

PREGUNTA NUMERO 1

```
--numero 01
-- Crear una vista para combinar Estudiantes, Inscripciones y Cursos
CREATE VIEW VistaEstudiantesInscripciones AS
SELECT
    e.EstudianteID,
    e.Nombre AS NombreEstudiante,
    e.Apellido,
    c.NombreCurso,
    c.Creditos,
    i.FechaInscripcion
FROM
    Estudiantes e
JOIN
    Inscripciones i ON e.EstudianteID = i.EstudianteID
JOIN
    Cursos c ON i.CursoID = c.CursoID;
GO

-- Consultar la vista creada
SELECT * FROM VistaEstudiantesInscripciones;
```

100 %							
		Results		Messages			
	EstudianteID	NombreEstudiante	Apellido	NombreCurso	Creditos	FechaInscripcion	
1	1	Juan	Perez	Matemáticas	4	2024-01-15	
2	2	Maria	Lopez	Historia	3	2024-01-16	
3	3	Luis	Martinez	Ciencias	4	2024-01-17	
4	4	Ana	Gomez	Filosofía	3	2024-01-18	
5	5	Carlos	Diaz	Programación	5	2024-01-19	
6	1	Juan	Perez	Matemáticas	4	2024-01-15	
7	2	Maria	Lopez	Historia	3	2024-01-16	
8	3	Luis	Martinez	Ciencias	4	2024-01-17	
9	4	Ana	Gomez	Filosofía	3	2024-01-18	
10	5	Carlos	Diaz	Programación	5	2024-01-19	
11	6	Elena	Fernandez	Matemáticas	4	2024-01-20	
12	7	Pedro	Ramirez	Historia	3	2024-01-21	
13	8	Sofia	Torres	Ciencias	4	2024-01-22	
14	9	Javier	Gutierrez	Filosofía	3	2024-01-23	
15	10	Clara	Jimenez	Programación	5	2024-01-24	
16	11	Raul	Hernandez	Matemáticas	4	2024-01-25	
17	12	Laura	Garcia	Historia	3	2024-01-26	
18	13	Miguel	Rodriguez	Ciencias	4	2024-01-27	
19	14	Natalia	Sanchez	Filosofía	3	2024-01-28	

✓ Query executed successfully.

PREGUNTA NUMERO 2

```
-- NUMERO 2
CREATE PROCEDURE InsertarEstudiante
    @Nombre NVARCHAR(100),
    @Apellido NVARCHAR(100),
    @FechaNacimiento DATE
AS
BEGIN
    INSERT INTO Estudiantes (Nombre, Apellido, FechaNacimiento)
    VALUES (@Nombre, @Apellido, @FechaNacimiento);
END;
GO

-- Ejecutar el procedimiento para insertar un estudiante
EXEC InsertarEstudiante 'Juan', 'Pérez', '2000-05-15';
```

PREGUNTA NUMERO 3

```
-- NUMERO 3
CREATE PROCEDURE EliminarEstudiante
    @EstudianteID INT
AS
BEGIN
    DELETE FROM Estudiantes
    WHERE EstudianteID = @EstudianteID;
END;
GO

-- Ejecutar el procedimiento para eliminar un estudiante
EXEC EliminarEstudiante 1;
```

PREGUNTA NUMERO 4

```
--PREGUNTA NUMERO 4
CREATE PROCEDURE ActualizarEstudiante
    @EstudianteID INT,
    @NuevoNombre NVARCHAR(100),
    @NuevoApellido NVARCHAR(100),
    @NuevaFechaNacimiento DATE
AS
BEGIN
    UPDATE Estudiantes
    SET Nombre = @NuevoNombre,
        Apellido = @NuevoApellido,
        FechaNacimiento = @NuevaFechaNacimiento
    WHERE EstudianteID = @EstudianteID;
END;
GO

-- Ejecutar el procedimiento para actualizar datos de un estudiante
EXEC ActualizarEstudiante 1, 'Carlos', 'Gómez', '1999-10-20';
```

