

INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO



Data Base S Elected Topics

3CV14

BOCANEGRA HEZQUIO

YESTLANEZI

2019340090

ACTIVIDAD 5: FUNCIONES SQL



PROFESORA: GALEANA CHAVEZ ING. MARIA DEL ROSARIO

FECHA DE ENTREGA: 03/11/2023

índice

Ejercicio 1.....	2
Función.....	2
Ejecución.....	3
Ejercicio 2.....	3
Función.....	3
Ejecución.....	4
Ejercicio 3.....	5
Función.....	5
Ejecución.....	5
Ejercicio 4.....	6
Función.....	6
Ejecución.....	7
Ejercicio 5.....	7
Función.....	7
Ejecución.....	8
Ejercicio 6.....	8
Función.....	9
Ejecución.....	10
Ejercicio 7.....	11
Función.....	11
Ejecución.....	12
Ejercicio 8.....	12
Función.....	12
Ejecución.....	13
Caso 1	13
Caso 2	13
Ejercicio 9.....	14
Función.....	14
Ejecución.....	15
Caso 1	15
Caso 2	15



Ejercicio 1

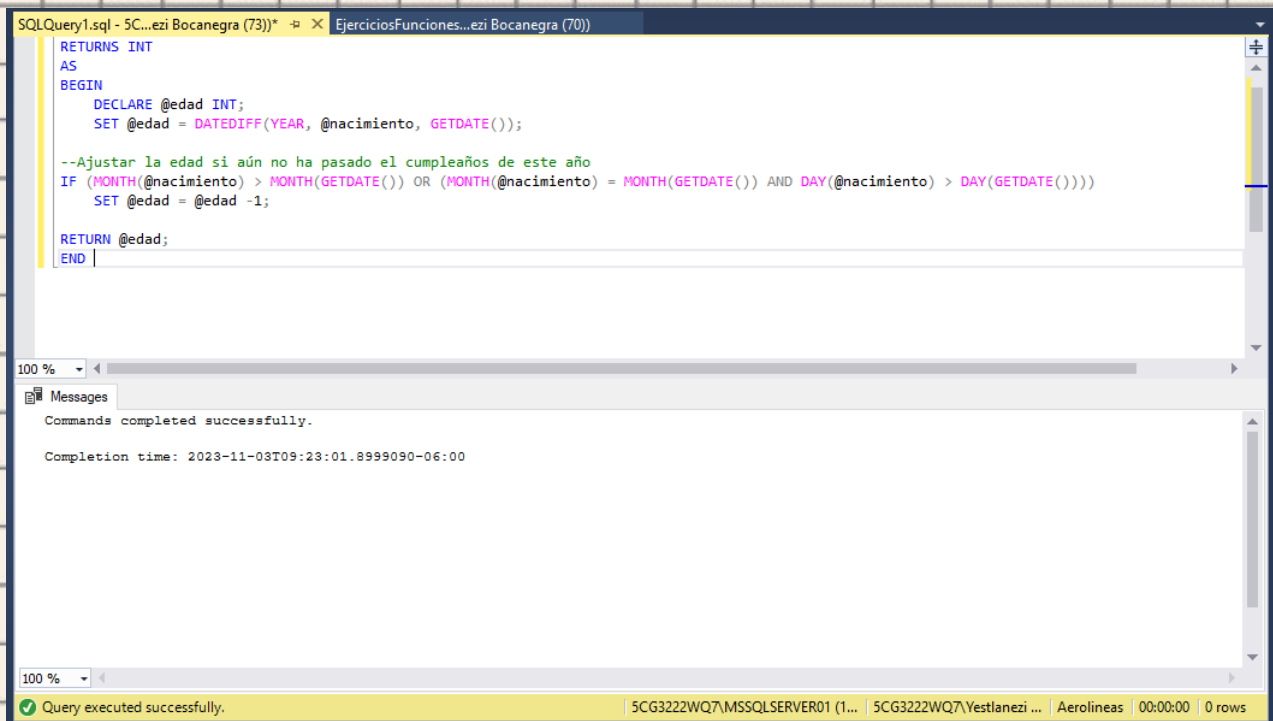
Crear una función que reciba como parámetro la fecha de nacimiento y devuelva la edad al día de hoy.

Funcion

```
CREATE FUNCTION CalcularEdad2(@nacimiento DATE)
RETURNS INT
AS
BEGIN
    DECLARE @edad INT;
    SET @edad = DATEDIFF(YEAR, @nacimiento, GETDATE());

    --Ajustar la edad si aún no ha pasado el cumpleaños de este año
    IF (MONTH(@nacimiento) > MONTH(GETDATE()) OR (MONTH(@nacimiento) = MONTH(GETDATE())
    AND DAY(@nacimiento) > DAY(GETDATE()))
        SET @edad = @edad -1;

    RETURN @edad;
END
```

