

Práctica 2. Procedimientos de almacenado.

Parte 1:

Una empresa almacena los datos de sus empleados en una tabla llamada "Empleado".

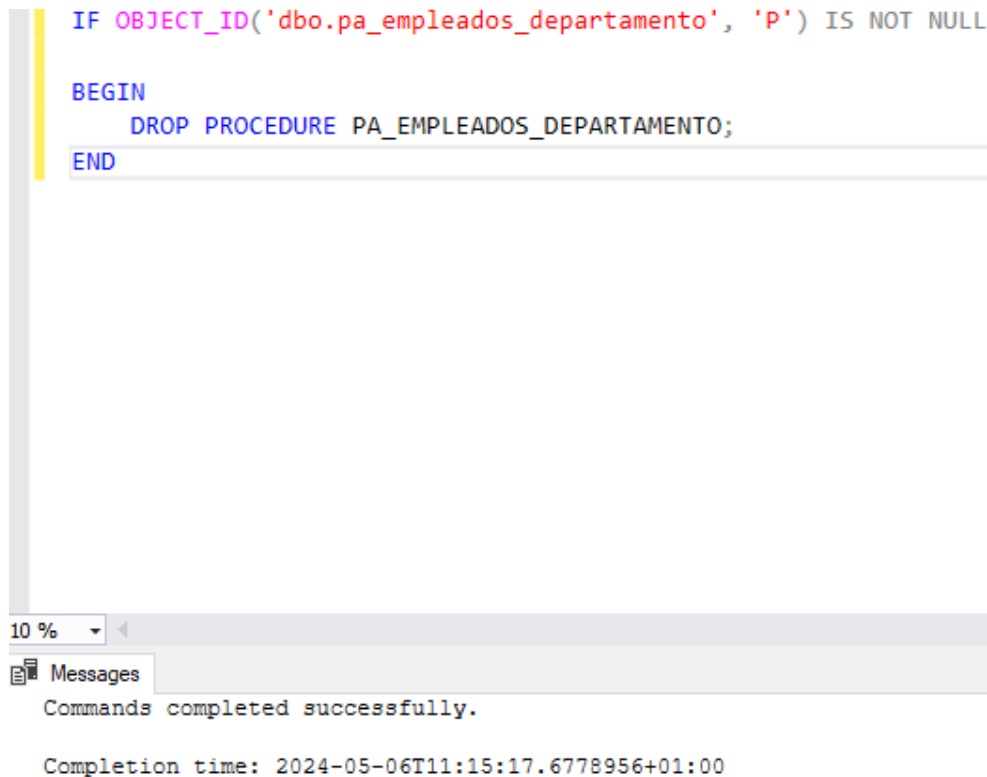
1- Elimine el procedimiento llamado "pa_empleados_departamento", si existe.

```
IF OBJECT_ID('dbo.pa_empleados_departamento', 'P') IS NOT NULL
```

```
BEGIN
```

```
DROP PROCEDURE PA_EMPLEADOS_DEPARTAMENTO;
```

```
END
```



```
IF OBJECT_ID('dbo.pa_empleados_departamento', 'P') IS NOT NULL

BEGIN
    DROP PROCEDURE PA_EMPLEADOS_DEPARTAMENTO;
END
```