PREGUNTA NUMERO 5

```
-- NUMERO 5
CREATE PROCEDURE CalcularCreditosPorEstudiante
    @EstudianteID INT
AS
BEGIN
    SELECT
        e.EstudianteID,
        e.Nombre AS NombreEstudiante,
        SUM(c.Creditos) AS TotalCreditos
    FROM
        Inscripciones i
    JOIN
        Cursos c ON i.CursoID = c.CursoID
    JOIN
        Estudiantes e ON i.EstudianteID = e.EstudianteID
    WHERE
        e.EstudianteID = @EstudianteID
    GROUP BY
        e.EstudianteID, e.Nombre;
END;
GO

-- Ejecutar el procedimiento para calcular créditos
EXEC CalcularCreditosPorEstudiante 1;
```

100 %

Results Messages

	EstudianteID	NombreEstudiante	TotalCreditos
1	1	Carlos	8

PREGUNTA NUMERO 6

```
--NUMERO 6
CREATE TABLE AuditoriaInscripciones (
    AuditoriaID INT IDENTITY(1,1) PRIMARY KEY,
    EstudianteID INT,
    CursoID INT,
    FechaInscripcion DATE,
    FechaRegistro DATETIME
);

CREATE TRIGGER InsertarAuditoria
ON Incripciones
AFTER INSERT
AS
BEGIN
    INSERT INTO AuditoriaInscripciones (EstudianteID, CursoID, FechaInscripcion, FechaRegistro)
    SELECT
        EstudianteID,
        CursoID,
        FechaInscripcion,
        GETDATE()
    FROM
        inserted;
END;
GO

-- Probar el disparador insertando una inscripción
INSERT INTO Incripciones (EstudianteID, CursoID, FechaInscripcion)
VALUES (1, 2, GETDATE());
```

PREGUNTA NUMERO 7

```
--NUMERO 7
CREATE TRIGGER EliminarInscripcionesRelacionadas
ON Cursos
AFTER DELETE
AS
BEGIN
    DELETE FROM Incripciones
    WHERE CursoID IN (SELECT CursoID FROM deleted);
END;
GO

-- Probar el disparador eliminando un curso
DELETE FROM Cursos WHERE CursoID = 1;
```

PREGUNTA NUMERO 8

```
--NUMERO 8
-- Agregar columna para registrar la última actualización
ALTER TABLE Estudiantes ADD UltimaActualizacion DATETIME;

-- Crear el disparador
CREATE TRIGGER ActualizarUltimaModificacion
ON Estudiantes
AFTER UPDATE
AS
BEGIN
    UPDATE Estudiantes
    SET UltimaActualizacion = GETDATE()
    WHERE EstudianteID IN (SELECT EstudianteID FROM inserted);
END;
GO

-- Probar el disparador
UPDATE Estudiantes
SET Apellido = 'Martínez'
WHERE EstudianteID = 1;

-- Verificar la actualización
SELECT * FROM Estudiantes;
```

100 %						
		Results		Messages		
	EstudianteID	Nombre	Apellido	FechaNacimiento	UltimaActualizacion	
1	1	Carlos	Martínez	1999-10-20	2024-11-21 11:15:48.250	
2	2	Maria	Lopez	1999-05-15	NULL	
3	3	Luis	Martinez	2001-03-22	NULL	
4	4	Ana	Gomez	2000-07-18	NULL	
5	5	Carlos	Diaz	1998-12-30	NULL	
6	6	Elena	Fernandez	1999-08-10	NULL	
7	7	Pedro	Ramirez	2002-04-05	NULL	
8	8	Sofia	Torres	2001-11-09	NULL	
9	9	Javier	Gutierrez	2000-06-17	NULL	
10	10	Clara	Jimenez	1998-02-20	NULL	
11	11	Raul	Hernandez	2000-09-12	NULL	

✓ Query executed successfully.

