Presentacion Técnica {

Campus: CEM

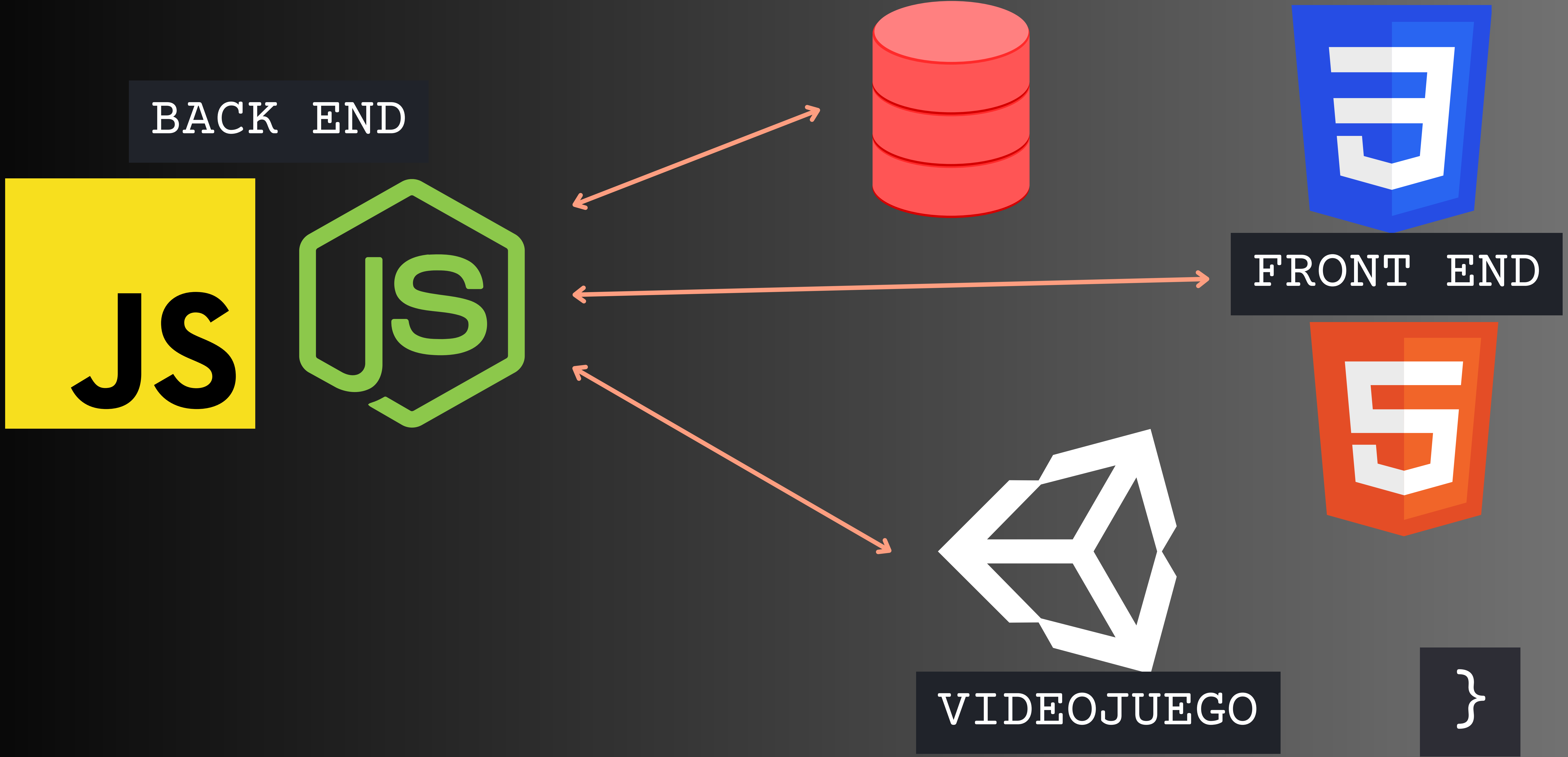
INTEGRANTES;

Héctor Gónzalez Sánchez - A01753863
Luis Adrián Abarca Gómez - A01798043
Gerardo Rios Mejía - A1753830
Juan Carlos Carro Cruz - A01748640
Alfredo Azamar López - A01798100

}

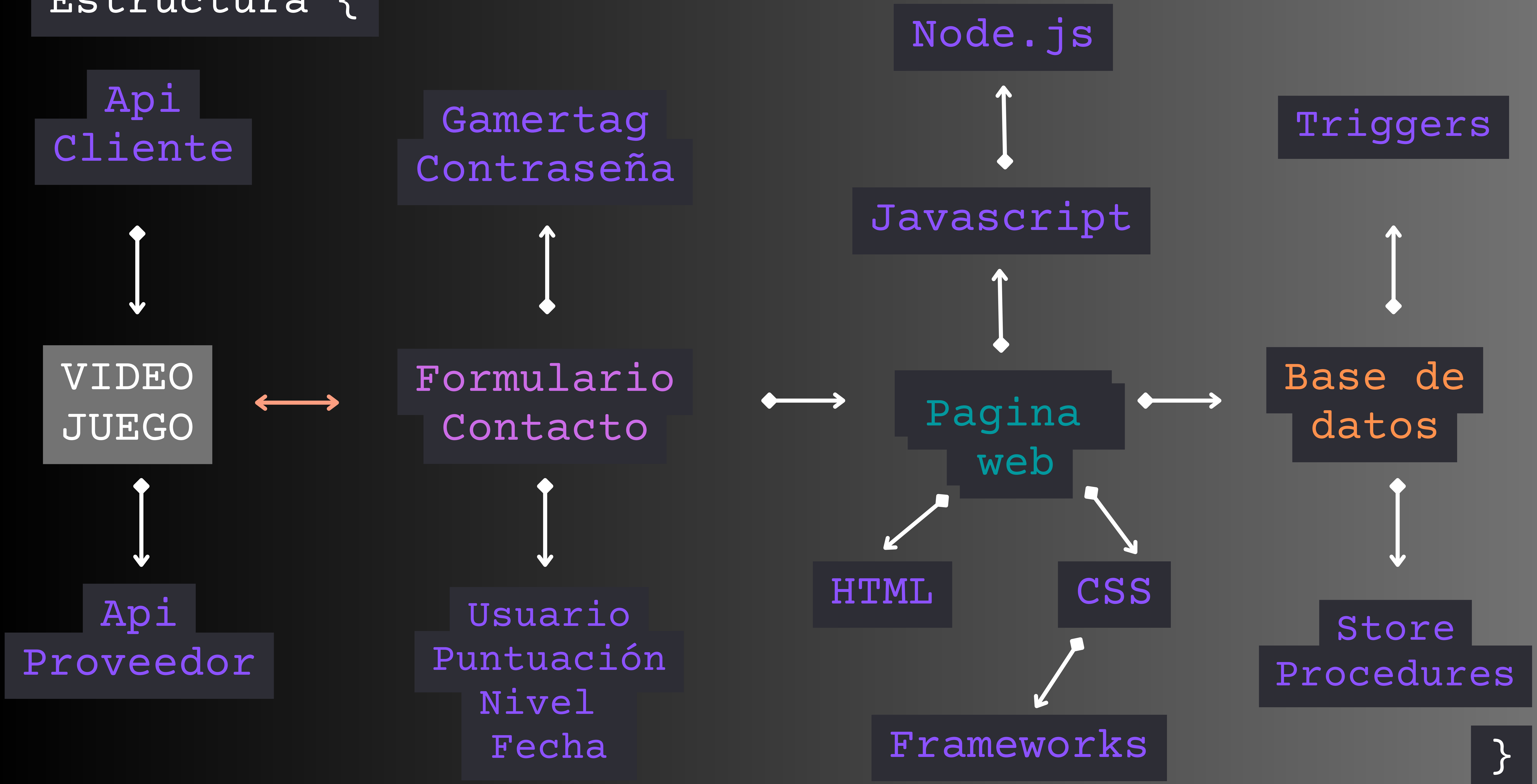
Contenidos

Estructura {



}

Estructura {


Api Cliente
VIDEO JUEGO
Api Proveedor
Gamertag Contraseña
Formulario Contacto
Usuario Puntuación Nivel Fecha
Node.js
Javascript
Pagina web
HTML
CSS
Frameworks
Triggers
Base de datos
Store Procedures

