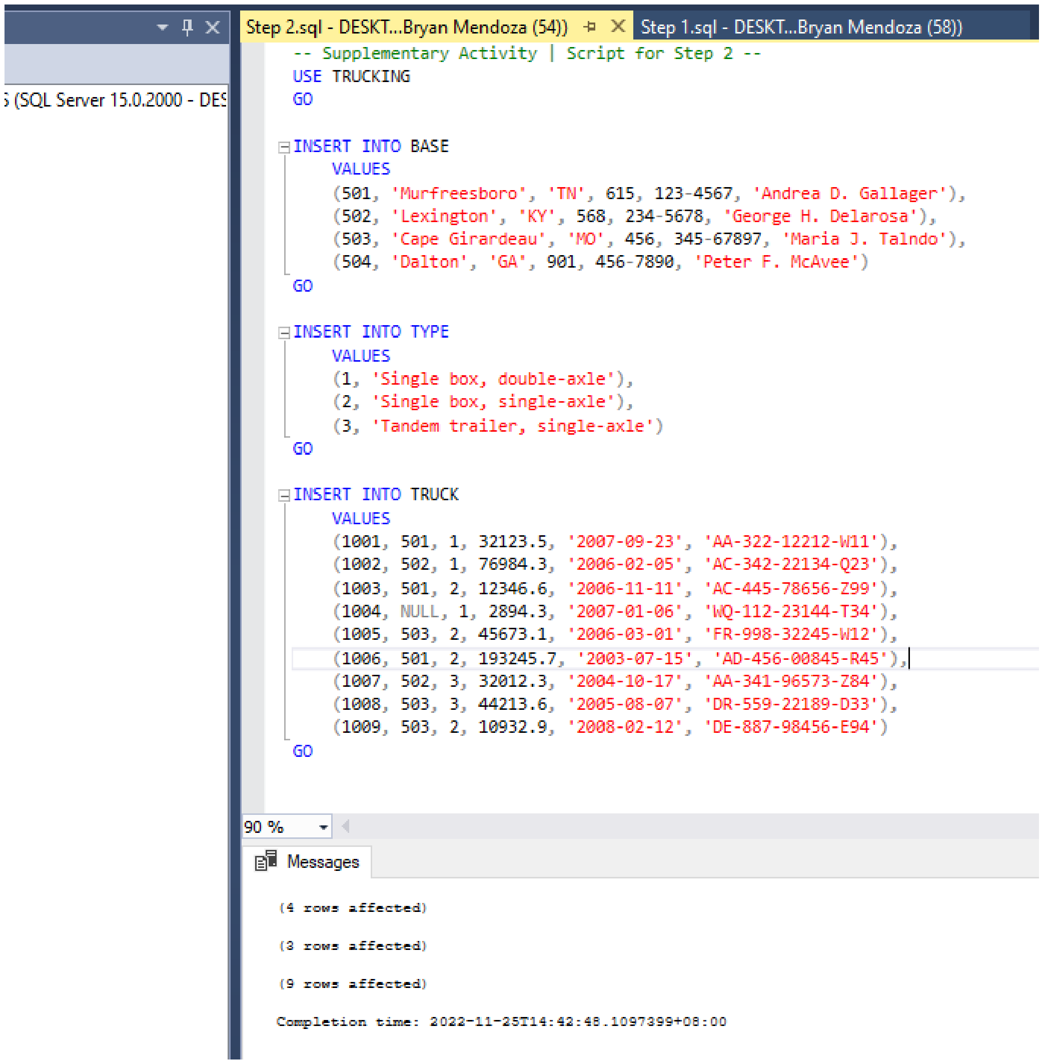
```sql
-- Supplementary Activity | Script for Step 1 --
CREATE DATABASE TRUCKING
GO

USE TRUCKING
GO

CREATE TABLE BASE(
    BASE_CODE INT PRIMARY KEY NOT NULL,
    BASE_CITY VARCHAR(50) NOT NULL,
    BASE_STATE VARCHAR(5) NOT NULL,
    BASE_AREA_CODE INT NOT NULL,
    BASE_PHONE INT NOT NULL,
    BASE_MANAGER VARCHAR(50)
    )
GO

CREATE TABLE TYPE(
    TYPE_CODE INT PRIMARY KEY NOT NULL,
    TYPE_DESCRIPTION TEXT NOT NULL
    )
GO

CREATE TABLE TRUCK(
    TRUCK_NUM INT PRIMARY KEY NOT NULL,
    BASE_CODE INT,
    TYPE_CODE INT NOT NULL,
    TRUCK_MILES DECIMAL(10,1) NOT NULL,
    TRUCK_BUY_DATE DATE NOT NULL,
    TRUCK_SERIAL_NUM VARCHAR(50) NOT NULL
    FOREIGN KEY (BASE_CODE) REFERENCES BASE(BASE_CODE),
    FOREIGN KEY (TYPE_CODE) REFERENCES TYPE(TYPE_CODE)
    )
GO
```

```
Commands completed successfully.

Completion time: 2022-11-25T14:41:23.7131422+08:00
```

Observation:
We created our database named TRUCKING using a new query following the data types given from the given table.

