DATA BASE

MANEJO DE PERMISOS A NIVEL DE USUARIOS

GAUNA, OCTAVIO
GIMENEZ, MAXIMILIANO
MARIN, MATIAS
LOVATO, MATIAS

GRUPO 7

Tabla de Contenidos:

1 Introducción: 2

1.1 Objetivo General: 2

1.2 Objetivos específicos: 2

2 Marco Conceptual: 3

2.1 Usuarios a nivel de servidor: 3

2.2 Usuarios a nivel de base de datos:....3

2.3 Manejo de Usuarios en SQL Server: 3

2.4 Permisos de Usuario: 4

3 Metodología Seguida: 5

3.1 Descripción de cómo se realizó el Trabajo Práctico: 5

3.2 Herramientas (Instrumentos y procedimientos): 5

3.2.1 Discord (Comunicaciones grupales):....5

3.2.2 WhatsApp (Seguimiento de avances): 5

3.2.3 Chat GPT (Apoyo en investigación y práctica): 5

3.2.4 GitHub (Gestión de código fuente): 6

3.2.5 SQL Server Management Studio (Administración de bases de datos): 6

3.2.6 Google Docs (Colaboración en documentación): 6

4 Desarrollo del Tema / Presentación de Resultado:....7

4.1 Aplicación de Otros Proyectos: 12

4.1.1 Procedimientos y Funciones Almacenadas: 12

4.1.2 Triggers de Autoría: 13

4.2 Conclusión del Desarrollo:....14

5 Conclusión: 15

6 Bibliografía: 16

1 Introducción:

En el ámbito de la gestión de datos de sistemas computacionales, las bases de datos SQL (Structured Query Language) han tenido una gran influencia en la organización y recuperación de la información de manera eficiente. Estas mismas, han ido evolucionado considerablemente a lo largo del tiempo, proporcionando a las organizaciones distintas ventajas como la capacidad de guardar y acceder a grandes volúmenes de datos de manera segura y efectiva. Sin embargo, para que esto sea así, la gestión de permisos es un aspecto fundamental que merece una atención continua y cuidadosa. Es por ello, que es importante diferenciar entre dos tipos que son, una a nivel **servidor** el cual abarca a los inicios de sesión y los roles dentro de este y la otra a nivel de **base de datos** asignados a usuarios y roles de la misma. [1], [2], [3]

El propósito de este trabajo de investigación es analizar en profundidad las prácticas actuales de manejo de permisos en bases de datos SQL, así como explorar algunas prácticas en este campo. Se examinarán aspectos clave, como la estructura de permisos en bases de datos, las formas de gestionarlos y las limitaciones que estos conllevan.

1.1 Objetivo General:

En esta investigación nos proponemos ahondar, concretamente, en el **Manejo de permisos a nivel de usuarios** con el objetivo de comprender, desarrollar y aplicar dichos conceptos dentro de bases de datos utilizando el motor de SQL Server para lograrlo.

1.2 Objetivos específicos:

1. Analizar la estructura de permisos en SQL Server a nivel de usuario y algunos de los tipos de permisos existentes como permisos de lectura, ejecución y administración.
2. Implementar diferentes roles de seguridad en una base de datos de SQL Server y asignar permisos específicos a dichos roles.
3. Crear usuarios de base de datos en SQL Server y asignarlos a roles con diferentes permisos.
4. Probar el funcionamiento de los permisos implementados mediante ejecución de consultas SQL con los diferentes usuarios creados.
5. Evaluar la efectividad de las políticas de seguridad implementadas a nivel de permisos de usuario.

6. Documentar el proceso de administración y prueba de permisos de usuarios implementado en la base de datos.

2 Marco Conceptual:

2.1 Usuarios a nivel de servidor:

Son aquellos que se encargan de la administración del servidor cómo seguridad o inicio de sesión, donde existen permisos ordenados jerárquicamente que son asignados a los mismos restringiendo su accionar, los cuales se pueden propagar a los permisos a nivel de base de datos. [2]

2.2 Usuarios a nivel de base de datos:

Son aquellos que se encargan de la administración a nivel de base de datos para los cuales existen permisos que delimitan sus funciones en toda la misma evitando su libre manipulación. Existen dos tipos de roles para los usuarios de base de datos: [3]