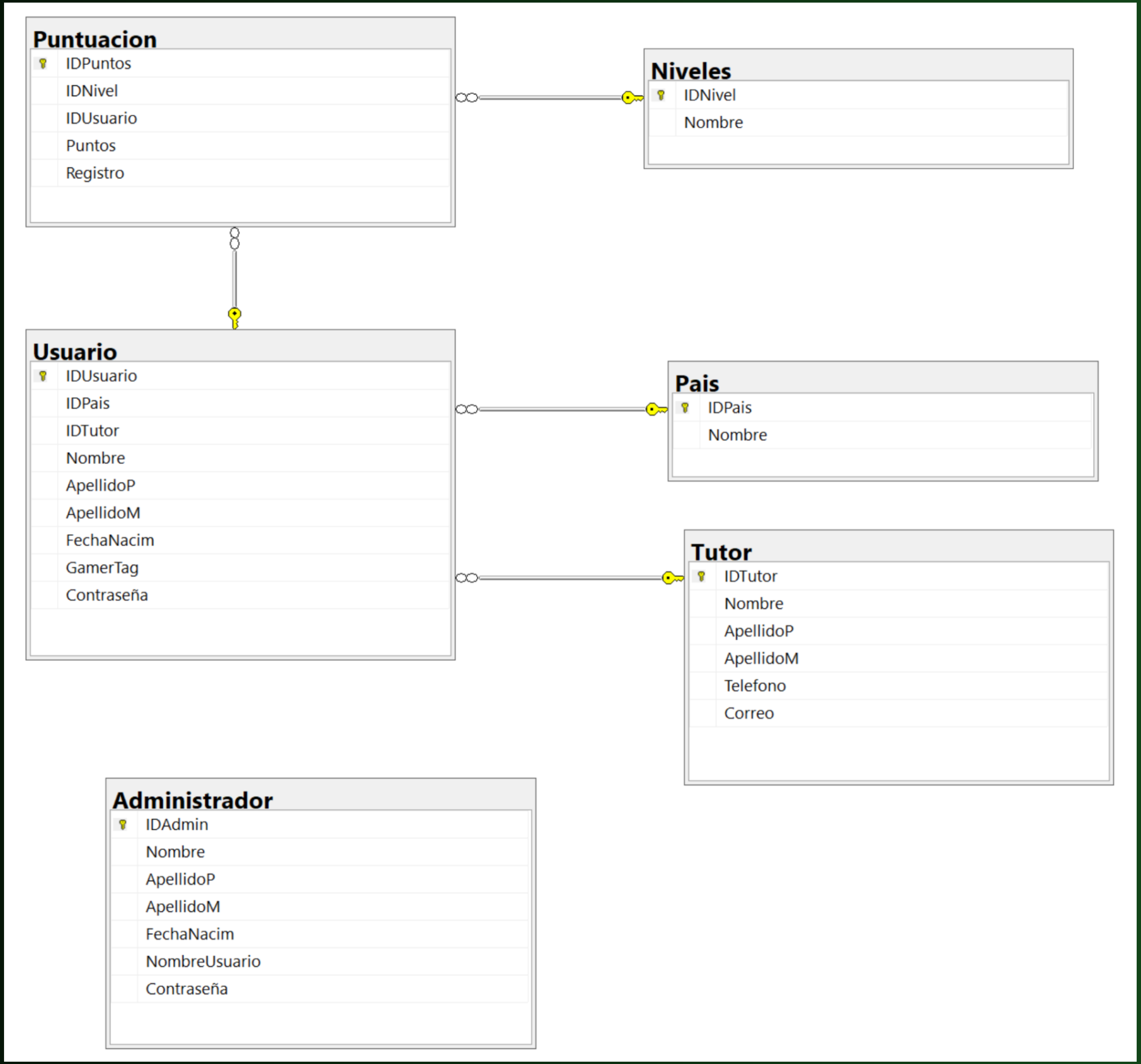
}

Base de datos

<SQL Server>

}

Modelo relacional {

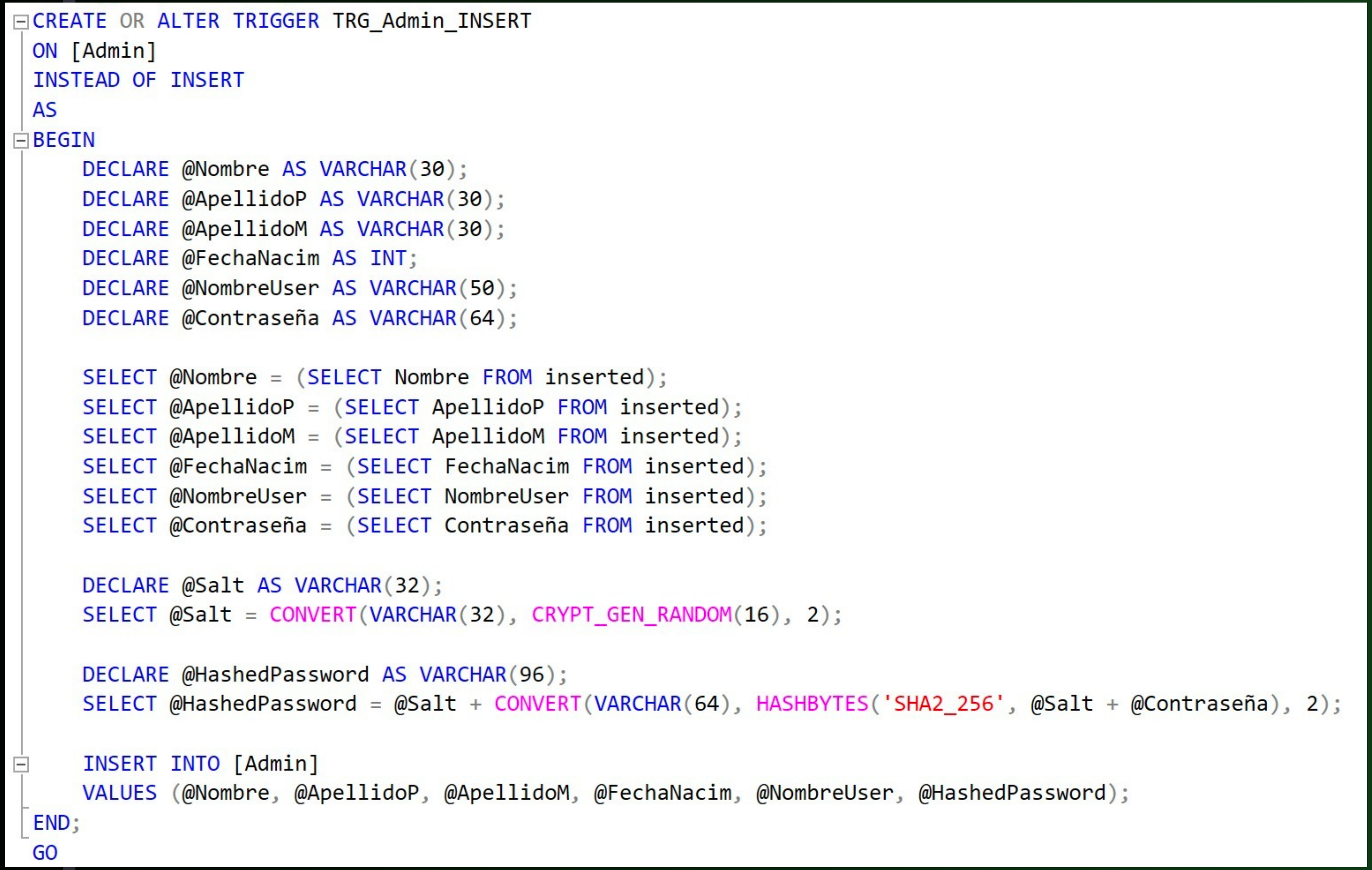


}

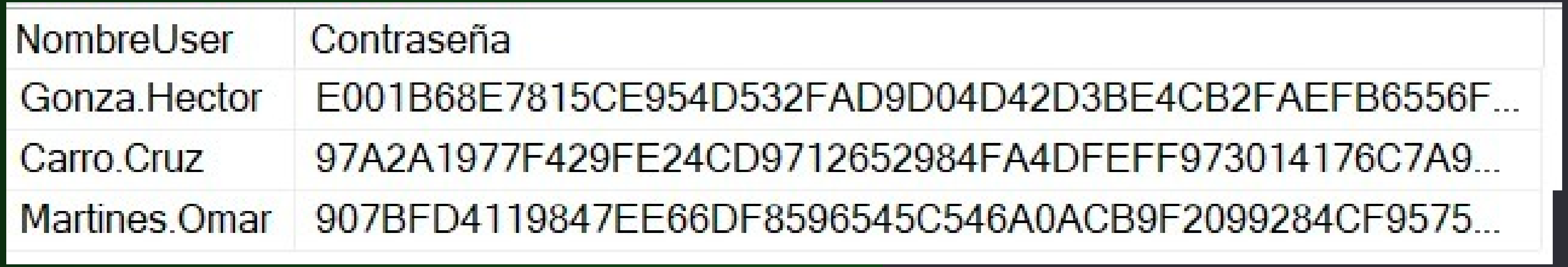
Insertando datos

{

Trigger



```
CREATE OR ALTER TRIGGER TRG_Admin_INSERT
ON [Admin]
INSTEAD OF INSERT
AS
BEGIN
    DECLARE @Nombre AS VARCHAR(30);
    DECLARE @ApellidoP AS VARCHAR(30);
    DECLARE @ApellidoM AS VARCHAR(30);
    DECLARE @FechaNacim AS INT;
    DECLARE @NombreUser AS VARCHAR(50);
    DECLARE @Contraseña AS VARCHAR(64);

    SELECT @Nombre = (SELECT Nombre FROM inserted);
    SELECT @ApellidoP = (SELECT ApellidoP FROM inserted);
    SELECT @ApellidoM = (SELECT ApellidoM FROM inserted);
    SELECT @FechaNacim = (SELECT FechaNacim FROM inserted);
    SELECT @NombreUser = (SELECT NombreUser FROM inserted);
    SELECT @Contraseña = (SELECT Contraseña FROM inserted);

    DECLARE @Salt AS VARCHAR(32);
    SELECT @Salt = CONVERT(VARCHAR(32), CRYPT_GEN_RANDOM(16), 2);

    DECLARE @HashedPassword AS VARCHAR(96);
    SELECT @HashedPassword = @Salt + CONVERT(VARCHAR(64), HASHBYTES('SHA2_256', @Salt + @Contraseña), 2);

    INSERT INTO [Admin]
    VALUES (@Nombre, @ApellidoP, @ApellidoM, @FechaNacim, @NombreUser, @HashedPassword);
END;
GO
```



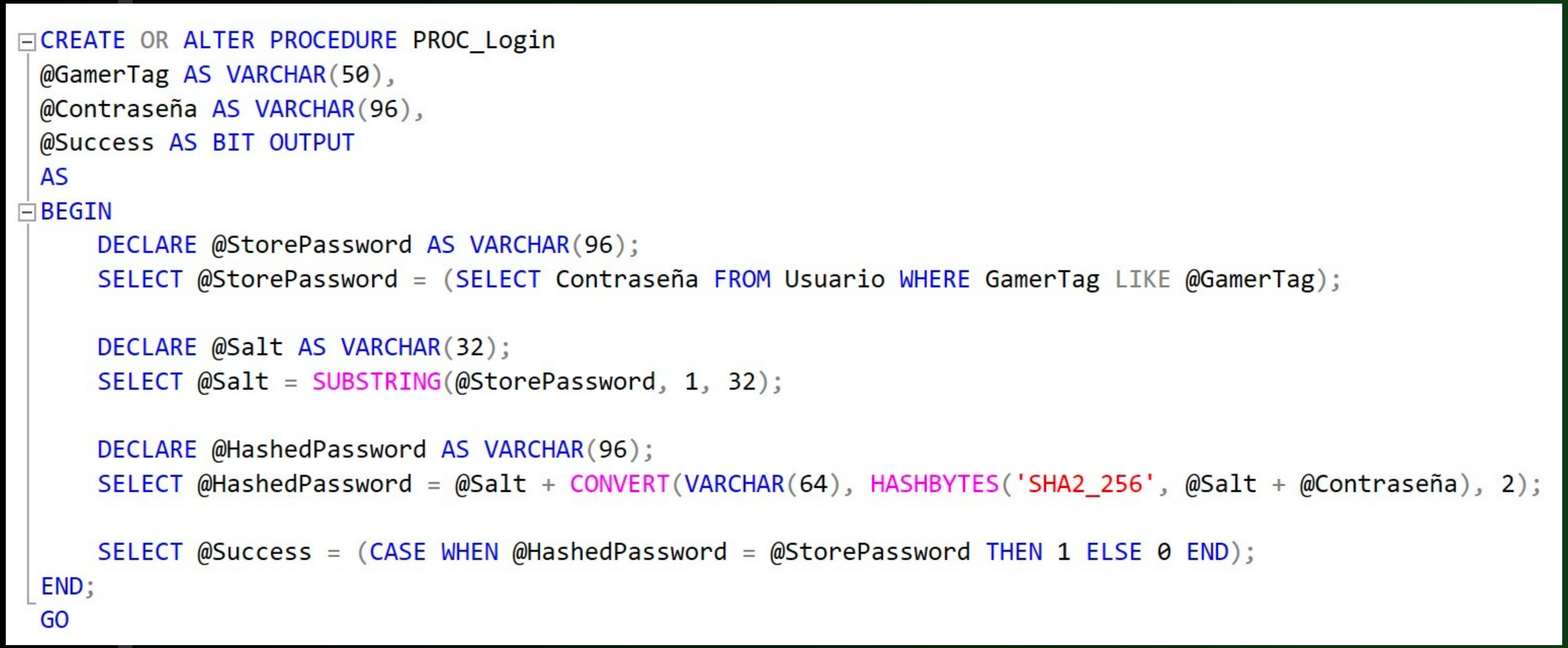
NombreUser	Contraseña
Gonza.Hector	E001B68E7815CE954D532FAD9D04D42D3BE4CB2FAEFB6556F...
Carro.Cruz	97A2A1977F429FE24CD9712652984FA4DFEFF973014176C7A9...
Martines.Omar	907BFD4119847EE66DF8596545C546A0ACB9F2099284CF9575...