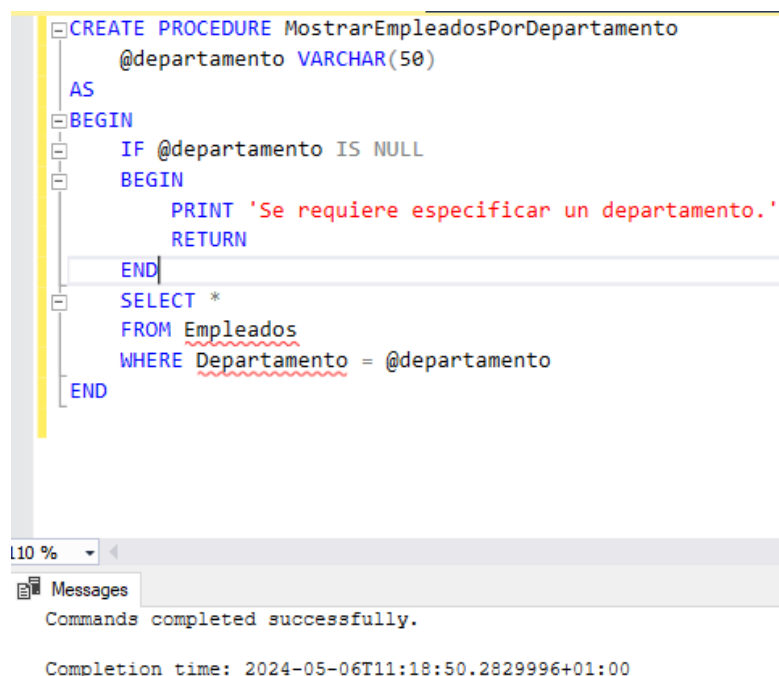
10 %

Messages

Commands completed successfully.

Completion time: 2024-05-06T11:15:17.6778956+01:00

2- Cree un procedimiento que muestre todos los empleados de un departamento determinado que se pase como parámetro. Si no se pone un valor, o se coloca "null", se muestra un mensaje y se sale del procedimiento.



```
CREATE PROCEDURE MostrarEmpleadosPorDepartamento
    @departamento VARCHAR(50)
AS
BEGIN
    IF @departamento IS NULL
    BEGIN
        PRINT 'Se requiere especificar un departamento.'
        RETURN
    END
    SELECT *
    FROM Empleados
    WHERE Departamento = @departamento
END
```

110 %

Messages

Commands completed successfully.

Completion time: 2024-05-06T11:18:50.2829996+01:00

```

CREATE PROCEDURE MostrarEmpleadosPorDepartamento
    @departamento VARCHAR(50)
AS
BEGIN
    IF @departamento IS NULL
    BEGIN
        PRINT 'Se requiere especificar un departamento.'
        RETURN END
    SELECT * FROM Empleados WHERE Departamento = @departamento
END

```

3- Ejecute el procedimiento enviándole un valor para el parámetro.

4- Ejecute el procedimiento sin parámetro.

```

exec MostrarEmpleadosPorDepartamento;

```

Msg 201, Level 16, State 4, Procedure MostrarEmpleadosPorDepartamento, Line 0 [Batch Start Line 0]
 El procedimiento o la función 'MostrarEmpleadosPorDepartamento' esperaba el parámetro '@departamento', que no se ha es

Completion time: 2024-05-06T11:20:33.1280840+01:00

exec MostrarEmpleadosPorDepartamento;

5- Elimine el procedimiento "pa_actualizarhijos", si existe.

```

IF OBJECT_ID('dbo.pa_actualizarhijos', 'P') IS NOT NULL
BEGIN
    DROP PROCEDURE PA_ACTUALIZARHIJOS;

```

END

```

IF OBJECT_ID('dbo.pa_actualizarhijos', 'P') IS NOT NULL
BEGIN
    DROP PROCEDURE PA_ACTUALIZARHIJOS;
END

```

Commands completed successfully.

Completion time: 2024-05-06T11:21:08.9061056+01:00

6- Cree un procedimiento almacenado que permita modificar la cantidad de hijos indicando el documento de un empleado y la nueva cantidad de hijos. Ambos parámetros DEBEN ponerse con un valor distinto de "null". El procedimiento retorna "1" si la actualización se realiza (si se insertan valores para ambos parámetros) y "0", en caso que uno o ambos parámetros no se insertan o son nulos.

```
CREATE PROCEDURE MODIFICARCANTIDADHIJOS @DOCUMENTO VARCHAR(20),  
@NUEVACANTIDADHIJOS INT AS  
BEGIN
```

```
    IF @DOCUMENTO IS NULL OR @NUEVACANTIDADHIJOS IS NULL  
    BEGIN
```