2. Create a script to insert the given values using the Trucking database.

Insertion of the Entries to each Table

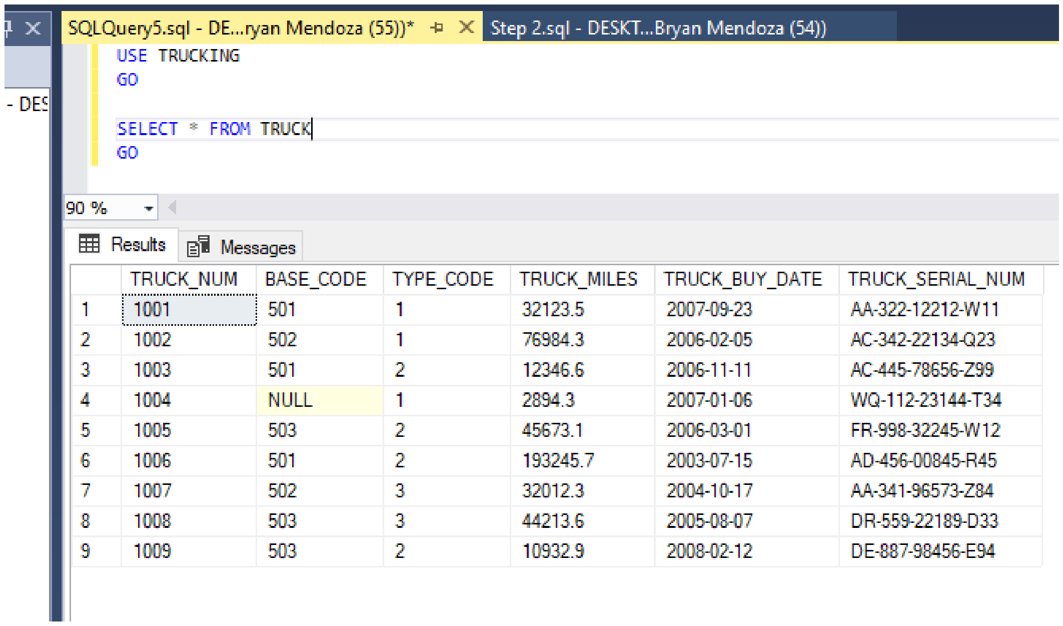

(SQL Server 15.0.2000 - DES
Step 2.sql - DESKT...Bryan Mendoza (54))
Step 1.sql - DESKT...Bryan Mendoza (58))
90 %
Messages

```sql
-- Supplementary Activity | Script for Step 2 --
USE TRUCKING
GO

INSERT INTO BASE
    VALUES
    (501, 'Murfreesboro', 'TN', 615, 123-4567, 'Andrea D. Gallager'),
    (502, 'Lexington', 'KY', 568, 234-5678, 'George H. Delarosa'),
    (503, 'Cape Girardeau', 'MO', 456, 345-67897, 'Maria J. Talndo'),
    (504, 'Dalton', 'GA', 901, 456-7890, 'Peter F. McAvee')
GO

INSERT INTO TYPE
    VALUES
    (1, 'Single box, double-axle'),
    (2, 'Single box, single-axle'),
    (3, 'Tandem trailer, single-axle')
GO

INSERT INTO TRUCK
    VALUES
    (1001, 501, 1, 32123.5, '2007-09-23', 'AA-322-12212-W11'),
    (1002, 502, 1, 76984.3, '2006-02-05', 'AC-342-22134-Q23'),
    (1003, 501, 2, 12346.6, '2006-11-11', 'AC-445-78656-Z99'),
    (1004, NULL, 1, 2894.3, '2007-01-06', 'WQ-112-23144-T34'),
    (1005, 503, 2, 45673.1, '2006-03-01', 'FR-998-32245-W12'),
    (1006, 501, 2, 193245.7, '2003-07-15', 'AD-456-00845-R45'),
    (1007, 502, 3, 32012.3, '2004-10-17', 'AA-341-96573-Z84'),
    (1008, 503, 3, 44213.6, '2005-08-07', 'DR-559-22189-D33'),
    (1009, 503, 2, 10932.9, '2008-02-12', 'DE-887-98456-E94')
GO
```

```
(4 rows affected)

(3 rows affected)

(9 rows affected)

Completion time: 2022-11-25T14:42:48.1097399+08:00
```

Displaying Table Contents

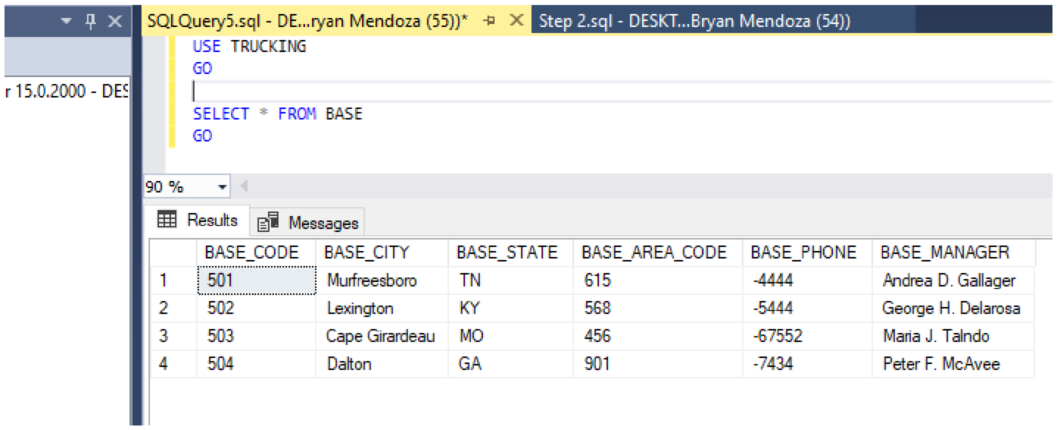
TRUCK Table


SQLQuery5.sql - DE...ryan Mendoza (55))*
Step 2.sql - DESKT...Bryan Mendoza (54))
- DES
90 %
Results
Messages

```
USE TRUCKING
GO

SELECT * FROM TRUCK
GO
```

	TRUCK_NUM	BASE_CODE	TYPE_CODE	TRUCK_MILES	TRUCK_BUY_DATE	TRUCK_SERIAL_NUM
1	1001	501	1	32123.5	2007-09-23	AA-322-12212-W11
2	1002	502	1	76984.3	2006-02-05	AC-342-22134-Q23
3	1003	501	2	12346.6	2006-11-11	AC-445-78656-Z99
4	1004	NULL	1	2894.3	2007-01-06	WQ-112-23144-T34
5	1005	503	2	45673.1	2006-03-01	FR-998-32245-W12
6	1006	501	2	193245.7	2003-07-15	AD-456-00845-R45
7	1007	502	3	32012.3	2004-10-17	AA-341-96573-Z84
8	1008	503	3	44213.6	2005-08-07	DR-559-22189-D33
9	1009	503	2	10932.9	2008-02-12	DE-887-98456-E94