}

Iniciar sesión

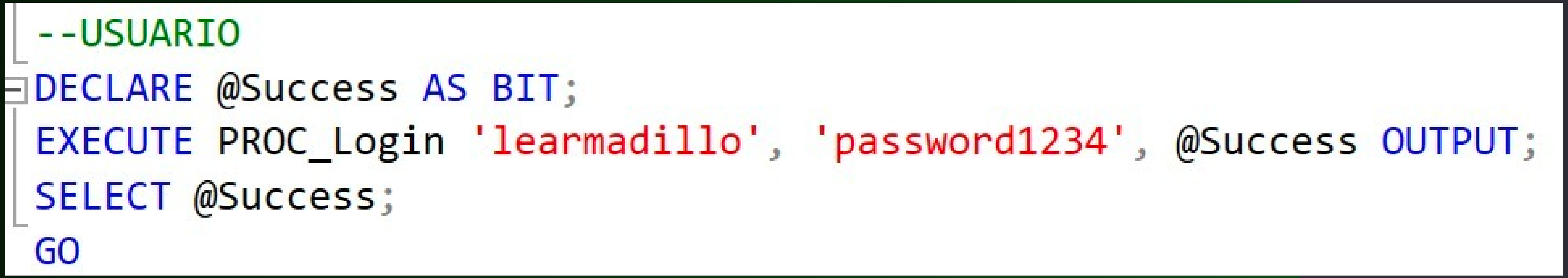
{

Stored Procedure



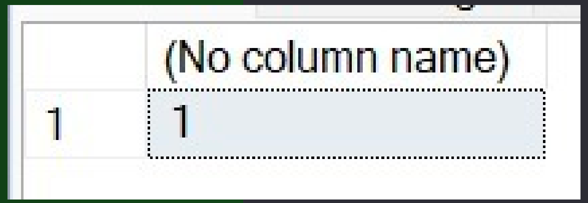
```
CREATE OR ALTER PROCEDURE PROC_Login
@GamerTag AS VARCHAR(50),
@Contraseña AS VARCHAR(96),
@Success AS BIT OUTPUT
AS
BEGIN
    DECLARE @StorePassword AS VARCHAR(96);
    SELECT @StorePassword = (SELECT Contraseña FROM Usuario WHERE GamerTag LIKE @GamerTag);

    DECLARE @Salt AS VARCHAR(32);
    SELECT @Salt = SUBSTRING(@StorePassword, 1, 32);

    DECLARE @HashedPassword AS VARCHAR(96);
    SELECT @HashedPassword = @Salt + CONVERT(VARCHAR(64), HASHBYTES('SHA2_256', @Salt + @Contraseña), 2);

    SELECT @Success = (CASE WHEN @HashedPassword = @StorePassword THEN 1 ELSE 0 END);
END;
GO
```

Ejemplo



```
--USUARIO
DECLARE @Success AS BIT;
EXECUTE PROC_Login 'learmadillo', 'password1234', @Success OUTPUT;
SELECT @Success;
GO
```

Salida



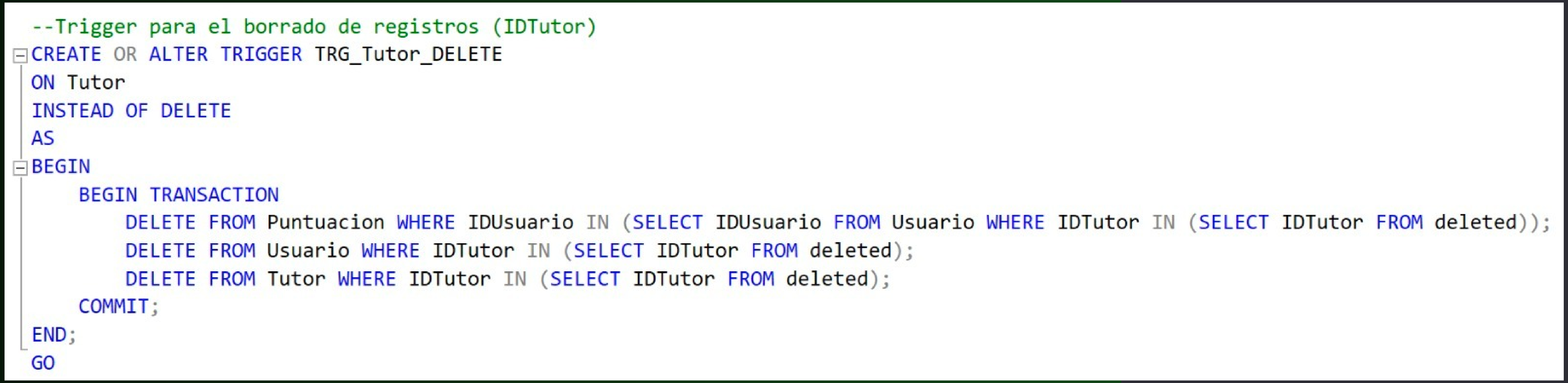
```
   (No column name)
1  1
```

}

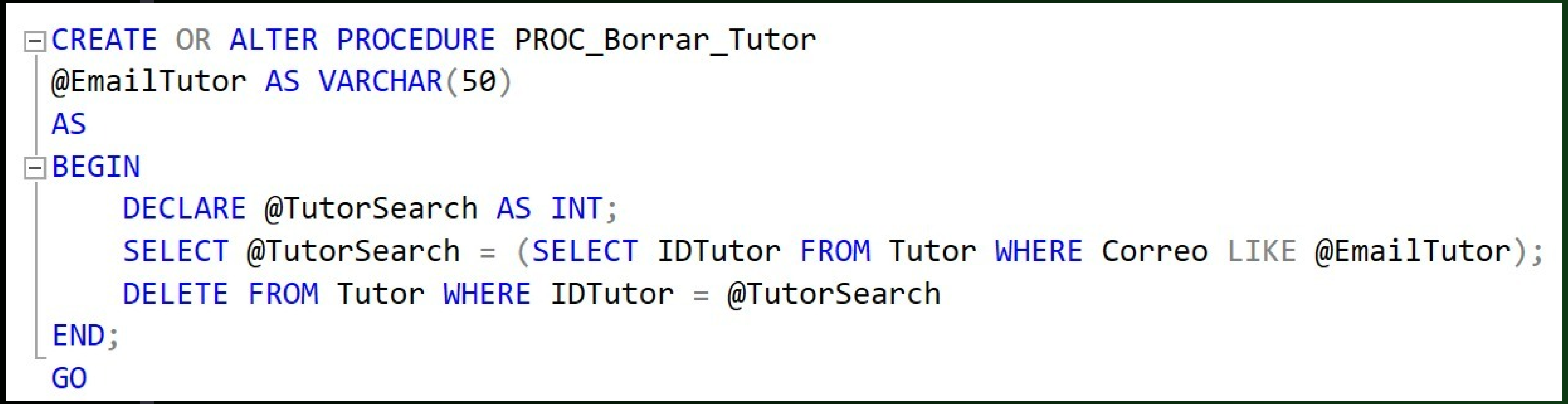
Borrando datos

{

Trigger



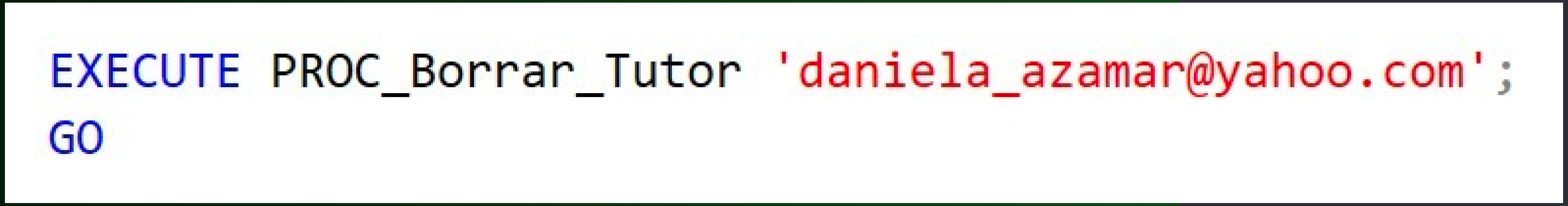
```sql
--Trigger para el borrado de registros (IDTutor)
CREATE OR ALTER TRIGGER TRG_Tutor_DELETE
ON Tutor
INSTEAD OF DELETE
AS
BEGIN
    BEGIN TRANSACTION
        DELETE FROM Puntuacion WHERE IDUsuario IN (SELECT IDUsuario FROM Usuario WHERE IDTutor IN (SELECT IDTutor FROM deleted));
        DELETE FROM Usuario WHERE IDTutor IN (SELECT IDTutor FROM deleted);
        DELETE FROM Tutor WHERE IDTutor IN (SELECT IDTutor FROM deleted);
    COMMIT;
END;
GO
```



```sql
CREATE OR ALTER PROCEDURE PROC_Borrar_Tutor
@EmailTutor AS VARCHAR(50)
AS
BEGIN
    DECLARE @TutorSearch AS INT;
    SELECT @TutorSearch = (SELECT IDTutor FROM Tutor WHERE Correo LIKE @EmailTutor);
    DELETE FROM Tutor WHERE IDTutor = @TutorSearch
END;
GO
```

Stored Procedure

Ejecución



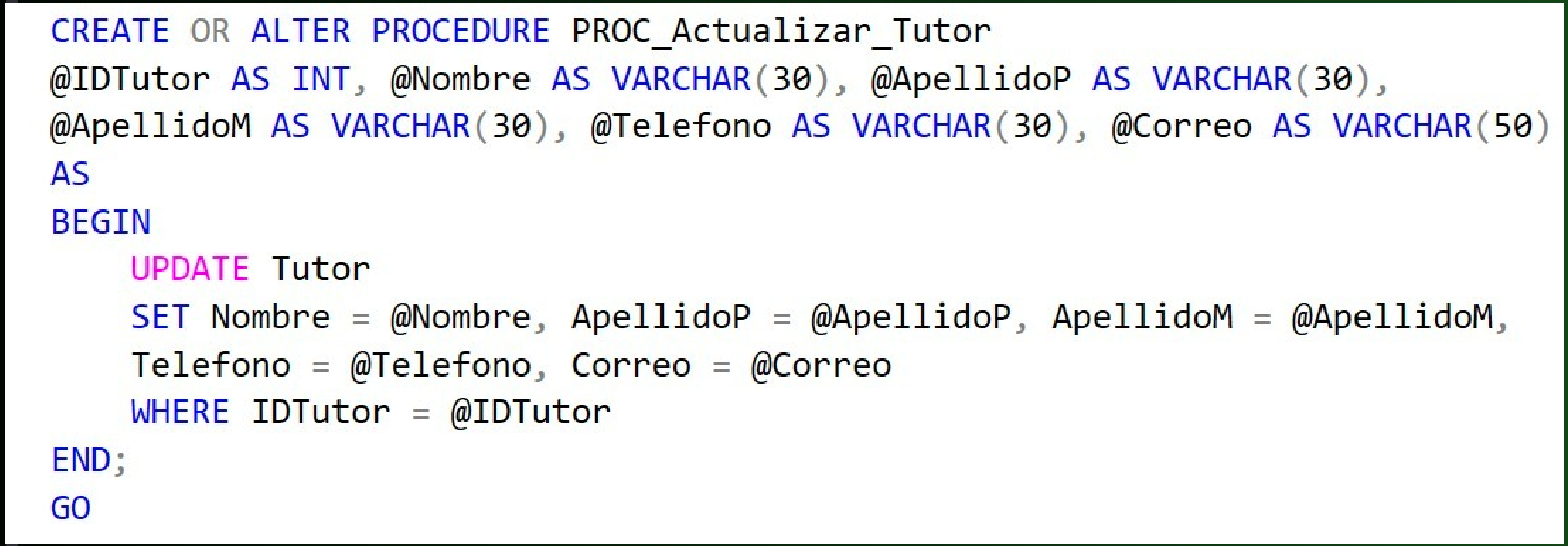
```sql
EXECUTE PROC_Borrar_Tutor 'daniela_azamar@yahoo.com';
GO
```

}

Actualizando datos

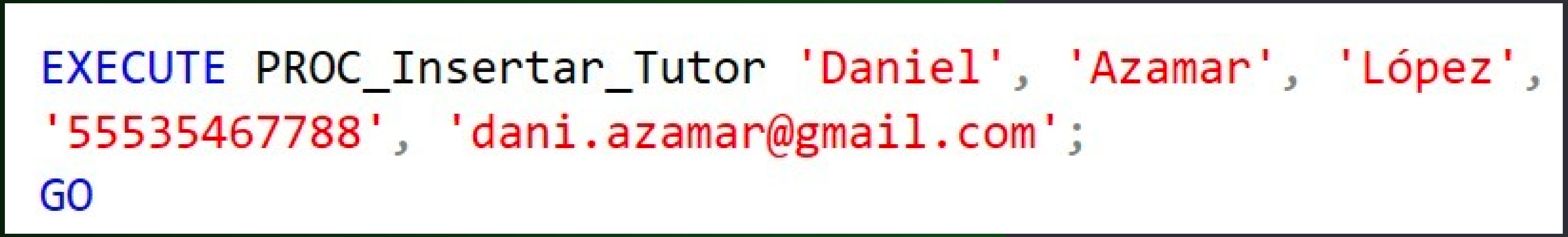
{

Stored Procedure



```sql
CREATE OR ALTER PROCEDURE PROC_Actualizar_Tutor
@IDTutor AS INT, @Nombre AS VARCHAR(30), @ApellidoP AS VARCHAR(30),
@ApellidoM AS VARCHAR(30), @Telefono AS VARCHAR(30), @Correo AS VARCHAR(50)
AS
BEGIN
    UPDATE Tutor
    SET Nombre = @Nombre, ApellidoP = @ApellidoP, ApellidoM = @ApellidoM,
    Telefono = @Telefono, Correo = @Correo
    WHERE IDTutor = @IDTutor
END;
GO
```