PREGUNTA NUMERO 9

```

--NUMERO 9
CREATE TRIGGER ValidarFechaInscripcion
ON Inscripciones
INSTEAD OF INSERT, UPDATE
AS
BEGIN
    -- Validar que la fecha de inscripción no sea futura
    IF EXISTS (
        SELECT *
        FROM inserted
        WHERE FechaInscripcion > GETDATE()
    )
    BEGIN
        RAISERROR ('La fecha de inscripción no puede ser futura.', 16, 1);
        ROLLBACK TRANSACTION;
    END
    ELSE
    BEGIN
        -- Si los datos son válidos, insertar o actualizar
        IF EXISTS (SELECT * FROM inserted)
        BEGIN
            -- Operación INSERT o UPDATE válida
            MERGE Inscripciones AS target
            USING inserted AS source
            ON target.InscripcionID = source.InscripcionID
            WHEN MATCHED THEN
                UPDATE SET EstudianteID = source.EstudianteID,
                           CursoID = source.CursoID,
                           FechaInscripcion = source.FechaInscripcion
            WHEN NOT MATCHED BY TARGET THEN
                INSERT (EstudianteID, CursoID, FechaInscripcion)
                VALUES (source.EstudianteID, source.CursoID, source.FechaInscripcion);
        END
    END
END;
GO

INSERT INTO Inscripciones (EstudianteID, CursoID, FechaInscripcion)
VALUES (1, 1, DATEADD(DAY, 1, GETDATE())); -- Esto generará un error

INSERT INTO Inscripciones (EstudianteID, CursoID, FechaInscripcion)
VALUES (1, 1, GETDATE()); -- Esto será válido

```

%

Messages

(1 row affected)

(1 row affected)

(1 row affected)

Completion time: 2024-11-21T11:21:18.1558107-05:00

PREGUNTA NUMERO 10


```

--NUMERO 10
-- Crear los usuarios en el servidor
CREATE LOGIN UsuarioJeremy WITH PASSWORD = 'Password123!';
CREATE LOGIN UsuarioAngeles WITH PASSWORD = 'Password123!';
CREATE LOGIN UsuarioJaztin WITH PASSWORD = 'Password123!';
CREATE LOGIN UsuarioBrayan WITH PASSWORD = 'Password123!';

-- Asignar usuarios a la base de datos
USE GestionEstudiantes;
CREATE USER jeremy FOR LOGIN UsuarioJeremy;
CREATE USER Anfeles FOR LOGIN UsuarioAngeles;
CREATE USER jaztin FOR LOGIN UsuarioJaztin;
CREATE USER Brayan FOR LOGIN UsuarioBrayan;

-- Verificar usuarios
SELECT name AS Usuario, create_date AS FechaCreacion
FROM sys.database_principals
WHERE type = 'S';

```

%

Results Messages

Usuario	FechaCreacion
INFORMATION_SCHEMA	2009-04-13 12:59:11.717
sys	2009-04-13 12:59:11.717
jeremy	2024-11-21 11:24:24.527
Anfeles	2024-11-21 11:24:24.530
jaztin	2024-11-21 11:24:24.530
Brayan	2024-11-21 11:24:24.530

PREGUNTA NUMERO 11

```


--NUMERO 11
-- Realizar una copia de seguridad de la base de datos original
BACKUP DATABASE GestionEstudiantes
TO DISK = 'C:\Backups\GestionEstudiantes.bak'
WITH FORMAT, INIT;

-- Restaurar la base de datos con un nuevo nombre
RESTORE DATABASE GestionEstudiantesCopia
FROM DISK = 'C:\Backups\GestionEstudiantes.bak'
WITH MOVE 'GestionEstudiantes' TO 'C:\Data\GestionEstudiantesCopia.mdf',
      MOVE 'GestionEstudiantes_log' TO 'C:\Data\GestionEstudiantesCopia_log.ldf',
      REPLACE;
GO

```

Compartir Vista

> Este equipo > Windows (C:) > Backups

Nombre	Fecha de modificación	Tipo	Tamaño
 GestionEstudiantes	21/11/2024 11:26	Archivo BAK	4,825 KB

Buscar en Backups

PREGUNTA NUMERO 12

```
--NUMERO 12
-- Crear copia de seguridad completa
BACKUP DATABASE GestionEstudiantes
TO DISK = 'C:\Backups\GestionEstudiantes.bak'
WITH FORMAT, INIT;

-- Verificar la copia de seguridad
RESTORE VERIFYONLY
FROM DISK = 'C:\Backups\GestionEstudiantes.bak';
```