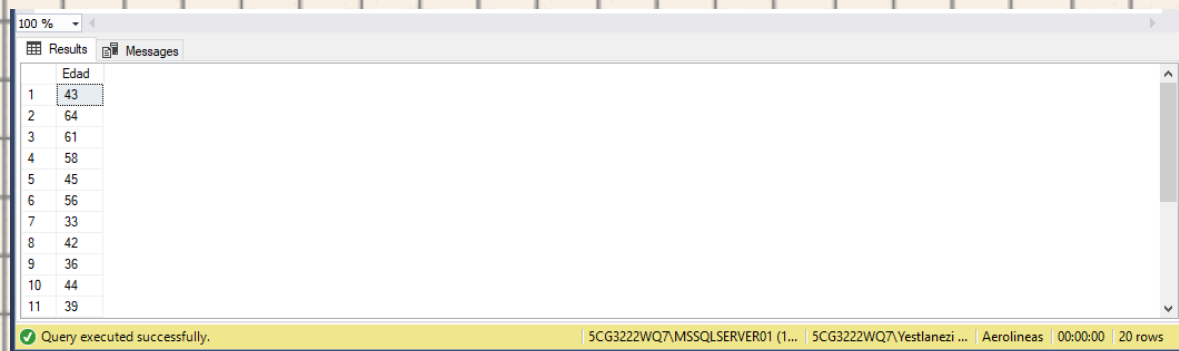


The screenshot shows the SQL Server Enterprise Manager interface. The top pane displays the SQL code for the 'CalcularEdad2' function, which is identical to the code provided in the previous block. The bottom pane shows the 'Messages' window with the following text: 'Commands completed successfully.' and 'Completion time: 2023-11-03T09:23:01.8999090-06:00'. The status bar at the bottom indicates 'Query executed successfully.' and shows the server name '5CG3222WQ7.MSSQLSERVER01 (1...' and the execution time '00:00:00 | 0 rows'.



Ejecucion

```
SELECT dbo.CalcularEdad2(nacimiento) AS Edad  
FROM Persona;
```



The screenshot shows a SQL Server query window with the 'Results' tab selected. The query results are displayed in a table with one column named 'Edad'. The table contains 11 rows of data. The status bar at the bottom indicates 'Query executed successfully.' and '20 rows'.

	Edad
1	43
2	64
3	61
4	58
5	45
6	56
7	33
8	42
9	36
10	44
11	39

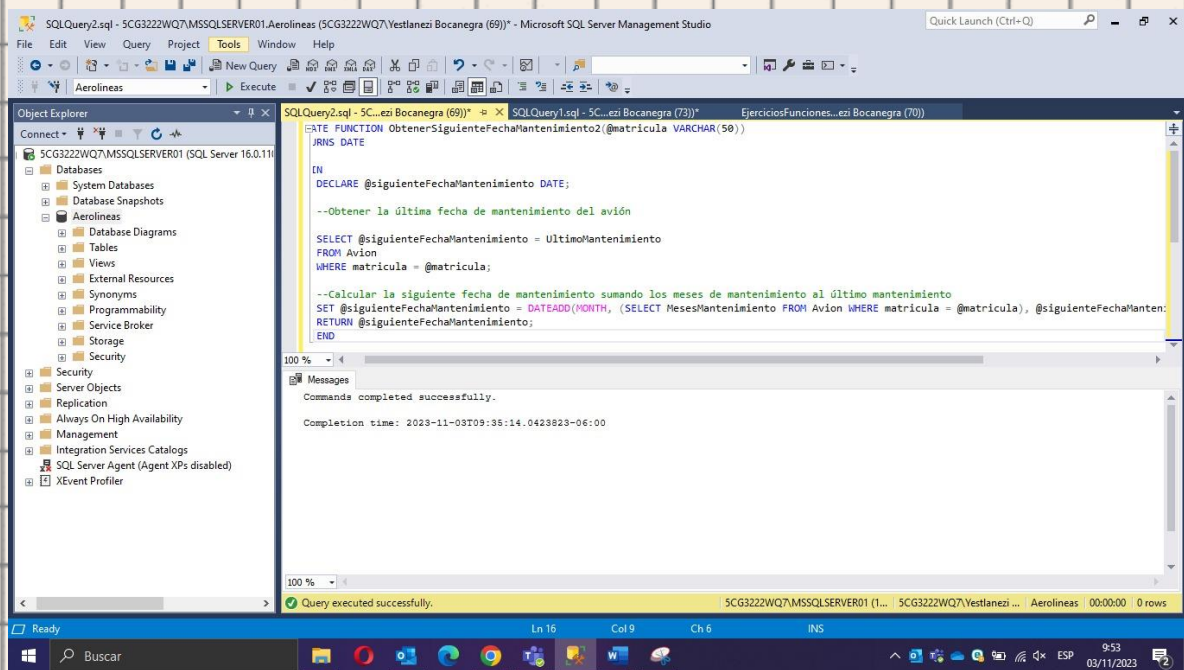
Ejercicio 2

Crear una función que reciba la matrícula de un avión y devuelva la siguiente fecha de mantenimiento.

Funcion

```
CREATE FUNCTION ObtenerSiguienteFechaMantenimiento2(@matricula VARCHAR(50))  
RETURNS DATE  
AS  
BEGIN  
    DECLARE @siguienteFechaMantenimiento DATE;  
  
    --Obtener la última fecha de mantenimiento del avión  
  
    SELECT @siguienteFechaMantenimiento = UltimoMantenimiento  
    FROM Avion  
    WHERE matricula = @matricula;  
  
    --Calcular la siguiente fecha de mantenimiento sumando los meses de  
    mantenimiento al último mantenimiento  
    SET @siguienteFechaMantenimiento = DATEADD(MONTH, (SELECT MesesMantenimiento  
    FROM Avion WHERE matricula = @matricula), @siguienteFechaMantenimiento);  
    RETURN @siguienteFechaMantenimiento;  
END
```





Ejecucion

```

SELECT dbo.ObtenerSiguienteFechaMantenimiento2(matricula) AS
SiguienteFechaMantenimiento
FROM Avion;

```

The screenshot shows the Results pane of the SQL Server Management Studio. The query has been executed successfully, and the results are displayed in a table with 9 rows. The table has a single column named 'SiguienteFechaMantenimiento'.

