ADMINISTRACIÓN DE BASES DE DATOS (SQL)

BIENVENIDA

Bienvenido(a) a la asignatura *Administración de Bases de Datos (SQL)*, con la cual podrás aprender a gestionar una base de datos en lenguaje SQL, permitiéndote así administrar usuarios, mantener la seguridad en la base de datos y a su vez, conocer los ataques SQL y la encriptación de datos.

UNIDAD 1

GESTIÓN DINÁMICA

TEMARIO

1.1 PROCEDIMIENTOS Y DISPARADORES

1.2 VISTAS Y TABLAS TEMPORALES

GESTIÓN DINÁMICA

INTRODUCCIÓN

La asignatura de *Administración de Bases de Datos (SQL)* tiene como objetivo apoyarte a comprender la administración de una base de datos permitiéndote tener una mejor gestión de la misma.

En esta primera unidad aprenderás a desarrollar procedimientos y disparadores, y crearás vistas y tablas temporales.

COMPETENCIAS A DESARROLLAR

- El alumno será capaz de implementar los procedimientos y disparadores.
- El alumno será capaz de ejecutar las vistas y tablas temporales.

GESTIÓN DINÁMICA

VIDEO

Te invitamos a ver el siguiente video:

Gestión dinámica

Técnica que permite construir consultas SQL de manera dinámica en tiempo de ejecución, lo que significa que la consulta se construye según la necesidad del momento y puede cambiar cada vez que se ejecuta.

Esto se usa en aplicaciones en las que se necesitan construir consultas SQL en tiempo real. La gestión dinámica SQL puede utilizarse para construir consultas que incluyan variables y condiciones que no son conocidas hasta que se ejecuta el código. Esto es útil en situaciones en las que se deben realizar búsquedas complejas y personalizadas en una BD.

Ejemplo para comprender mejor:

Tenemos la tabla: *Libros*

ID_libro	Nombre_libro	Autor_libro	Año_publicado
1	Programación	Francisco Pérez	2013
2	Bases de datos	Manuel Cruz	2011
3	Algoritmos	Pedro Paz	1988
4	Variables	Ernesto Villa	2018

Empezamos con una declaración de selección bastante simple, construiremos un punto de inicio para comprender el SQL dinámico:

```
SELECT TOP(2) ID_libros  FROM Libros;
```

Esta instrucción regresa 2 filas de ID_libros de la tabla Libros. ¿Qué sucedería si quisiéramos seleccionar datos de una tabla, pero no supiéramos el nombre de la tabla hasta el tiempo de ejecución? ¿Cómo sustituiríamos el nombre de la tabla de variables en nuestro T-SQL?

Antes de responder a esa pregunta, introduzcamos SQL dinámico simplemente reescribiendo la consulta anterior para que la ejecutemos como una cadena de caracteres, en lugar de T-SQL estándar:

```
DECLARE @sql_command NVARCHAR(MAX);
SELECT @sql_command = 'SELECT TOP 2 ID_libros FROM Libros;
EXEC (@sql_command);
```

En este ejemplo, hemos definido una cadena de caracteres llamada @sql_command que se usará para contener nuestro SQL dinámico.

¿Qué es el SQL dinámico?

Es la cadena que estamos creando y luego ejecutando. En este caso, es la misma instrucción SELECT de antes, sin modificaciones. Después de establecer el valor de nuestro @sql_command, se ejecuta y proporciona los mismos resultados que antes.

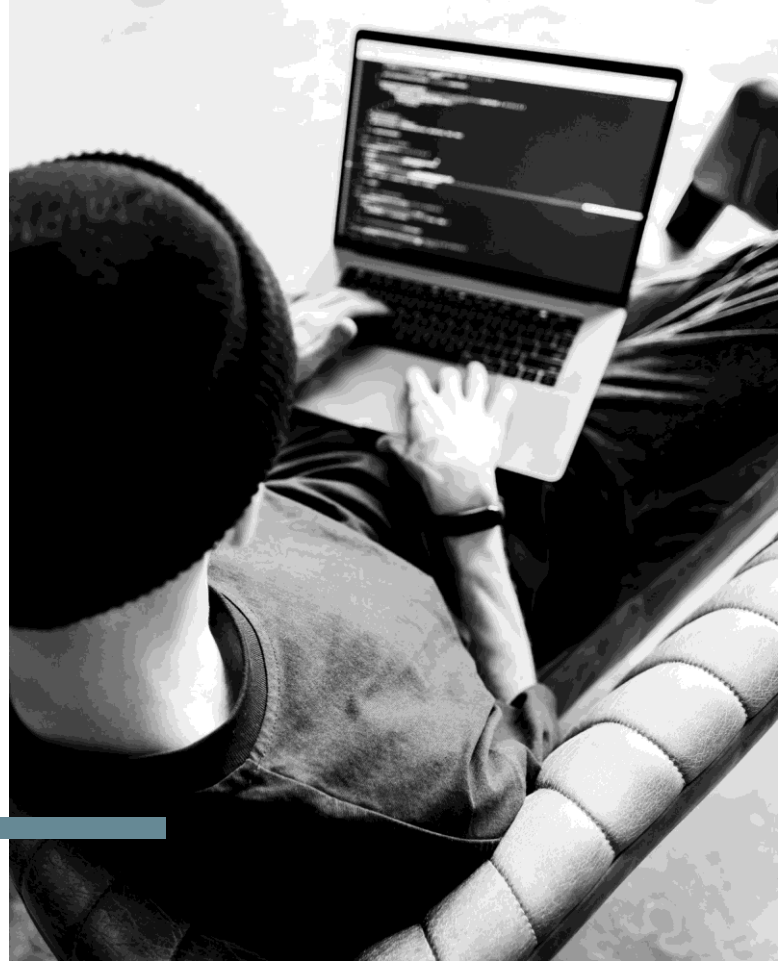
Procedimientos

Un procedimiento almacenado en SQL es un conjunto de instrucciones SQL que se almacenan en el servidor de la base de datos y se pueden llamar desde una aplicación o desde otras consultas. Los procedimientos almacenados permiten agrupar una serie de instrucciones en una sola unidad lógica y ejecutarlas en conjunto.

Los procedimientos almacenados tienen varias ventajas, entre ellas:

- Modularidad
- Reutilización
- Eficiencia
- Seguridad

Los procedimientos almacenados se crean utilizando el lenguaje de programación Transact-SQL (T-SQL) y se almacenan en la base de datos. Para llamar un procedimiento almacenado, se utiliza la instrucción EXECUTE seguida del nombre del procedimiento y de los parámetros, si los hay.



SQL Server combina un procedimiento almacenado del sistema, que se encuentra en una base de datos maestra y se reconoce por su nombre, todos tienen un nombre que comienza con SP_ (prefijo de procedimiento almacenado).

Este permite restaurar la información de la tabla del sistema y se puede ejecutar en cualquier servidor de base de datos.

Los procedimientos almacenados se crean mediante la sentencia CREATE PROCEDURE y se ejecutan con EXEC (o EXECUTE).

Sintaxis:

CREATE PROCEDURE nombre_procedimiento AS sql_statement

...

GO;

Para ejecutarlo: EXEC nombre_procedimiento;

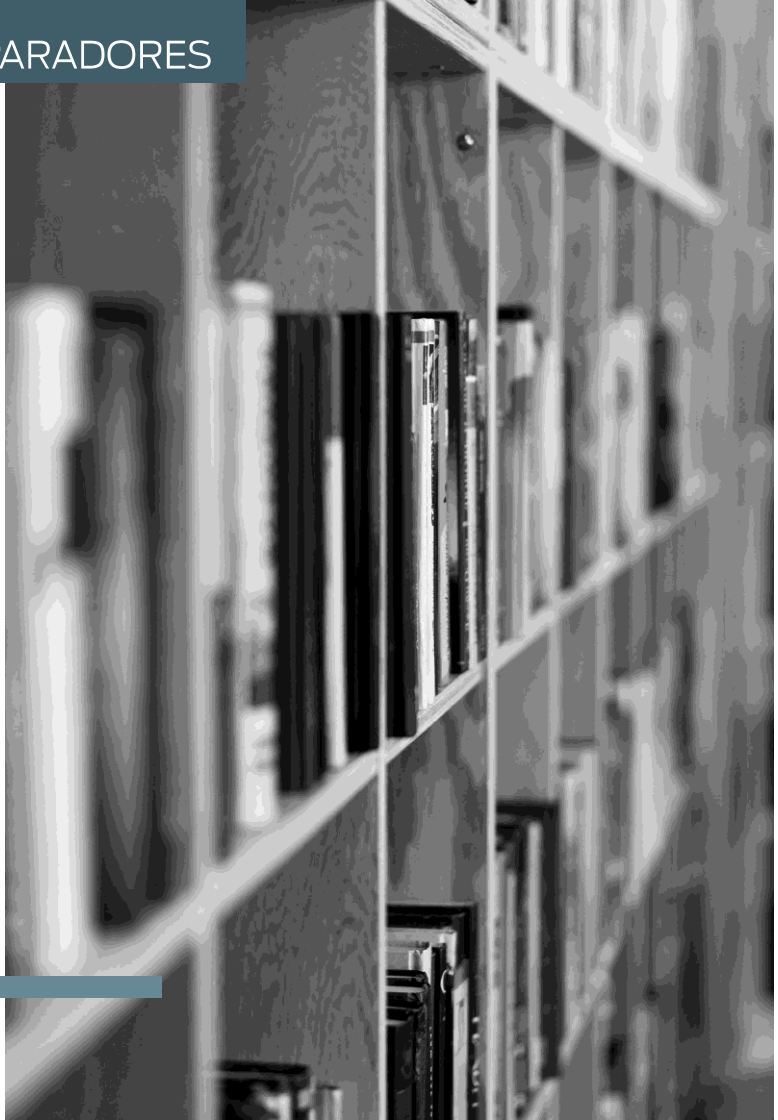
Ejemplo 1:

Con base en la tabla: *Libros*

ID_libro	Nombre_libro	Autor_libro	Año_publicado
1	Programación	Francisco Pérez	2013
2	Bases de datos	Manuel Cruz	2011
3	Algoritmos	Pedro Paz	1988
4	Variables	Ernesto Villa	2018

```
CREATE PROCEDURE TodosLosLibros
AS
SELECT ID_libro, Nombre_libro FROM Libros
GO;
EXEC TodosLosLibros
```

Este procedimiento almacenado nos muestra todos los ID_libros y Nombre_libro de nuestra tabla Libros.



Ejemplo 2:

Este procedimiento almacenado recibe 2 parámetros, nombre de un autor y nombre del libro:

```
CREATE procedure Ejemplo2
@autor varchar(30),
@nombreLibro varchar(20)
AS
SELECT ID_libro
FROM Libros
WHERE autor_libro= @autor AND
Nombre_libro=@nombreLibro;
```

Para ejecutarlo:

EXEC Ejemplo2 ‘ Francisco Pérez ‘,’Programación’;

Disparadores (trigger)

Son un tipo de objeto que se asocia con una tabla en una base de datos y se activa automáticamente cuando se realiza una acción en la tabla, como una inserción, actualización o eliminación de registros. El trigger puede contener un conjunto de instrucciones SQL que se ejecutan en respuesta a la acción que lo activó.

Los triggers se utilizan comúnmente para automatizar ciertas acciones a realizar en una base de datos, en respuesta a cambios específicos en los datos. Algunos ejemplos comunes del uso de triggers incluyen:

Actualización de campos calculados
Auditoría de cambios
Validación de datos
Enviar notificaciones

Veamos un ejemplo de trigger en SQL que se activa después de la inserción de un nuevo registro en la tabla “clientes” y actualiza automáticamente el campo “total_gastos” en la tabla “resumen_clientes”:

```
CREATE TRIGGER tr_actualizar_resumen_clientes
AFTER INSERT ON clientes
FOR EACH ROW
BEGIN
    UPDATE resumen_clientes
    SET total_gastos = total_gastos + NEW.gastos
    WHERE cliente_id = NEW.cliente_id;
END
```

En este ejemplo, el trigger “tr_actualizar_resumen_clientes” se activará automáticamente después de que se haya insertado un nuevo registro en la tabla “clientes”. La cláusula “FOR EACH ROW” indica que el trigger se ejecutará para cada fila insertada en la tabla.

SQL Server admite los siguientes tipos de disparadores:

Disparadores de evento DDL (Data Definition Language): estos se ejecutan en respuesta a eventos de modificación del esquema de la base de datos, como la creación, modificación o eliminación de tablas, vistas, procedimientos almacenados u otros objetos de la base de datos.

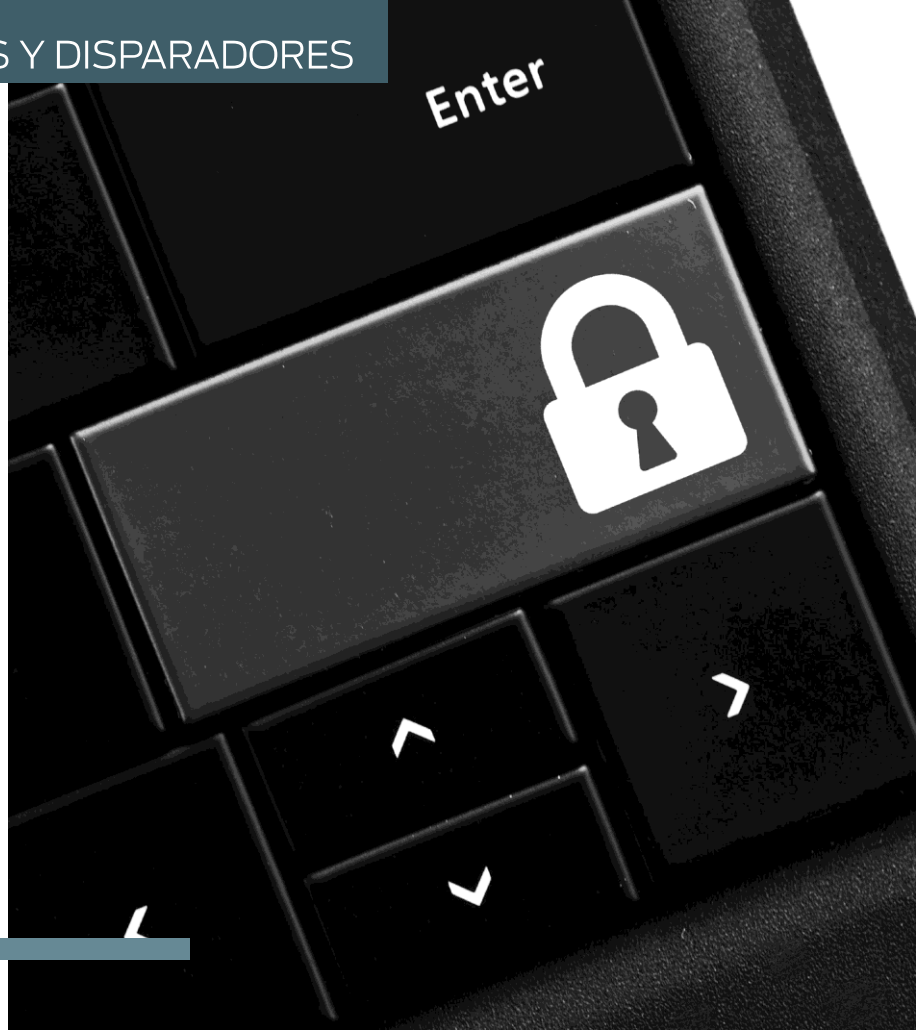
Ejemplo:

```
CREATE TRIGGER seguridad
ON DATABASE
FOR DROP_TABLE, ALTER_TABLE
AS
  PRINT ‘¡Debe deshabilitar la “seguridad” de Trigger para eliminar o modificar tablas!’
  ROLLBACK;
```

Se muestra un mensaje cuando desea ejecutar una instrucción en CREATE, ALTER o DROP.