PREGUNTA NUMERO 13

```
--NUMERO 13
-- Crear una clave maestra
CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'ClaveFuerte123!';

-- Crear un certificado
CREATE CERTIFICATE CertificadoEstudiantes
WITH SUBJECT = 'Certificado para encriptar datos de Estudiantes';

-- Crear una clave simétrica
CREATE SYMMETRIC KEY ClaveEstudiantes
WITH ALGORITHM = AES_256
ENCRYPTION BY CERTIFICATE CertificadoEstudiantes;

-- Abrir la clave simétrica
OPEN SYMMETRIC KEY ClaveEstudiantes
DECRYPTION BY CERTIFICATE CertificadoEstudiantes;

-- Actualizar los datos encriptados
UPDATE Estudiantes
SET Nombre = EncryptByKey(Key_GUID('ClaveEstudiantes'), Nombre),
    Apellido = EncryptByKey(Key_GUID('ClaveEstudiantes'), Apellido);

-- Cerrar la clave
CLOSE SYMMETRIC KEY ClaveEstudiantes;
GO

-- Abrir la clave simétrica
OPEN SYMMETRIC KEY ClaveEstudiantes
DECRYPTION BY CERTIFICATE CertificadoEstudiantes;

-- Seleccionar los datos desencriptados
SELECT
    EstudianteID,
    CAST(DecryptByKey(Nombre) AS NVARCHAR(100)) AS Nombre,
    CAST(DecryptByKey(Apellido) AS NVARCHAR(100)) AS Apellido,
    FechaNacimiento
FROM Estudiantes;

-- Cerrar la clave
CLOSE SYMMETRIC KEY ClaveEstudiantes;
GO
```

0 %

Results		Messages		
	EstudianteID	Nombre	Apellido	FechaNacimiento
1	1	Carlos	Martínez	1999-10-20
2	2	Maria	Lopez	1999-05-15
3	3	Luis	Martinez	2001-03-22
4	4	Ana	Gomez	2000-07-18
5	5	Carlos	Diaz	1998-12-30
6	6	Elena	Fernandez	1999-08-10
7	7	Pedro	Ramirez	2002-04-05

PREGUNTA NUMERO 14

```
--NUMEOR 14
-- Denegar acceso a la columna FechaNacimiento
DENY SELECT ON Estudiantes(FechaNacimiento) TO Jeremy;

-- Probar la restricción
EXECUTE AS USER = 'Jeremy';
SELECT EstudianteID, Nombre, FechaNacimiento FROM Estudiantes; -- Fallará
REVERT;

-- Permitir acceso nuevamente (opcional)
GRANT SELECT ON Estudiantes(FechaNacimiento) TO Jeremy;
```

PREGUNTA NUMERO 15

```
--NUMERO 15
-- Denegar acceso a la columna FechaNacimiento
DENY SELECT ON Estudiantes(FechaNacimiento) TO Jeremy;

-- Verificar la restricción (cambiar a contexto del usuario Jeremy)
EXECUTE AS USER = 'Jeremy';
SELECT EstudianteID, Nombre, FechaNacimiento FROM Estudiantes; -- Este debería generar un error
REVERT;
```

0 %
Messages

Msg 229, Level 14, State 5, Line 598
The SELECT permission was denied on the object 'Estudiantes', database 'GestionEstudiantes', schema 'dbo'.

Completion time: 2024-11-21T11:36:47.5724537-05:00

PREGUNTA NUMERO 16

```

--NUMERO 16
-- Crear un certificado para el TDE
CREATE CERTIFICATE CertificadoTDE
WITH SUBJECT = 'Certificado para TDE';

-- Crear una clave de cifrado
CREATE DATABASE ENCRYPTION KEY
WITH ALGORITHM = AES_256
ENCRYPTION BY SERVER CERTIFICATE CertificadoTDE;

-- Habilitar el cifrado en la base de datos
ALTER DATABASE GestionEstudiantes
SET ENCRYPTION ON;

```

PREGUNTA NUMERO 17

```

--NUMERO 17
-- Otorgar permisos
GRANT SELECT, INSERT, UPDATE, DELETE ON Estudiantes TO Brayan ;
GRANT SELECT, INSERT, UPDATE, DELETE ON Cursos TO Brayan;
GRANT SELECT, INSERT, UPDATE, DELETE ON Inscripciones TO Brayan;

-- Verificar permisos
EXECUTE AS USER = 'Brayan';
SELECT * FROM Estudiantes; --
REVERT;