- **Roles Fijos de Base de Datos:** son roles predefinidos proporcionados por SQL Server que permiten asignar permisos y tareas específicas a los usuarios en una base de datos, los cuales no pueden ser modificados ni eliminados. [3]
- **Roles de Base de Datos Definidos por el Usuario:** son roles que los administradores o desarrolladores de bases de datos, a diferencia del anterior, pueden crear en estas para satisfacer requisitos específicos de seguridad y organización de la misma, los cuales pueden ser modificados o eliminados. [3]

2.3 Manejo de Usuarios en SQL Server:

El Management Studio de SQL Server ofrece seis opciones al crear un usuario de base de datos. Estas opciones se dividen en dos categorías: credenciales de inicio de sesión y usuarios no asignados a un inicio de sesión. [5]

- **Credenciales de Inicio de Sesión:**

  Si la persona o grupo que necesita acceso tiene credenciales de inicio de sesión y administra SQL Server o necesita acceder a muchas bases de datos, se debe crear un "usuario de SQL con inicio de sesión." Este usuario de base de datos es la identidad del inicio de sesión cuando se conecta a una base de datos. Se asume que ya existe un inicio de sesión en SQL Server. [5]

- **Usuarios No Asignados a un Inicio de Sesión:**

  Si la persona o grupo no tiene un inicio de sesión y solo necesita acceso a una o varias bases de datos, se puede crear un "usuario de Windows" o un "usuario SQL con contraseña", también conocido como "usuario de base de datos independiente." Es decir que no está asociado a un inicio de sesión en la base de datos principal. [5]

2.4 Permisos de Usuario:

Los permisos de usuario son reglas que definen qué operaciones pueden realizar los usuarios en una base de datos. Esto incluye permisos para leer, escribir, modificar o eliminar datos.

3 Metodología Seguida:

3.1 Descripción de cómo se realizó el Trabajo Práctico:

Para iniciar el desarrollo, planificamos y coordinamos el proyecto mediante una reunión a través de la plataforma Discord. Durante esta sesión, discutimos y trazamos un plan de acción que nos serviría como hoja de ruta para el trabajo.

Tras esta fase inicial, comenzamos con la investigación sobre el tema central de nuestro proyecto. Cada uno de los participantes se dedicó a investigar de manera individual, explorando diversas fuentes de información, documentos relevantes y estudios relacionados lo que nos permitió obtener una comprensión más profunda del tema en cuestión.

Una vez que contamos con los conocimientos necesarios, procedimos a la fase de implementación práctica de nuestro proyecto. En este punto, cada miembro del equipo se centró en desarrollar código y herramientas de manera individual, lo que nos brindó la oportunidad de sumergirnos en los detalles técnicos y entender a fondo las complejidades de lo que estábamos construyendo. Esta aproximación individual permitió a cada uno de nosotros adquirir conocimiento de la implementación, lo que a su vez contribuyó a la calidad y robustez del proyecto en su conjunto.

Finalmente, una vez que cada miembro había completado su contribución, unificamos nuestro trabajo y conocimientos para crear una solución coherente.

3.2 Herramientas (Instrumentos y procedimientos):

3.2.1 Discord (Comunicaciones grupales):

- Creamos un servidor en Discord para trabajo.
- Establecimos un canal específico para la realización y actualización del proyecto.
- Por último, utilizamos las llamadas de Discord para discusiones generales, actualizaciones rápidas y comunicación en tiempo real.

3.2.2 WhatsApp (Seguimiento de avances):

- Creamos un grupo de WhatsApp para el equipo.
- En este se compartió actualizaciones diarias sobre avances.

3.2.3 Chat GPT (Apoyo en investigación y práctica):

- Utilizamos Chat GPT para obtener perspectivas adicionales sobre conceptos abordados.

- Preguntamos sobre sintaxis y cuestiones prácticas en la programación.
- Solicitamos guía cuando desconocemos la forma correcta de estructurar el informe.

3.2.4 GitHub (Gestión de código fuente):

- Creamos un repositorio en GitHub para el proyecto.
- Utilizamos ramas para trabajar en funciones o características específicas.
- Fuimos subiendo commits con los avances en el proyecto.

3.2.5 SQL Server Management Studio (Administración de bases de datos):

- Utilizamos SQL Server Management Studio para diseñar, desarrollar y administrar bases de datos.
- Se emplea para realizar consultas SQL para extraer información relevante.