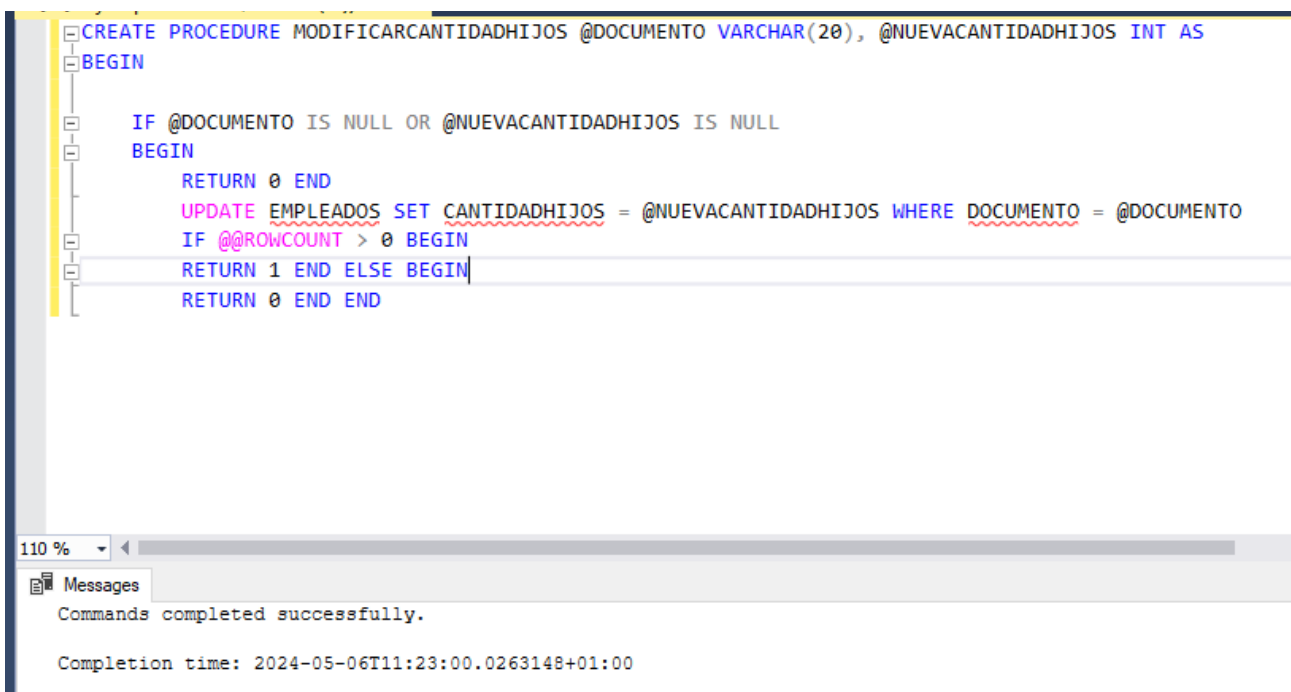
```
        RETURN 0 END
```

```
    UPDATE EMPLEADOS SET CANTIDADHIJOS = @NUEVACANTIDADHIJOS WHERE  
DOCUMENTO = @DOCUMENTO
```

```
    IF @@ROWCOUNT > 0 BEGIN
```

```
        RETURN 1 END ELSE BEGIN
```

```
        RETURN 0 END END
```



The screenshot shows a SQL Server Enterprise Manager interface. The top pane displays the SQL code for creating a stored procedure named MODIFICARCANTIDADHIJOS. The code is as follows:

```
CREATE PROCEDURE MODIFICARCANTIDADHIJOS @DOCUMENTO VARCHAR(20), @NUEVACANTIDADHIJOS INT AS  
BEGIN  
    IF @DOCUMENTO IS NULL OR @NUEVACANTIDADHIJOS IS NULL  
    BEGIN  
        RETURN 0 END  
    UPDATE EMPLEADOS SET CANTIDADHIJOS = @NUEVACANTIDADHIJOS WHERE DOCUMENTO = @DOCUMENTO  
    IF @@ROWCOUNT > 0 BEGIN  
        RETURN 1 END ELSE BEGIN  
        RETURN 0 END END
```

The bottom pane shows a message box with the text "Commands completed successfully." and the completion time "2024-05-06T11:23:00.0263148+01:00".