```

Results					
Messages					
EstudianteID	Nombre	Apellido	FechaNacimiento	UltimaActualizacion	
1	Brayan	Brayan	1999-10-20	2024-11-21 11:30:54.393	
2	Brayan	Brayan	1999-05-15	2024-11-21 11:30:54.393	
3	Brayan	Brayan	2001-03-22	2024-11-21 11:30:54.393	
4	Brayan	Brayan	2000-07-18	2024-11-21 11:30:54.393	
5	Brayan	Brayan	1998-12-30	2024-11-21 11:30:54.393	
6	Brayan	Brayan	1999-08-10	2024-11-21 11:30:54.393	
7	Brayan	Brayan	2002-04-05	2024-11-21 11:30:54.393	

Query executed successfully.

PREGUNTA NUMERO 18

```
--NUMERO 18
CREATE PROCEDURE InsertarEstudianteSeguro
    @Nombre NVARCHAR(100),
    @Apellido NVARCHAR(100),
    @FechaNacimiento DATE
AS
BEGIN
    -- Verificar entradas maliciosas
    IF @Nombre LIKE '%''%' OR @Nombre LIKE '%--%' OR @Nombre LIKE '%;%' OR
        @Apellido LIKE '%''%' OR @Apellido LIKE '%--%' OR @Apellido LIKE '%;%'
    BEGIN
        RAISERROR ('Caracteres no permitidos detectados.', 16, 1);
        RETURN;
    END

    -- Insertar datos seguros
    INSERT INTO Estudiantes (Nombre, Apellido, FechaNacimiento)
    VALUES (@Nombre, @Apellido, @FechaNacimiento);
END;
GO

-- Probar con datos seguros
EXEC InsertarEstudianteSeguro 'Juan', 'Gómez', '2000-01-01'; -- Correcto

-- Probar con datos maliciosos
EXEC InsertarEstudianteSeguro 'Juan'; DROP TABLE Estudiantes;--, 'Gómez', '2000-01-01'; -- Fallará
```

) %

Messages

Msg 50000, Level 16, State 1, Procedure InsertarEstudianteSeguro, Line 11 [Batch Start Line 650]
Caracteres no permitidos detectados.

Completion time: 2024-11-21T11:41:28.9979483-05:00

PREGUNTA NUMERO 19

```
--NUMERO 19
CREATE PROCEDURE InsertarEstudianteConValidacion
    @Nombre NVARCHAR(100),
    @Apellido NVARCHAR(100),
    @FechaNacimiento DATE
AS
BEGIN
    -- Validar formato
    IF LEN(@Nombre) > 100 OR LEN(@Apellido) > 100
    BEGIN
        RAISERROR ('El nombre o apellido exceden la longitud máxima.', 16, 1);
        RETURN;
    END

    IF ISNUMERIC(@Nombre) = 1 OR ISNUMERIC(@Apellido) = 1
    BEGIN
        RAISERROR ('El nombre o apellido no deben contener números.', 16, 1);
        RETURN;
    END

    -- Insertar datos válidos
    INSERT INTO Estudiantes (Nombre, Apellido, FechaNacimiento)
    VALUES (@Nombre, @Apellido, @FechaNacimiento);
END;
GO

-- Probar el procedimiento
EXEC InsertarEstudianteConValidacion 'Pedro', 'López', '1999-05-10'; -- Correcto
EXEC InsertarEstudianteConValidacion '123', '456', '1999-05-10'; -- Fallará
```

Messages

```
sg 50000, Level 16, State 1, Procedure InsertarEstudianteConValidacion, Line 16 [Batch Start Line 680]
1 nombre o apellido no deben contener números.
```