	SiguienteFechaMantenimiento
1	2014-04-01
2	2013-07-15
3	2014-07-06
4	2015-06-10
5	2015-01-01
6	2013-04-01
7	2014-07-23
8	2014-06-30
9	2015-02-03

The status bar at the bottom indicates: 'Query executed successfully. 5CG3222WQ7\MSSQLSERVER01 (1... 5CG3222WQ7\Yestlanezi ... Aerolineas 00:00:00 9 rows'.



Ejercicio 3

Crear una función que reciba una fecha y muestre la siguiente información del vuelo más cercano: matrícula del avión, nombre del vendedor, nombre del cliente, fecha de vuelo y precio del asiento.

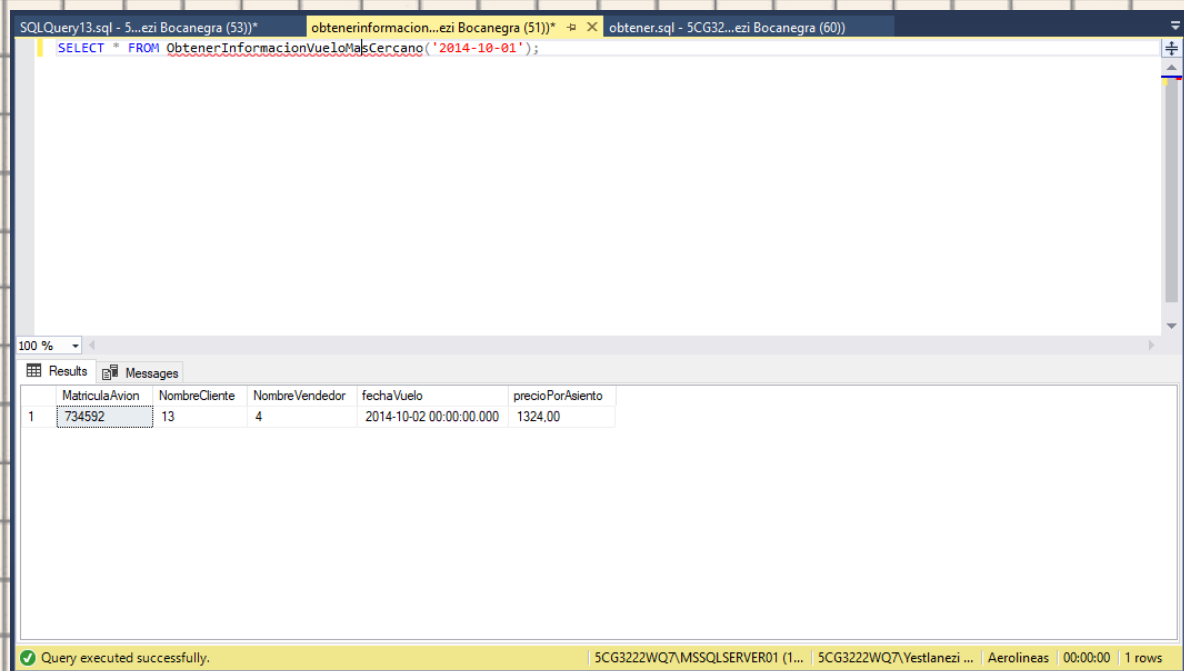
Funcion

```
CREATE FUNCTION ObtenerInformacionVueloMasCercano(@fecha DATE)

RETURNS TABLE
AS
RETURN
(
    SELECT TOP 1 Avion.matricula AS MatriculaAvion, Cliente.clienteId AS
NombreCliente, Vendedor.vendedorId AS NombreVendedor, Vuelo.fechaVuelo,
Vuelo.precioPorAsiento
    FROM Vuelo
    INNER JOIN Avion ON Vuelo.matriculaAvion = Avion.matricula
    INNER JOIN Venta ON Vuelo.idVuelo = Venta.idVuelo
    INNER JOIN Vendedor ON Venta.idVendedor = Vendedor.vendedorId
    INNER JOIN Cliente ON Venta.idCliente = Cliente.clienteId
    WHERE Vuelo.fechaVuelo >= @fecha
    ORDER BY Vuelo.fechaVuelo ASC
)
```

Ejecucion

```
SELECT * FROM ObtenerInformacionVueloMasCercano('2014-10-01');
```



	MatriculaAvion	NombreCliente	NombreVendedor	fechaVuelo	precioPorAsiento
1	734592	13	4	2014-10-02 00:00:00.000	1324.00

Query executed successfully. 5CG3222WQ7\MSSQLSERVER01 (1... 5CG3222WQ7\Yestlanezi ... Aerolineas 00:00:00 1 rows

Ejercicio 4

Crear una función que reciba el id de un cliente y el id de un vuelo y regrese el precio del vuelo considerando el descuento por tipo de cliente.