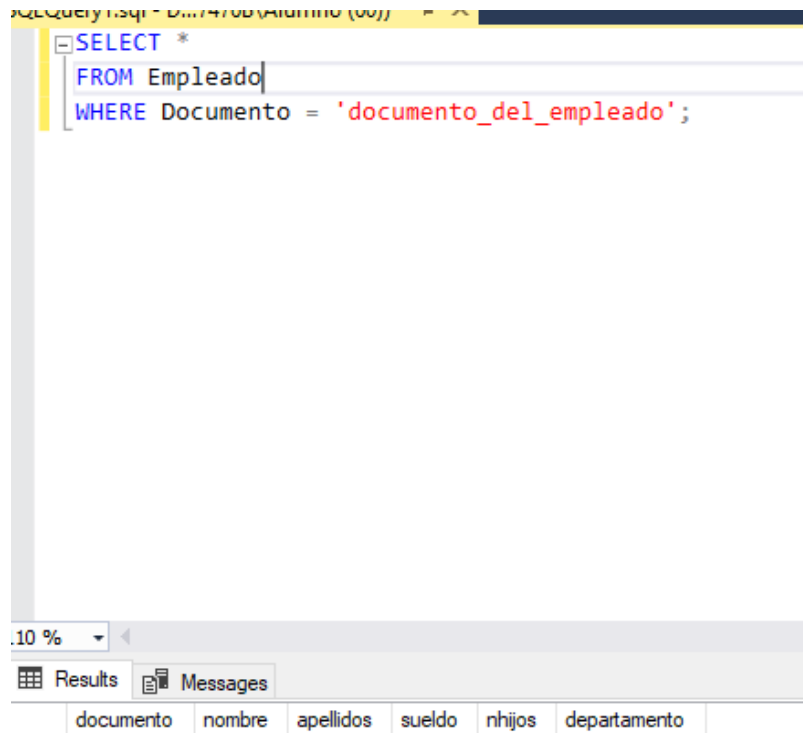
7- Declare una variable en la cual se almacenará el valor devuelto por el procedimiento, ejecute el procedimiento enviando los dos parámetros y vea el contenido de la variable. El procedimiento retorna "1", con lo cual indica que fue actualizado.

```
DECLARE @resultado INT
```

```
EXEC @resultado = ModificarCantidadHijos @documento = 'documento_del_empleado',  
@nuevaCantidadHijos = 3
```

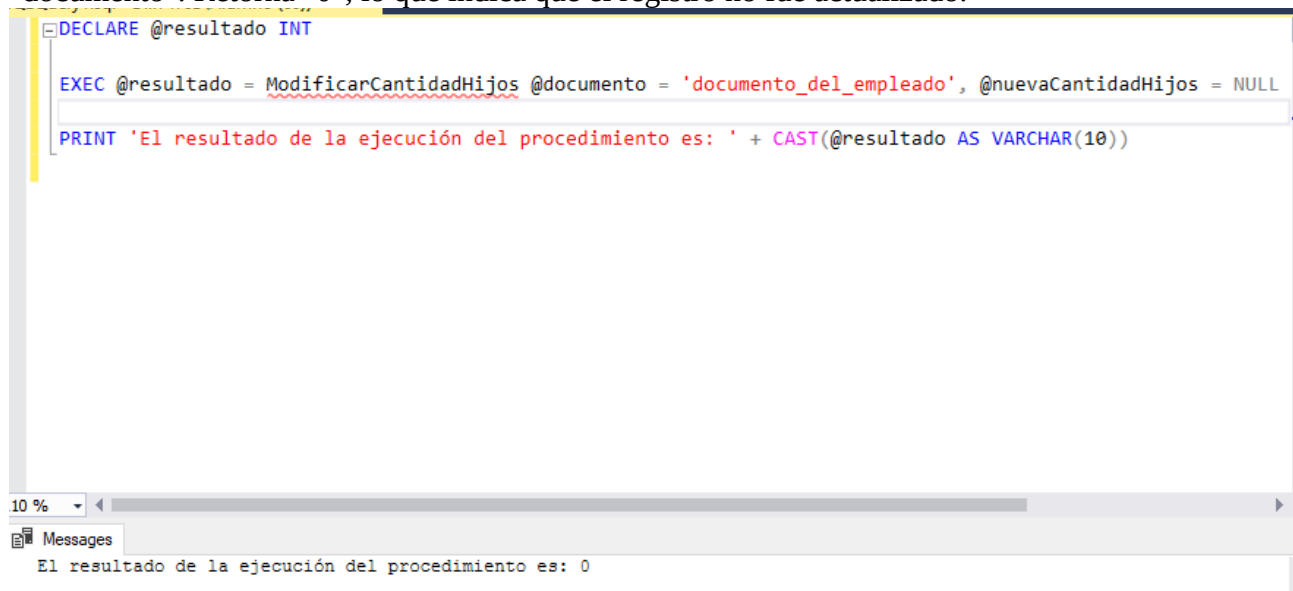
```
PRINT 'El resultado de la ejecución del procedimiento es: ' + CAST(@resultado AS  
VARCHAR(10))
```