ompletion time: 2024-11-21T11:42:16.9824969-05:00

PREGUNTA NUMERO 20

```
--NUMERO 20
CREATE TABLE Auditoria (
    AuditoriaID INT IDENTITY(1,1) PRIMARY KEY,
    Tabla NVARCHAR(50),
    Operacion NVARCHAR(50),
    RegistroID INT,
    Fecha DATETIME,
    Usuario NVARCHAR(50)
);

-- Auditar inserciones en Estudiantes
CREATE TRIGGER AuditarInserciones
ON Estudiantes
AFTER INSERT
AS
BEGIN
    INSERT INTO Auditoria (Tabla, Operacion, RegistroID, Fecha, Usuario)
    SELECT 'Estudiantes', 'INSERT', EstudianteID, GETDATE(), SYSTEM_USER
    FROM inserted;
END;

-- Auditar actualizaciones en Estudiantes
CREATE TRIGGER AuditarActualizaciones
ON Estudiantes
AFTER UPDATE
AS
BEGIN
    INSERT INTO Auditoria (Tabla, Operacion, RegistroID, Fecha, Usuario)
    SELECT 'Estudiantes', 'UPDATE', EstudianteID, GETDATE(), SYSTEM_USER
    FROM inserted;
END;

-- Auditar eliminaciones en Estudiantes
CREATE TRIGGER AuditarEliminaciones
ON Estudiantes
AFTER DELETE
AS
BEGIN
    INSERT INTO Auditoria (Tabla, Operacion, RegistroID, Fecha, Usuario)
    SELECT 'Estudiantes', 'DELETE', EstudianteID, GETDATE(), SYSTEM_USER
    FROM deleted;
END;
GO
```

%

PREGUNTA NUMERO 21

```
--NUMERO 21
-- Configurar memoria máxima (en MB)
EXEC sp_configure 'max server memory (MB)', 2048;
RECONFIGURE;

-- Configurar archivo de datos con tamaño máximo
ALTER DATABASE GestionEstudiantes
MODIFY FILE (NAME = 'GestionEstudiantes', MAXSIZE = 500MB);
GO
```

Messages



Commands completed successfully.

Completion time: 2024-11-21T11:44:28.7698839-05:00

PREGUNTA NUMERO 22

```
--NUMERO 22
BACKUP DATABASE GestionEstudiantes
TO DISK = 'C:\Backups\GestionEstudiantes_Full.bak'
WITH FORMAT, INIT;
GO
```





Este equipo > Windows (C:) > Backups

Nombre	Fecha de modificación	Tipo	Tamaño
 GestionEstudiantes	21/11/2024 11:28	Archivo BAK	4,825 KB
 GestionEstudiantes_Full	21/11/2024 11:44	Archivo BAK	5,465 KB

PREGUNTA NUMERO 24

--NUMERO 24

```
RESTORE DATABASE GestionEstudiantesRestaurada
FROM DISK = 'C:\Backups\GestionEstudiantes_Full.bak'
WITH MOVE 'GestionEstudiantes' TO 'C:\Data\GestionEstudiantesRestaurada.mdf',
MOVE 'GestionEstudiantes_log' TO 'C:\Data\GestionEstudiantesRestaurada_log.ldf',
REPLACE;
GO
```

nombre	fecha de modificacion	tipo	tamaño
 GestionEstudiantesRestaurada	21/11/2024 11:46	SQL Server Databa...	8,192 KB
 GestionEstudiantesRestaurada_log	21/11/2024 11:46	SQL Server Databa...	8,192 KB
 GestionEstudiantesCopia	21/11/2024 11:27	SQL Server Databa...	8,192 KB
 GestionEstudiantesCopia_log	21/11/2024 11:27	SQL Server Databa...	8,192 KB

PREGUNTA NUMERO 26

--NUMERO 26

-- Insertar datos en la base de datos primaria

```
INSERT INTO Estudiantes (Nombre, Apellido, FechaNacimiento)
VALUES ('Luis', 'Ramírez', '2000-02-15');
```

PREGUNTA NUMERO 27

--NUMERO 27

-- Otorgar permisos

```
GRANT SELECT ON Estudiantes TO Jaztin;
```

-- Programar revocación de permisos después de una hora

```
WAITFOR DELAY '00:01:00'; -- Esperar una hora
```

```
REVOKE SELECT ON Estudiantes FROM Jaztin;
```

PREGUNTA NUMERO 28

```
--NUMERO 28
-- Configuración inicial (realizar desde SQL Server Management Studio)
EXEC sp_replicationdboption @dbname = 'GestionEstudiantes',
                             @optname = 'publish',
                             @value = 'true';
```