3.2.6 Google Docs (Colaboración en documentación):

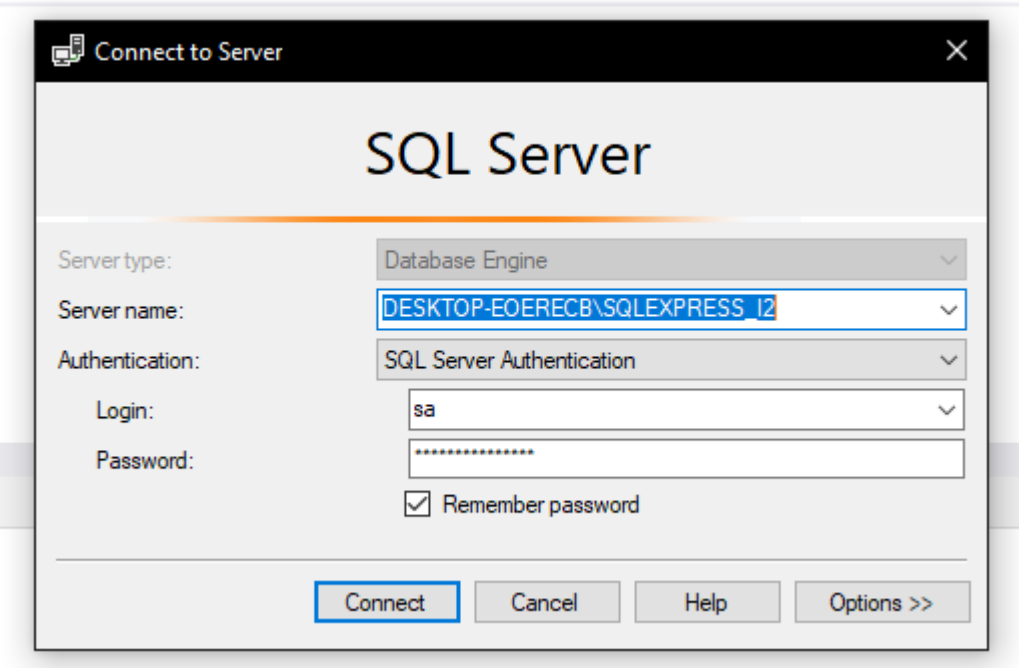
- Creamos un documento compartido en donde vamos recopilando la información y redacción final del informe.
- Nos permite y facilita la colaboración en tiempo real para la redacción y documentación del informe.

.

4 Desarrollo del Tema / Presentación de Resultado:

El desarrollo de la parte práctica constó de diferentes pasos y procedimientos, que nos encargamos de registrar para luego detallarlos a continuación:

1) Iniciamos una instancia del motor SQL Server en modo mixto para poder manejar usuarios:



2) Con el modo mixto creamos dos usuarios a nivel servidor desde código:

```
-- se necesita tener una instancia MIXTA para realizar esto
-- Crear dos usuarios de base de datos:
CREATE LOGIN UsuarioAdmin WITH PASSWORD = 'pwAdmin';
CREATE LOGIN UsuarioSoloLectura WITH PASSWORD = 'pwSoloLectura';
```

3) Creamos usuarios a nivel base de datos y los limitamos para que asi, uno sea administrador y el otro tenga permiso de **solo lectura.**

```
--Crea usuario dentro de la base consorcio
CREATE USER UsuarioAdmin FOR LOGIN UsuarioAdmin;
--Asignar permisos al usuario de administrador:
ALTER ROLE db_owner ADD MEMBER UsuarioAdmin;

--Crea usuario dentro de la base consorcio
CREATE USER UsuarioSoloLectura FOR LOGIN UsuarioSoloLectura;
--Asignar permisos al usuario de solo lectura:
ALTER ROLE db_datareader ADD MEMBER UsuarioSoloLectura;
```