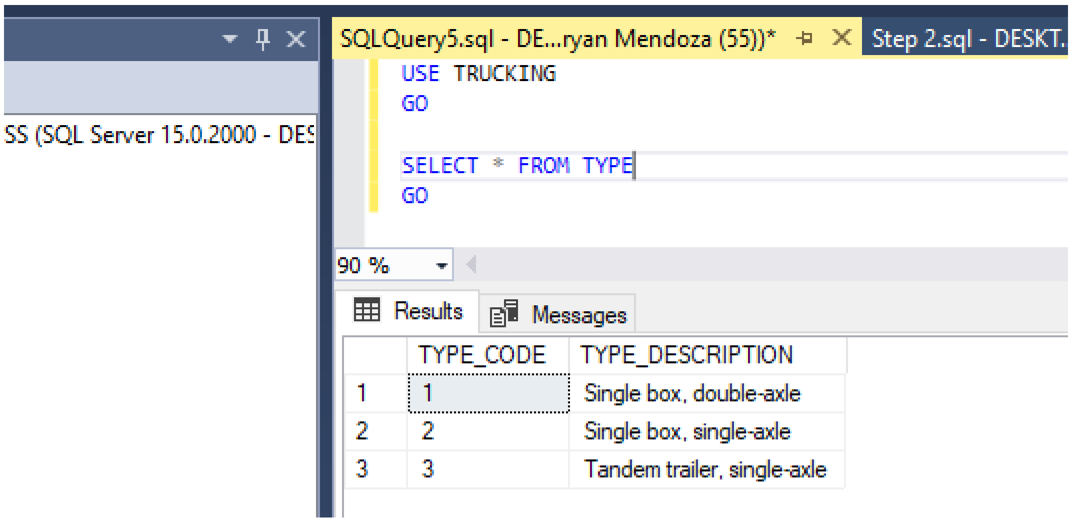
BASE Table


SQLQuery5.sql - DE...ryan Mendoza (55))*
Step 2.sql - DESKT...Bryan Mendoza (54))
r 15.0.2000 - DES
90 %
Results
Messages

```
USE TRUCKING
GO

SELECT * FROM BASE
GO
```

	BASE_CODE	BASE_CITY	BASE_STATE	BASE_AREA_CODE	BASE_PHONE	BASE_MANAGER
1	501	Murfreesboro	TN	615	-4444	Andrea D. Gallager
2	502	Lexington	KY	568	-5444	George H. Delarosa
3	503	Cape Girardeau	MO	456	-67552	Maria J. Talndo
4	504	Dalton	GA	901	-7434	Peter F. McAvee

TYPE Table


SQLQuery5.sql - DE...ryan Mendoza (55))*
Step 2.sql - DESKT
SS (SQL Server 15.0.2000 - DES
90 %
Results
Messages

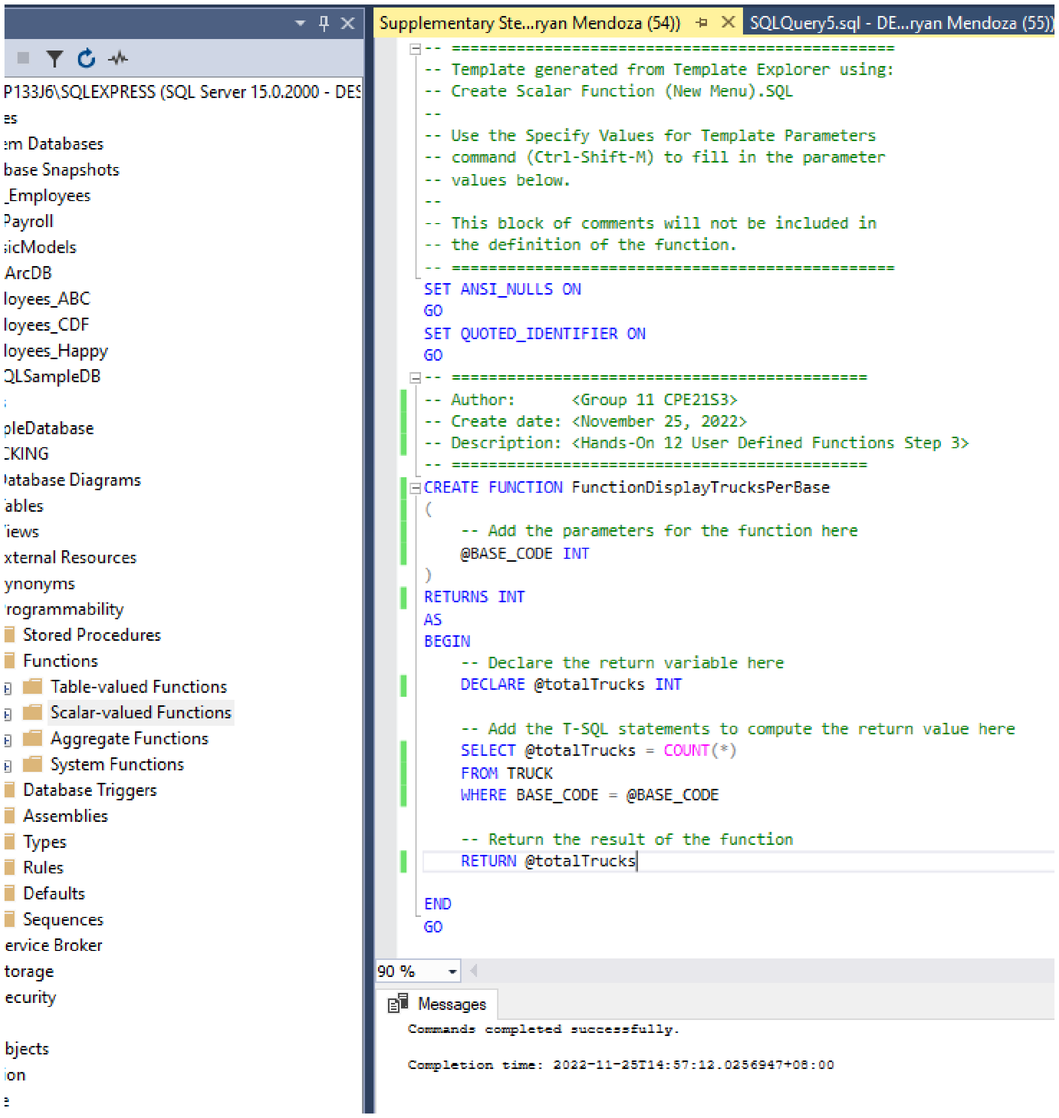
```
USE TRUCKING
GO

SELECT * FROM TYPE
GO
```