¡El desencadenante “seguridad” debe desactivarse para eliminar o cambiar la tabla!

No permite cambios.



DML (Data Manipulation Language)
Es un lenguaje de programación utilizado para realizar operaciones en los datos almacenados de una base de datos. Las principales operaciones que se realizan con DML son la inserción, actualización y eliminación de datos en una tabla. También se utilizan para consultar los datos almacenados en la tabla.

Los desarrolladores utilizan disparadores DML para conservar una lógica compleja y reglas de integridad de datos.

Los comandos DML más comunes son:

INSERT
se utiliza para agregar nuevos datos a una tabla.

DELETE
se utiliza para eliminar registros de una tabla.

DML

UPDATE
se utiliza para actualizar datos existentes en una tabla.

SELECT
se utiliza para recuperar datos de una o varias tablas.

Ejemplo:

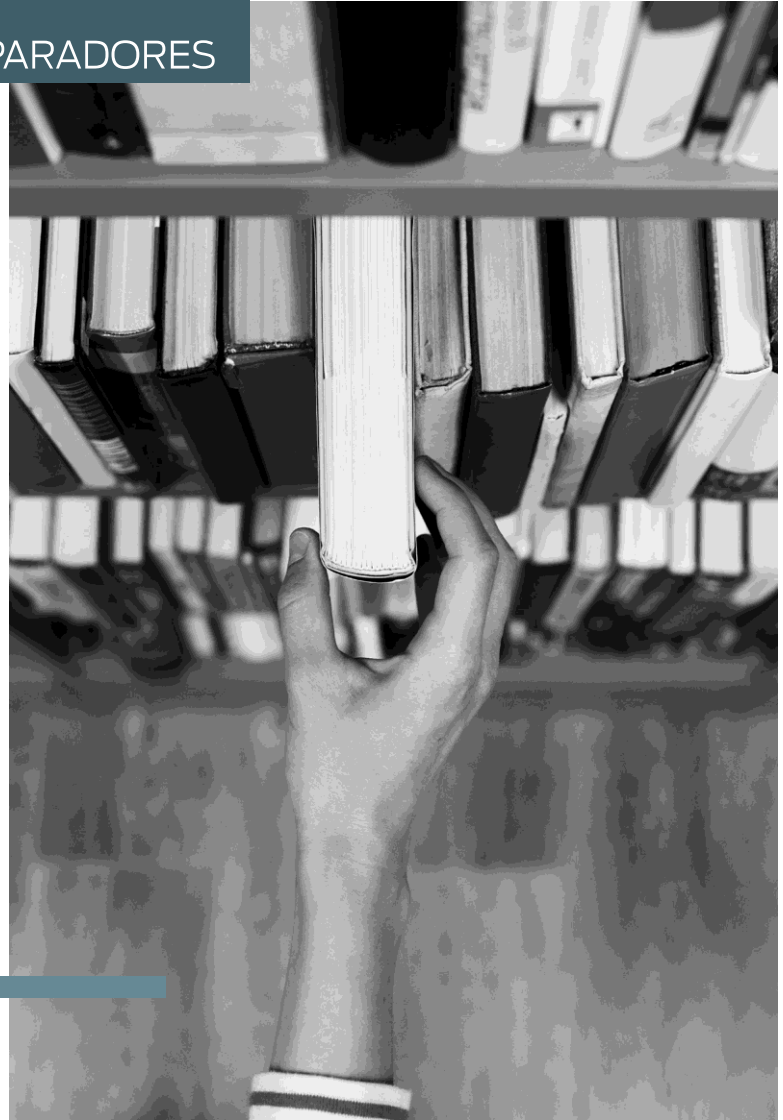
```
CREATE TRIGGER Ejemplo1
ON Libros --Tabla donde se ejecuta el disparador
--El disparador se activará cuando las operaciones Insert y Update sean correctas
FOR INSERT, UPDATE
AS
    --Instrucciones del disparador
    PRINT 'Libro actualizado correctamente'

Go
```

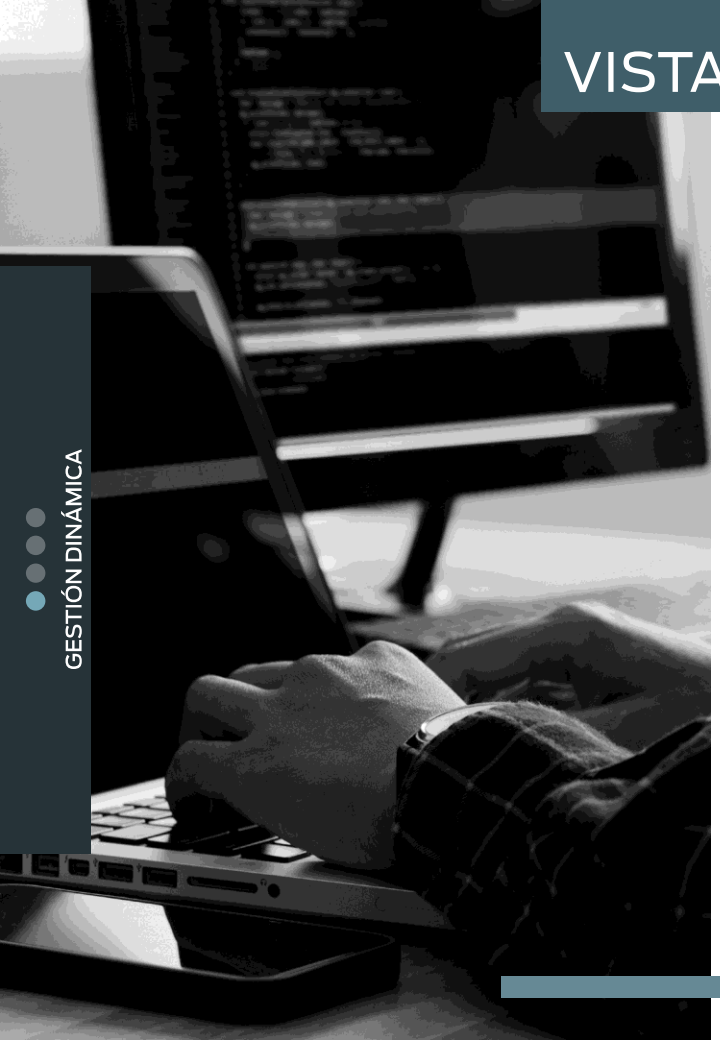
Para verlo en acción:

```
update Libros set Nombre_libro='Base de datos
2'
where ID_libro='2'
Go
```

Al dar clic en Execute, mostrará el mensaje:
Libro actualizado correctamente



VISTAS Y TABLAS TEMPORALES



Vistas

Es un objeto que se crea a partir de una o varias tablas existentes de una base de datos y que representa una tabla virtual. Una vista no es una tabla física, sino una definición de una tabla que se almacena en la base de datos y se utiliza como si fuera una tabla real.

Las vistas se utilizan comúnmente para simplificar el acceso a los datos y para proporcionar una capa adicional de seguridad y privacidad. Algunos ejemplos comunes de uso de vistas incluyen:

Simplificación de consultas complejas: una vista puede simplificar una consulta compleja combinando varias tablas en una sola vista.

Reducción de la redundancia de datos: una vista puede reducir la redundancia de datos al combinar varias tablas en una sola vista.

Restricción de acceso a datos: una vista puede restringir el acceso a ciertos datos al permitir solo ciertas columnas o filas para ser vistos por usuarios específicos.

Ocultación de la complejidad de la base de datos: una vista puede ocultar la complejidad de la base de datos al proporcionar una interfaz simplificada a los usuarios.

GESTIÓN DINÁMICA

VIDEO

Te invitamos a ver el siguiente video:

Las vistas se crean utilizando la instrucción CREATE VIEW, que se utiliza para definir la consulta que se utilizará para crear la vista. La consulta puede incluir una o varias tablas y puede contener cualquier instrucción SQL válida, como SELECT, JOIN, WHERE, GROUP BY y ORDER BY.

Por ejemplo, se puede crear una vista que combine las tablas de clientes y pedidos en una sola vista utilizando la siguiente instrucción:

```
CREATE VIEW customer_order_view AS
SELECT c.customer_id, c.customer_name, o.order_id, o.order_date
FROM customers c
INNER JOIN orders o ON c.customer_id = o.customer_id
```

Esta vista combina las tablas de clientes y pedidos en una sola vista llamada customer_order_view, que puede ser consultada como si fuera una tabla real. Por ejemplo, se podría consultar la vista para obtener una lista de todos los pedidos realizados por un cliente específico utilizando la siguiente instrucción:

```
SELECT order_id, order_date
FROM customer_order_view
WHERE customer_id = '12345'
```

Veamos otro ejemplo:
Tabla: Libros

ID_libro	Nombre_libro	Autor_libro	Año_publicado
1	Programación	Francisco Pérez	2013
2	Bases de datos	Manuel Cruz	2011
3	Algoritmos	Pedro Paz	1988
4	Variables	Ernesto Villa	2018

Podemos crear una vista simple de la siguiente forma, escribimos el código:

```
CREATE VIEW VISTA_LIBRO AS
  SELECT ID_libro, Nombre_libro
   FROM Libros;
```

Para ejecutarla, debemos escribir el código:

```
SELECT * FROM VISTA_LIBRO
```

Nos mostrará las columnas requeridas, de esta manera podemos acceder a información de una forma más concreta.

Tablas temporales

Es una tabla que existe solo durante la sesión de la conexión que la crea y se elimina automáticamente cuando la sesión se cierra. Las tablas temporales son útiles para almacenar datos temporales o intermedios que se utilizan en consultas complejas o en procesos de varios pasos.

En SQL, existen dos tipos de tablas temporales:

tablas temporales locales
tablas temporales globales.

1. **Tablas temporales locales:**
Estas tablas se crean con el prefijo “#” en su nombre y son visibles solo dentro de la sesión de la conexión que las crea. Cuando la sesión se cierra, la tabla temporal se elimina automáticamente.

A continuación vemos un ejemplo de creación de una tabla temporal local:

```
CREATE TABLE #temporal (
    id INT,
    nombre VARCHAR(50),
    fecha DATE
);
```

2. **Tablas temporales globales:**
Estas tablas se crean con el prefijo "##" en su nombre y son visibles para todas las conexiones a la base de datos. Estas tablas temporales persisten después de que la sesión que las crea se ha cerrado, y se eliminan cuando se cierra la última sesión que las utiliza.

A continuación vemos un ejemplo de creación de una tabla temporal global:

```
CREATE TABLE ##temporal (
    id INT,
    nombre VARCHAR(50),
    fecha DATE
);
```

Ambos tipos de tablas temporales se pueden utilizar de la misma manera que las tablas normales, permitiendo insertar, actualizar y eliminar datos.

Sin embargo, es importante tener en cuenta que las tablas temporales son útiles para almacenar datos temporales o intermedios y no deben ser utilizadas para almacenar datos permanentes.

FORO

Con los conocimientos que aprendiste en esta unidad, responde la siguiente pregunta:

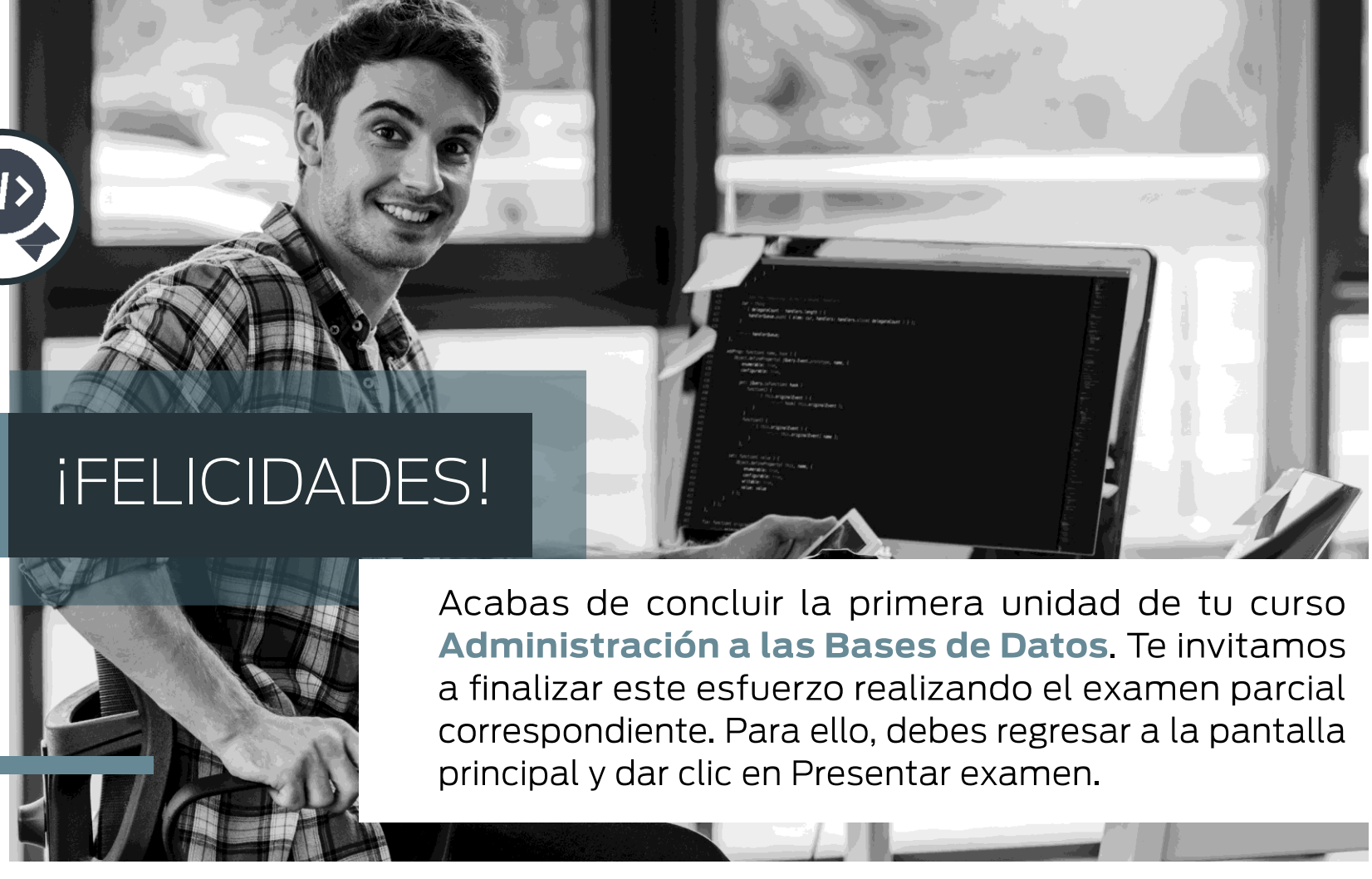
¿Cómo influye la optimización de la información en el uso de una base de datos en una empresa o negocio?