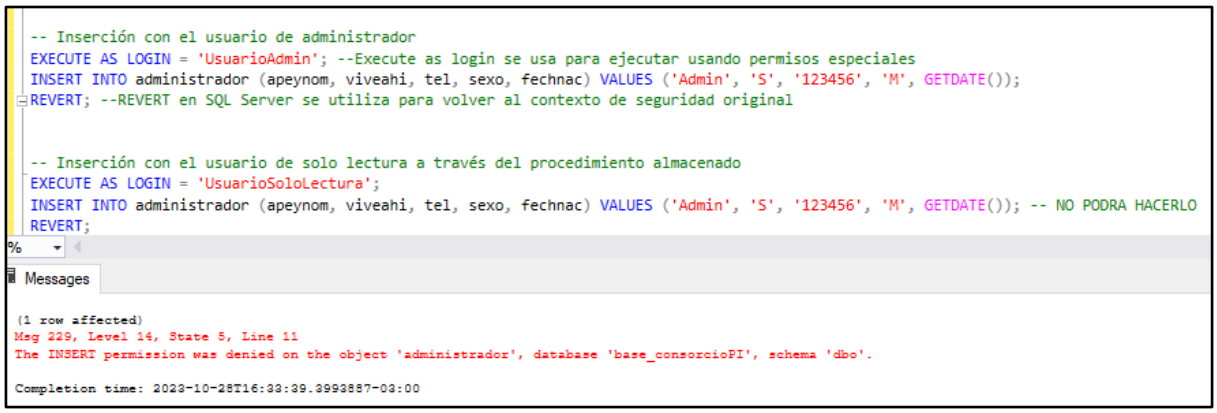
4) Creamos un **procedimiento almacenado** donde se insertan cierta cantidad de registros con datos:

```
CREATE PROCEDURE procedimiento_insert_administrador
AS
BEGIN
    -- Aqui va la logica del procedimiento
    Insert into administrador(apeynom,viveahi,tel,sexo,fechnac) values ('GONZ♦LEZ JUAN', 'S', '3624235689', 'M', '19801015');
    Insert into administrador(apeynom,viveahi,tel,sexo,fechnac) values ('RODR♦GUEZ MAR♦A', 'N', '3624236689', 'F', '19751203');
    Insert into administrador(apeynom,viveahi,tel,sexo,fechnac) values ('L♦PEZ CARLOS', 'S', '3624237689', 'M', '19790822');
    Insert into administrador(apeynom,viveahi,tel,sexo,fechnac) values ('MART♦NEZ LUC♦A', 'N', '3624238689', 'F', '19820614');
    Insert into administrador(apeynom,viveahi,tel,sexo,fechnac) values ('S♦NCHEZ ANDR♦S', 'S', '3624239689', 'M', '19740520');
    Insert into administrador(apeynom,viveahi,tel,sexo,fechnac) values ('FERN♦NDEZ ELENA', 'N', '3624240689', 'F', '19790110');
    Insert into administrador(apeynom,viveahi,tel,sexo,fechnac) values ('D♦AZ RODRIGO', 'S', '3624241689', 'M', '19831007');
    Insert into administrador(apeynom,viveahi,tel,sexo,fechnac) values ('PEREZ ANA', 'N', '3624242689', 'F', '19720430');
    Insert into administrador(apeynom,viveahi,tel,sexo,fechnac) values ('G♦MEZ LAURA', 'S', '3624243689', 'F', '19810719');
    Insert into administrador(apeynom,viveahi,tel,sexo,fechnac) values ('TORRES JOS♦', 'S', '3624244689', 'M', '19780711');
    Insert into administrador(apeynom,viveahi,tel,sexo,fechnac) values ('RODRIGUEZ HUGO', 'N', '3624245689', 'M', '19810502');
    Insert into administrador(apeynom,viveahi,tel,sexo,fechnac) values ('SILVA MARTINA', 'S', '3624246689', 'F', '19760625');
    Insert into administrador(apeynom,viveahi,tel,sexo,fechnac) values ('LOPEZ JUAN', 'N', '3624247689', 'M', '19800217');
    Insert into administrador(apeynom,viveahi,tel,sexo,fechnac) values ('MARTINEZ MARIA', 'S', '3624248689', 'F', '19731028');
    Insert into administrador(apeynom,viveahi,tel,sexo,fechnac) values ('SANCHEZ PEDRO', 'N', '3624249689', 'M', '19830906');
    Insert into administrador(apeynom,viveahi,tel,sexo,fechnac) values ('GOMEZ ISABEL', 'S', '3624250689', 'F', '19770723');

END;
```