	TYPE_CODE	TYPE_DESCRIPTION
1	1	Single box, double-axle
2	2	Single box, single-axle
3	3	Tandem trailer, single-axle

Observation:
Using another query, we inserted the entries as required by the instructions. The data inserted on the table is from the given table of this activity. We can observe that the data are successfully inserted as the SELECT statements displayed the desired results.

3. Create and execute a function that returns the total number of trucks per base. Use BASE_CODE as input parameter

Creation of the function

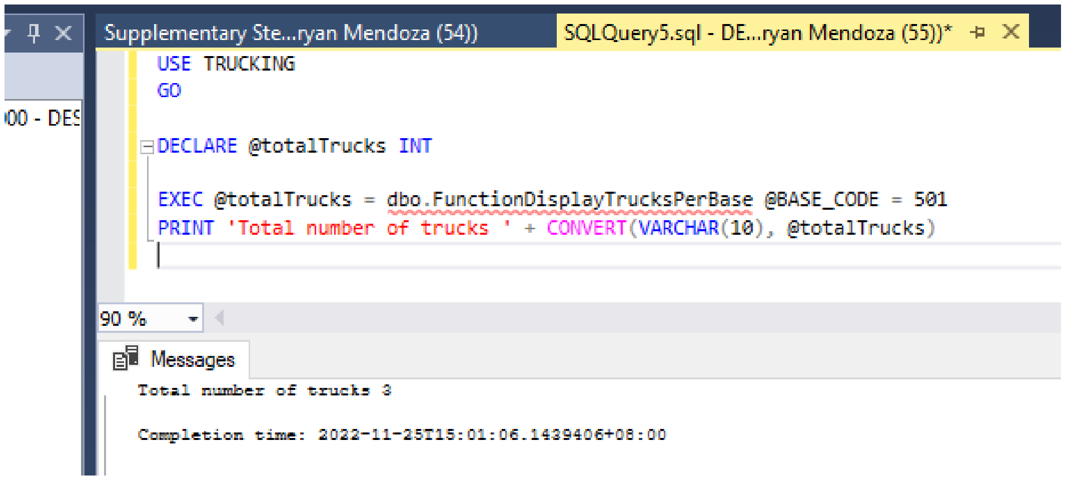


Supplementary Ste...ryan Mendoza (54))   SQLQuery5.sql - DE...ryan Mendoza (55))

```
-- ================================================
-- Template generated from Template Explorer using:
-- Create Scalar Function (New Menu).SQL
--
-- Use the Specify Values for Template Parameters
-- command (Ctrl-Shift-M) to fill in the parameter
-- values below.
--
-- This block of comments will not be included in
-- the definition of the function.
-- ================================================
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
-- =============================================
-- Author:      <Group 11 CPE21S3>
-- Create date: <November 25, 2022>
-- Description: <Hands-On 12 User Defined Functions Step 3>
-- =============================================
CREATE FUNCTION FunctionDisplayTrucksPerBase
(
    -- Add the parameters for the function here
    @BASE_CODE INT
)
RETURNS INT
AS
BEGIN
    -- Declare the return variable here
    DECLARE @totalTrucks INT

    -- Add the T-SQL statements to compute the return value here
    SELECT @totalTrucks = COUNT(*)
    FROM TRUCK
    WHERE BASE_CODE = @BASE_CODE

    -- Return the result of the function
    RETURN @totalTrucks

END
GO
```

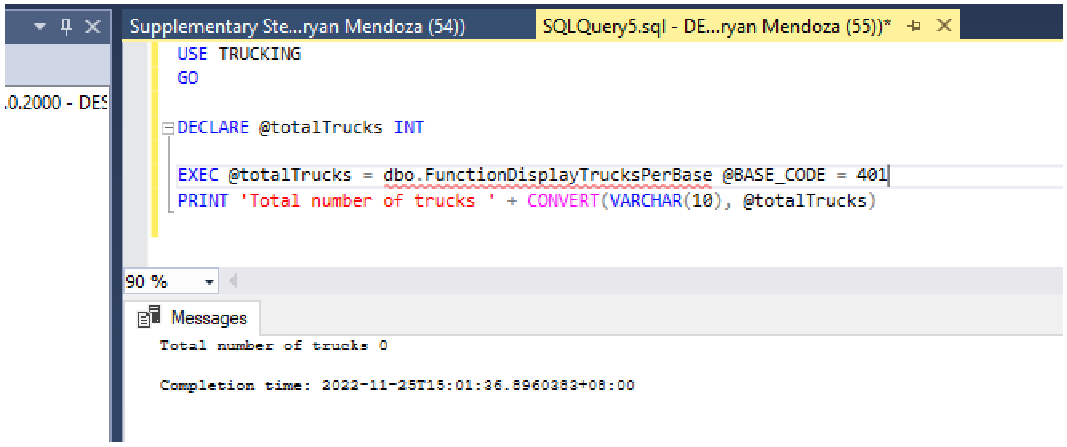
```
Commands completed successfully.

Completion time: 2022-11-25T14:57:12.0256947+08:00
```