Presiona el botón para participar en el foro.

CONCLUSIÓN

Durante esta unidad comprendiste qué son los procedimientos, su forma de utilizarlos en conjunto con los parámetros, y también aprendiste los tipos de disparadores para SQL Server.

De igual manera, aprendiste la forma de utilizar las vistas y las ventajas de crearlas, así como el modo de usar las tablas temporales según los datos a manejar.



GESTIÓN DINÁMICA

¡FELICIDADES!

Acabas de concluir la primera unidad de tu curso **Administración a las Bases de Datos**. Te invitamos a finalizar este esfuerzo realizando el examen parcial correspondiente. Para ello, debes regresar a la pantalla principal y dar clic en Presentar examen.

UNIDAD 2

GESTIÓN DE USUARIOS Y GRUPOS

TEMARIO

2.1 CREACIÓN Y ASIGNACIÓN DE PERMISOS

2.2 REPORTES DE USUARIO Y GRUPOS

INTRODUCCIÓN

En esta segunda unidad aprenderás a crear y asignar permisos a usuarios, así como hacer reportes de usuarios y de grupos.

Todo esto con el fin de añadir fácilmente a tu base de datos diversos usuarios que tengan diferentes roles dentro de la misma, además podrás generar reportes que ayuden a visualizar la información de manera distinta.

COMPETENCIAS A DESARROLLAR

- El alumno será capaz de implementar la creación y asignación de permisos.
- El alumno será capaz de desarrollar reportes de usuario y grupos.

VIDEO

Te invitamos a ver el siguiente video:

CREACIÓN Y ASIGNACIÓN DE PERMISOS

Gestión de usuarios y grupos

La gestión de usuarios y grupos en una base de datos es una tarea importante para garantizar la seguridad y el acceso adecuado a los datos almacenados. A continuación se describen algunos conceptos y acciones clave relacionadas con la gestión de usuarios y grupos en una base de datos:

Usuario
Crear usuarios
Eliminar usuarios
Crear grupos
Eliminar grupos
Asignar usuarios a grupos
Revocar permisos
Conceder permisos

1. **Usuario:** es una entidad que se utiliza para autenticar y autorizar a las personas que acceden a la base de datos. Un usuario puede tener permisos específicos para realizar ciertas acciones en la base de datos, como leer, escribir o modificar datos.

2. **Grupo:** es una entidad que agrupa a uno o varios usuarios para que puedan ser administrados juntos. Los permisos de grupo se aplican a todos los usuarios que pertenecen al grupo, lo que simplifica la administración de permisos.

3. **Crear usuarios:** los usuarios se pueden crear mediante la instrucción **CREATE USER**, especificando un nombre de usuario y una contraseña. También se pueden asignar roles y permisos a usuarios al crearlos.

4. **Eliminar usuarios:** los usuarios se pueden eliminar utilizando la instrucción DROP USER.

5. **Crear grupos:** los grupos se pueden crear utilizando la instrucción CREATE ROLE, especificando un nombre de grupo. También se pueden asignar permisos y roles a grupos al crearlos.

6. **Asignar usuarios a grupos:** los usuarios se pueden asignar a grupos utilizando la instrucción **ALTER ROLE**, especificando el nombre del grupo y el nombre del usuario que se va a agregar al grupo.

7. **Revocar permisos:** los permisos se pueden revocar utilizando la instrucción **REVOKE**, especificando el permiso que se va a revocar y el usuario o grupo al que se le está revocando el permiso.

8. **Conceder permisos:** los permisos se pueden conceder utilizando la instrucción **GRANT**, especificando el permiso que se va a conceder y el usuario o grupo al que se le está concediendo el permiso.

Creación y asignación de permisos

En SQL Server, los servicios se crean y asignan a través de la configuración de SQL Server Configuration Manager. A continuación se presentan los pasos generales para crear y asignar servicios en SQL Server:

1. Abrir SQL Server Configuration Manager:

- En el menú Inicio, buscar “SQL Server Configuration Manager”.
- Seleccionar la versión de SQL Server correspondiente.

2. Crear un nuevo servicio:

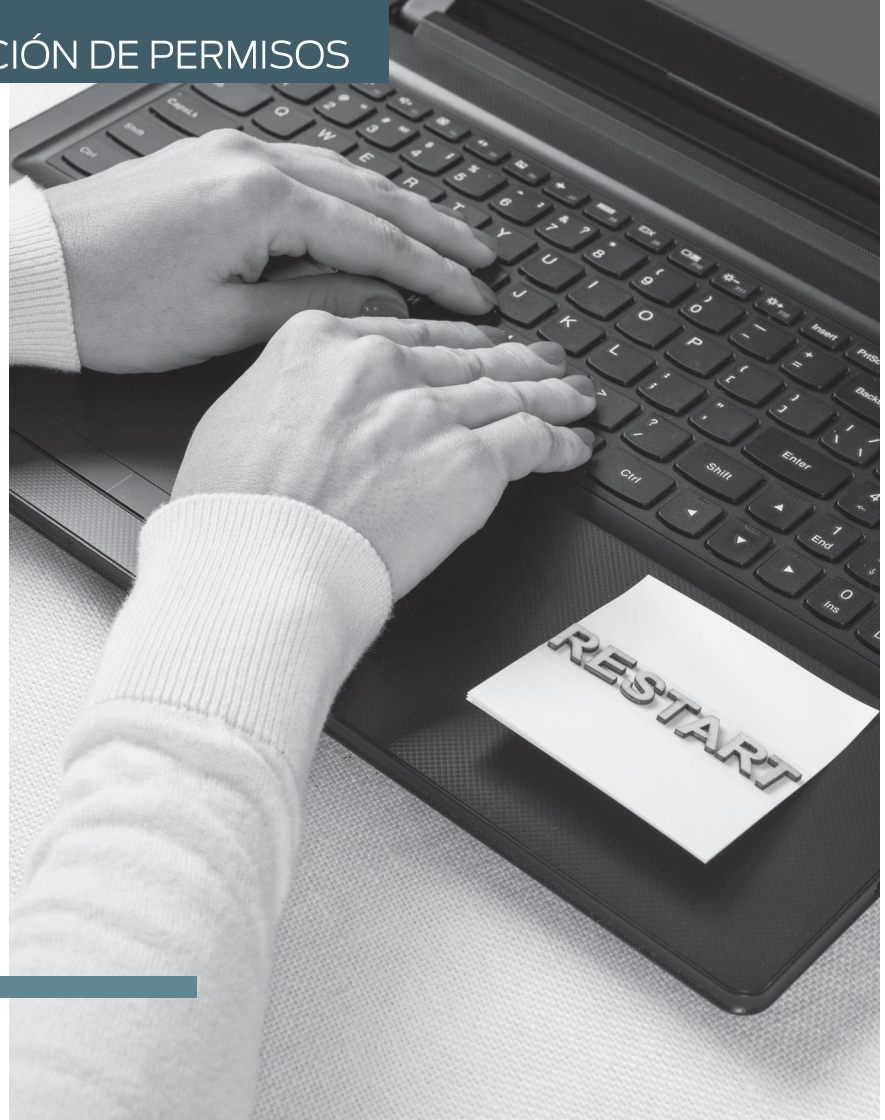
- En la ventana de SQL Server Configuration Manager, seleccionar la opción “SQL Server Services”.
- Hacer clic en el botón “Add” para agregar un nuevo servicio.
- Seleccionar el tipo de servicio a crear (por ejemplo, “SQL Server Database Engine”).
- Especificar un nombre para el servicio y una cuenta de inicio de sesión.

3. Asignar permisos al servicio:

- En la ventana de SQL Server Configuration Manager, seleccionar el servicio creado.
- Hacer clic con el botón derecho del mouse y seleccionar “Properties”.
- En la pestaña “Log On”, especificar la cuenta de inicio de sesión a utilizar.
- En la pestaña “Permissions”, agregar las cuentas de usuario o grupos que tendrán permisos para utilizar el servicio.

4. Reiniciar el servicio:

- En la ventana de SQL Server Configuration Manager, seleccionar el servicio creado.
- Hacer clic con el botón derecho del mouse y seleccionar “Restart”.



Creación de un inicio de sesión

Para acceder al motor de la base de datos, se debe iniciar sesión. El inicio de sesión puede ser la del usuario, como una cuenta de Windows o miembro del grupo de Windows, o el inicio de sesión puede ser un registro del servidor SQL que existe solo en el servidor SQL. Siempre que sea posible, utilice la autenticación de Windows.

De forma predeterminada, el administrador del equipo tiene acceso total al servidor SQL.

Creación de una cuenta de Windows

1. Hacer clic derecho sobre el icono de Windows y dar clic en Ejecutar, escribe %SystemRoot%\system32\compmgmt.msc /s, luego, haz clic en Aceptar para abrir el programa Administración de equipos.

2. En Herramientas del sistema, expande Usuarios y grupos locales, haz clic con el botón derecho en Usuarios y luego haz clic en Nuevo usuario.

3. En el cuadro Nombre de usuario, escribe Gabriel como ejemplo.

4. En Herramientas del sistema, expande Usuarios y grupos locales, haz clic con el botón derecho en Usuarios y luego haz clic en Nuevo usuario.

En caso de tener Windows 10:

Abre el Panel de control, haz clic en Cuentas de usuario, después haz clic en Agregar un nuevo usuario en Configuración, y te mostrará si deseas agregar una cuenta que sea un familiar o no (será a tu elección).

Creación de un inicio de sesión de SQL

En el programa Microsoft SQL Server Management Studio, escribe y ejecuta el siguiente código que reemplaza Computer_Name con el nombre del comando. Desde Windows indica que Windows autenticará al usuario. Un argumento opcional para default_database conecta a Oliver con la base de datos de la biblioteca (que usaremos como ejemplo), a menos que la cadena de conexión no indique otra base de datos.

Esta instrucción introduce el punto y coma como una terminación opcional de una instrucción Transact-SQL.

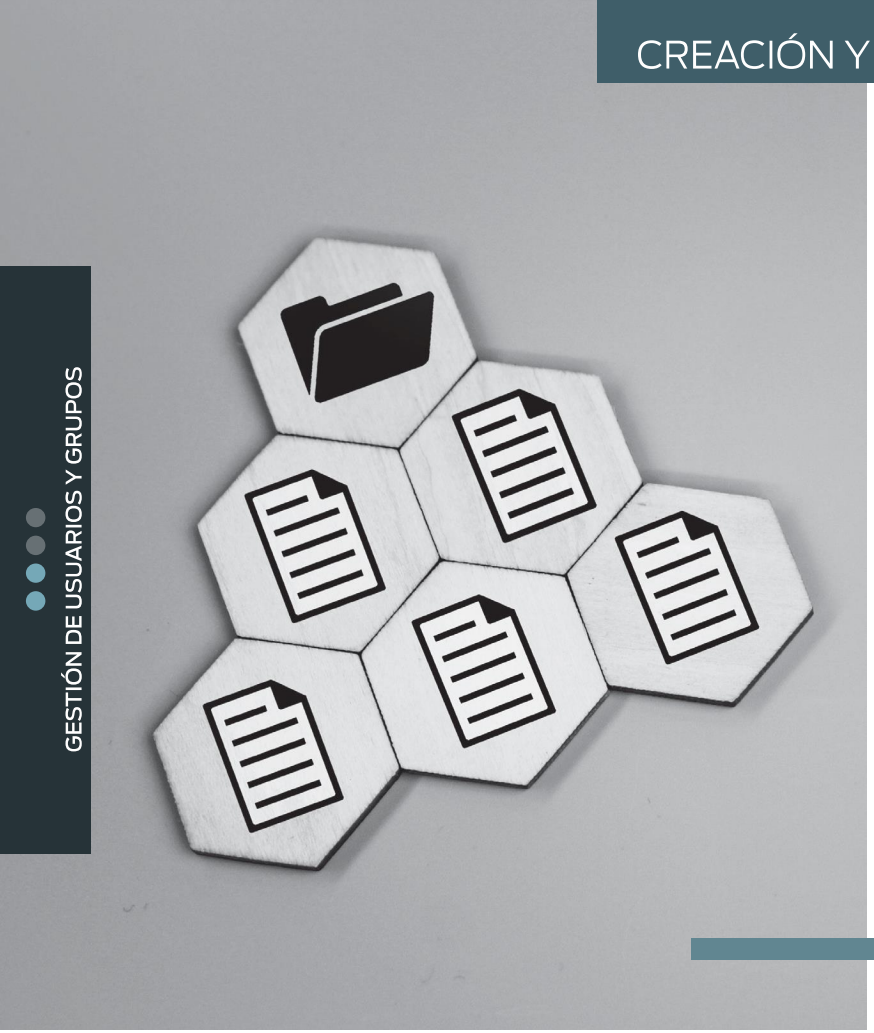
```
CREATE LOGIN [computer_name\Oliver]
   FROM WINDOWS
   WITH DEFAULT_DATABASE = [Biblioteca];
GO
```

Esto autoriza al nombre de usuario Oliver, autenticado por el equipo, para tener acceso a esta instancia de SQL Server. Si existe más de una instancia de SQL Server en el equipo, debe crear el inicio de sesión en cada instancia a la que Oliver deba tener acceso.

Concesión de acceso a una base de datos
Oliver ahora tiene acceso a esta instancia de SQL Server, pero no tiene permiso para acceder a la base de datos. Ni siquiera tiene acceso a su propia base de datos de la biblioteca hasta que se le permita como usuario de la base de datos.

Para dar acceso a Oliver, cambia a la base de datos de la biblioteca y luego usa las instrucciones para que el usuario establezca el inicio de sesión en el usuario llamado Oliver.

CREACIÓN Y ASIGNACIÓN DE PERMISOS



Para crear roles de bases de datos definidos por el usuario desde el Management Studio realizaremos los siguientes pasos:

- Posicionarnos en la base de datos a la cual le agregaremos los roles definidos por el usuario y expandirla.
- Expandir la carpeta Seguridad.
- Expandir la carpeta Roles.
- Posicionarnos sobre la opción Roles de base de datos y dar clic con el botón derecho del mouse y seleccionar la opción Nuevo rol de base de datos.