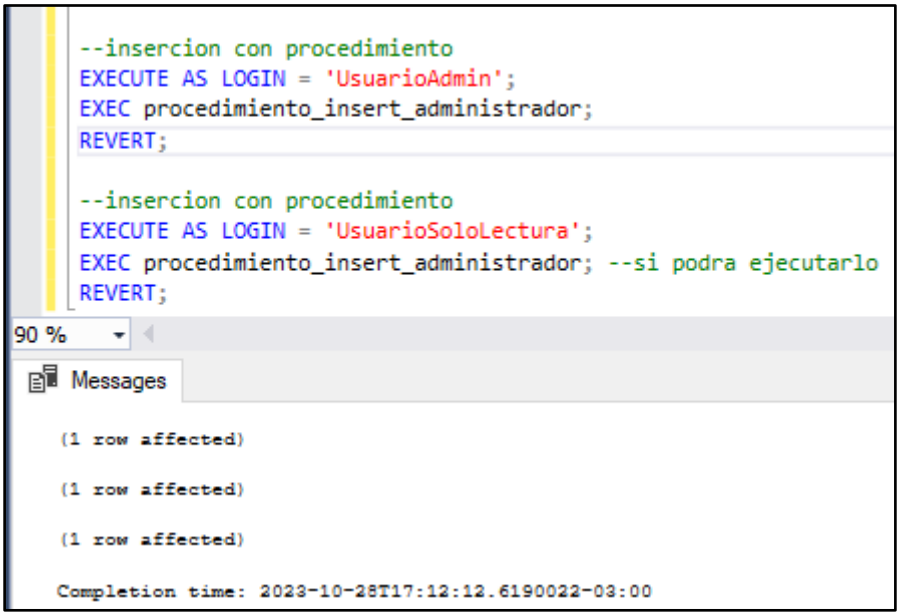
5) Con la sentencia **GRANT** permitimos al usuario de solo lectura la utilización del **procedimiento almacenado**.

```
--damos acceso al usuario de solo lectura al procedimiento
GRANT EXECUTE ON dbo.procedimiento_insert_administrador TO UsuarioSoloLectura;
```

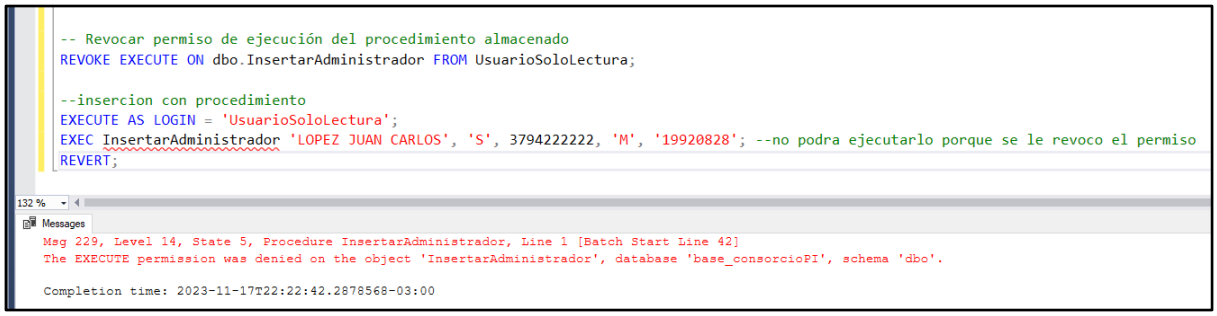
6) En la siguiente imagen, se ejecuta un insert con ambos usuarios y podemos observar que el administrador si puede insertar pero el “UsuarioSoloLectura” **no tiene permiso** para realizar la operación.



En cambio, si ejecutamos el **procedimiento almacenado** al que le dimos permiso de usar al “UsuarioSoloLectura” este si podrá ejecutarse correctamente al igual que el administrador. Esto significa que, a pesar de que "UsuarioSoloLectura" inicialmente tenía permisos limitados de **solo lectura**, al otorgarle permisos de ejecución en el procedimiento almacenado específico, ahora tiene la capacidad de ejecutar ese procedimiento almacenado, lo que le permite realizar acciones más allá de la lectura de datos.



7) El comando “**REVOKE**” es utilizado para revocar los permisos con los que cuenta el usuario y de esa forma ya no podrá usar el procedimiento.