Execution of the function


Supplementary Ste...ryan Mendoza (54))
SQLQuery5.sql - DE...ryan Mendoza (55))*
00 - DES
90 %
Messages

```
USE TRUCKING
GO

DECLARE @totalTrucks INT

EXEC @totalTrucks = dbo.FunctionDisplayTrucksPerBase @BASE_CODE = 501
PRINT 'Total number of trucks ' + CONVERT(VARCHAR(10), @totalTrucks)
```

```
Total number of trucks 3

Completion time: 2022-11-25T15:01:06.1439406+08:00
```


Supplementary Ste...ryan Mendoza (54))
SQLQuery5.sql - DE...ryan Mendoza (55))*
.0.2000 - DES
90 %
Messages

```
USE TRUCKING
GO

DECLARE @totalTrucks INT

EXEC @totalTrucks = dbo.FunctionDisplayTrucksPerBase @BASE_CODE = 401
PRINT 'Total number of trucks ' + CONVERT(VARCHAR(10), @totalTrucks)
```

```
Total number of trucks 0

Completion time: 2022-11-25T15:01:36.8960383+08:00
```

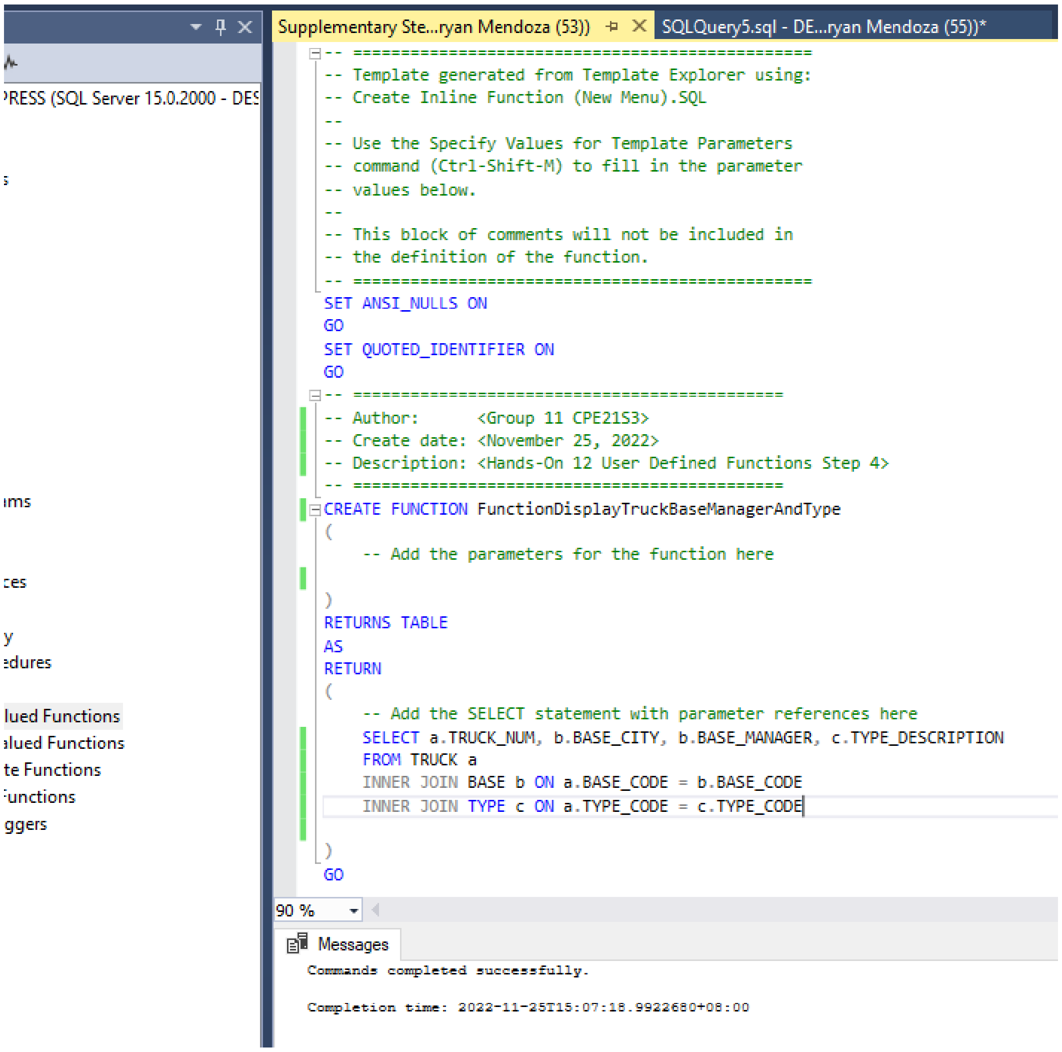
Observation:
Since the instruction was to display the total number of trucks using BASE_CODE as the parameter, we decided to use Scalar Function as this returns an INT data type based on the procedures.

After initializing the function, we called the function using a new query in order to test the results. We added a string to further emphasize the results since the function only returns a scalar value which is the integer.

We can also observe that having a BASE_CODE non-existing on the table was not displayed, thus giving a result of zero (0).