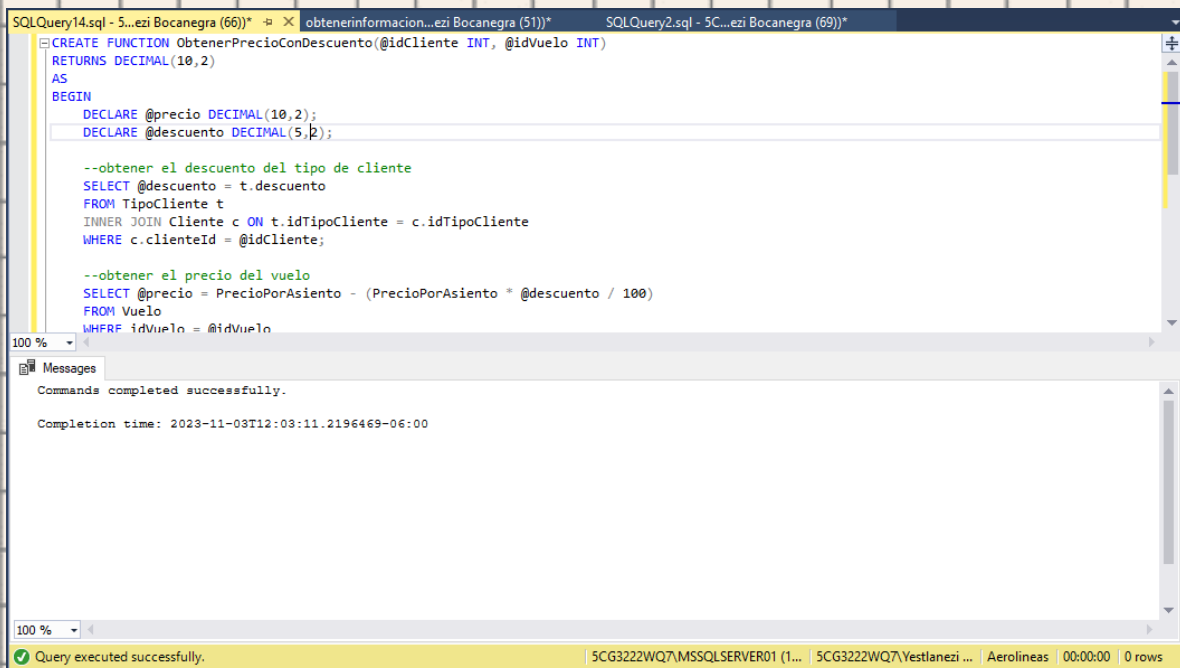
Funcion

```
CREATE FUNCTION ObtenerPrecioConDescuento(@idCliente INT, @idVuelo INT)
RETURNS DECIMAL(10,2)
AS
BEGIN
    DECLARE @precio DECIMAL(10,2);
    DECLARE @descuento DECIMAL(5,2);

    --obtener el descuento del tipo de cliente
    SELECT @descuento = t.descuento
    FROM TipoCliente t
    INNER JOIN Cliente c ON t.idTipoCliente = c.idTipoCliente
    WHERE c.clienteId = @idCliente;

    --obtener el precio del vuelo
    SELECT @precio = PrecioPorAsiento - (PrecioPorAsiento * @descuento / 100)
    FROM Vuelo
    WHERE idVuelo = @idVuelo

    RETURN @precio;
END
```



The screenshot displays the SQL Server Enterprise Manager interface. The main window shows the SQL script for creating the function 'ObtenerPrecioConDescuento'. The script is as follows:

```
CREATE FUNCTION ObtenerPrecioConDescuento(@idCliente INT, @idVuelo INT)
RETURNS DECIMAL(10,2)
AS
BEGIN
    DECLARE @precio DECIMAL(10,2);
    DECLARE @descuento DECIMAL(5,2);

    --obtener el descuento del tipo de cliente
    SELECT @descuento = t.descuento
    FROM TipoCliente t
    INNER JOIN Cliente c ON t.idTipoCliente = c.idTipoCliente
    WHERE c.clienteId = @idCliente;

    --obtener el precio del vuelo
    SELECT @precio = PrecioPorAsiento - (PrecioPorAsiento * @descuento / 100)
    FROM Vuelo
    WHERE idVuelo = @idVuelo;

    RETURN @precio;
END
```

The Messages pane at the bottom shows the following output:

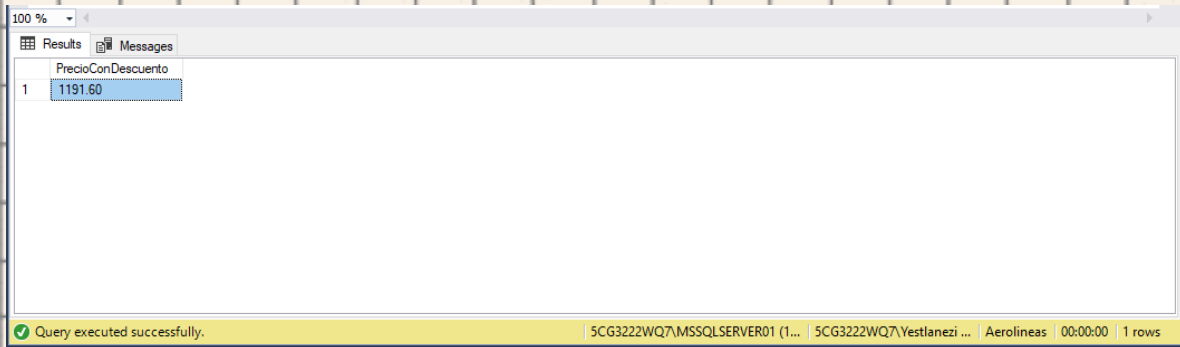
```
Commands completed successfully.
Completion time: 2023-11-03T12:03:11.2196469-06:00
```