Ejecución



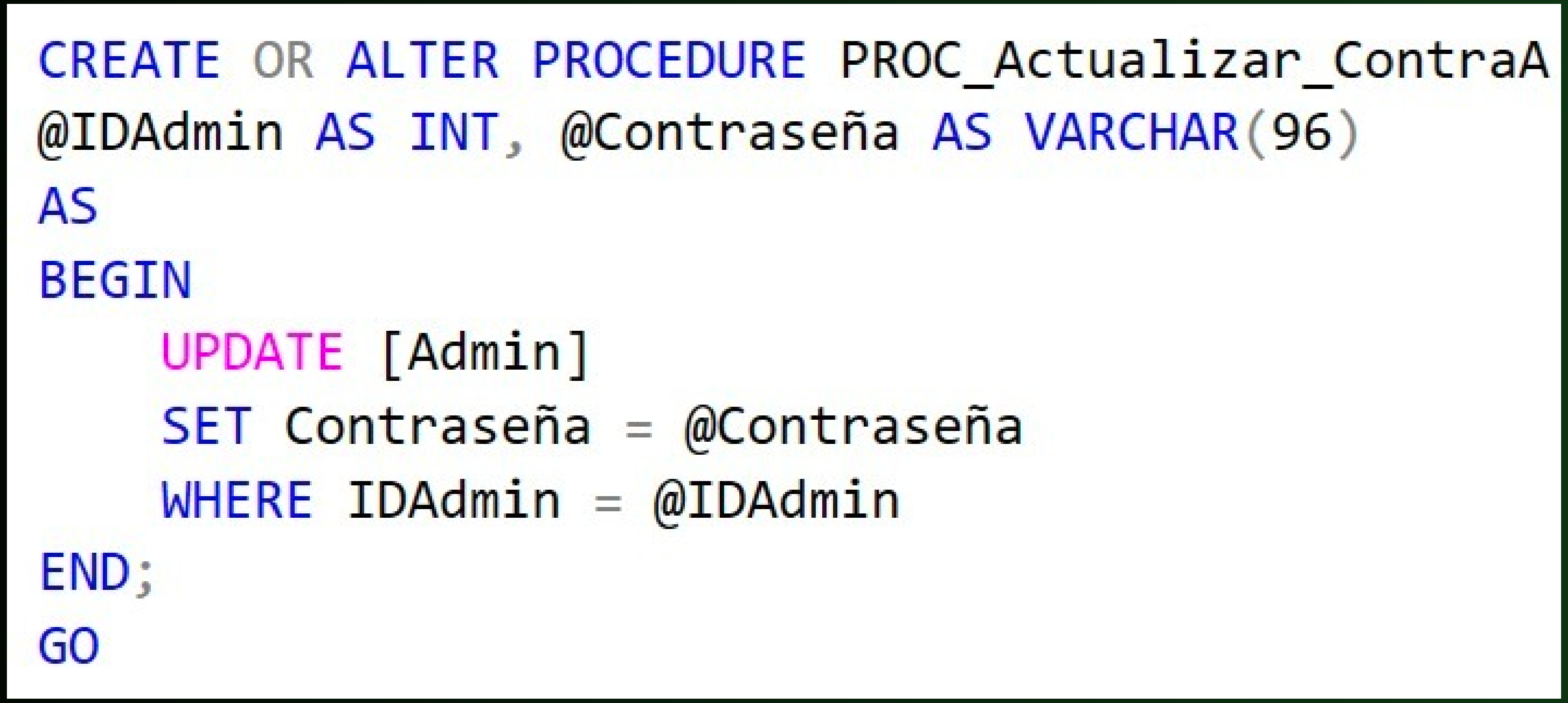
```sql
EXECUTE PROC_Insertar_Tutor 'Daniel', 'Azamar', 'López',
'55535467788', 'dani.azamar@gmail.com';
GO
```

}

Actualizando datos

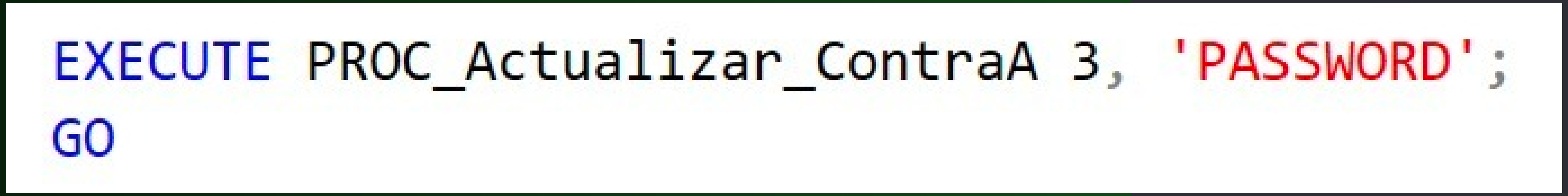
{

Stored Procedure



```sql
CREATE OR ALTER PROCEDURE PROC_Actualizar_ContraA
@IDAdmin AS INT, @Contraseña AS VARCHAR(96)
AS
BEGIN
    UPDATE [Admin]
    SET Contraseña = @Contraseña
    WHERE IDAdmin = @IDAdmin
END;
GO
```

Ejecución

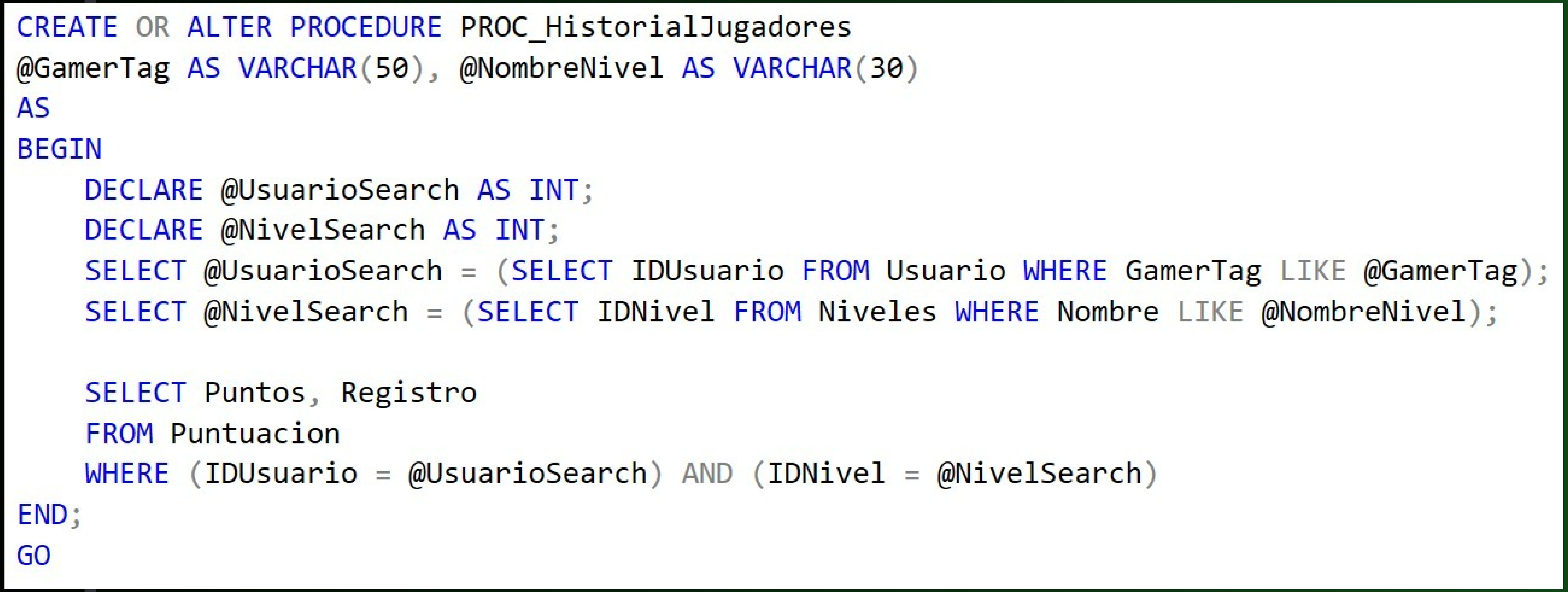


```sql
EXECUTE PROC_Actualizar_ContraA 3, 'PASSWORD';
GO
```

}

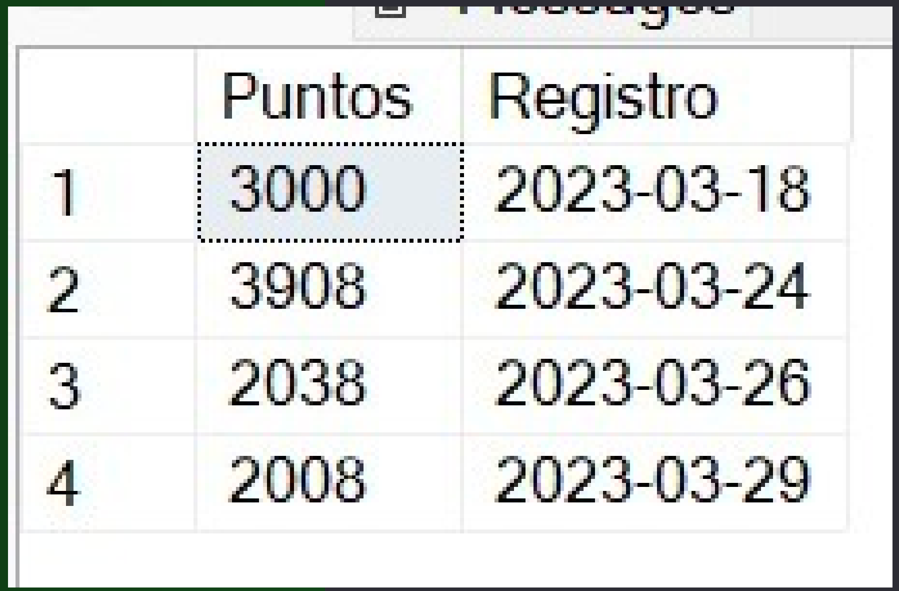
Queries {

Stored Procedure



```
CREATE OR ALTER PROCEDURE PROC_HistorialJugadores
@GamerTag AS VARCHAR(50), @NombreNivel AS VARCHAR(30)
AS
BEGIN
    DECLARE @UsuarioSearch AS INT;
    DECLARE @NivelSearch AS INT;
    SELECT @UsuarioSearch = (SELECT IDUsuario FROM Usuario WHERE GamerTag LIKE @GamerTag);
    SELECT @NivelSearch = (SELECT IDNivel FROM Niveles WHERE Nombre LIKE @NombreNivel);

    SELECT Puntos, Registro
    FROM Puntuacion
    WHERE (IDUsuario = @UsuarioSearch) AND (IDNivel = @NivelSearch)
END;
GO
```

Salida



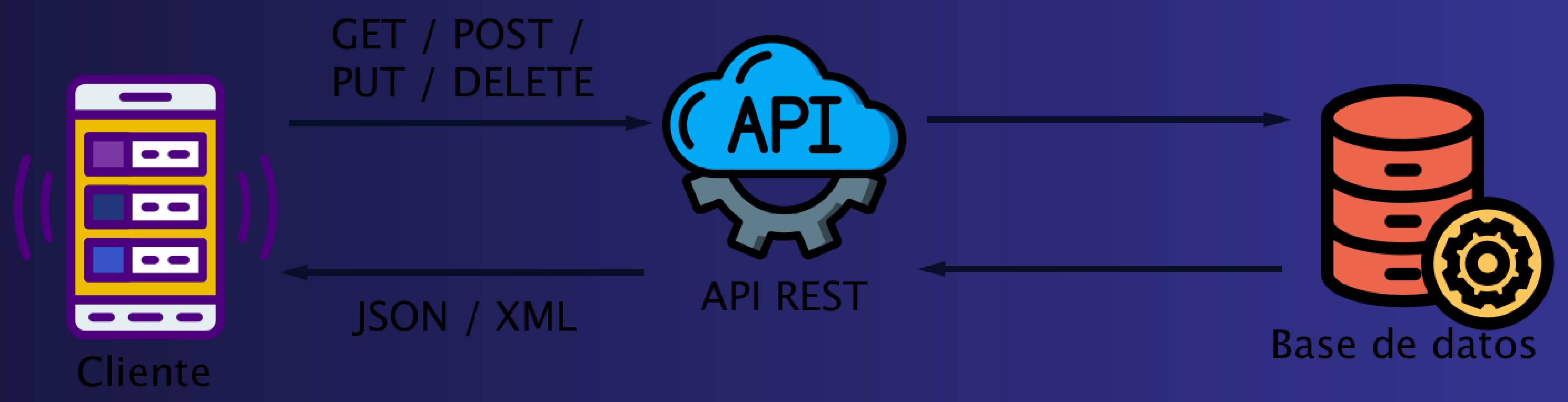
	Puntos	Registro
1	3000	2023-03-18
2	3908	2023-03-24
3	2038	2023-03-26
4	2008	2023-03-29