8) Otro escenario donde se pueden implementar los permisos es mediante el uso de **vistas**, para lo cual creamos otro usuario.

```
--////////////////////   EJEMPLO CON VISTAS  ////////////////////////---

--Creamos otro usuario para probar
CREATE LOGIN UsuarioVista WITH PASSWORD = 'pwVista';
CREATE USER UsuarioVista FOR LOGIN UsuarioVista;
```

a) Creamos la vista y elegimos las columnas que queremos que el usuario visualice.

```
-- Creamos una vista que permitira que solo se visualize las columnas especificadas
CREATE VIEW dbo.VistaAdministrador AS
SELECT idadmin, apeynom, tel, sexo, fechnac
FROM dbo.administrador;
```

b) Otorgamos el permiso para que el usuario acceda a la vista.

```
--Otorgamos permisos SELECT al usuario en la vista creada, permitiéndole consultar datos a través de esta vista.
GRANT SELECT ON dbo.VistaAdministrador TO UsuarioVista;
```

c) Ejecutamos la vista usando el usuario elegido.

```
--Probamos la vista
EXECUTE AS LOGIN = 'UsuarioVista';
SELECT * FROM dbo.VistaAdministrador; --aparecera la vista que creamos antes
REVERT;
```

Results | Messages

idadmin	apeynom	sexo	fechnac
1	GONZ?LEZ JUAN	M	1980-10-15 00:00:00.000
2	LOPEZ JUAN CARLOS	M	1992-08-28 00:00:00.000

9) También podemos ceder o revocar el permiso al uso de SELECT dentro de una tabla específica.

```
--Otrogamos permiso de select en la tabla admin
GRANT SELECT ON dbo.administrador TO UsuarioVista;
```

```
EXECUTE AS LOGIN = 'UsuarioVista';
SELECT * from administrador --comprobamos que puede usar select en administrador
REVERT;
```

Results | Messages

	idadmin	apeynom	viveahi	tel	sexo	fechnac
1	1	GONZ?LEZ JUAN	S	3624235689	M	1980-10-15 00:00:00.000
2	2	LOPEZ JUAN CARLOS	S	3794222222	M	1992-08-28 00:00:00.000

```
--Revocamos los permisos directos de SELECT en la tabla administrador para evitar que el usuario acceda directamente a ella.
REVOKE SELECT ON dbo.administrador FROM UsuarioVista;

EXECUTE AS LOGIN = 'UsuarioVista';
SELECT * from administrador --comprobamos que no puede usar select en administrador
REVERT;
```

Messages

```
Msg 229, Level 14, State 5, Line 21
The SELECT permission was denied on the object 'administrador', database 'base_consorcioPI', schema 'dbo'.
```

4.1 Aplicación de Otros Proyectos:

4.1.1 Procedimientos y Funciones Almacenadas:

1) Aplicamos el procedimiento de inserción:

```
--Funcion para insertar un nuevo registro a la tabla administrador
CREATE PROCEDURE InsertarAdministrador
(
    @apeynom varchar(50),
    @viveahi varchar(1),
    @tel varchar(20),
    @sexo varchar(1),
    @fechnac datetime
)
AS
BEGIN -- "base_consorcio.dbo.administrador" se debe cambiar si se uso otro nombre para la base
    INSERT INTO base_consorcioPI.dbo.administrador (apeynom, viveahi, tel, sexo, fechnac)
    VALUES (@apeynom, @viveahi, @tel, @sexo, @fechnac);
END;
```

2) Ejecutamos con el usuario de sólo lectura (ya que anteriormente se le brindaron los permisos):

```
--insercion con procedimiento Solo lectura
EXECUTE AS LOGIN = 'UsuarioSoloLectura';
EXEC InsertarAdministrador 'GONZALES ERNESTO CARLOS', 'S', 3794222222, 'M', '19920828'; --si podra ejecutarlo
REVERT;
```

```
Messages

(1 row affected)

Completion time: 2023-11-20T12:05:19.1665446-03:00
```

3) Observamos el resultado:

```
Select * from administrador
WHERE apeynom LIKE ('GONZALES%')
```

Results | Messages

	idadmin	apeynom	viveahi	tel	sexo	fechnac
1	233	GONZALES ERNESTO CARLOS	S	3794222222	M	Aug 28 1992 12:00AM