The status bar at the bottom indicates: 'Query executed successfully. | SCG3222WQ7\MSSQLSERVER01 (1... | SCG3222WQ7\Yestlanezi ... | Aerolineas | 00:00:00 | 0 rows

Ejecucion

```
DECLARE @clienteId INT = 1;  
DECLARE @vueloId INT = 100;
```

```
SELECT dbo.ObtenerPrecioConDescuento(15,20) AS PrecioConDescuento;
```



	PrecioConDescuento
1	1191.60

Query executed successfully. | 5CG3222WQ7\MSSQLSERVER01 (1... | 5CG3222WQ7\Vestlanezi ... | Aerolineas | 00:00:00 | 1 rows

Ejercicio 5

Crear una función que regrese el RFC de todas las personas de la base de datos o se especifique algún IdPersona para solo obtener el de una.

Funcion

```
CREATE FUNCTION ObtenerNombrePersonas(@idPersona INT = NULL)  
  
RETURNS TABLE  
AS  
RETURN  
(  
    SELECT nombre  
    FROM Persona  
    WHERE @idPersona IS NULL OR IdPersona = @idPersona  
)
```




```
SQLQuery15.sql - 5...ezi Bocanegra (65)))* X obtenerinformacion...ezi Bocanegra (51))* SQLQuery14.sql - 5...ezi Bocanegra (66))*
CREATE FUNCTION ObtenerNombrePersonas(@idPersona INT = NULL)
RETURNS TABLE
AS
RETURN
(
    SELECT nombre
    FROM Persona
    WHERE @idPersona IS NULL OR IdPersona = @idPersona
)
```

100 %

Messages

Commands completed successfully.

Completion time: 2023-11-03T12:16:21.4204926-06:00

100 %

Query executed successfully. | 5CG3222WQ7\MSSQLSERVER01 (1... | 5CG3222WQ7\Vestlanezi ... | Aerolineas | 00:00:00 | 0 rows

Ejecucion

```
DECLARE @idPersona INT
SELECT nombre
FROM dbo.ObtenerNombresPersonas(@idPersona);
```

100 %

Results Messages

	nombre
1	Edgardo
2	Edith
3	Javier
4	Eduardo
5	Efrain
6	Beatriz
7	Adrian
8	Eleonora
9	Marco
10	Elsa
11	Elizabeth

Query executed successfully. | 5CG3222WQ7\MSSQLSERVER01 (1... | 5CG3222WQ7\Vestlanezi ... | Aerolineas | 00:00:00 | 20 rows



Ejercicio 6

Crear una función que reciba como parámetros el Id de un cliente, y una fecha inicio y fin. Como salida debe indicar cuantas veces ha viajado.

Funcion

```
CREATE FUNCTION ContarViajesCliente(  
    @idCliente INT,  
    @fechaInicio DATE,  
    @fechaFin DATE  
)  
  
RETURNS INT  
AS  
BEGIN  
    DECLARE @cantidadViajes INT;  
  
    SELECT @cantidadViajes = COUNT(*)  
    FROM Venta  
    INNER JOIN Vuelo ON Venta.idVuelo = Vuelo.idVuelo  
    WHERE Venta.idCliente = @idCliente  
    AND Vuelo.fechaVuelo >= @fechaInicio  
    AND Vuelo.fechaVuelo <= @fechaFin;  
  
    RETURN @cantidadViajes;  
END
```



```
SQLQuery1.sql - DES...-E69JFFI\yestl (57)) * X
CREATE FUNCTION ContarViajesCliente(
    @idCliente INT,
    @fechaInicio DATE,
    @fechaFin DATE
)
RETURNS INT
AS
BEGIN
    DECLARE @cantidadViajes INT;

    SELECT @cantidadViajes = COUNT(*)
    FROM Venta
    INNER JOIN Vuelo ON Venta.idVuelo = Vuelo.idVuelo
    WHERE Venta.idCliente = @idCliente
    AND Vuelo.fechaVuelo >= @fechaInicio
    AND Vuelo.fechaVuelo <= @fechaFin;
END
```

100 %

Messages

Commands completed successfully.

Completion time: 2023-11-03T20:45:03.4578127-06:00

100 %

Query executed successfully. DESKTOP-E69JFFI (16.0 RTM) DESKTOP-E69JFFI\yestl ... Aerolineas 00:00:00 0 rows

Ejecucion

```
SELECT dbo.ContarViajesCliente(15, '2014-09-11', '2014-09-11') AS CantidadViaje;
```