8- Verifique la actualización consultando la tabla.



```
SELECT *  
FROM Empleado  
WHERE Documento = 'documento_del_empleado';
```

9- Ejecute los mismos pasos, pero esta vez envíe solamente un valor para el parámetro "documento". Retorna "0", lo que indica que el registro no fue actualizado.



```
DECLARE @resultado INT
```

```
EXEC @resultado = ModificarCantidadHijos @documento = 'documento_del_empleado',  
@nuevaCantidadHijos = NULL
```

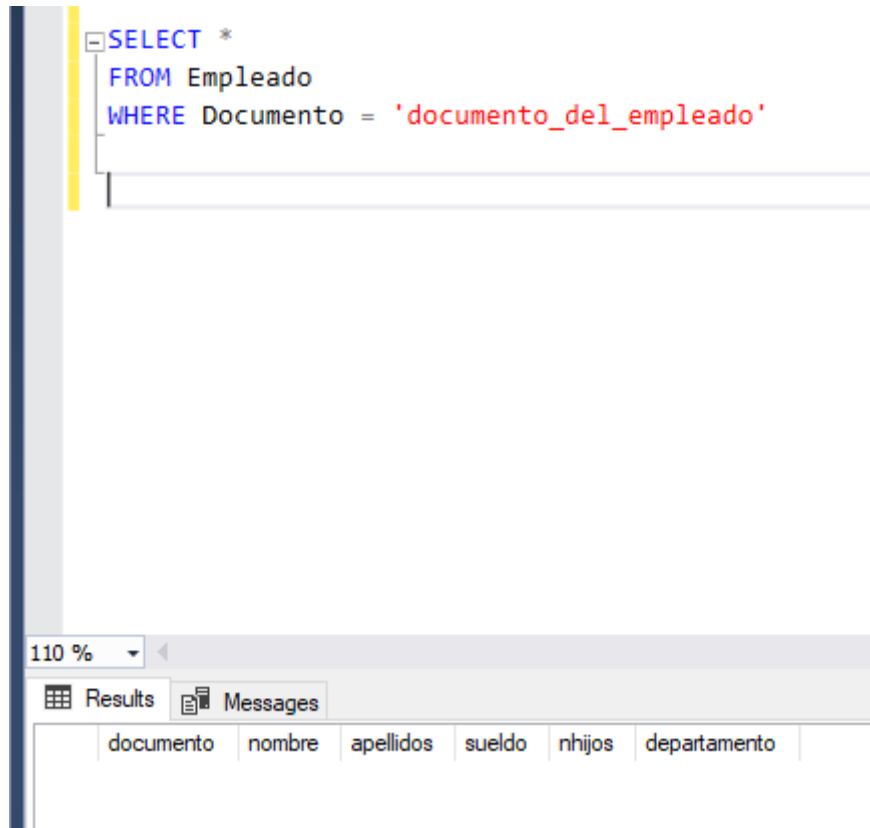
```
PRINT 'El resultado de la ejecución del procedimiento es: ' + CAST(@resultado AS  
VARCHAR(10))
```

10- Verifique que el registro no se actualizó consultando la tabla.

```
SELECT *
```

```
FROM Empleado
```

```
WHERE Documento = 'documento_del_empleado'
```



11- Emplee un "if" para controlar el valor de la variable de retorno. Enviando al procedimiento valores para los parámetros.