4.1.2 Triggers de Autoría:

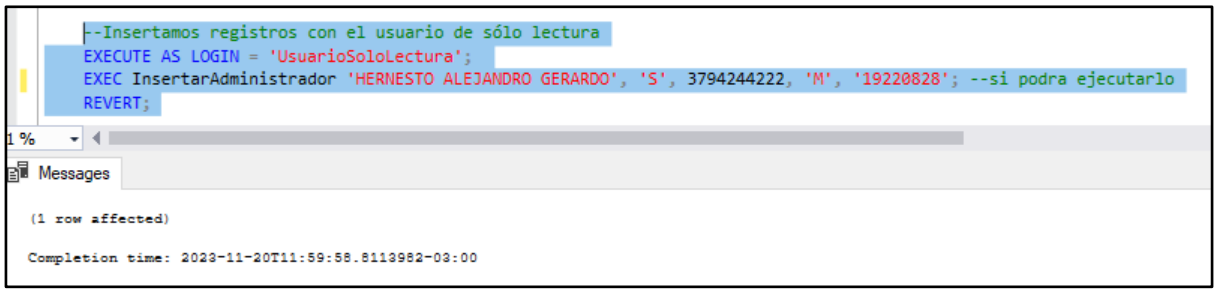
1) Creamos una tabla de auditoría donde se guardaran los datos de cualquier inserción que se haga en la tabla administrador:

```
USE base_consorcioPI
go


CREATE TABLE auditoria (
    id_auditoria int identity primary key,
    tabla_afectada varchar(100),
    columna_afectada varchar(100),
    accion varchar(10),
    fecha_hora datetime,
    usuario varchar(50),
    valor_anterior varchar(max),
    valor_actual varchar(max),
);
```

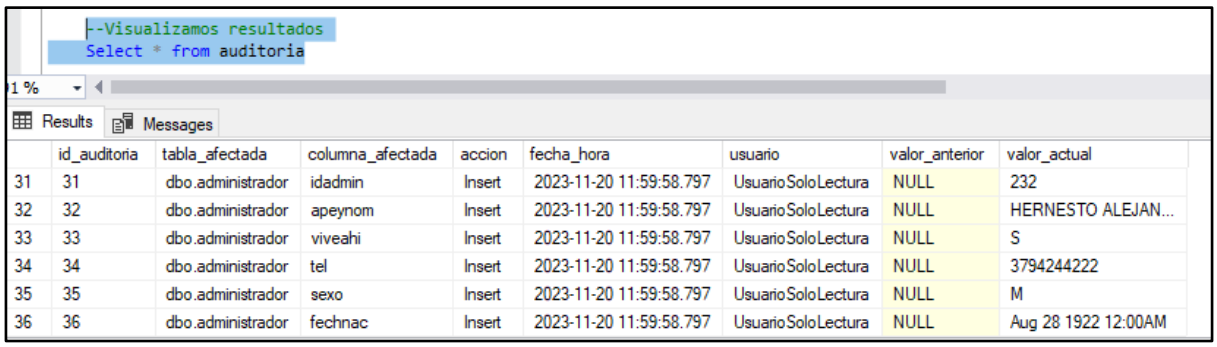
2) Luego creamos el trigger con los datos que van a ser tomados de la inserción que se realice:

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER TRIGGER [dbo].[tradministradorInsert]
   ON  [dbo].[administrador]
   AFTER Insert
AS
BEGIN
    -- SET NOCOUNT ON added to prevent extra result sets from
    -- interfering with SELECT statements.
    SET NOCOUNT ON;
    SET NOCOUNT ON;

    -- Insert statements for trigger here
    INSERT INTO dbo.auditoria(tabla_afectada, columna_afectada, valor_Anterior, valor_actual, usuario, fecha_hora, accion)
    SELECT 'dbo.administrador','idadmin', NULL, inserted.idadmin, SYSTEM_USER, CURRENT_TIMESTAMP, 'Insert' FROM inserted;

    INSERT INTO dbo.auditoria(tabla_afectada, columna_afectada, valor_Anterior, valor_actual, usuario, fecha_hora, accion)
    SELECT 'dbo.administrador','apeynom', NULL, inserted.apeynom, SYSTEM_USER, CURRENT_TIMESTAMP, 'Insert' FROM inserted;

    INSERT INTO dbo.auditoria(tabla_afectada, columna_afectada, valor_Anterior, valor_actual, usuario, fecha_hora, accion)
    SELECT 'dbo.administrador','viveahi', NULL, inserted.viveahi, SYSTEM_USER, CURRENT_TIMESTAMP, 'Insert' FROM inserted;

    INSERT INTO dbo.auditoria(tabla_afectada, columna_afectada, valor_Anterior, valor_actual, usuario, fecha_hora, accion)
    SELECT 'dbo.administrador','tel', NULL, inserted.tel, SYSTEM_USER, CURRENT_TIMESTAMP, 'Insert' FROM inserted;