100 %

Results Messages

	CantidadViaje
1	1

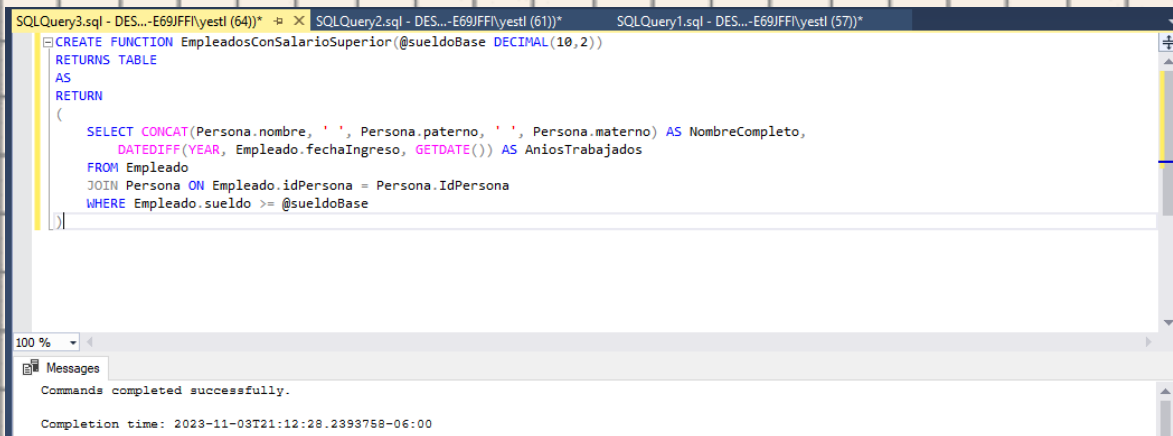


Ejercicio 7

Crear una función que reciba como parámetro un sueldo base y que muestre el nombre completo de los empleados que tengan un salario mayor o igual al parámetro de entrada de mostrar la cantidad de años que lleva el empleado desde su fecha de ingreso.

Funcion

```
CREATE FUNCTION EmpleadosConSalarioSuperior(@sueldoBase DECIMAL(10,2))
RETURNS TABLE
AS
RETURN
(
    SELECT CONCAT(Persona.nombre, ' ', Persona.paterno, ' ', Persona.materno) AS
    NombreCompleto,
           DATEDIFF(YEAR, Empleado.fechaIngreso, GETDATE()) AS AniosTrabajados
    FROM Empleado
    JOIN Persona ON Empleado.idPersona = Persona.IdPersona
    WHERE Empleado.sueldo >= @sueldoBase
)
```



The screenshot shows a SQL Server Enterprise Manager window with three tabs: 'SQLQuery3.sql - DES...-E69JFFI\yestl (64)', 'SQLQuery2.sql - DES...-E69JFFI\yestl (61)', and 'SQLQuery1.sql - DES...-E69JFFI\yestl (57)'. The active tab, 'SQLQuery3.sql', contains the SQL script for creating the function 'EmpleadosConSalarioSuperior'. The script is as follows:

```
CREATE FUNCTION EmpleadosConSalarioSuperior(@sueldoBase DECIMAL(10,2))
RETURNS TABLE
AS
RETURN
(
    SELECT CONCAT(Persona.nombre, ' ', Persona.paterno, ' ', Persona.materno) AS NombreCompleto,
           DATEDIFF(YEAR, Empleado.fechaIngreso, GETDATE()) AS AniosTrabajados
    FROM Empleado
    JOIN Persona ON Empleado.idPersona = Persona.IdPersona
    WHERE Empleado.sueldo >= @sueldoBase
)
```

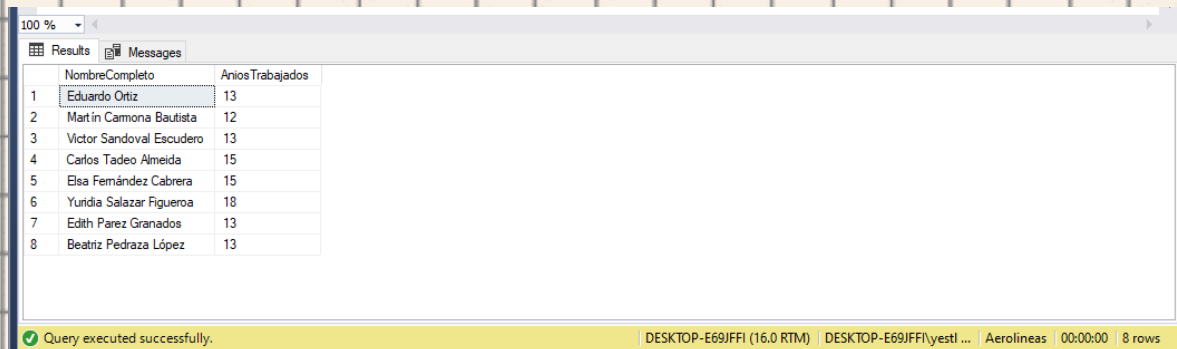
Below the script, the 'Messages' pane shows the following output:

```
Commands completed successfully.
Completion time: 2023-11-03T21:12:28.2393758-06:00
```



Ejecucion

```
SELECT *  
FROM EmpleadosConSalarioSuperior(5000)
```



The screenshot shows a SQL Server query window with the results of the query 'SELECT * FROM EmpleadosConSalarioSuperior(5000)'. The results are displayed in a table with two columns: 'NombreCompleto' and 'AniosTrabajados'. There are 8 rows of data. The status bar at the bottom indicates 'Query executed successfully.' and '8 rows'.

	NombreCompleto	AniosTrabajados
1	Eduardo Ortiz	13
2	Martin Cammona Bautista	12
3	Victor Sandoval Escudero	13
4	Carlos Tadeo Almeida	15
5	Elsa Fernández Cabrera	15
6	Yuridia Salazar Figueroa	18
7	Edith Perez Granados	13
8	Beatriz Pedraza López	13

Ejercicio 8

Modificar la función ObtenerDireccionEmpleado, en caso de que el Id proporcionado no tenga jefe, mostrar en la columna direccion completa la siguiente leyenda "información confidencial".