Retorna 1.

```
DECLARE @resultado INT
```

```
EXEC @resultado = ModificarCantidadHijos @documento = 'documento_del_empleado',
```

```
@nuevaCantidadHijos = 3
```

```
IF @resultado = 1
```

```
BEGIN
```

```
    PRINT 'La actualización fue exitosa.'
```

```
END
```

```
ELSE
```

```
BEGIN
```

```
    PRINT 'La actualización no fue exitosa.'
```

```
END
```

12- Verifique la actualización consultando la tabla.

```
SELECT *
```

```
FROM Empleado
```

```
WHERE Documento = 'documento_del_empleado'
```

13- Emplee nuevamente un "if" y envíe solamente valor para el parámetro "hijos".
Retorna 0.

```
DECLARE @resultado INT

EXEC @resultado = ModificarCantidadHijos @documento = NULL, @nuevaCantidadHijos = 5

IF @resultado = 1
BEGIN
    PRINT 'La actualización fue exitosa.'
END
ELSE
BEGIN
    PRINT 'La actualización no fue exitosa.'
END
```

Messages

La actualización no fue exitosa.

Completion time: 2024-05-06T11:28:13.4245128+01:00

```
DECLARE @resultado INT
```

```
EXEC @resultado = ModificarCantidadHijos @documento = NULL, @nuevaCantidadHijos = 5
```

```
IF @resultado = 1
```

```
BEGIN
```

```
    PRINT 'La actualización fue exitosa.'
```

```
END
```

```
ELSE
```

```
BEGIN
```

```
    PRINT 'La actualización no fue exitosa.'
```

```
END
```

Parte 2:

Un profesor guarda en una tabla llamada "Alumno" el nombre de los alumnos y su nota.

1- Elimine la tabla si existe y créela con los siguientes campos: documento char(8), nombre varchar(40), nota decimal(4,2), primary key(documento)

```
IF OBJECT_ID('Alumno', 'U') IS NOT NULL
```

```
DROP TABLE ALUMNO
```

```
CREATE TABLE ALUMNO (
```

```
    DOCUMENTO CHAR(8) PRIMARY KEY,
```

```
    NOMBRE VARCHAR(40),
```

```
    NOTA DECIMAL(4, 2)
```

```
)
```

```
IF OBJECT_ID('Alumno', 'U') IS NOT NULL

DROP TABLE ALUMNO
CREATE TABLE ALUMNO (
    DOCUMENTO CHAR(8) PRIMARY KEY,
    NOMBRE VARCHAR(40),
    NOTA DECIMAL(4, 2)
)
```

110 %

Messages

Commands completed successfully.

Completion time: 2024-05-06T11:30:41.7445642+01:00

2- Inserte algunos registros:

```
insert into alumnos values ('22222222','Pedro López',5);
insert into alumnos values ('23333333','Ana López',4);
insert into alumnos values ('24444444','María Santana',8);
insert into alumnos values ('25555555','Juan García',5.6);
insert into alumnos values ('26666666','Carlos Torres',2);
insert into alumnos values ('27777777','Noelia Torres',7.5);
insert into alumnos values ('28888888','Mariano Herreros',3.5);
```

```
INSERT INTO Alumno VALUES ('22222222','Pedro López',5);
INSERT INTO Alumno VALUES ('23333333','Ana López',4);
INSERT INTO Alumno VALUES ('24444444','María Santana',8);
INSERT INTO Alumno VALUES ('25555555','Juan García',5.6);
INSERT INTO Alumno VALUES ('26666666','Carlos Torres',2);
INSERT INTO Alumno VALUES ('27777777','Noelia Torres',7.5);
INSERT INTO Alumno VALUES ('28888888','Mariano Herreros',3.5);
```

110 %

Messages

(1 row affected)

(1 row affected)

(1 row affected)

(1 row affected)

(1 row affected)