REPORTES DE USUARIO Y GRUPOS

Un reporte de usuarios en Microsoft SQL Server Management Studio es un listado de los usuarios y sus respectivos permisos en una instancia de SQL Server. Este reporte es útil para administradores de bases de datos, ya que les permite ver quiénes tienen acceso a la base de datos, qué permisos tienen y en qué objetos pueden trabajar.

El reporte de usuarios puede incluir la siguiente información:

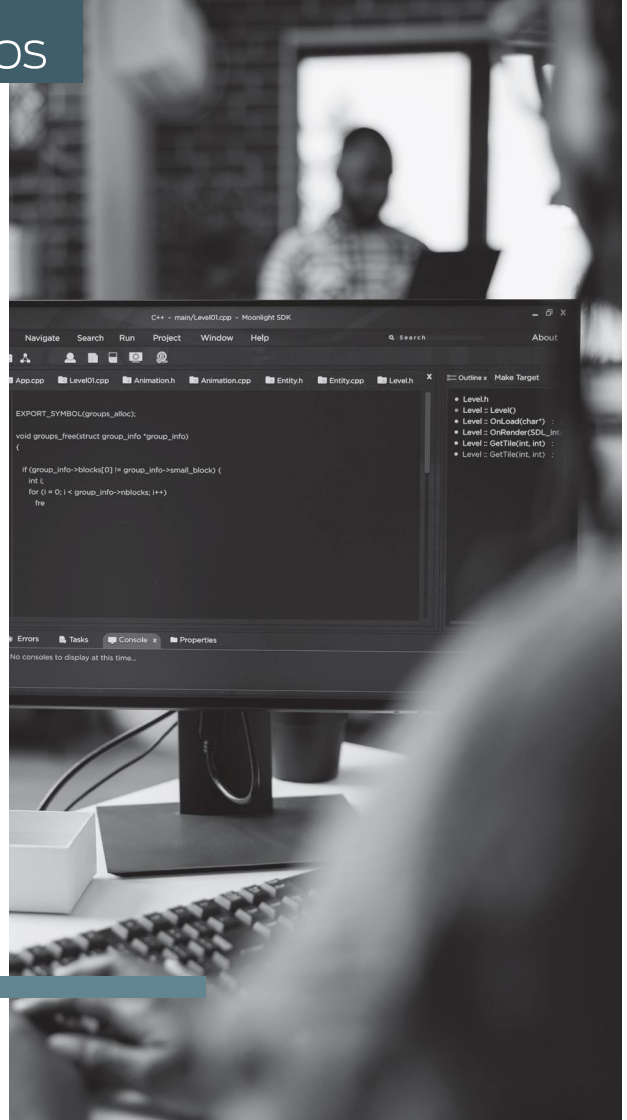
- el nombre de usuario
- el inicio de sesión
- el rol asignado
- los permisos
- las bases de datos a las que tienen acceso.

VIDEO

Te invitamos a ver el siguiente video:

Este informe es muy útil para auditar la seguridad de la base de datos y garantizar que los usuarios tengan los permisos necesarios para realizar su trabajo, sin tener acceso a información confidencial o a recursos que no deberían tener.

En SQL Server Management Studio, se puede obtener el reporte de usuarios mediante la exploración de las carpetas de seguridad en el explorador de objetos y la visualización de las propiedades de los inicios de sesión de usuario. También puedes ejecutar consultas Transact-SQL para obtener información más detallada sobre los usuarios y sus permisos.



REPORTES DE USUARIO Y GRUPOS

Para obtener un reporte de usuarios en Microsoft SQL Server Management Studio, es importante seguir los siguientes pasos:

1. Conéctate a la instancia de SQL Server en la que deseas obtener el reporte.
2. En el Explorador de objetos, expande la carpeta de Seguridad y luego haz clic en la carpeta de Inicios de sesión.
3. Aparecerá una lista de todos los inicios de sesión configurados en esa instancia de SQL Server.
4. Se puede hacer clic derecho en cualquier inicio de sesión y seleccionar la opción **“Propiedades”** para ver más detalles sobre ese inicio de sesión, como su nombre de usuario, permisos, etc.
5. También se puede hacer clic derecho en la carpeta de Inicios de sesión y seleccionar **“Nuevo inicio de sesión”** para crear un nuevo inicio de sesión.
6. Si necesitas obtener un reporte más detallado, se puede ejecutar consultas de Transact-SQL para obtener información específica sobre los inicios de sesión y sus permisos.

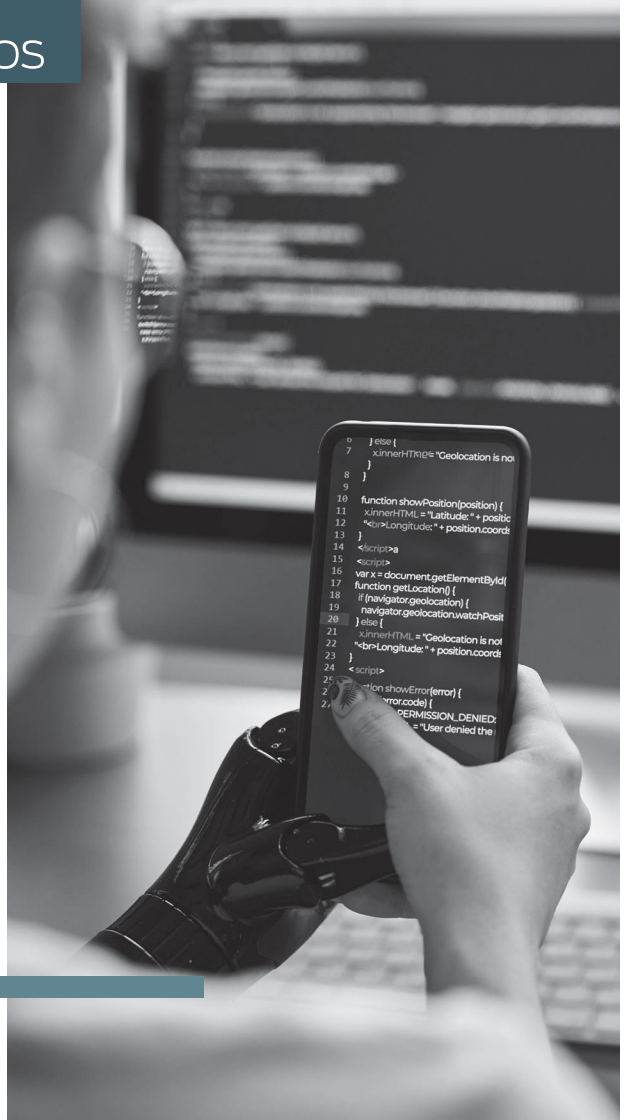
Implementación

Los informes personalizados se guardan como archivos de informe (.RDL) y generan informes (RDL).

RDL tiene información de diseño y recuperación de datos para un informe en formato XML. RDL es un esquema abierto.

Los desarrolladores tienen la oportunidad de expandir RDL con atributos y recursos adicionales. Los informes pueden realizar todas las transacciones Transact SQL válidas dentro del informe.

Si Object Explorer está conectado a un servidor, los informes personalizados se pueden ejecutar en el contexto de la selección actual de Object Explorer si los informes hacen referencia a los parámetros de informe de este nodo. De esta forma, un informe puede utilizar el contexto actual, p. B. La base de datos real. O un contexto coherente, como B. Especifique una base de datos con nombre como parte de la instrucción SQL Transact, que se incluye en el informe personalizado.



Ejecución de un informe personalizado

Puedes ejecutar un informe personalizado en Management Studio de la siguiente manera:

1. En el Explorador de objetos, seleccionar la base de datos y expandir la carpeta "Informes" para ver los informes disponibles.
2. Hacer clic con el botón derecho del ratón en la carpeta "Informes" y seleccionar "Agregar informe existente" para agregar el informe personalizado que se desea ejecutar.
3. Seleccionar el archivo de informe personalizado (.rdl) y hacer clic en "Abrir". El informe se agregará a la carpeta de informes de la base de datos.
4. Hacer clic con el botón derecho del ratón en el informe personalizado que se desea ejecutar y seleccionar "Ver informe" para ejecutarlo.
5. En el visor de informes, se solicitará que se ingresen los parámetros necesarios para ejecutar el informe. Ingresar los valores requeridos y hacer clic en "Aceptar".
6. El informe personalizado se ejecutará y se mostrará en el visor de informes.

Reportes estándar

Microsoft SQL Server Management Studio proporciona varios reportes estándar que permiten a los usuarios ver información detallada sobre varios aspectos de la base de datos y su configuración. Estos reportes se dividen en varias categorías, como Administración, Seguridad, Rendimiento, etc.

Para utilizar los reportes estándar de Microsoft SQL Server Management Studio, se deben seguir los siguientes pasos:

- Abrir Microsoft SQL Server Management Studio y conectarse a la instancia de SQL Server que se desea administrar.
- En el Explorador de objetos, hacer clic con el botón derecho en la base de datos para la que desea ver el informe y seleccionar “Informes” en el menú contextual.
- En el submenú Informes, seleccionar la categoría relevante para el informe que se desea ver. Por ejemplo, si se desea ver un informe sobre el rendimiento de la base de datos, seleccionar la categoría “Rendimiento”.

ACTIVIDAD INTEGRADORA 1

Te invitamos a realizar la siguiente actividad.

Presiona el botón para descargar la actividad.

Presiona el botón para entregar la actividad.

LIBROS RECOMENDADOS

- Pérez, M. (2011). *Microsoft SQL Server 2008 R2. Motor de base de datos y administración. G.*
- Degollado, A, et al. (2019). *Manual SQL Server –Transact SQL Básico /Avanzado [Tesis].* Universidad Tecnológica de Santa Catarina.

CONCLUSIÓN

La gestión de usuarios, grupos y permisos, junto con los reportes de usuarios, son elementos fundamentales en la administración de bases de datos en SQL Server, y es importante que los administradores estén familiarizados con estas herramientas y técnicas para garantizar la seguridad y el buen funcionamiento de la base de datos.

¡FELICIDADES!

Acabas de concluir la segunda unidad de tu curso **Administración a las Bases de Datos**. Te invitamos a finalizar este esfuerzo realizando el examen parcial correspondiente. Para ello, debes regresar a la pantalla principal y dar clic en Presentar examen.

UNIDAD 3

SEGURIDAD DEL GESTOR DE BASES DE DATOS

TEMARIO

3.1 RESPALDOS LOCALES Y EN LA NUBE

3.2 ATAQUES SQL Y ENCRIPTACIÓN DE DATOS

INTRODUCCIÓN

En esta tercera unidad aprenderás a crear respaldos locales y en la nube de bases de datos en SQL Server, a su vez conocerás los ataques a SQL y cómo realizar encriptación de datos.

Todo esto con el fin de lograr mayor seguridad en tu base de datos para que tu información esté resguardada y segura ante cualquier ataque malicioso.

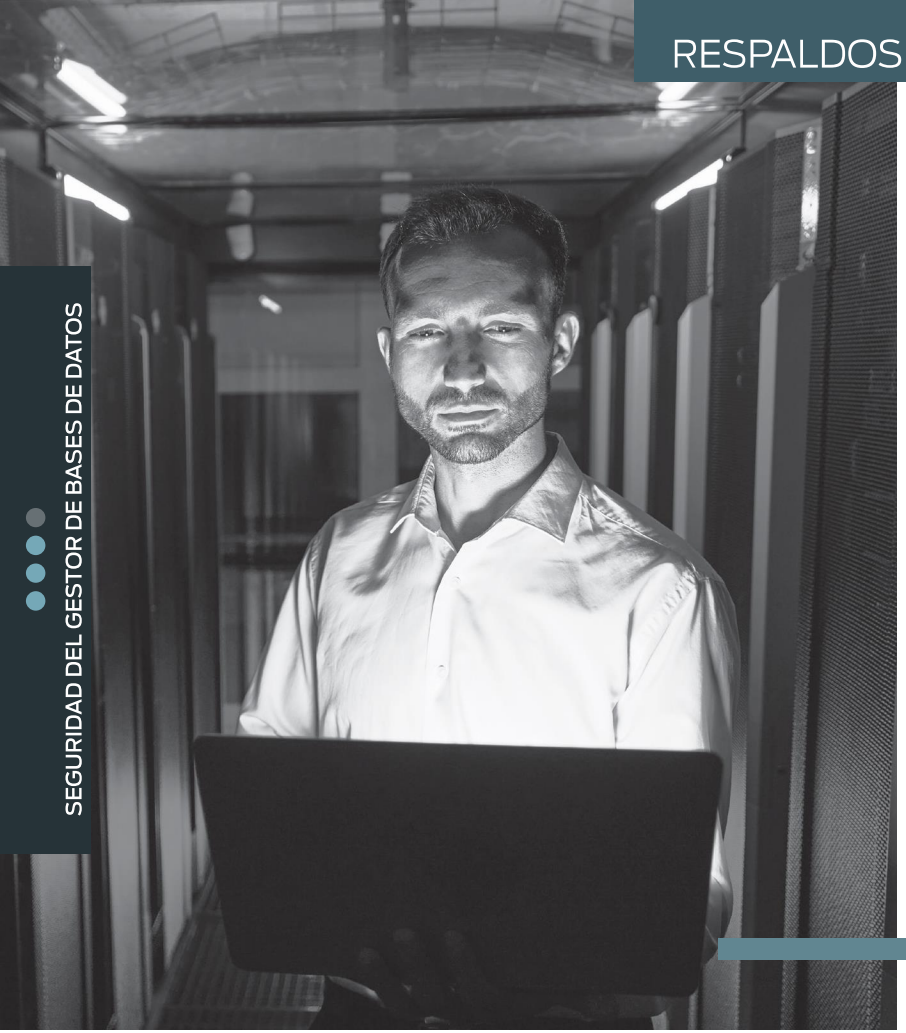
COMPETENCIAS A DESARROLLAR

- El alumno será capaz de implementar respaldos locales y en la nube.
- El alumno será capaz de reconocer ataques SQL y ejecutar encriptación de datos.

VIDEO

Te invitamos a ver el siguiente video:

Seguridad del Gestor de Bases de Datos
La seguridad en SQL Server abarca tanto la autenticación como la autorización de usuarios frente a datos e instancia del servidor SQL Server. Es importante siempre crear un entorno seguro para SQL Server, el cual debe contar con seguridad en las etapas de diseño, desarrollo e implementación del software y bases de datos.