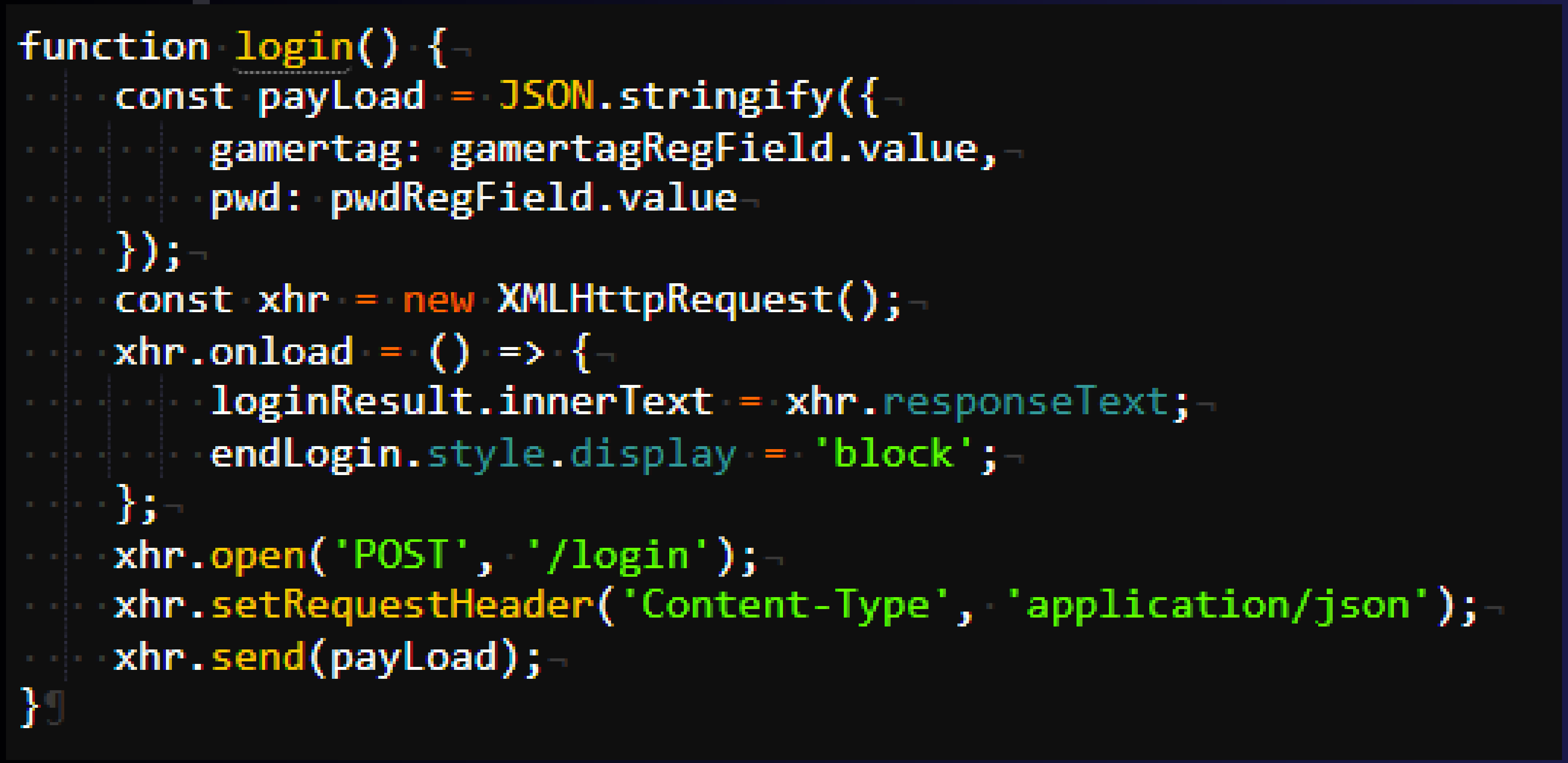
}

Pagina web

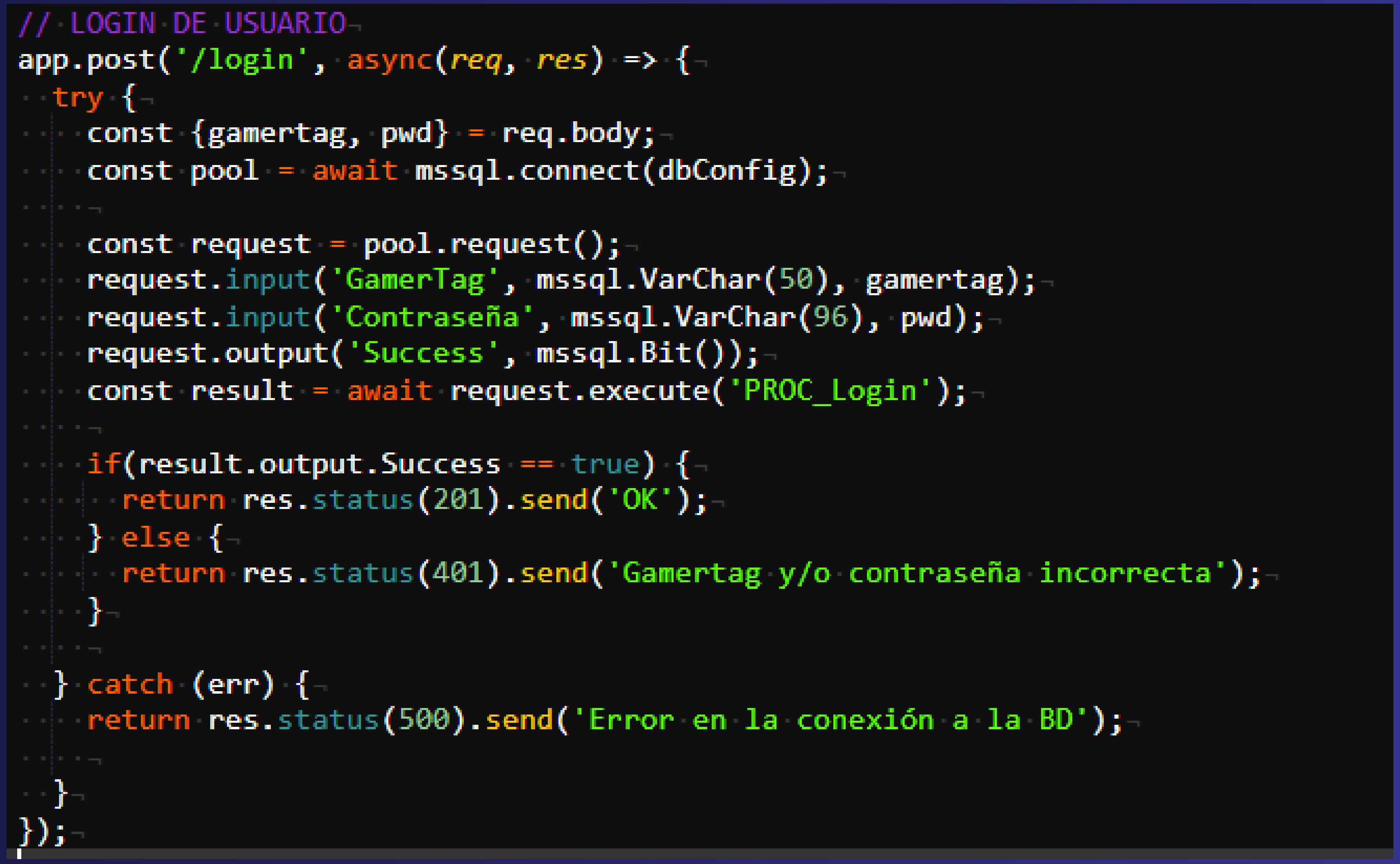
```
<="Montada en AWS y construida
con HTML y CSS"/>
}
```

Servicio web para el inicio de sesión {





```javascript
function login() {
    const payLoad = JSON.stringify({
        gamertag: gamertagRegField.value,
        pwd: pwdRegField.value
    });
    const xhr = new XMLHttpRequest();
    xhr.onload = () => {
        loginResult.innerText = xhr.responseText;
        endLogin.style.display = 'block';
    };
    xhr.open('POST', '/login');
    xhr.setRequestHeader('Content-Type', 'application/json');
    xhr.send(payLoad);
}
```



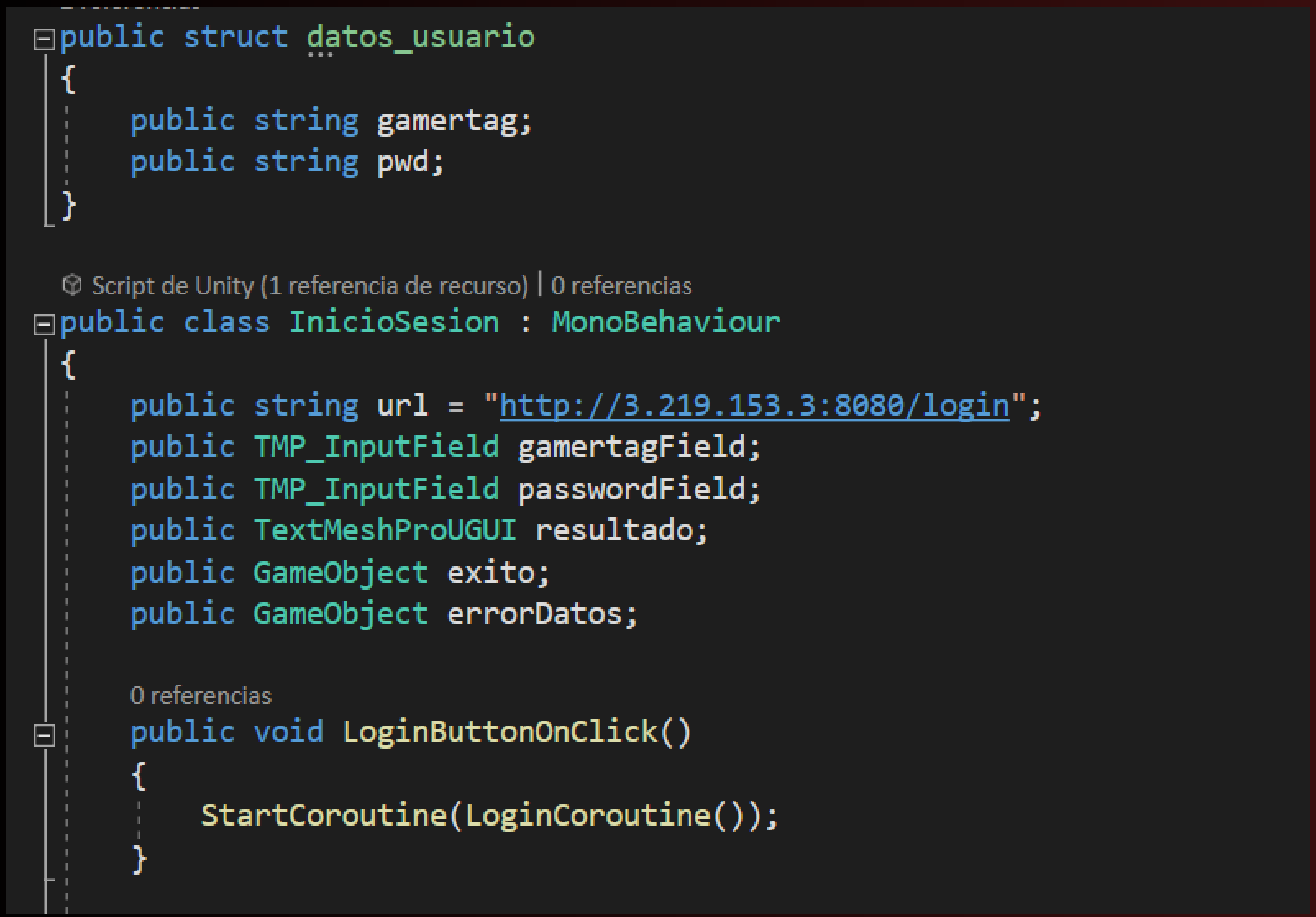
```javascript
// LOGIN DE USUARIO
app.post('/login', async(req, res) => {
  try {
    const {gamertag, pwd} = req.body;
    const pool = await mssql.connect(dbConfig);

    const request = pool.request();
    request.input('GamerTag', mssql.VarChar(50), gamertag);
    request.input('Contraseña', mssql.VarChar(96), pwd);
    request.output('Success', mssql.Bit());
    const result = await request.execute('PROC_Login');

    if(result.output.Success == true) {
      return res.status(201).send('OK');
    } else {
      return res.status(401).send('Gamertag y/o contraseña incorrecta');
    }

  } catch (err) {
    return res.status(500).send('Error en la conexión a la BD');

  }
});
```