Funcion

```
CREATE FUNCTION ObtenerDireccionEmpleado(@IdEmpleado INT)  
RETURNS @tbEmpleadoDireccion TABLE  
(  
    IdEmpleado INT,  
    nombreCompleto VARCHAR(100),  
    direccioncompleta varchar(100)  
)  
AS  
BEGIN  
    INSERT INTO @tbEmpleadoDireccion  
    SELECT a.idEmpleado,  
           nombre + ' ' + paterno + ' ' + ISNULL(materno, ''),  
           CASE  
               WHEN a.idJefe IS NULL THEN 'Información confidencial'  
               ELSE calle + ' ' + numero + ' ' + colonia + ' ' + codigoPostal  
           END  
    FROM Empleado a  
    INNER JOIN persona b ON a . idPersona = b.IdPersona  
    LEFT JOIN Direccion c ON b.direccionId = c.direccionId  
    WHERE a.idEmpleado = @IdEmpleado;  
    RETURN;  
END;
```

```

SQLQuery6.sql - DES...-E69JFFI\yestl (62))*  SQLQuery5.sql - DES...-E69JFFI\yestl (59))*  SQLQuery4.sql - DES...-E69JFFI\yestl (56))*
CREATE FUNCTION ObtenerDireccionEmpleado(@IdEmpleado INT)
RETURNS @tbEmpleadoDireccion TABLE
(
    IdEmpleado INT,
    nombreCompleto CHAR(100),
    direccionCompleta varchar(100)
)
AS
BEGIN
    INSERT INTO @tbEmpleadoDireccion
    SELECT a.idEmpleado,
        nombre + ' ' + paterno + ' ' + ISNULL(materno, ''),
        CASE
            WHEN a.idJefe IS NULL THEN 'Información confidencial'
            ELSE calle + ' ' + numero + ' ' + colonia + ' ' + codigoPostal
        END
    FROM Empleado a
    INNER JOIN persona b ON a . idPersona = b.IdPersona
    LEFT JOIN Direccion c ON b.direccionId = c.direccionId
    WHERE a.idEmpleado = @IdEmpleado;
    RETURN;
END;

```

100 %

Messages

Commands completed successfully.

Completion time: 2023-11-03T21:38:34.5485684-06:00

100 %

Query executed successfully. DESKTOP-E69JFFI (16.0 RTM) DESKTOP-E69JFFI\yestl ... Aerolíneas 00:00:00 0 rows

Ejecucion

```

SELECT *
FROM ObtenerDireccionEmpleado(10)

```

Caso 1

100 %

Results Messages

	IdEmpleado	nombreCompleto	direccionCompleta
1	10	Beatriz Pedraza López	Av. 20 de Noviembre 2 Centro 06060

Caso 2

```

SELECT *
FROM ObtenerDireccionEmpleado(11)

```

Results Messages

	IdEmpleado	nombreCompleto	direccionCompleta
	11	Juan Pérez García	Información confidencial

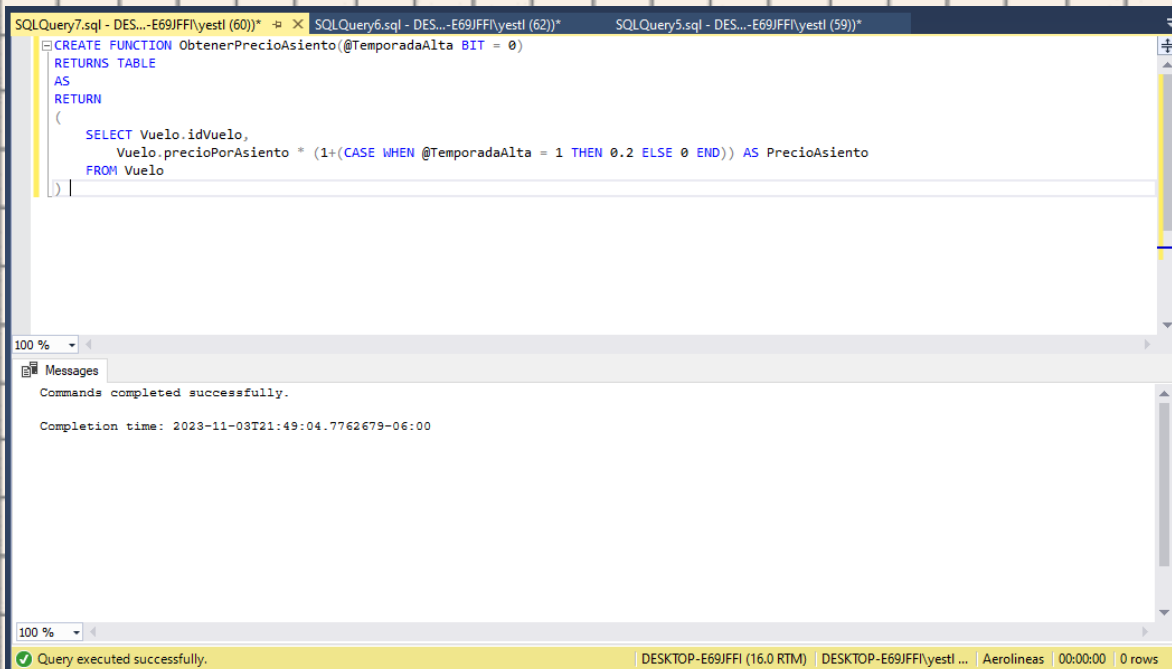


Ejercicio 9

Crear una función que devuelva el precio por asiento de todos los vuelos y que reciba un parámetro opcional que indique si es temporada alta, en su caso aumentar 20% el precio por asiento.