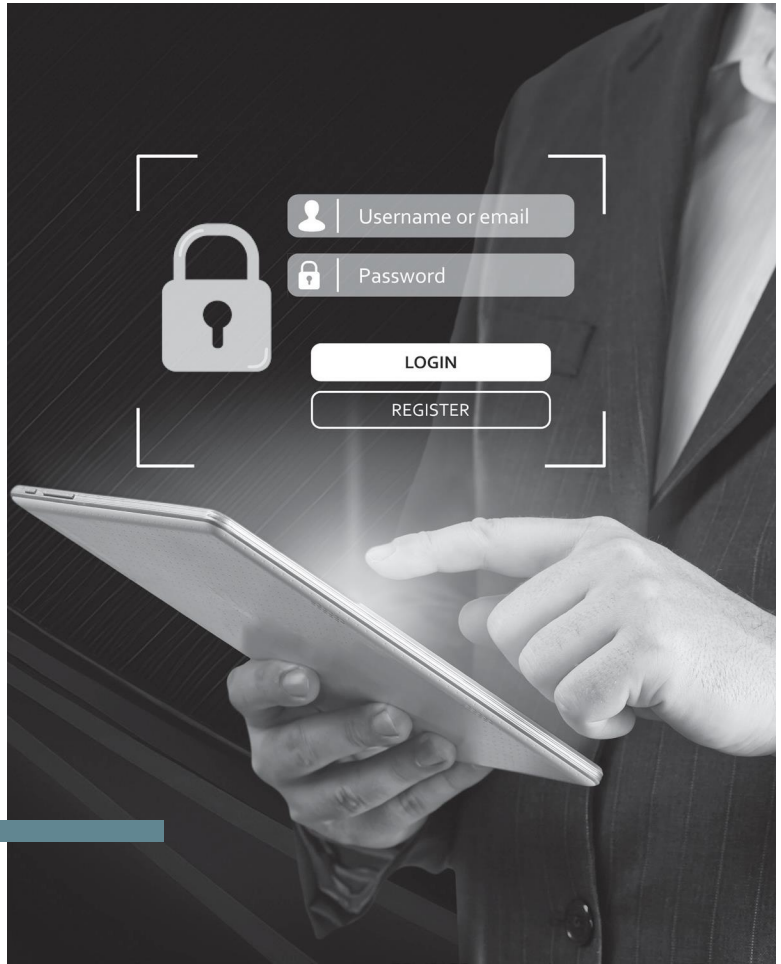


Autenticación

La autenticación se refiere al proceso de verificar la identidad de un usuario que intenta acceder a una base de datos o a una instancia de SQL Server. SQL Server ofrece diferentes opciones de autenticación, como:

- **Autenticación de Windows:** Permite a los usuarios utilizar sus credenciales de inicio de sesión de Windows para acceder a la base de datos. Esto implica que la autenticación se realiza a través de la cuenta de Windows del usuario.

- **Autenticación de SQL Server:** También conocida como autenticación mediante inicio de sesión, permite a los usuarios ingresar un nombre de usuario y contraseña específicos para acceder a la base de datos. Estas credenciales se almacenan dentro de la misma base de datos o en el servidor de SQL Server.



Autorización

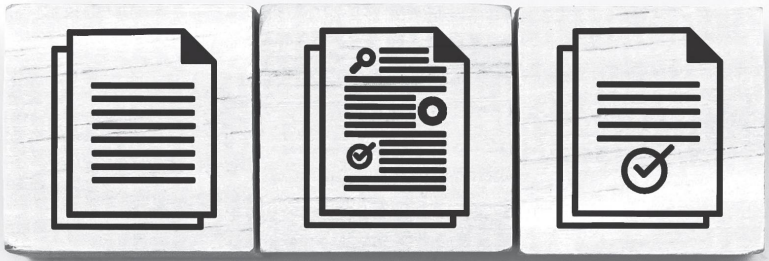
Una vez que un usuario ha sido autenticado y se ha establecido su identidad, la autorización determina los permisos y los niveles de acceso que se le conceden en la base de datos o en la instancia de SQL Server. Esto garantiza que los usuarios solo puedan acceder y manipular los datos a los que tienen permiso.

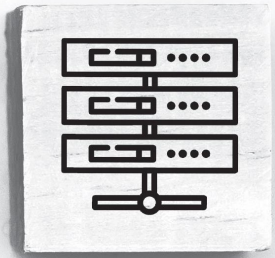
SQL Server proporciona un sistema de autorización basado en roles. Los roles permiten agrupar permisos y asignarlos a usuarios o a grupos de usuarios. Algunos roles predefinidos en SQL Server incluyen el rol de administrador del sistema (sysadmin), que tiene privilegios completos y control total sobre la instancia, y el rol de db_owner, que tiene control total sobre una base de datos específica.

Respaldo local

Realizar copias de seguridad de una base de datos de SQL Server es una práctica fundamental para proteger la integridad y disponibilidad de los datos en caso de fallos del sistema, errores humanos, corrupción de datos u otros eventos inesperados. A continuación, veremos cómo realizar un respaldo local de una base de datos en SQL Server:

Una de las ventajas de la copia de seguridad local es la capacidad de realizar una copia de seguridad de los datos detrás de un firewall de red.
La copia de seguridad local también es más rápida y proporciona un mayor control sobre quién puede acceder a los datos.





Creación de respaldo local

1. Abrir SQL Server Management Studio y conectar a la instancia de SQL Server donde se encuentra la base de datos que se desea respaldar. En Herramientas del sistema, expande Usuarios y grupos locales, haz clic con el botón derecho en Usuarios y luego haz clic en Nuevo usuario.

2. En el Explorador de objetos, expandir la carpeta “Bases de datos” y seleccionar la base de datos que se desea respaldar.

3. Hacer clic derecho sobre la base de datos y seleccionar “Tareas” y luego “Hacer copia de seguridad...”.

4. En la ventana de configuración de la copia de seguridad, se puede elegir la ubicación del archivo de respaldo, establecer opciones de compresión y cifrado, y configurar la programación si se desea realizar respaldos periódicos.

5. Hacer clic en “Aceptar” para comenzar el respaldo. Una vez finalizado, se habrá creado un archivo de respaldo (.bak) que contiene la base de datos en su estado actual.

Transact-SQL:

1. Abrir SQL Server Management Studio y conectar a la instancia de SQL Server.

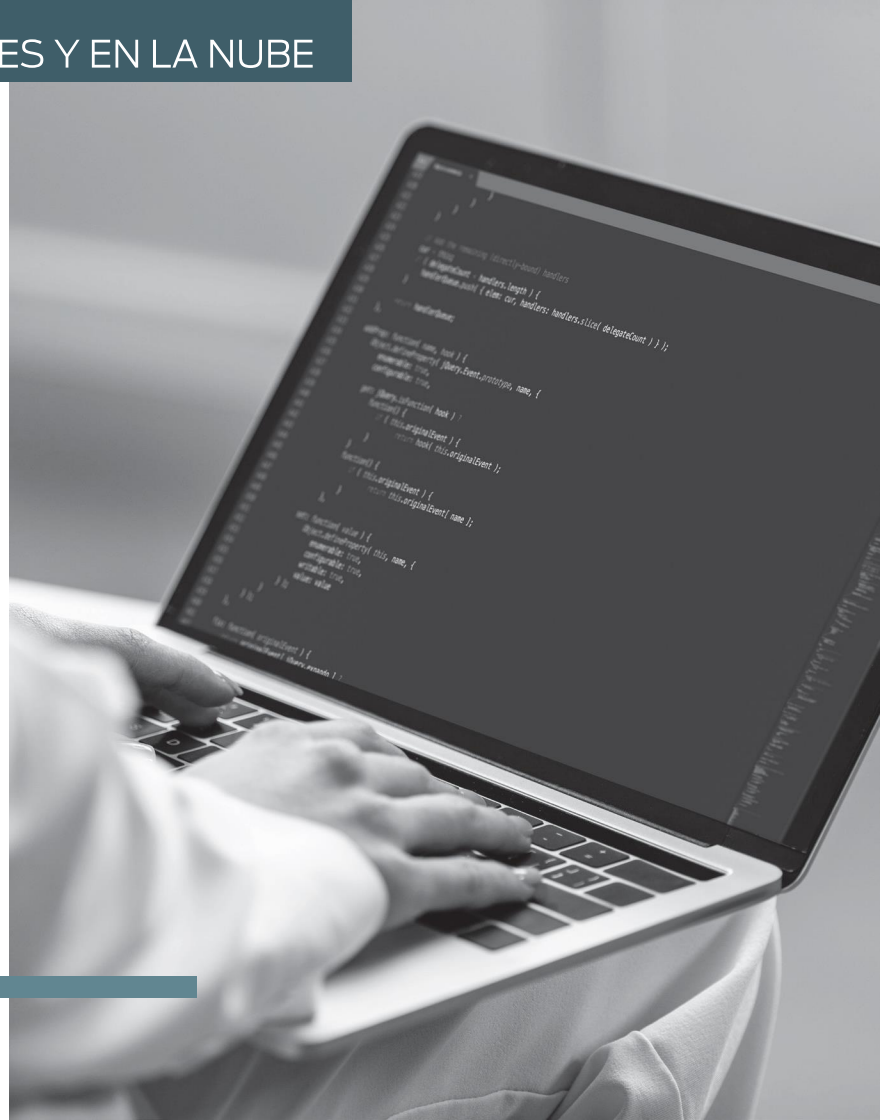
2. Abrir una nueva consulta y ejecutar el siguiente comando para realizar un respaldo completo de la base de datos:

```
BACKUP DATABASE [NombreBaseDatos] TO DISK = 'C:\Ruta\archivo.bak' WITH COMPRESSION, INIT;
```

3. Es importante recordar reemplazar “NombreBaseDatos” con el nombre de la base de datos y “C:\Ruta\archivo.bak” con la ubicación y el nombre de archivo deseado.

4. Ejecutar la consulta y el respaldo, se creará en la ubicación especificada.

Es importante tener en cuenta que el respaldo local debe almacenarse en un lugar seguro, preferiblemente en un dispositivo de almacenamiento externo o en un sistema de respaldo centralizado. Además, es recomendable establecer una estrategia de respaldo regular que se ajuste a las necesidades de la aplicación y asegurarse de que los respaldos sean probados y se puedan restaurar correctamente en caso de necesidad.



Respaldo en la nube

Realizar un respaldo de una base de datos SQL en la nube generalmente implica utilizar las herramientas y servicios específicos proporcionados por el proveedor de la nube en la que se encuentra alojada la base de datos. A continuación, se muestra un ejemplo de cómo realizar un respaldo de una base de datos SQL en dos de los proveedores de la nube más populares: AWS (Amazon Web Services) y Azure (Microsoft Azure).

AWS (Amazon Web Services):
AWS ofrece un servicio llamado Amazon RDS (Relational Database Service), que proporciona bases de datos SQL en la nube. Se pueden realizar respaldos de una base de datos de RDS utilizando las funciones de respaldo integradas. A continuación, se muestra un ejemplo de cómo realizar un respaldo utilizando la consola de AWS:

1. Acceder a la consola de AWS y navegar hasta el servicio de Amazon RDS.
2. Seleccionar la instancia de base de datos SQL que se desea respaldar.
3. En la pestaña “Punto de recuperación y restauración”, hacer clic en “Crear punto de recuperación”.
4. Seleccionar la opción de respaldo automático si se desea programar respaldos periódicos, o seleccionar “Crear instantánea de la base de datos” para realizar un respaldo manual.
5. Configurar los parámetros adicionales según necesidades y hacer clic en “Crear punto de recuperación” o “Crear instantánea de la base de datos” para iniciar el respaldo.

Una vez completado el respaldo, tendrás una copia de seguridad de tu base de datos SQL en la nube de AWS.

Azure (Microsoft Azure):
Azure ofrece el servicio Azure SQL Database, que proporciona bases de datos SQL en la nube. Se pueden realizar respaldos de una base de datos de Azure SQL utilizando las opciones de respaldo integradas en Azure Portal. A continuación, veremos un ejemplo de cómo realizar un respaldo utilizando Azure Portal:

1. Acceder a Azure Portal (portal.azure.com) y navegar hasta el servicio de Azure SQL Database.
2. Seleccionar la base de datos SQL que se desea respaldar.
3. En el menú de la base de datos, seleccionar “Copias de seguridad” en la sección “Operaciones”.
4. Hacer clic en “Crear” para crear una nueva copia de seguridad.
5. Configurar los parámetros de la copia de seguridad, como el nombre, la ubicación y la frecuencia, y hacer clic en “Crear” para iniciar el respaldo.
6. Una vez completado el respaldo, ya habrá una copia de seguridad de la base de datos SQL en la nube de Azure.

VIDEO

Te invitamos a ver el siguiente video:

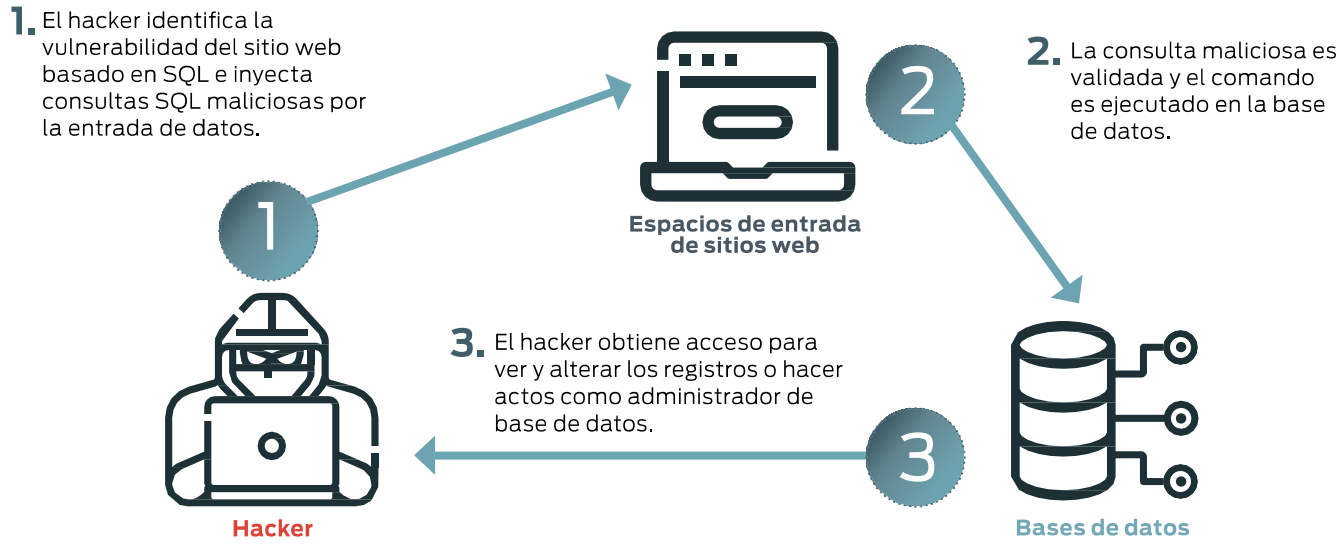
ATAQUES SQL Y ENCRIPTACIÓN DE DATOS

Ataques SQL

Los ataques SQL son intentos malintencionados de explotar vulnerabilidades en una aplicación o sistema que utiliza una base de datos SQL. Estos ataques tienen como objetivo acceder, manipular o eliminar datos confidenciales almacenados en la base de datos, así como obtener acceso no autorizado al sistema.

Inyección SQL
Encriptación de datos

Inyección SQL: Es uno de los ataques más comunes. Los atacantes insertan código SQL malicioso en las entradas de una aplicación web con el fin de engañar al sistema y ejecutar comandos SQL no deseados. Esto puede permitir al atacante obtener acceso a datos confidenciales, modificar o eliminar información en la base de datos, e incluso tomar el control del sistema.