Videojuego

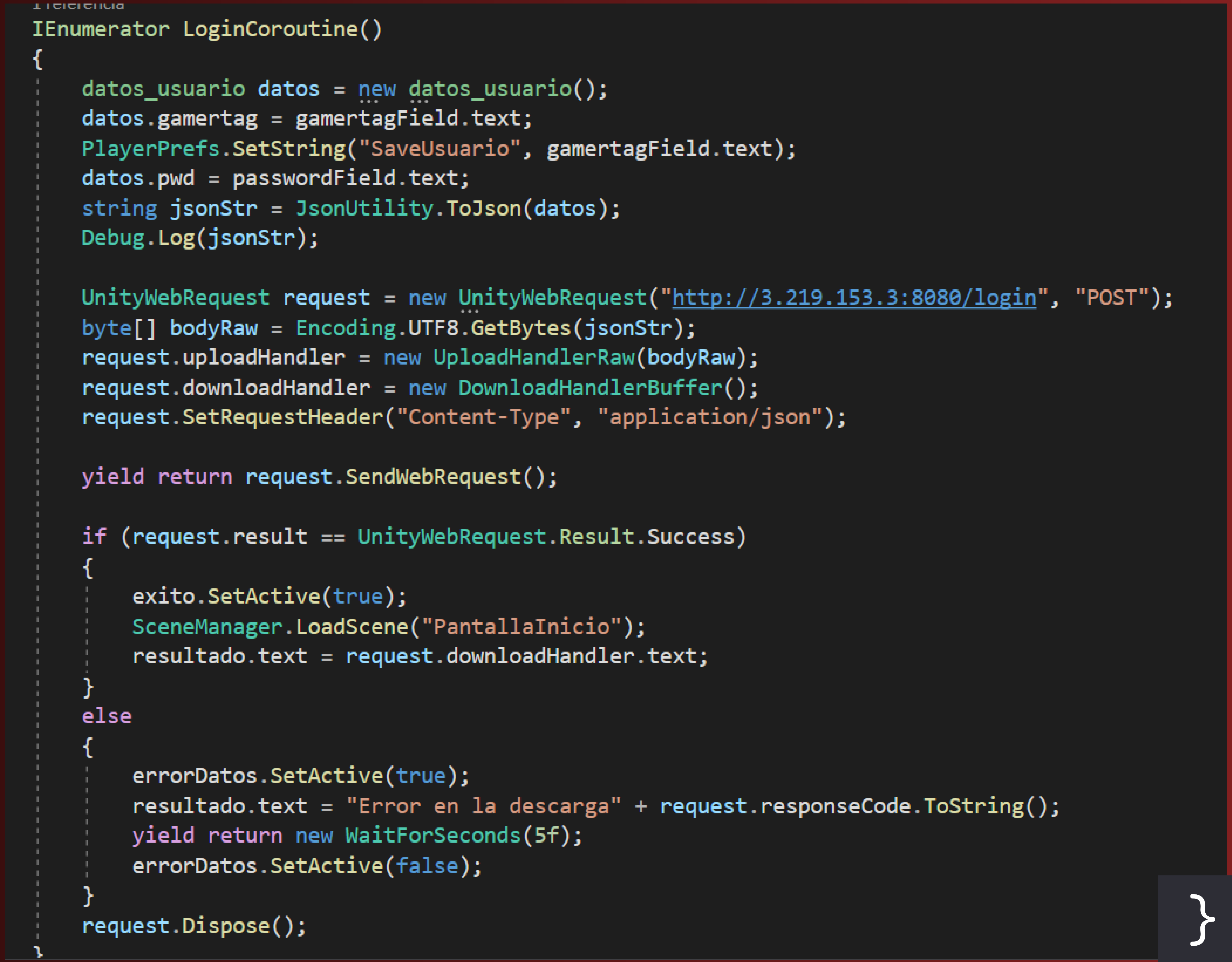
<Unity>

}

Envio de datos al iniciar sesion {



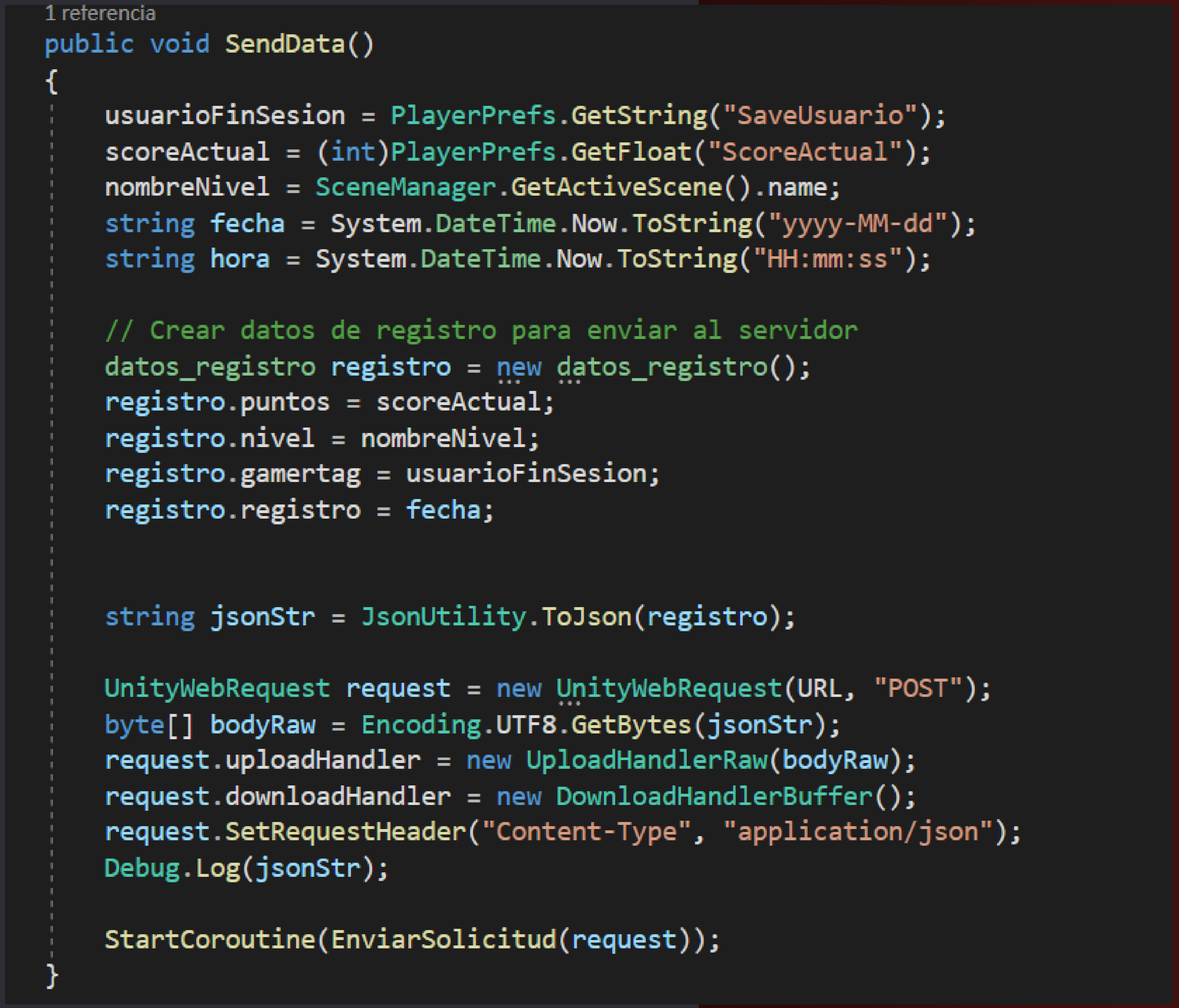
```
public struct datos_usuario
{
    public string gamertag;
    public string pwd;
}

Script de Unity (1 referencia de recurso) | 0 referencias
public class InicioSesion : MonoBehaviour
{
    public string url = "http://3.219.153.3:8080/login";
    public TMP_InputField gamertagField;
    public TMP_InputField passwordField;
    public TextMeshProUGUI resultado;
    public GameObject exito;
    public GameObject errorDatos;

    0 referencias
    public void LoginButtonOnClick()
    {
        StartCoroutine(LoginCoroutine());
    }
```