4. Create and execute a function to display the truck number, base city, base manager and type description.

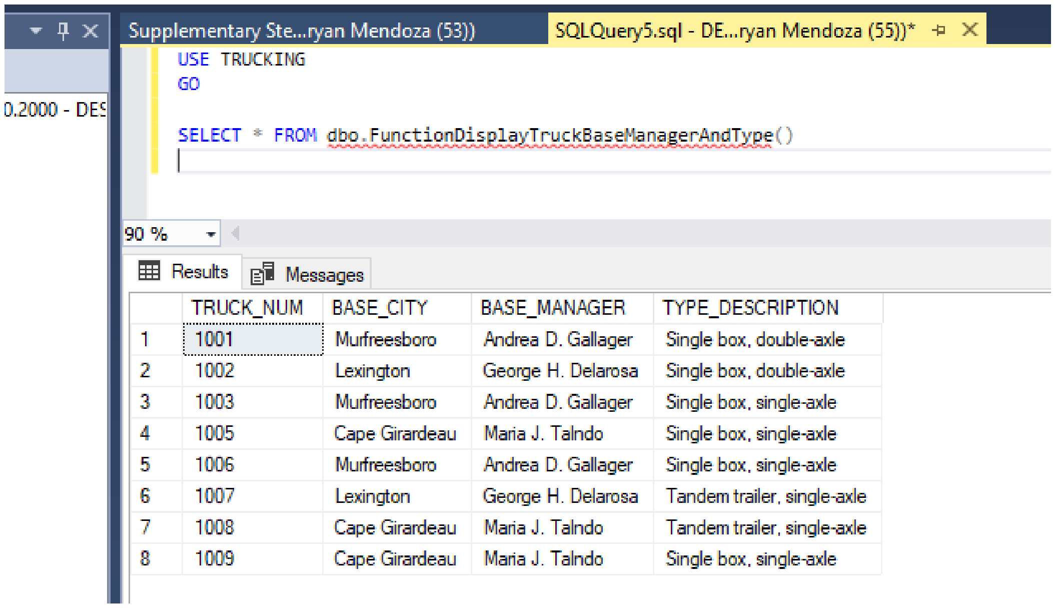
Creation of the function


Supplementary Ste...ryan Mendoza (53))
SQLQuery5.sql - DE...ryan Mendoza (55))*
PRESS (SQL Server 15.0.2000 - DES
ms
ces
y
edures
lued Functions
alued Functions
te Functions
Functions
ggers
90 %
Messages

```
-- ================================================
-- Template generated from Template Explorer using:
-- Create Inline Function (New Menu).SQL
--
-- Use the Specify Values for Template Parameters
-- command (Ctrl-Shift-M) to fill in the parameter
-- values below.
--
-- This block of comments will not be included in
-- the definition of the function.
-- ================================================
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
-- =============================================
-- Author:      <Group 11 CPE21S3>
-- Create date: <November 25, 2022>
-- Description: <Hands-On 12 User Defined Functions Step 4>
-- =============================================
CREATE FUNCTION FunctionDisplayTruckBaseManagerAndType
(
    -- Add the parameters for the function here

)
RETURNS TABLE
AS
RETURN
(
    -- Add the SELECT statement with parameter references here
    SELECT a.TRUCK_NUM, b.BASE_CITY, b.BASE_MANAGER, c.TYPE_DESCRIPTION
    FROM TRUCK a
    INNER JOIN BASE b ON a.BASE_CODE = b.BASE_CODE
    INNER JOIN TYPE c ON a.TYPE_CODE = c.TYPE_CODE

)
GO
```

```
Commands completed successfully.

Completion time: 2022-11-25T15:07:18.9922680+08:00
```

Function execution


Supplementary Ste...ryan Mendoza (53))
SQLQuery5.sql - DE...ryan Mendoza (55))*
0.2000 - DES
90 %
Results
Messages

```
USE TRUCKING
GO

SELECT * FROM dbo.FunctionDisplayTruckBaseManagerAndType()
```

	TRUCK_NUM	BASE_CITY	BASE_MANAGER	TYPE_DESCRIPTION
1	1001	Murfreesboro	Andrea D. Gallager	Single box, double-axle
2	1002	Lexington	George H. Delarosa	Single box, double-axle
3	1003	Murfreesboro	Andrea D. Gallager	Single box, single-axle
4	1005	Cape Girardeau	Maria J. Talndo	Single box, single-axle
5	1006	Murfreesboro	Andrea D. Gallager	Single box, single-axle
6	1007	Lexington	George H. Delarosa	Tandem trailer, single-axle
7	1008	Cape Girardeau	Maria J. Talndo	Tandem trailer, single-axle
8	1009	Cape Girardeau	Maria J. Talndo	Single box, single-axle

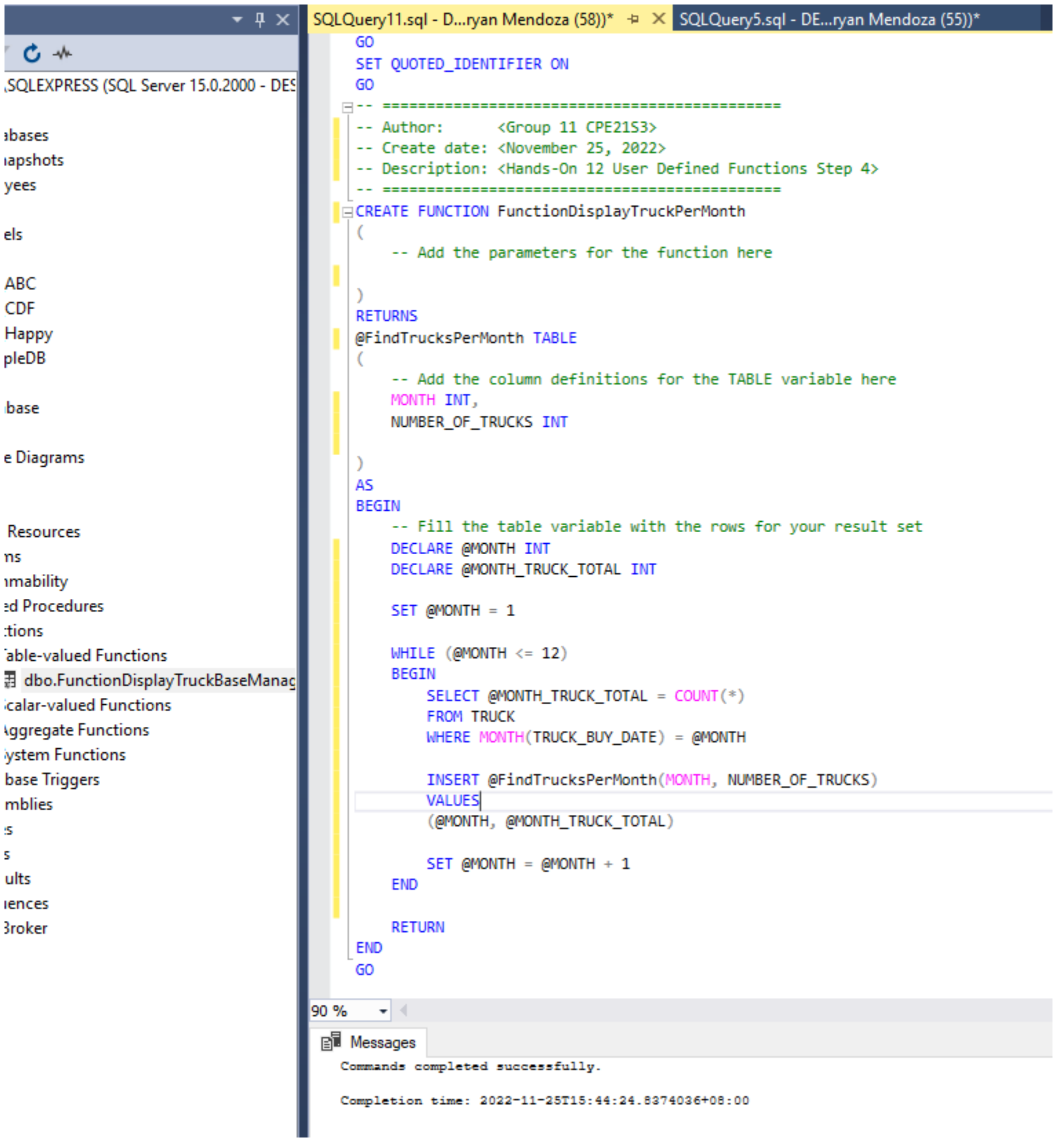
Observation:
On this procedure we are asked to display the TRUCK_NUM, BASE_CITY, BASE_MANAGER, and TYPE_DESCRIPTION of the database. We used Table Valued Function in order to make a function since this returns a table instead of a scalar value compared with the previous procedure.