Ejemplo:

Con base en la tabla: *Libros*

ID_libro	Nombre_libro	Autor_libro	Año_publicado
1	Programación	Francisco Pérez	2013
2	Bases de datos	Manuel Cruz	2011
3	Algoritmos	Pedro Paz	1988
4	Variables	Ernesto Villa	2018

Se tiene la siguiente consulta:
SELECT Nombre_libro FROM Libros WHERE Nombre_libros = ‘Variables’;

Ahora, si la cláusula proporcionada por el atacante es “Algoritmos’” (observa la comilla simple al final), la consulta SQL se verá de esta manera:

SELECT Nombre_libro FROM Libros WHERE Nombre_libros = ‘Algoritmos’ ‘;

Por lo general, esto dará como resultado un error de sintaxis porque la comilla simple al final confunde a SQL Server.

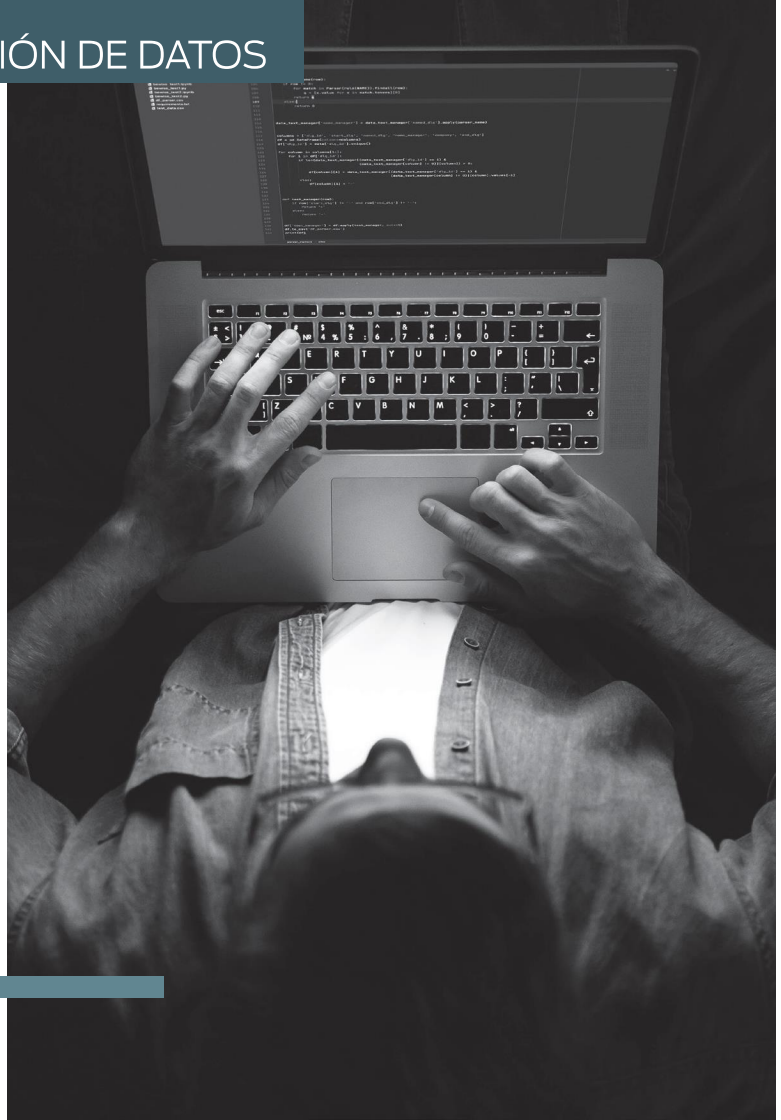
Entonces, ¿qué pasará si intentamos cambiar la naturaleza de la cláusula WHERE inyectando una cláusula diseñada para que siempre sea verdadera, como “anything” o “z’=’z”?

La consulta entonces se convierte en:

SELECT Nombre_libro FROM Libros WHERE Nombre_libro = ‘anything’ o ‘z’=’z’;

De esta forma siempre se cumplirán las condiciones, devolverá siempre todas las filas de la tabla.

El atacante puede continuar desde allí con algunas conjeturas en los nombres de las tablas e incluso llegar a donde, en lugar de simplemente copiar o robar la información de la base, el atacante está escribiendo datos en la base, si eso no está restringido.



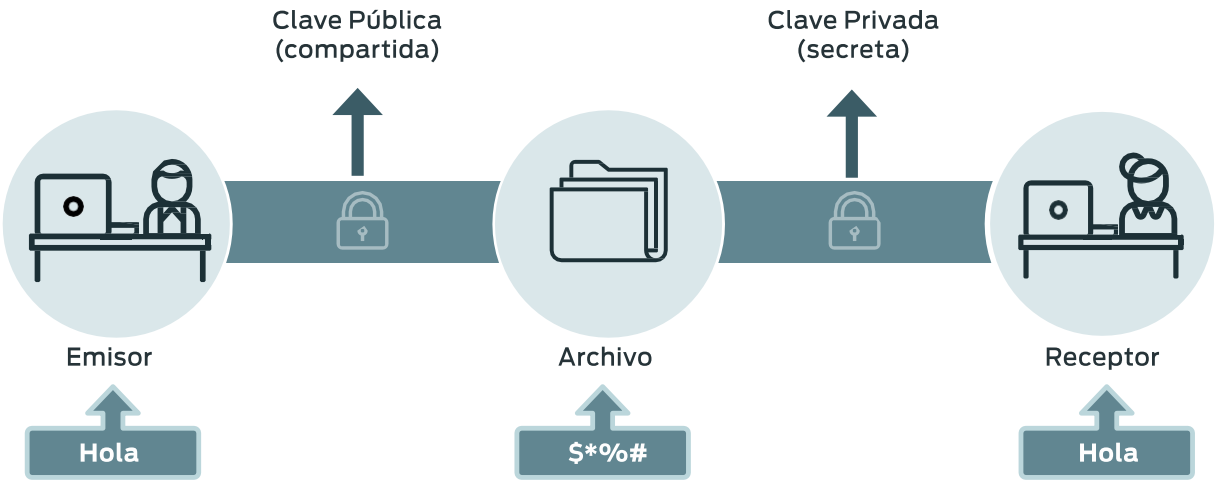
VIDEO

Te invitamos a ver el siguiente video:

Encriptación de datos

Es un intento de obtener acceso no autorizado a los datos encriptados almacenados en una base de datos SQL. Normalmente, los ataques de encriptación de datos se centran en obtener la clave de cifrado o descifrar los datos encriptados sin tener la autorización adecuada.

Veamos posibles escenarios de ataques de encriptación de datos en SQL:



1. **Ataque de fuerza bruta:** En este tipo de ataque, un atacante intenta adivinar la clave de cifrado probando diferentes combinaciones posibles. Si la clave es débil o se puede adivinar, el atacante podría descifrar los datos encriptados.

2. **Ataque de diccionario:** En lugar de probar todas las combinaciones posibles, un atacante puede utilizar un diccionario de palabras comunes o contraseñas para intentar descifrar la clave de cifrado. Esto es especialmente efectivo si la clave está relacionada con información personal o utiliza palabras fáciles de adivinar.

3. **Ataque de descifrado lateral:** En algunos casos, un atacante puede obtener acceso a la clave de cifrado a través de métodos indirectos, como obtener acceso a las credenciales de un administrador del sistema o mediante ataques a otros componentes del sistema que tienen acceso a las claves de cifrado.

4. **Ataque de inyección de código:** Si un sistema utiliza consultas dinámicas para realizar operaciones de encriptación, un atacante podría intentar aprovechar las vulnerabilidades de inyección de código para manipular las consultas y extraer información confidencial, incluyendo las claves de cifrado.

Para protegerse contra los ataques de encriptación de datos en SQL, es recomendable seguir las mejores prácticas de seguridad:

- Utilizar algoritmos de cifrado robustos y actualizados.
- Generar claves de cifrado fuertes y almacenarlas de forma segura, utilizando técnicas como el almacenamiento seguro de claves y la gestión adecuada de contraseñas.
- Implementar medidas de protección de la infraestructura, como cortafuegos, sistemas de detección de intrusiones y control de acceso adecuado a los servidores de bases de datos.
- Mantener los sistemas y aplicaciones actualizados con los últimos parches de seguridad.
- Limitar el acceso a las claves de cifrado solo a personal autorizado y seguir buenas prácticas de gestión de claves.
- Realizar pruebas de seguridad y auditorías periódicas para identificar y mitigar posibles vulnerabilidades.

Always Encrypted

Es una característica de Microsoft SQL Server diseñada para proteger datos sensibles almacenados en una base de datos SQL. Esta característica permite que los datos permanezcan encriptados tanto en reposo como en tránsito, incluso cuando están siendo procesados por aplicaciones y consultas.

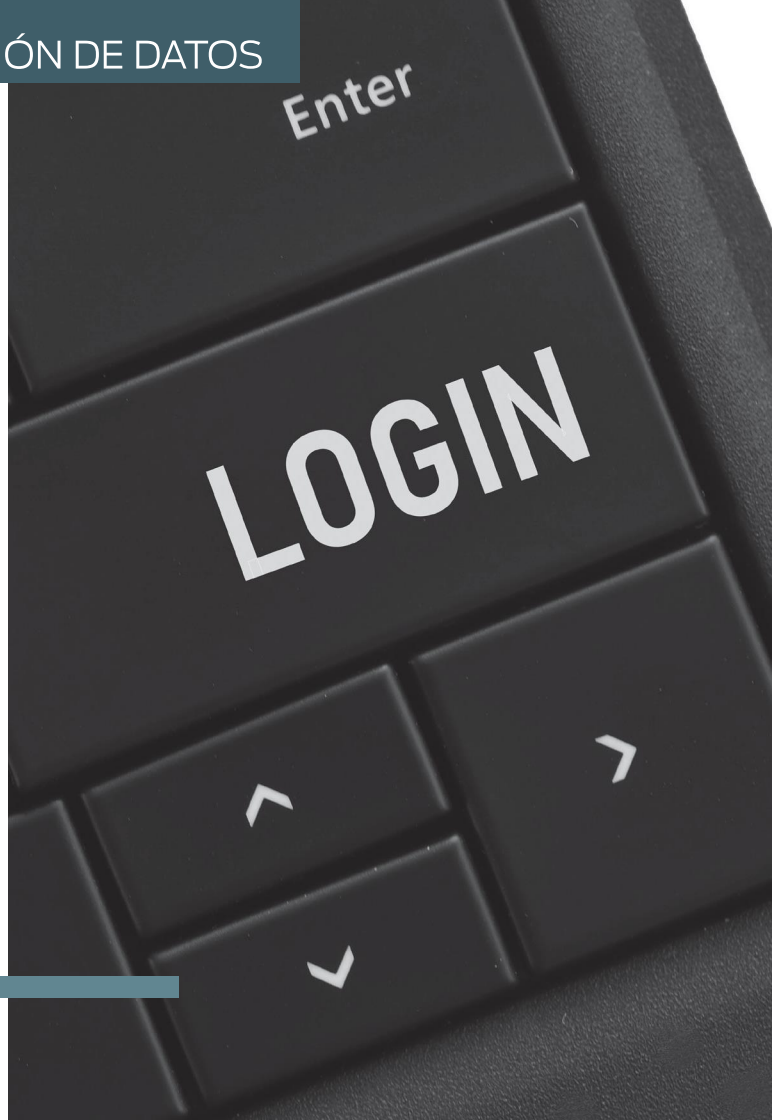
La principal ventaja de Always Encrypted es que proporciona un nivel adicional de seguridad al proteger los datos confidenciales, incluso si los atacantes logran acceder a la base de datos o interceptar el tráfico de red. Veamos unos conceptos clave importantes:

- Encriptación de columna
- Claves de encriptación
- Procesamiento de datos encriptados
- Separación de responsabilidades

Encriptación de columna:
Always Encrypted permite encriptar columnas específicas que contienen datos sensibles. Los datos se encriptan utilizando claves de encriptación generadas y gestionadas fuera de la base de datos. Esto significa que los datos encriptados se almacenan en la base de datos, y solo los clientes autorizados pueden descifrarlos y ver su contenido.

Claves de encriptación:
Always Encrypted utiliza un par de claves de encriptación asimétrica para proteger los datos. La clave de encriptación de columna (Column Encryption Key) se utiliza para encriptar y descifrar los datos almacenados en la columna, mientras que la clave maestra (Master Key) se utiliza para proteger la clave de encriptación de columna. Estas claves se almacenan en un almacén de claves externo y se gestionan por fuera de la base de datos.

Procesamiento de datos encriptados:
Con Always Encrypted, los datos permanecen encriptados en todo momento, incluso cuando se procesan por aplicaciones y consultas. Las aplicaciones cliente que necesitan acceder a los datos encriptados deben utilizar un controlador de cliente compatible con Always Encrypted y tener acceso a las claves de encriptación necesarias para descifrar los datos.



Separación de responsabilidades:
Always Encrypted permite separar las responsabilidades de los administradores de bases de datos y los desarrolladores de aplicaciones. Los administradores de bases de datos pueden administrar la infraestructura de base de datos sin tener acceso a los datos sensibles encriptados. Los desarrolladores de aplicaciones pueden trabajar con los datos encriptados sin necesidad de conocer las claves de encriptación.

ACTIVIDAD INTEGRADORA 2

Te invitamos a realizar la siguiente actividad.

Presiona el botón para descargar la actividad.

Presiona el botón para entregar la actividad.

LIBROS RECOMENDADOS

- Yaseen, A. (2019). *Respaldar y restaurar una base de datos SQL Server usando múltiples archivos.* SQL Shack - articles about database auditing, server performance, data recovery, and more.

- D5hyz. (2023). *Seguridad en Microsoft SQL Server.* Sothis.

CONCLUSIÓN

Los respaldos de bases de datos SQL son esenciales para la continuidad del negocio y la recuperación de datos, mientras que los ataques SQL representan un riesgo para la seguridad de los datos. Al implementar prácticas sólidas de respaldo y adoptar medidas de seguridad adecuadas, se puede fortalecer la protección de las bases de datos y mitigar los riesgos asociados con los ataques SQL.

¡FELICIDADES!

Acabas de concluir la tercera unidad de tu curso **Administración a las Bases de Datos**. Te invitamos a finalizar este esfuerzo realizando el examen parcial correspondiente. Para ello, debes regresar a la pantalla principal y dar clic en Presentar examen.

UNIDAD 4

DISPONIBILIDAD 24X7

TEMARIO

4.1 MONITOREO DE BASES DE DATOS

4.2 GRANJAS DE BASES DE DATOS DISTRIBUIDAS

INTRODUCCIÓN

En esta cuarta unidad aprenderás a gestionar la disponibilidad de bases de datos, a monitorearlas y, a su vez, a identificar cómo son las bases de datos distribuidas.

Todo esto con el fin de monitorear de mejor manera tu base de datos, para conocer el tiempo que le toma a la misma realizar ciertas tareas.