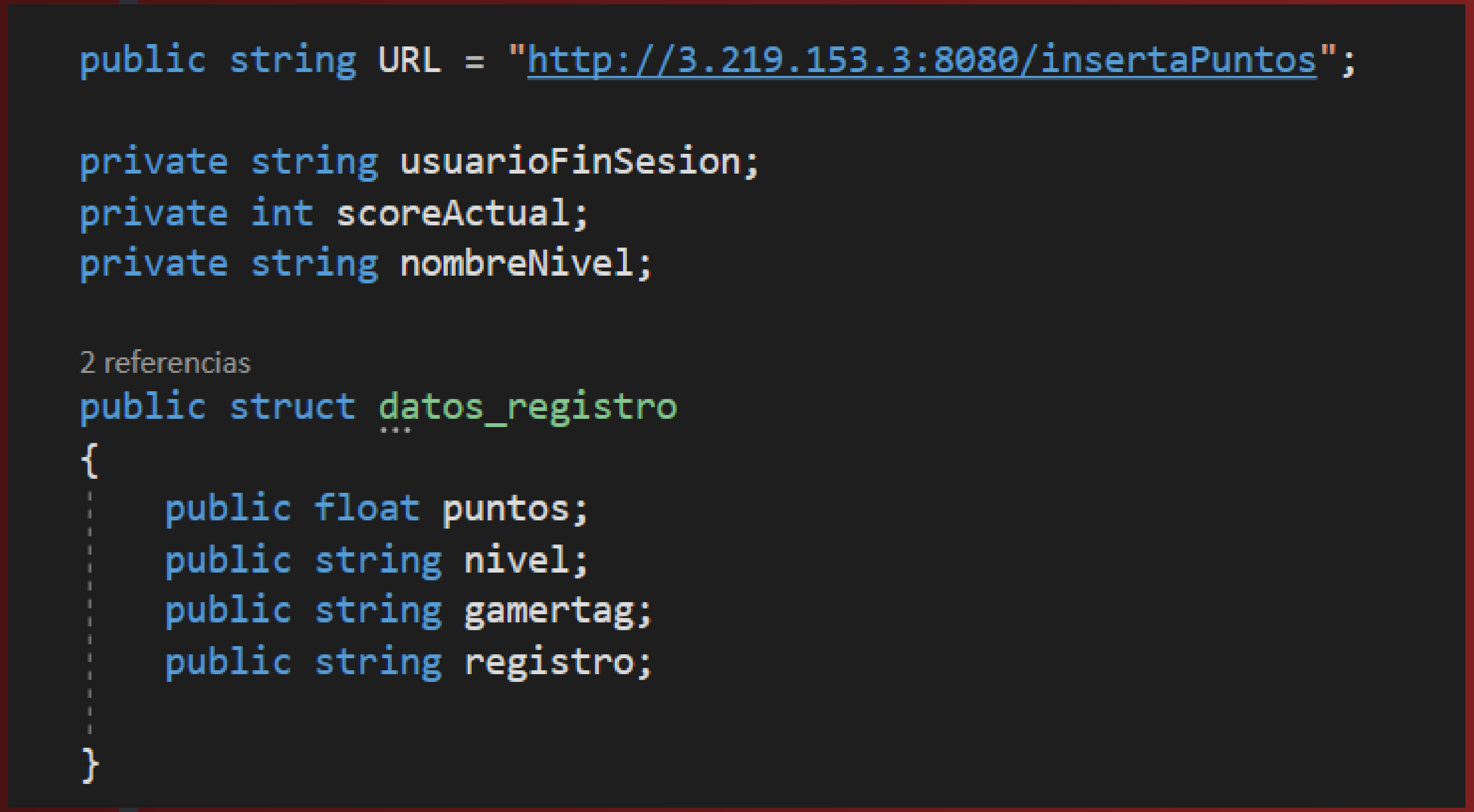
```
1 referencia
IEnumerator LoginCoroutine()
{
    datos_usuario datos = new datos_usuario();
    datos.gamertag = gamertagField.text;
    PlayerPrefs.SetString("SaveUsuario", gamertagField.text);
    datos.pwd = passwordField.text;
    string jsonStr = JsonUtility.ToJson(datos);
    Debug.Log(jsonStr);

    UnityWebRequest request = new UnityWebRequest("http://3.219.153.3:8080/login", "POST");
    byte[] bodyRaw = Encoding.UTF8.GetBytes(jsonStr);
    request.uploadHandler = new UploadHandlerRaw(bodyRaw);
    request.downloadHandler = new DownloadHandlerBuffer();
    request.SetRequestHeader("Content-Type", "application/json");

    yield return request.SendWebRequest();

    if (request.result == UnityWebRequest.Result.Success)
    {
        exito.SetActive(true);
        SceneManager.LoadScene("PantallaInicio");
        resultado.text = request.downloadHandler.text;
    }
    else
    {
        errorDatos.SetActive(true);
        resultado.text = "Error en la descarga" + request.responseCode.ToString();
        yield return new WaitForSeconds(5f);
        errorDatos.SetActive(false);
    }
    request.Dispose();
}
```

}

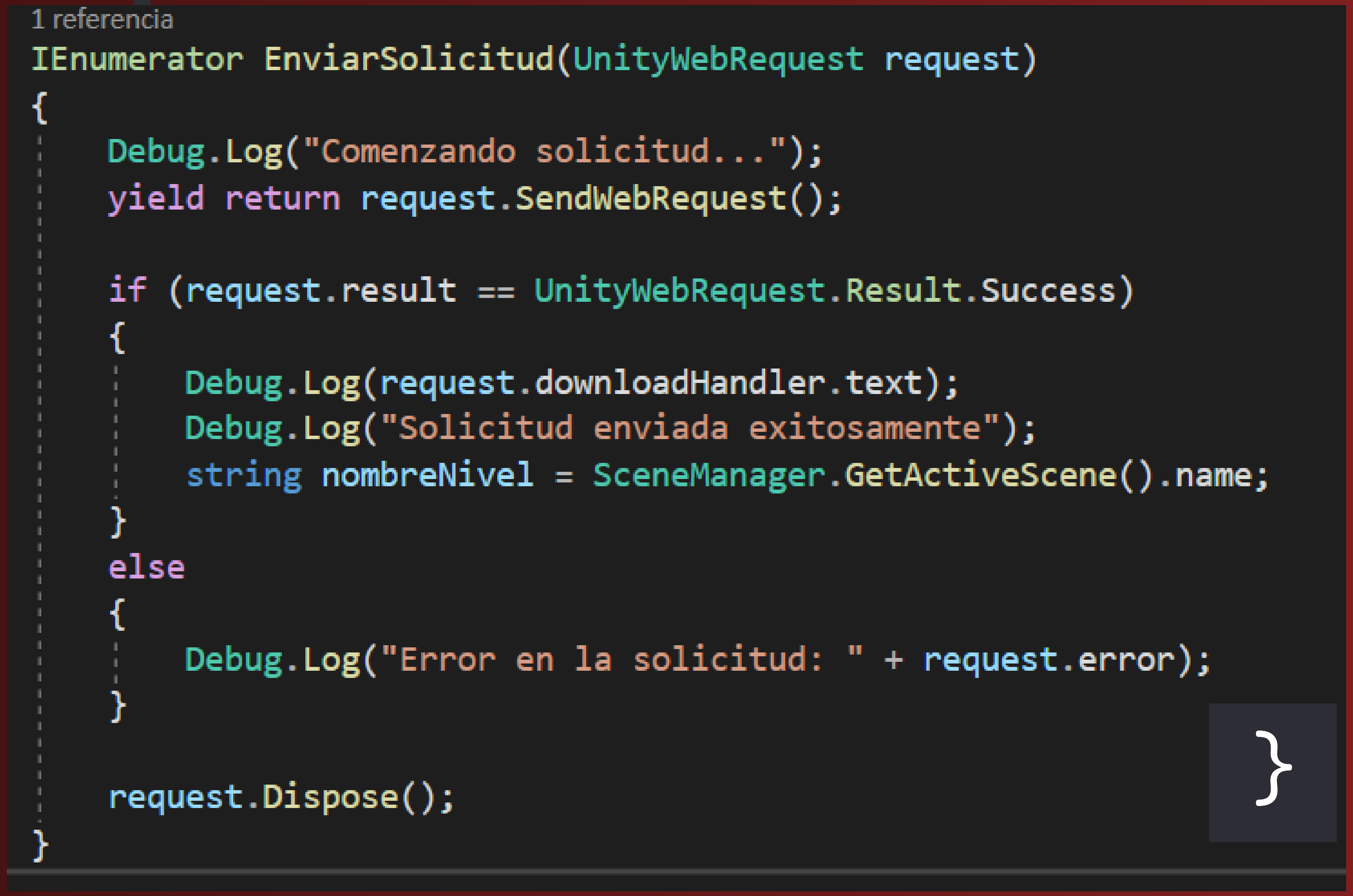
Envio de datos sobre niveles y puntuacion dentro del el juego {



```
1 referencia
public void SendData()
{
    usuarioFinSesion = PlayerPrefs.GetString("SaveUsuario");
    scoreActual = (int)PlayerPrefs.GetFloat("ScoreActual");
    nombreNivel = SceneManager.GetActiveScene().name;
    string fecha = System.DateTime.Now.ToString("yyyy-MM-dd");
    string hora = System.DateTime.Now.ToString("HH:mm:ss");

    // Crear datos de registro para enviar al servidor
    datos_registro registro = new datos_registro();
    registro.puntos = scoreActual;
    registro.nivel = nombreNivel;
    registro.gamertag = usuarioFinSesion;
    registro.registro = fecha;


    string jsonStr = JsonUtility.ToJson(registro);

    UnityWebRequest request = new UnityWebRequest(URL, "POST");
    byte[] bodyRaw = Encoding.UTF8.GetBytes(jsonStr);
    request.uploadHandler = new UploadHandlerRaw(bodyRaw);
    request.downloadHandler = new DownloadHandlerBuffer();
    request.SetRequestHeader("Content-Type", "application/json");
    Debug.Log(jsonStr);

    StartCoroutine(EnviarSolicitud(request));
}
```



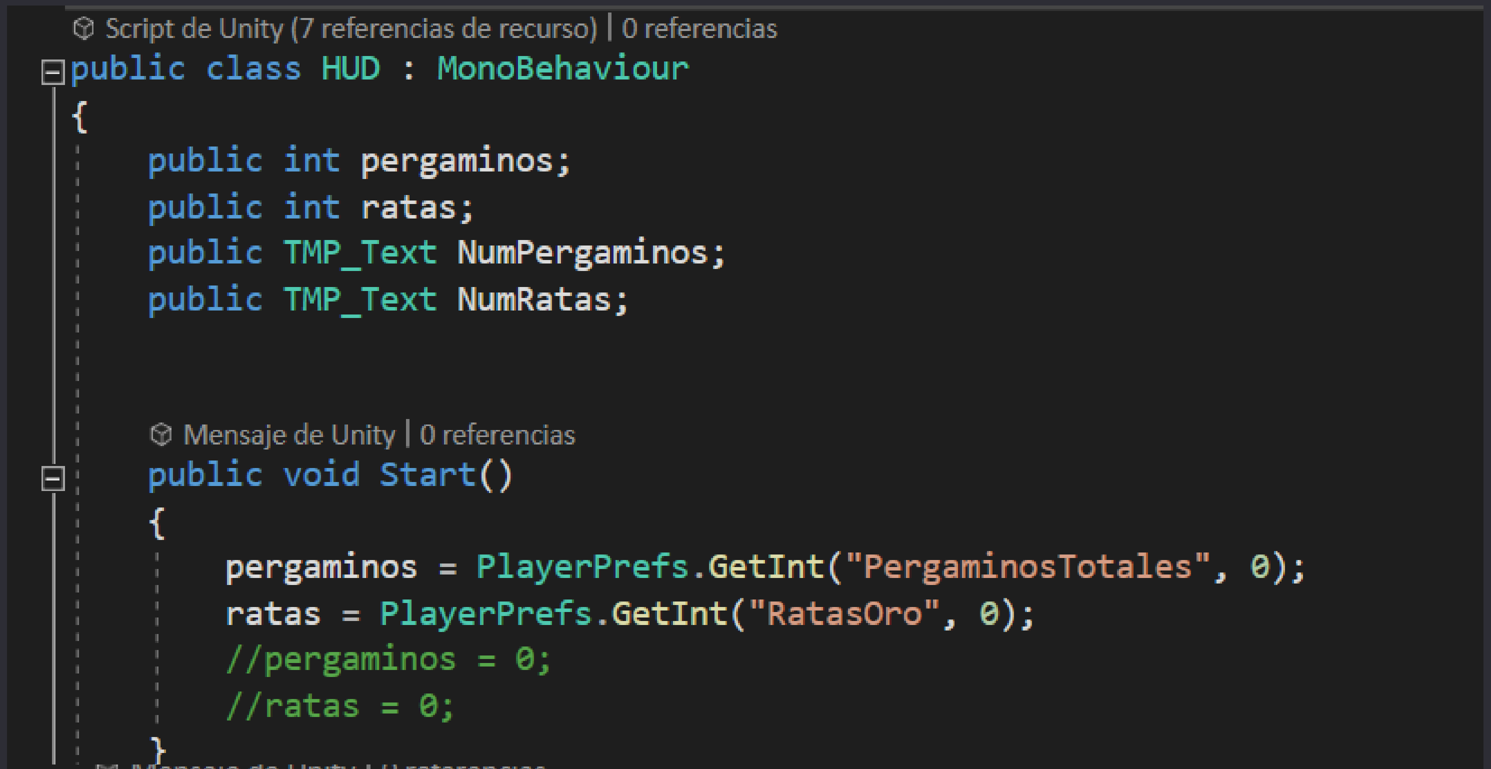
```
public string URL = "http://3.219.153.3:8080/insertaPuntos";

private string usuarioFinSesion;
private int scoreActual;
private string nombreNivel;

2 referencias
public struct datos_registro
{
    public float puntos;
    public string nivel;
    public string gamertag;
    public string registro;

}
```