    INSERT INTO dbo.auditoria(tabla_afectada, columna_afectada, valor_Anterior, valor_actual, usuario, fecha_hora, accion)
    SELECT 'dbo.administrador','sexo', NULL, inserted.sexo, SYSTEM_USER, CURRENT_TIMESTAMP, 'Insert' FROM inserted;

    INSERT INTO dbo.auditoria(tabla_afectada, columna_afectada, valor_Anterior, valor_actual, usuario, fecha_hora, accion)
    SELECT 'dbo.administrador','fechnac', NULL, inserted.fechnac, SYSTEM_USER, CURRENT_TIMESTAMP, 'Insert' FROM inserted;

END
```

3) Por último, hacemos las pruebas de inserción:



4) Observamos los resultados del funcionamiento del trigger:



	id_auditoria	tabla_afectada	columna_afectada	accion	fecha_hora	usuario	valor_anterior	valor_actual
31	31	dbo.administrador	idadmin	Insert	2023-11-20 11:59:58.797	UsuarioSoloLectura	NULL	232
32	32	dbo.administrador	apeynom	Insert	2023-11-20 11:59:58.797	UsuarioSoloLectura	NULL	HERNESTO ALEJAN...
33	33	dbo.administrador	viveahi	Insert	2023-11-20 11:59:58.797	UsuarioSoloLectura	NULL	S
34	34	dbo.administrador	tel	Insert	2023-11-20 11:59:58.797	UsuarioSoloLectura	NULL	3794244222
35	35	dbo.administrador	sexo	Insert	2023-11-20 11:59:58.797	UsuarioSoloLectura	NULL	M
36	36	dbo.administrador	fechnac	Insert	2023-11-20 11:59:58.797	UsuarioSoloLectura	NULL	Aug 28 1922 12:00AM

4.2 Conclusión del Desarrollo:

Luego de realizar la sección práctica del tema planteado en el proyecto hemos comprendido la gran importancia de los permisos a nivel de base de datos, ya que al brindarle el rol de administrador a un usuario identificamos que es capaz de realizar una gran cantidad de funciones solamente en la respectiva base de datos a la que se le asignó dichos permisos, como escritura, modificación y eliminación de registros, entre otras cosas, cosa que no es posible cuando se asigna a un usuario el permiso de sólo lectura, ya que este podrá ver la información pero no realizar ninguna de las acciones anteriormente mencionadas exceptuando que a este se le brinde la posibilidad de ejecución de un procedimiento almacenado que en este caso sólo podrá ejecutar las instrucciones que se encuentren dentro de este, cosa que en nuestro caso es la inserción de registros.

5 Conclusión:

Los permisos de usuario en una base de datos son elementos esenciales que nos ayudan a garantizar la seguridad, integridad y privacidad de los datos. Estas reglas establecen de manera precisa las acciones que los usuarios pueden llevar a cabo, tales como la lectura, escritura, modificación o eliminación de información. La adecuada gestión de los permisos es crucial para proteger la información sensible y garantizar un uso eficiente y controlado de los recursos de la base de datos. Al establecer y mantener cuidadosamente los permisos de usuario, se puede lograr un gran equilibrio entre la accesibilidad de los datos y la protección de los mismos, lo que es fundamental en entornos empresariales y de seguridad de la información.

6 Bibliografía:

[1] Microsoft. (2023, 9 de julio). Permisos en el Motor de Base de Datos. Recuperado de https://learn.microsoft.com/es-es/sql/relational-databases/security/permissions-database-engine?view=sql-server-ver16

[2] Microsoft. (2023, 10 de julio). Roles a nivel de servidor. Recuperado de https://learn.microsoft.com/es-es/sql/relational-databases/security/authentication-access/server-level-roles?view=sql-server-ver16

[3] Microsoft. (2023, 9 de marzo). Roles a nivel de base de datos. Recuperado de https://learn.microsoft.com/es-es/sql/relational-databases/security/authentication-access/database-level-roles?view=sql-server-ver16

[4] Microsoft. (2023, 4 de febrero). Procedimientos Almacenados - SQL Server Database Engine. Recuperado de: https://learn.microsoft.com/es-es/sql/relational-databases/stored-procedures/stored-procedures-database-engine?view=sql-server-ver16

[5] Microsoft. (2023, 8 de febrero). Crear un usuario de base de datos. Recuperado de: https://learn.microsoft.com/es-es/sql/relational-databases/security/authentication-access/create-a-database-user?view=sql-server-ver16