Table Valued Function may or may not take a parameter, on this procedure, there were no requirements of a parameter therefore, we set it to non parameterized function.

On a new query, we called the function and executed it without passing a parameter, and we successfully displayed the table with only the desired columns.

5. Create and execute a function that displays the total number of trucks purchased per month. Display the month and the total number of trucks. Arrange the list from highest to lowest number of truck.

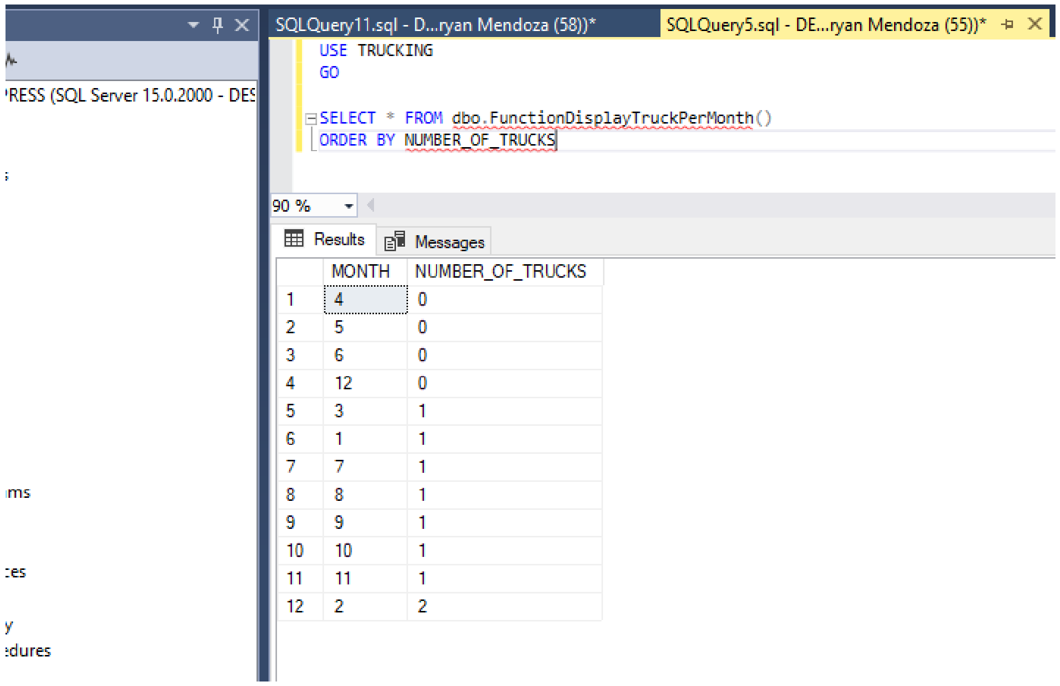
Creation of the Function



```
GO
SET QUOTED_IDENTIFIER ON
GO
-- =============================================
-- Author:      <Group 11 CPE2153>
-- Create date: <November 25, 2022>
-- Description: <Hands-On 12 User Defined Functions Step 4>
-- =============================================
CREATE FUNCTION FunctionDisplayTruckPerMonth
(
    -- Add the parameters for the function here

)
RETURNS
@FindTrucksPerMonth TABLE
(
    -- Add the column definitions for the TABLE variable here
    MONTH INT,
    NUMBER_OF_TRUCKS INT

)
AS
BEGIN
    -- Fill the table variable with the rows for your result set
    DECLARE @MONTH INT
    DECLARE @MONTH_TRUCK_TOTAL INT

    SET @MONTH = 1

    WHILE (@MONTH <= 12)
    BEGIN
        SELECT @MONTH_TRUCK_TOTAL = COUNT(*)
        FROM TRUCK
        WHERE MONTH(TRUCK_BUY_DATE) = @MONTH

        INSERT @FindTrucksPerMonth(MONTH, NUMBER_OF_TRUCKS)
        VALUES
        (@MONTH, @MONTH_TRUCK_TOTAL)

        SET @MONTH = @MONTH + 1
    END

    RETURN
END
GO
```

```
Commands completed successfully.

Completion time: 2022-11-25T15:44:24.8374036+08:00
```

Execution of the Function


SQLQuery11.sql - D...ryan Mendoza (58))*
SQLQuery5.sql - DE...ryan Mendoza (55))*
'RESS (SQL Server 15.0.2000 - DES
90 %
Results
Messages

```
USE TRUCKING
GO

SELECT * FROM dbo.FunctionDisplayTruckPerMonth()
ORDER BY NUMBER_OF_TRUCKS
```

	MONTH	NUMBER_OF_TRUCKS
1	4	0
2	5	0
3	6	0
4	12	0
5	3	1
6	1	1
7	7	1
8	8	1
9	9	1
10	10	1
11	11	1
12	2	2

Observation:
In this procedure, we used Table-Valued Function since this procedure requires us to return a table having the Number of Trucks purchased on each Month.