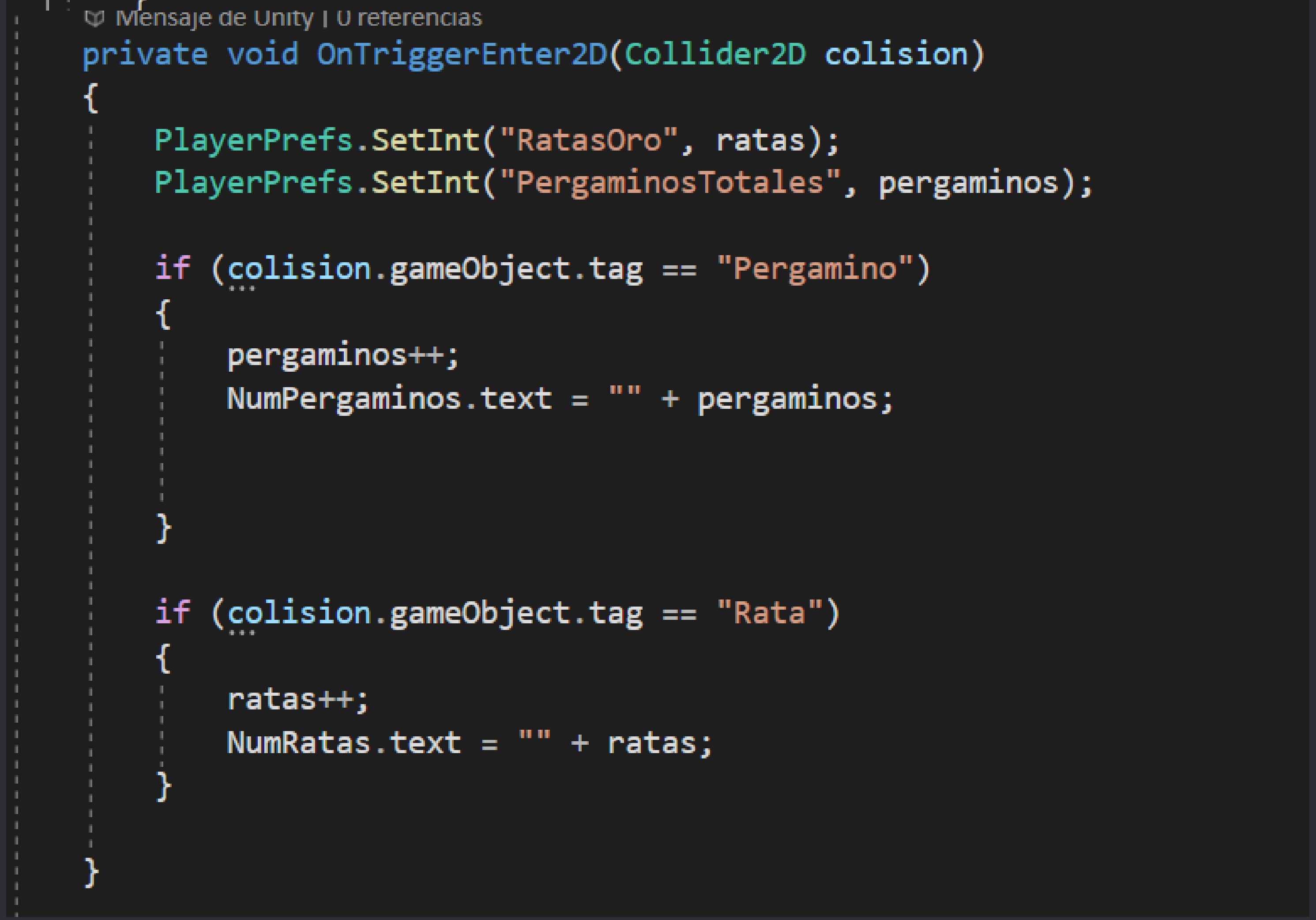
```
1 referencia
IEnumerator EnviarSolicitud(UnityWebRequest request)
{
    Debug.Log("Comenzando solicitud...");
    yield return request.SendWebRequest();

    if (request.result == UnityWebRequest.Result.Success)
    {
        Debug.Log(request.downloadHandler.text);
        Debug.Log("Solicitud enviada exitosamente");
        string nombreNivel = SceneManager.GetActiveScene().name;
    }
    else
    {
        Debug.Log("Error en la solicitud: " + request.error);
    }

    request.Dispose();
}
```

}

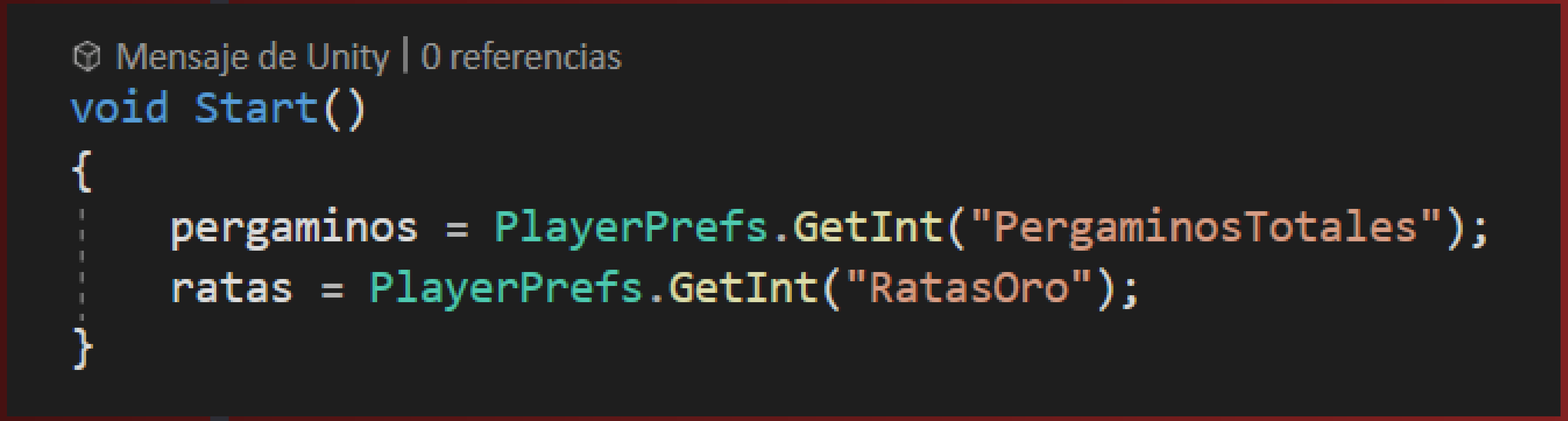
Guardar items importantes dentro del juego {



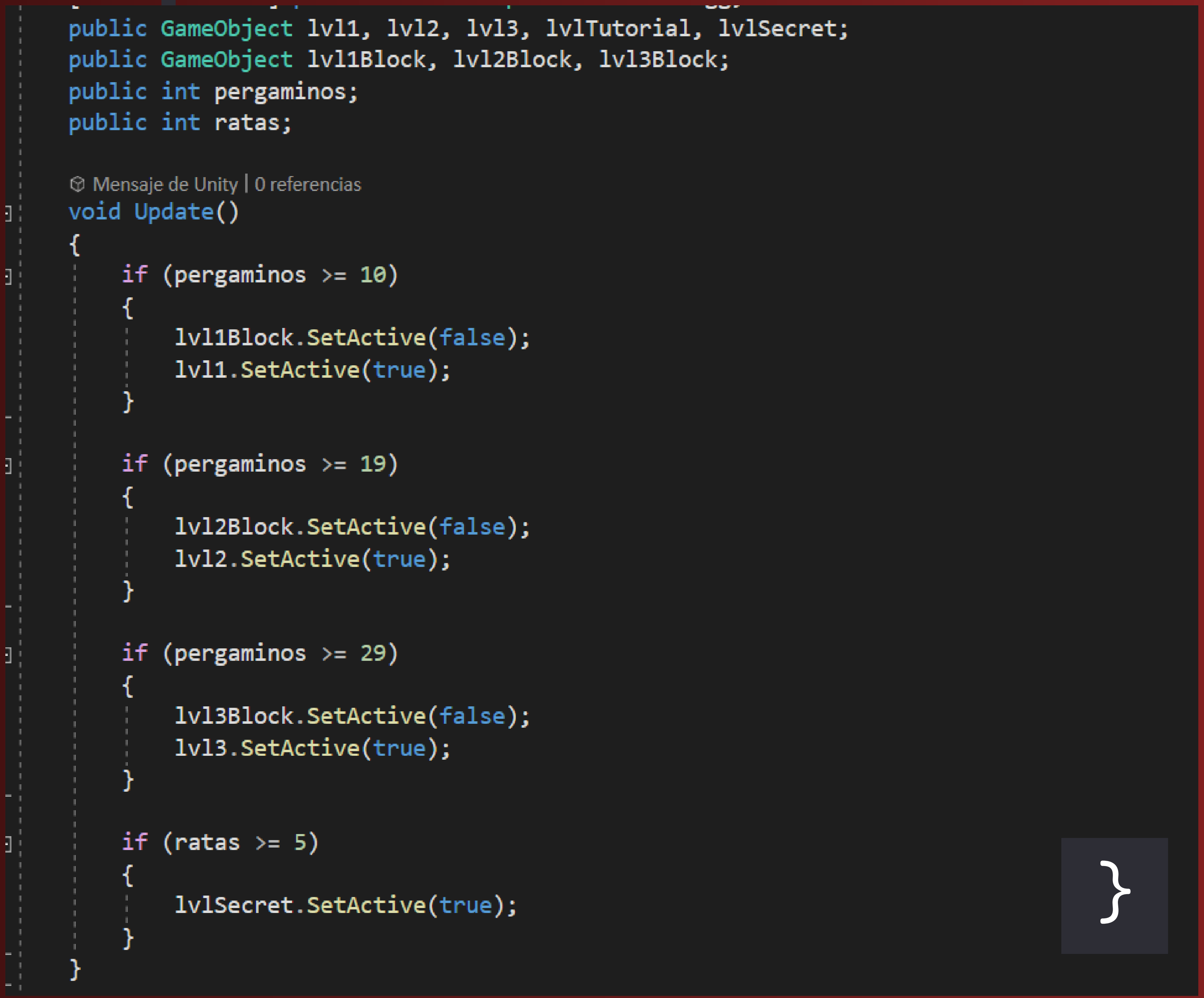
```
Script de Unity (7 referencias de recurso) | 0 referencias
public class HUD : MonoBehaviour
{
    public int pergaminos;
    public int ratas;
    public TMP_Text NumPergaminos;
    public TMP_Text NumRatas;


    Mensaje de Unity | 0 referencias
    public void Start()
    {
        pergaminos = PlayerPrefs.GetInt("PergaminosTotales", 0);
        ratas = PlayerPrefs.GetInt("RatasOro", 0);
        //pergaminos = 0;
        //ratas = 0;
    }
```



```
  Mensaje de Unity | 0 referencias
  private void OnTriggerEnter2D(Collider2D colision)
  {
      PlayerPrefs.SetInt("RatasOro", ratas);
      PlayerPrefs.SetInt("PergaminosTotales", pergaminos);

      if (colision.gameObject.tag == "Pergamino")
      {
          pergaminos++;
          NumPergaminos.text = "" + pergaminos;


      }

      if (colision.gameObject.tag == "Rata")
      {
          ratas++;
          NumRatas.text = "" + ratas;
      }

  }
```



```
Mensaje de Unity | 0 referencias
void Start()
{
    pergaminos = PlayerPrefs.GetInt("PergaminosTotales");
    ratas = PlayerPrefs.GetInt("RatasOro");
}
```



```
public GameObject lvl1, lvl2, lvl3, lvlTutorial, lvlSecret;
public GameObject lvl1Block, lvl2Block, lvl3Block;
public int pergaminos;
public int ratas;

Mensaje de Unity | 0 referencias
void Update()
{
    if (pergaminos >= 10)
    {
        lvl1Block.SetActive(false);
        lvl1.SetActive(true);
    }

    if (pergaminos >= 19)
    {
        lvl2Block.SetActive(false);
        lvl2.SetActive(true);
    }

    if (pergaminos >= 29)
    {
        lvl3Block.SetActive(false);
        lvl3.SetActive(true);
    }

    if (ratas >= 5)
    {
        lvlSecret.SetActive(true);
    }
}
```

}

Gracias por su atencion {

<Equipo 4>

}