Funcion

```
CREATE FUNCTION ObtenerPrecioAsiento(@TemporadaAlta BIT = 0)
RETURNS TABLE
AS
RETURN
(
    SELECT Vuelo.idVuelo,
           Vuelo.precioPorAsiento * (1+(CASE WHEN @TemporadaAlta = 1 THEN 0.2 ELSE 0 END)) AS PrecioAsiento
    FROM Vuelo
)
```



The screenshot shows the SQL Server Enterprise Manager interface. The top pane displays the SQL query used to create the function, which is identical to the one in the previous block. The bottom pane shows a message indicating that the commands were completed successfully. The completion time is 2023-11-03T21:49:04.7762679-06:00. The status bar at the bottom indicates that the query was executed successfully.

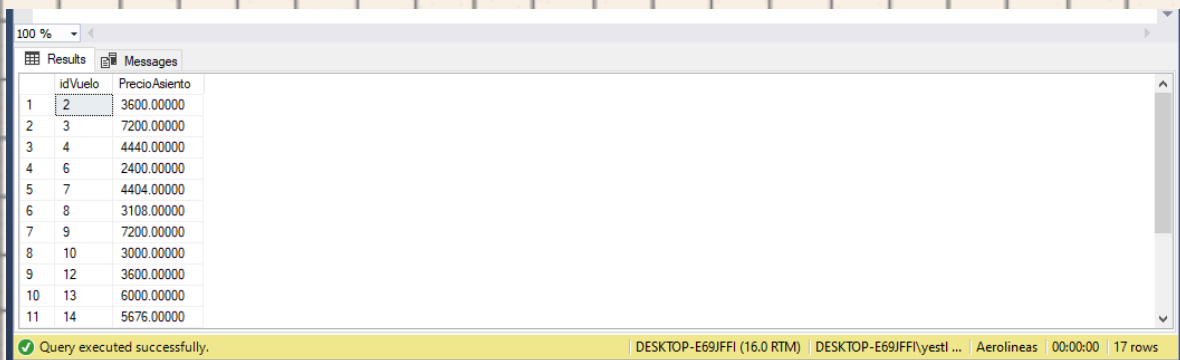
```
SQLQuery7.sql - DES...-E69JFFI\yestl (60))* SQLQuery6.sql - DES...-E69JFFI\yestl (62))* SQLQuery5.sql - DES...-E69JFFI\yestl (59))*
CREATE FUNCTION ObtenerPrecioAsiento(@TemporadaAlta BIT = 0)
RETURNS TABLE
AS
RETURN
(
    SELECT Vuelo.idVuelo,
           Vuelo.precioPorAsiento * (1+(CASE WHEN @TemporadaAlta = 1 THEN 0.2 ELSE 0 END)) AS PrecioAsiento
    FROM Vuelo
)

100 %
Messages
Commands completed successfully.
Completion time: 2023-11-03T21:49:04.7762679-06:00
100 %
Query executed successfully. DESKTOP-E69JFFI (16.0 RTM) DESKTOP-E69JFFI\yestl ... Aerolineas 00:00:00 0 rows
```

Ejecucion

Caso 1

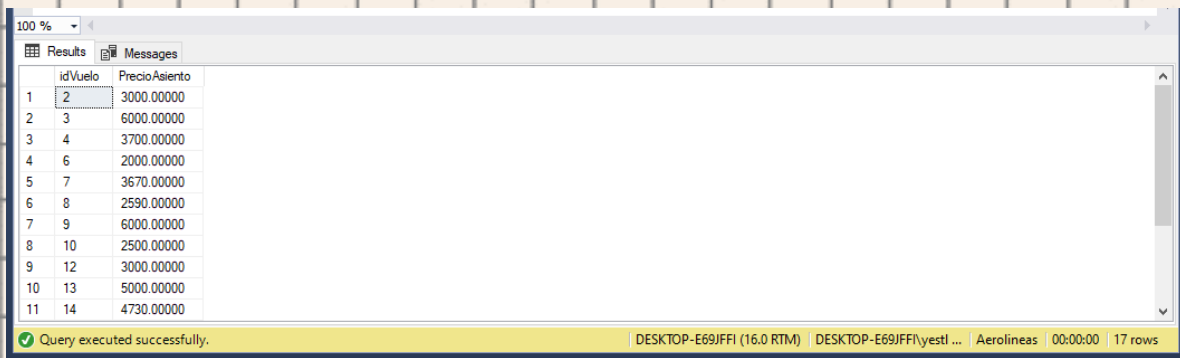
```
SELECT *  
FROM ObtenerPrecioAsiento(1)
```



	idVuelo	PrecioAsiento
1	2	3600.00000
2	3	7200.00000
3	4	4440.00000
4	6	2400.00000
5	7	4404.00000
6	8	3108.00000
7	9	7200.00000
8	10	3000.00000
9	12	3600.00000
10	13	6000.00000
11	14	5676.00000

Caso 2

```
SELECT *  
FROM ObtenerPrecioAsiento(0)
```



	idVuelo	PrecioAsiento
1	2	3000.00000
2	3	6000.00000
3	4	3700.00000
4	6	2000.00000
5	7	3670.00000
6	8	2590.00000
7	9	6000.00000
8	10	2500.00000
9	12	3000.00000
10	13	5000.00000
11	14	4730.00000