To accomplish this, we first declared the columns or attributes that we would be having on the new table. We initially set the MONTH = 1 to indicate January, then we used a WHILE loop in order to count the number of trucks purchased from MONTH 1 to MONTH 12. While iterating, we inserted the obtained values one by one.

On the execution of the function, we added an ORDER BY statement to accomplish Displaying the new table based on the NUMBER_OF_TRUCKS descendingly.

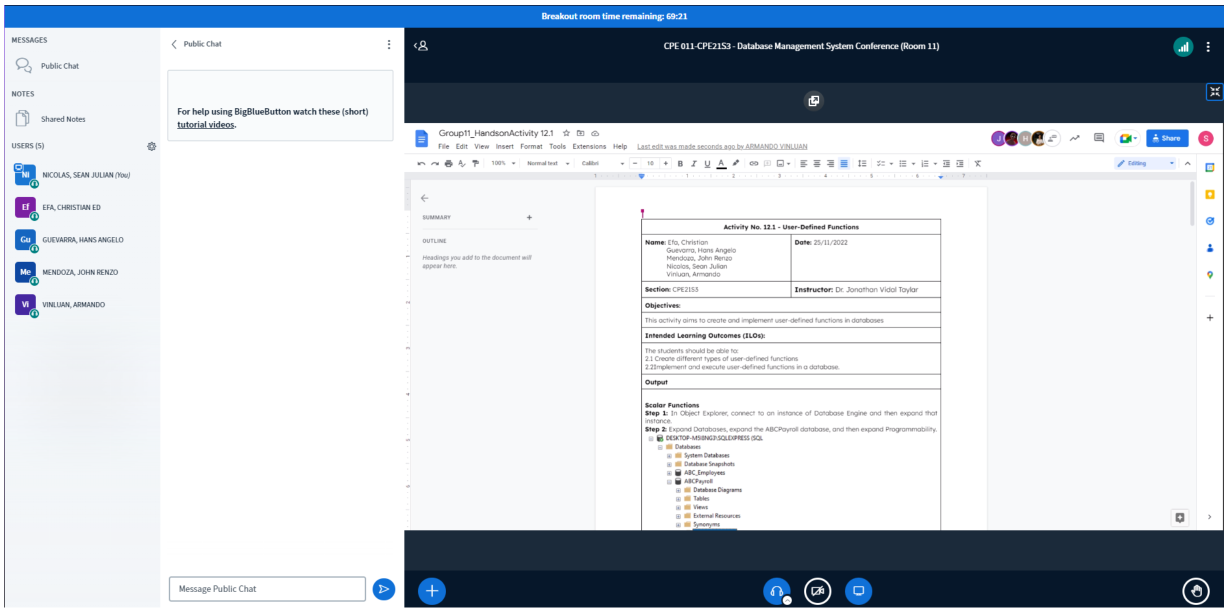
Conclusion

The group was able to create various types of user defined functions and implement them in a database during this activity. User-defined functions, like functions in other coding languages, accept parameters, perform actions, and return a value.

Moreover, there are two types of functions introduced in this laboratory activity, the scalar-valued and table-valued function. Scalar function returns a single data type wherein it could be an integer, varchar, text, and etc. On the contrary, table-valued function returns a table wherein it could be inline with the existing tables, or a new table created within a function.

In addition, this is useful whenever we need a function to perform a specific process for us and return the desired value. It is also important because we can use user-defined functions if we need to repeat calculations.

Proof of Collaboration


Breakout room time remaining: 69:21
MESSAGES
Public Chat
NOTES
Shared Notes
USERS (5)
Public Chat
CPE 011-CPE21S3 - Database Management System Conference (Room 11)
Group11_HandsonActivity 12.1
File Edit View Insert Format Tools Extensions Help
Last edit was made seconds ago by ARMANDO VINLUAN
Share
Editing
SUMMARY
OUTLINE
Headings you add to the document will appear here.
Message Public Chat

For help using BigBlueButton watch these (short) tutorial videos.

NICOLAS, SEAN JULIAN (You)
EFA, CHRISTIAN ED
GUEVARRA, HANS ANGELO
MENDOZA, JOHN RENZO
VINLUAN, ARMANDO

Activity No. 12.1 - User-Defined Functions	
Name: Efa, Christian Guevarra, Hans Angelo Mendoza, John Renzo Nicolas, Sean Julian Vinluan, Armando	**Date:** 25/11/2022
Section: CPE21S3	**Instructor:** Dr. Jonathan Vidal Taylar
Objectives:	
This activity aims to create and implement user-defined functions in databases	
Intended Learning Outcomes (ILOs):	
The students should be able to: 2.1 Create different types of user-defined functions 2.2Implement and execute user-defined functions in a database.	
Output	

Scalar Functions
Step 1: In Object Explorer, connect to an instance of Database Engine and then expand that instance.
Step 2: Expand Databases, expand the ABCPayroll database, and then expand Programmability.

```
DESKTOP-M5I8NG3\SQLEXPRESS (SQL
  Databases
    System Databases
    Database Snapshots
    ABC_Employees
    ABCPayroll
      Database Diagrams
      Tables
      Views
      External Resources
      Synonyms
```

Honor Pledge

"I accept responsibility for my role in ensuring the integrity of the work submitted by the group in which I participated."