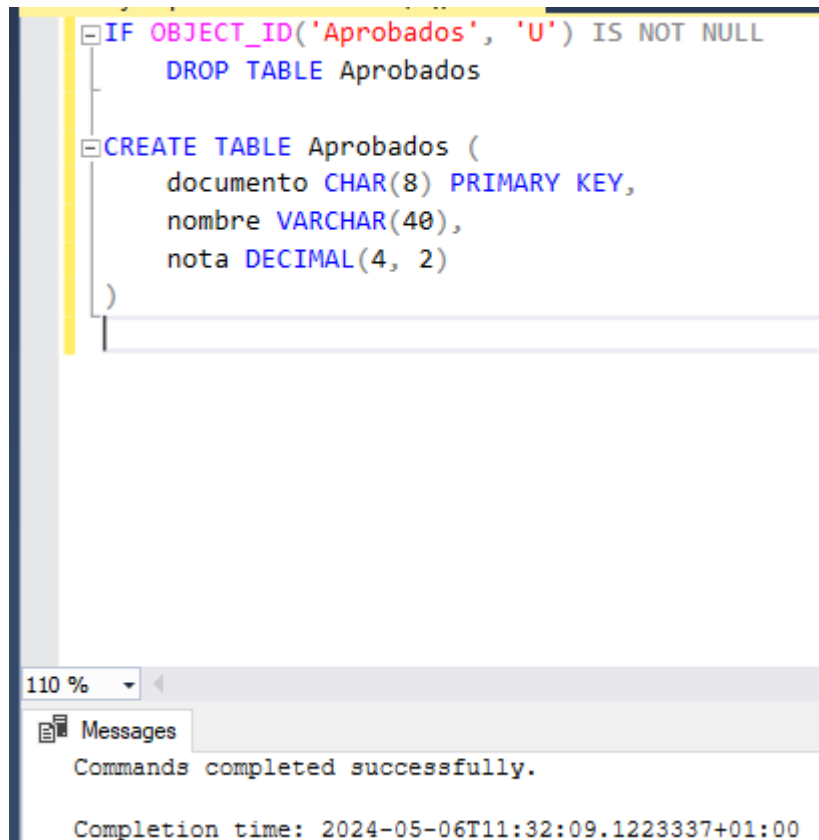
(1 row affected)

(1 row affected)

Completion time: 2024-05-06T11:31:34.1907628+01:00

3- Elimine la tabla "Aprobados" si existe y créela con los mismos campos de la tabla "alumnos".

```
IF OBJECT_ID('Aprobados', 'U') IS NOT NULL DROP TABLE Aprobados CREATE  
TABLE Aprobados ( documento CHAR(8) PRIMARY KEY, nombre VARCHAR(40), nota  
DECIMAL(4, 2) )
```



```
IF OBJECT_ID('Aprobados', 'U') IS NOT NULL  
    DROP TABLE Aprobados  
  
CREATE TABLE Aprobados (  
    documento CHAR(8) PRIMARY KEY,  
    nombre VARCHAR(40),  
    nota DECIMAL(4, 2)  
)
```

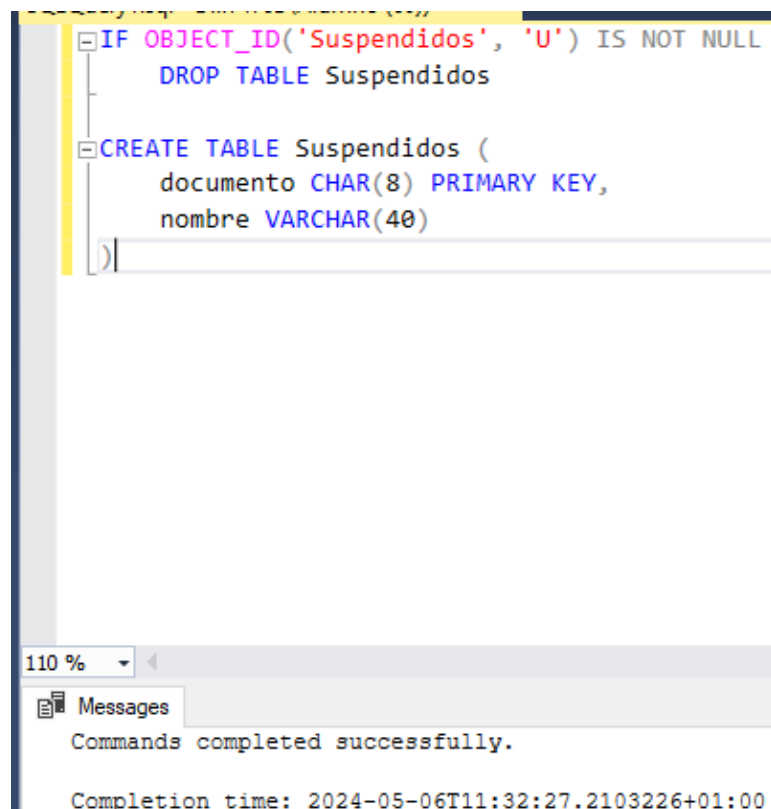
110 %

Messages

Commands completed successfully.