DISPONIBILIDAD 24X7

COMPETENCIAS A DESARROLLAR

- El alumno será capaz de implementar lo aprendido para monitorear bases de datos.
- El alumno será capaz de ejecutar granjas de bases de datos distribuidas.

MONITOREO DE BASES DE DATOS

DISPONIBILIDAD 24X7

Monitoreo de Base de Datos
Es una práctica fundamental para garantizar la disponibilidad, rendimiento y seguridad de los sistemas de bases de datos.
Los principales aspectos del monitoreo de bases de datos son:

- Supervisión del rendimiento
- Monitoreo de disponibilidad
- Seguimiento de eventos y registros
- Detección de problemas de seguridad
- Alertas y notificaciones
- Análisis y reportes

DISPONIBILIDAD 24X7

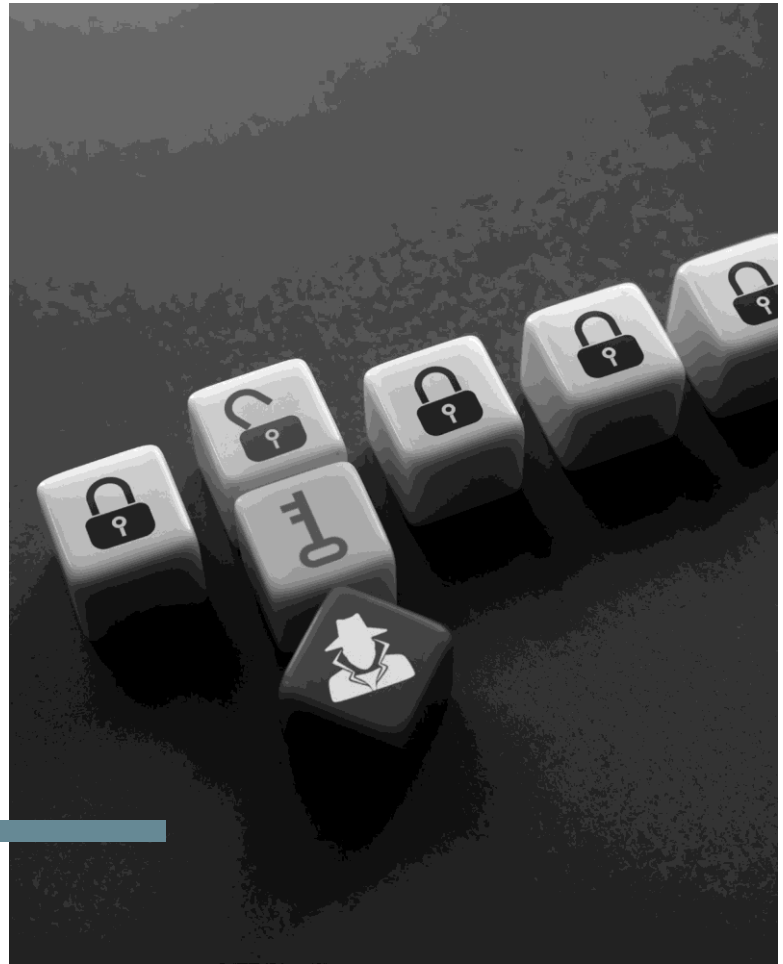
1. **Supervisión del rendimiento:** El monitoreo de bases de datos incluye el seguimiento y análisis del rendimiento del sistema, como el uso de recursos (CPU, memoria, almacenamiento), tiempos de respuesta de consultas, índices de utilización y latencia. Esto permite identificar cuellos de botella, problemas de rendimiento y optimizar la configuración de la base de datos para garantizar un rendimiento óptimo.

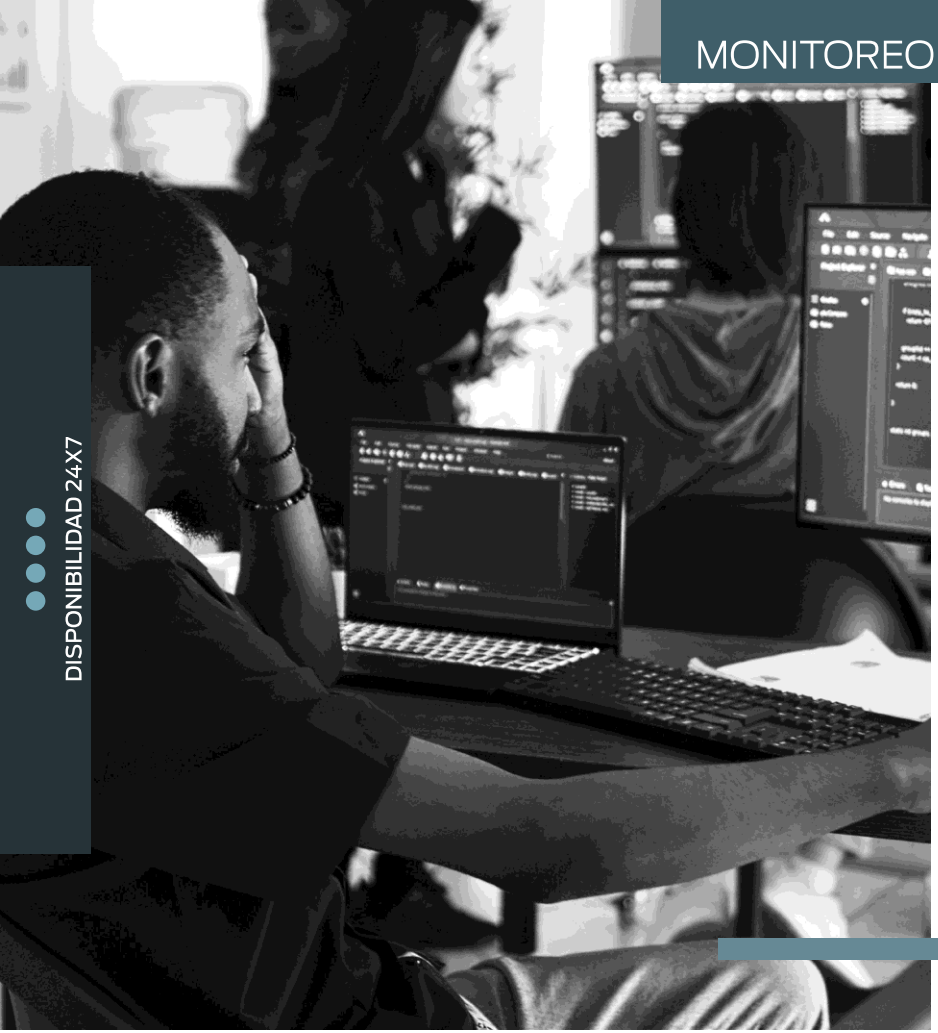
2. **Monitoreo de disponibilidad:** Se supervisa el estado y la disponibilidad de la base de datos para detectar y responder rápidamente a problemas como caídas del servidor, fallas de hardware, problemas de red o errores de configuración. Esto implica la monitorización de servicios, conexiones y disponibilidad de los componentes de la infraestructura relacionados con la base de datos.

DISPONIBILIDAD 24X7

3. **Seguimiento de eventos y registros:** El monitoreo de bases de datos implica la revisión de eventos y registros, como registros de transacciones, registros de errores y registros de auditoría. Esto permite detectar actividades anormales, intrusiones, errores de aplicación o accesos no autorizados a la base de datos.

4. **Detección de problemas de seguridad:** El monitoreo de bases de datos también abarca la detección de posibles vulnerabilidades y amenazas de seguridad. Esto puede incluir la supervisión de intentos de acceso no autorizados, ataques de inyección SQL, cambios no autorizados en los permisos de acceso o actividades sospechosas en la base de datos.





DISPONIBILIDAD 24X7

5. **Alertas y notificaciones:** El monitoreo de bases de datos se complementa con la configuración de alertas y notificaciones que informan sobre problemas críticos, umbrales de rendimiento superados o eventos de seguridad. Esto permite a los administradores y equipos de operaciones responder de manera proactiva a los problemas y tomar medidas correctivas rápidamente.

6. **Análisis y reportes:** El monitoreo de bases de datos también incluye el análisis de datos recopilados y la generación de informes periódicos sobre el rendimiento, la disponibilidad, la seguridad y otros aspectos relevantes de la base de datos.

AlwaysOn (SQL Server Management Studio)

Es una característica de alta disponibilidad y recuperación ante desastres en SQL Server Management Studio (SSMS) que permite la configuración de grupos de disponibilidad para bases de datos. Proporciona una solución robusta para mantener la disponibilidad de las bases de datos en caso de fallas del servidor o interrupciones planificadas.

Los conceptos y características principales de AlwaysOn en SQL Server Management Studio son los siguientes:

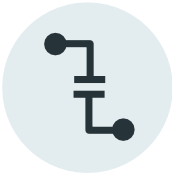
Grupos de disponibilidad
Réplicas primarias y secundarias
Sincronización de datos
Conmutación por error automática
Supervisión y administración

1. **Grupos de disponibilidad:** Un grupo de disponibilidad es una colección de bases de datos que se replican y se mantienen disponibles en varios servidores de SQL Server. Un grupo de disponibilidad incluye un conjunto de réplicas, donde una réplica primaria acepta las transacciones y las replica a las réplicas secundarias.

2. **Réplicas primarias y secundarias:** La réplica primaria es la instancia principal donde se realizan las operaciones de escritura y donde se originan las transacciones. Las réplicas secundarias son instancias de SQL Server que reciben y aplican las transacciones replicadas desde la réplica primaria.

3. **Sincronización de datos:** AlwaysOn admite dos modos de sincronización de datos: sincronización completa y sincronización asincrónica:

La sincronización completa garantiza que las réplicas secundarias estén completamente sincronizadas con la réplica primaria antes de confirmar una transacción.



Sincronización asincrónica permite un retraso en la replicación de las transacciones, lo que puede resultar en una menor latencia, pero con el riesgo de pérdida de datos en caso de una falla de la réplica primaria.

4. **Conmutación por error automática:** AlwaysOn proporciona la capacidad de conmutación por error automática, lo que significa que, si la réplica primaria experimenta una falla o se desconecta, una réplica secundaria se promociona automáticamente a réplica primaria, lo que minimiza el tiempo de inactividad y garantiza la continuidad del servicio.

5. **Supervisión y administración:** SSMS ofrece herramientas para supervisar y administrar los grupos de disponibilidad de AlwaysOn. Estas herramientas permiten realizar tareas como agregar o quitar réplicas, cambiar la configuración de sincronización, realizar conmutaciones por error manuales, monitorear el rendimiento y generar informes sobre el estado y la salud de los grupos de disponibilidad.

Opciones (SQL Server Always On, página del panel)

Se utiliza la página Panel de SQL Server Always On del cuadro de diálogo Opciones de SQL Server Management Studio para configurar el Panel de control Always On.

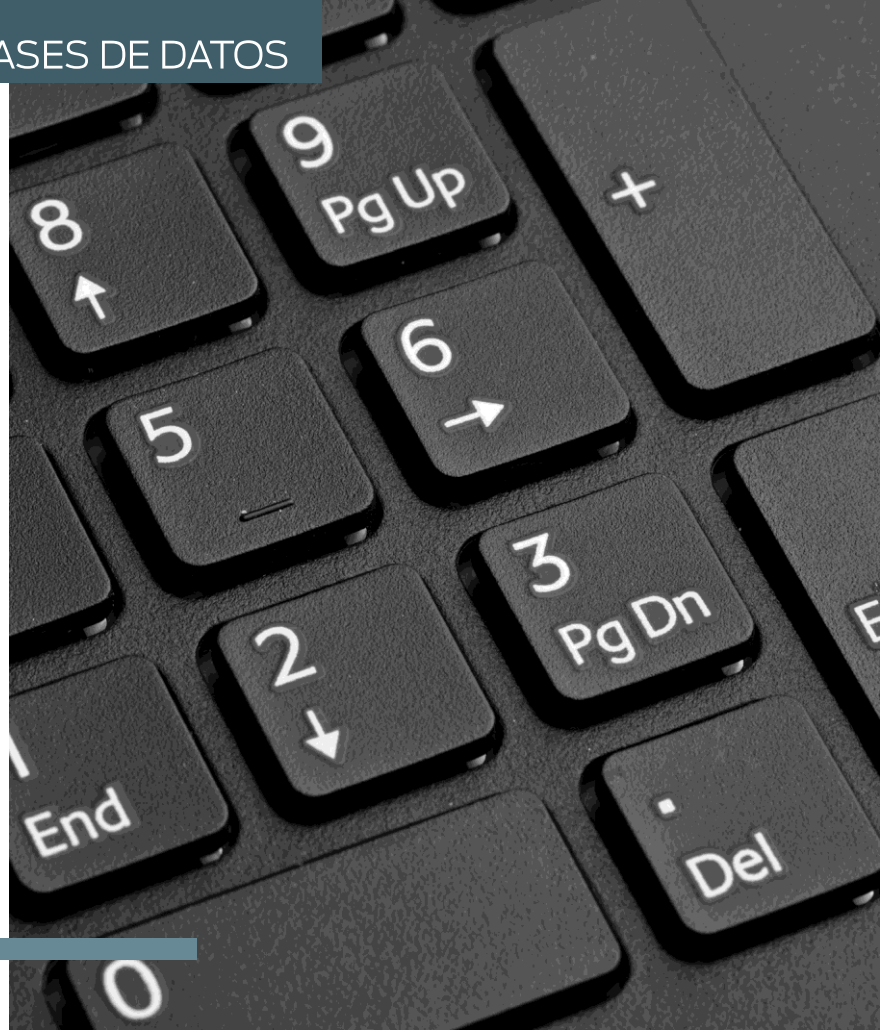
Para acceder a esta página:

1. Ingresa a SQL Server;
2. En el menú Tools, haz clic en Options;
3. Expande la carpeta de SQL Server Always On;
4. Haz clic en Dashboard.

Para activar la actualización automática
Haz clic para habilitar la actualización automática. Las opciones que muestra son:

- El campo *Intervalo de actualización (en segundos)* el cual muestra la cantidad de segundos en los que se actualizará el tablero. El valor predeterminado es de 30. Una vez que la actualización automática está habilitada, se puede editar este campo para cambiar el intervalo de actualización.

- El número de reintentos de conexión muestra el número de veces que el tablero intentará conectarse a una instancia de SQL Server que hospeda una réplica de disponibilidad para un grupo de disponibilidad que el tablero está monitoreando. El valor predeterminado es de 65535. Cuando la actualización automática está habilitada, se puede editar este campo para cambiar el número de reintentos de conexión.