Completion time: 2024-05-06T11:32:09.1223337+01:00

4- Elimine la tabla "Suspendidos" si existe y créela con los siguientes campos: documento char(8), nombre varchar(40)



```
IF OBJECT_ID('Suspendidos', 'U') IS NOT NULL  
    DROP TABLE Suspendidos  
  
CREATE TABLE Suspendidos (  
    documento CHAR(8) PRIMARY KEY,  
    nombre VARCHAR(40)  
)
```

110 %

Messages

Commands completed successfully.

Completion time: 2024-05-06T11:32:27.2103226+01:00

5- Elimine el procedimiento llamado "pa_aprobados", si existe.

```
IF OBJECT_ID('pa_aprobados', 'P') IS NOT NULL
    DROP PROCEDURE pa_aprobados
```

110 %

Messages

Commands completed successfully.

Completion time: 2024-05-06T11:32:36.7405053+01:00

6- Cree el procedimiento para que seleccione todos los datos de los alumnos cuya nota es igual o superior a 5.

```
CREATE PROCEDURE pa_aprobados
AS
BEGIN
    SELECT * FROM Alumno WHERE nota >= 5
END
```

110 %

Messages

Commands completed successfully.

Completion time: 2024-05-06T11:32:50.9083744+01:00

```
CREATE PROCEDURE pa_aprobados AS BEGIN SELECT * FROM Alumno WHERE nota
>= 5 END
```

7- Inserte en la tabla "aprobados" el resultado devuelto por el procedimiento almacenado "pa_aprobados".

```
INSERT INTO Aprobados EXEC pa_aprobados
```

.0 %

Messages

(4 rows affected)

Completion time: 2024-05-06T11:33:05.1330070+01:00

INSERT INTO Aprobados EXEC pa_aprobados

8- Vea el contenido de "Aprobados".

SELECT * FROM Aprobados

```
SELECT * FROM Aprobados
```

110 %

Results Messages

	documento	nombre	nota
1	22222222	Pedro López	5.00
2	24444444	María Santana	8.00
3	25555555	Juan García	5.60
4	27777777	Noelia Torres	7.50

9- Elimine el procedimiento llamado "pa_suspendidos", si existe.

IF OBJECT_ID('pa_suspendidos', 'P') IS NOT NULL DROP PROCEDURE pa_suspendidos

```
IF OBJECT_ID('pa_suspendidos', 'P') IS NOT NULL
    DROP PROCEDURE pa_suspendidos
```

110 %

Messages

Commands completed successfully.

Completion time: 2024-05-06T11:33:37.8601741+01:00

10- Cree el procedimiento para que seleccione el documento y nombre de los alumnos cuya nota es menor a 5.

```
CREATE PROCEDURE pa_suspendidos
AS
BEGIN
    SELECT documento, nombre FROM Alumno WHERE nota < 5
END
```

110 %

Messages

Commands completed successfully.

Completion time: 2024-05-06T11:33:57.7643581+01:00

```
CREATE PROCEDURE pa_suspendidos AS BEGIN SELECT documento, nombre FROM Alumno WHERE nota < 5 END
```

11- Inserte en la tabla "suspendidos" el resultado devuelto por el procedimiento almacenado "pa_suspendidos".

```
INSERT INTO Suspendidos EXEC pa_suspendidos
```

110 %

Messages

(3 rows affected)

Completion time: 2024-05-06T11:34:33.2359432+01:00

12- Vea el contenido de "Suspendidos".

```
SELECT * FROM Suspendidos
```

```
SELECT * FROM Suspendidos
```

110 %

	documento	nombre
1	23333333	Ana López
2	26666666	Carlos Torres
3	28888888	Mariano Herreros