Métricas SQL Server

Son indicadores clave que proporcionan información sobre el rendimiento, la utilización de recursos y la salud general de una instancia de SQL Server. Estas métricas son fundamentales para el monitoreo y la optimización del rendimiento del servidor. Las métricas más comunes son:

Uso de CPU
Uso de memoria
Uso de disco
Espacio en disco
Caché de datos
Bloqueos
Tiempo de respuesta de consultas
Conexiones activas

SQL Server provee dos formas integradas de monitoreo: **Activity Monitor y Data Collector.**

DISPONIBILIDAD 24X7

VIDEO

Te invitamos a ver el siguiente video:

Activity Monitor

Es una herramienta de monitoreo integrada en SQL Server Management Studio (SSMS) que permite supervisar y analizar la actividad en una instancia de SQL Server en tiempo real. Proporciona información detallada sobre el rendimiento, las sesiones, las consultas, el bloqueo y otros aspectos relacionados con la base de datos.

Al abrir el “Activity Monitor” en SSMS, se muestra una vista de resumen con diversas pestañas que muestran información relevante, como:

- Overview
- Processes
- Resource Waits
- Data File I/O
- Recent Expensive Queries
- Active Expensive Queries

1. **Overview:** Proporciona una vista general del rendimiento del servidor, incluyendo el uso de CPU, memoria, disco y red.

2. **Processes:** Muestra información detallada sobre las sesiones y procesos activos en el servidor, como el estado, la duración, el tiempo de CPU utilizado y la cantidad de lecturas y escrituras.

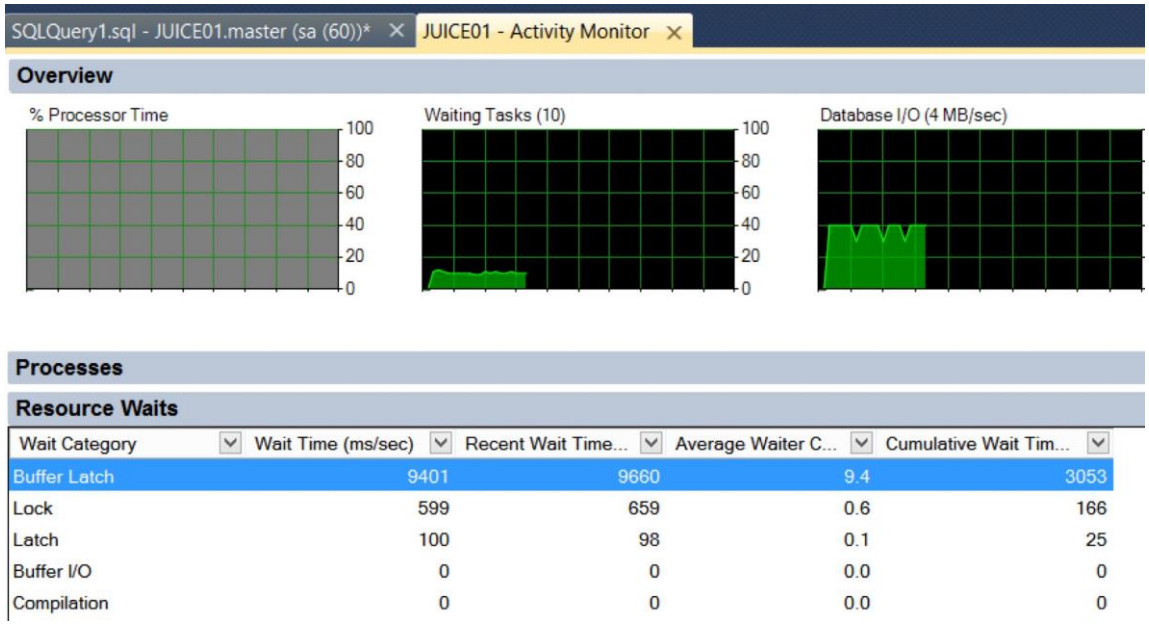
3. **Resource Waits:** Muestra las esperas de recursos que están ocurriendo en el servidor, lo cual puede ayudar a identificar cuellos de botella y áreas de mejora.

4. **Data File I/O:** Muestra información sobre la actividad de E/S en los archivos de datos, como el número de lecturas y escrituras por segundo, la latencia y el tamaño de los archivos.

5. **Recent Expensive Queries:** Enumera las consultas más costosas en términos de recursos utilizados, como tiempo de CPU, E/S y duración.

6. **Active Expensive Queries:** Muestra las consultas más costosas que se están ejecutando actualmente en el servidor.

Además de estas pestañas principales, el “Activity Monitor” también proporciona gráficos visuales y la capacidad de ordenar y filtrar la información para facilitar el análisis. Permite identificar rápidamente las consultas y sesiones que están consumiendo recursos de manera significativa, y ayuda a comprender el rendimiento general del servidor y las áreas que requieren atención.



Wait Category	Wait Time (ms/sec)	Recent Wait Time...	Average Waiter C...	Cumulative Wait Tim...
Buffer Latch	9401	9660	9.4	3053
Lock	599	659	0.6	166
Latch	100	98	0.1	25
Buffer I/O	0	0	0.0	0
Compilation	0	0	0.0	0

Data Collector

Es una característica de SQL Server que permite recopilar y almacenar datos de rendimiento y eventos de una instancia de SQL Server para su posterior análisis y diagnóstico. Proporciona una forma centralizada de recopilar datos de varias instancias de SQL Server en una ubicación común.

El Data Collector recopila información en tiempo real y la almacena en una base de datos especial llamada Management Data Warehouse (MDW). Algunos de los datos que se pueden recopilar incluyen:

Datos de rendimiento
Eventos del sistema
Ejecución de consultas
Estado de las bases de datos

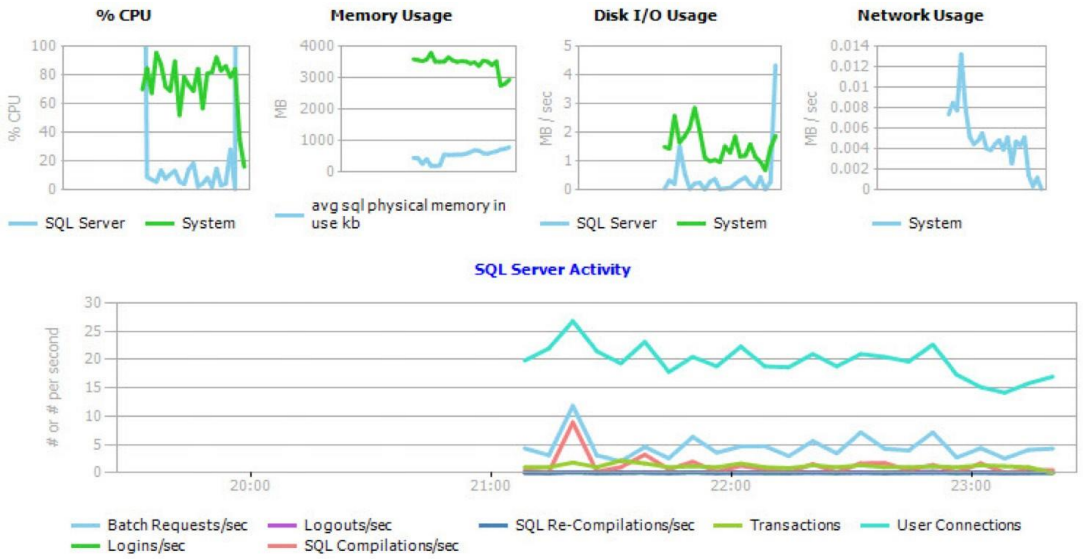
1. **Datos de rendimiento:** El Data Collector puede recopilar información sobre el uso de CPU, memoria, disco, red y otros recursos del sistema. Esto permite monitorear el rendimiento del servidor e identificar posibles cuellos de botella o problemas de capacidad.

2. **Eventos del sistema:** Se pueden recopilar eventos del sistema operativo y del servidor de SQL Server, como eventos de inicio o apagado, errores de inicio de sesión, errores de E/S, cambios en la configuración del servidor, entre otros. Esto ayuda a diagnosticar problemas y realizar un seguimiento del estado del servidor.

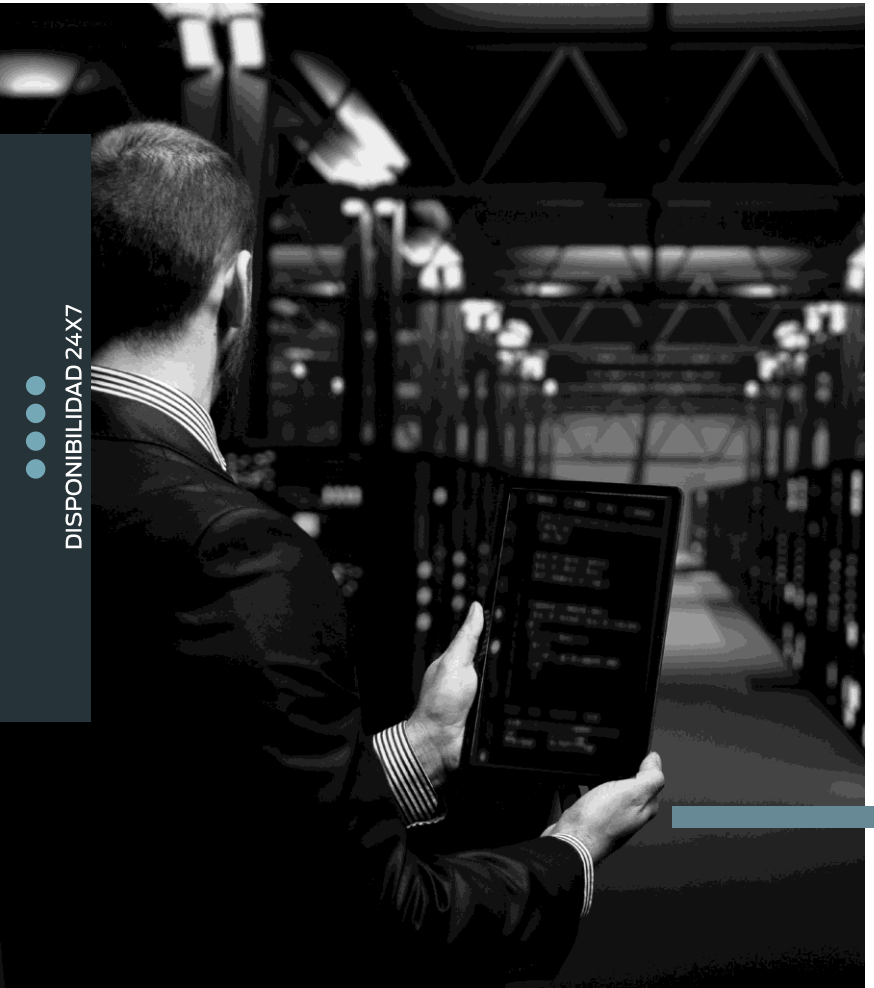
3. **Ejecución de consultas:** El Data Collector puede capturar información sobre las consultas ejecutadas en el servidor, como el tiempo de ejecución, el uso de CPU, los planes de ejecución y los bloqueos. Esto es útil para analizar y optimizar consultas problemáticas.

4. **Estado de las bases de datos:** Se puede recopilar información sobre el estado y el tamaño de las bases de datos, incluyendo el crecimiento de archivos, el espacio utilizado y los eventos de creación o modificación de bases de datos.

Además de la recopilación de datos, el Data Collector proporciona informes predefinidos y personalizables que permiten visualizar y analizar los datos recopilados. Estos informes ofrecen una visión general del rendimiento, la utilización de recursos y otros aspectos importantes de la instancia de SQL Server.



GRANJAS DE BASES DE DATOS DISTRIBUIDAS



Granjas de bases de datos distribuidas
Es una arquitectura en la que múltiples bases de datos están distribuidas en diferentes servidores o nodos de forma que trabajan de manera conjunta para procesar y almacenar datos. Esta configuración se utiliza para mejorar el rendimiento, la escalabilidad y la disponibilidad de las bases de datos en entornos de alto rendimiento.

GRANJAS DE BASES DE DATOS DISTRIBUIDAS

En una granja de bases de datos distribuidas, cada servidor o nodo contiene una copia de la base de datos o una parte específica de los datos. Los servidores están interconectados y trabajan de forma coordinada para procesar las consultas y transacciones de los usuarios. Algunas características y beneficios de las granjas de bases de datos distribuidas incluyen:

- Rendimiento escalable
- Alta disponibilidad
- Escalabilidad vertical y horizontal
- Tolerancia a fallos
- Mayor capacidad de almacenamiento

Para ver y modificar las propiedades del distribuidor

1. Conéctate al distribuidor en SQL Server Management Studio y luego expande el nodo del servidor.
2. Haz clic con el botón derecho en la carpeta Replication y luego haz clic en Configure Distribution...
3. Ve y modifica las propiedades en el cuadro de diálogo Propiedades del distribuidor - <Distribuidor> .
4. Para ver y modificar las propiedades de una base de datos de distribución, haz clic en el botón de propiedades (...) de la base de datos en la página General del cuadro de diálogo.
5. Para ver y modificar las propiedades del publicador asociadas con el distribuidor, haz clic en el botón de propiedades (...) del publicador en la página Publicadores del cuadro de diálogo.
6. Para acceder a los perfiles de los agentes de replicación, haz clic en el botón Valores predeterminados del perfil en la página General del cuadro de diálogo.
7. Modifica cualquier propiedad si es necesario y luego haz clic en Aceptar.

Para ver y modificar las propiedades de Publisher

1. Conéctate al publicador en SQL Server Management Studio y luego expande el nodo del servidor.

2. Haz clic con el botón derecho en la carpeta Replicación y luego haz clic en Propiedades del publicador.

3. Ve y modifica las propiedades en el cuadro de diálogo Propiedades de Publisher - < Publisher >

   - Un usuario con el rol de servidor fijo de administrador del sistema puede habilitar las bases de datos para la replicación en la página Bases de datos de publicación. Habilitar una base de datos no publica esa base; más bien, permite que cualquier usuario en el rol de base de datos fijo db_owner para esa base pueda crear una o más publicaciones en la base de datos.

4. Modifica cualquier propiedad si es necesario y luego haz clic en Aceptar.

PROYECTO FINAL

Te invitamos a realizar la siguiente actividad.

Presiona el botón para descargar la actividad.

Presiona el botón para entregar la actividad.

LIBROS RECOMENDADOS

- 4.3 Monitoreo. (2016, 2 diciembre). *Tópicos de Bases de Datos*.

- Curino, C., Jones, E. P., Madden, S., & Balakrishnan, H. (2011, June). *Workload-aware database monitoring and consolidation.* In Proceedings of the 2011 ACM SIGMOD International Conference on Management of data (pp. 313-324).

CONCLUSIÓN

Durante esta unidad aprendiste cómo una base de datos puede tener disponibilidad según sea dada por el creador de la misma.

De igual forma comprendiste cómo se puede monitorear una base de datos desde las dos formas que brinda SQL Server, y la forma de configurar bases de datos distribuidas, todo esto con el fin de administrar una base de datos tú solo.

DISPONIBILIDAD 24X7

¡FELICIDADES!

Acabas de concluir la cuarta unidad de tu curso **Administración a las Bases de Datos**. Te invitamos a finalizar este esfuerzo realizando el examen parcial correspondiente. Para ello, debes regresar a la pantalla principal y dar clic en Presentar examen.