“UNIVERSIDAD PERUANA LOS ANDES”

FACULTAD DE INGENIERÍA DE SISTEMAS Y COMPUTACIÓN

CURSO:

Base de Datos II

DOCENTE:

Mg.Raúl Fernandez Bejarano

CICLO:

V

AULA:

A1

INTEGRANTES:

Luis Ramirez Brayan Esnayder

HUANCAYO – PERU

2024

INTRODUCCION

El encriptado de datos es una técnica crucial en la seguridad de la información, que convierte datos legibles en un formato ilegible para protegerlos de accesos no autorizados. En SQL Server, existen funciones integradas que permiten encriptar y desencriptar información de manera eficiente, garantizando la confidencialidad e integridad de los datos almacenados en la base de datos. Este informe describe el proceso de encriptado y desencriptado de datos utilizando frases de paso en SQL Server, junto con los comandos y explicaciones detalladas.

Propósitos del Encriptado de Datos:

Confidencialidad:

- El principal propósito del encriptado es garantizar que solo las personas autorizadas puedan acceder y comprender los datos. Esto es especialmente importante en el manejo de información sensible, como contraseñas, datos financieros y datos personales. El encriptado asegura que incluso si los datos son interceptados, no puedan ser leídos sin la clave o frase de paso adecuada.

Integridad:

- El encriptado también ayuda a mantener la integridad de los datos. Al almacenar información de forma encriptada, podemos verificar si los datos han sido alterados durante su almacenamiento o transmisión. El uso de funciones de hash, por ejemplo, puede permitir detectar cambios en los datos y alertar sobre posibles manipulaciones no autorizadas.

Encriptado y Desencriptado en SQL Server

SQL Server proporciona diversas funciones para encriptar y desencriptar datos. En este caso, utilizamos las funciones *ENCRYPTBYPASSPHRASE* para encriptar datos y *DECRYPTBYPASSPHRASE* para desencriptarlos, utilizando una frase de paso como clave para ambos procesos.

Pasos para encriptar y desencriptar datos:

1. Crear la Tabla de Usuarios:

Primero, se debe crear una tabla en la base de datos que contendrá los datos del usuario, como su nombre y contraseña. En este ejemplo, la tabla se llama USUARIOS y tiene tres campos: usuario, password, y password_encriptado.

```
-- Crear la tabla USUARIOS con los campos necesarios
CREATE TABLE USUARIOS (
    usuario NVARCHAR(50),  -- Campo para almacenar el nombre de usuario
    password NVARCHAR(50),  -- Campo para almacenar la contraseña sin encriptar
    password_encriptado VARBINARY(128)  -- Campo para almacenar la contraseña encriptada
);
```

2. Encriptar la Contraseña con una Frase de Paso:

El siguiente paso es encriptar las contraseñas de los usuarios utilizando una frase de paso. En este ejemplo, utilizamos la función *ENCRYPTBYPASSPHRASE*, que requiere una frase de paso y el dato a encriptar (la contraseña).

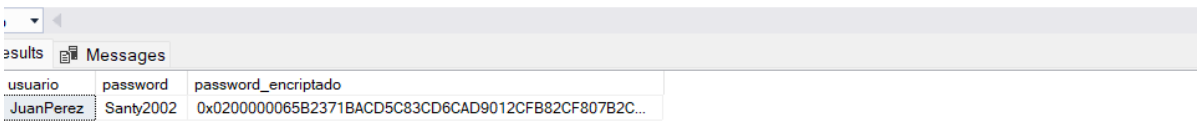
```
-- Declaramos las variables para la frase de paso y la contraseña a encriptar
DECLARE @frase NVARCHAR(50) = 'FraseSecreta';  -- Frase secreta utilizada para encriptar la contraseña
DECLARE @contraseña NVARCHAR(50) = 'Santy2002';  -- Contraseña a encriptar

-- Insertamos un usuario en la tabla con la contraseña encriptada
INSERT INTO USUARIOS (usuario, password, password_encriptado)
VALUES ('JuanPerez', @contraseña, ENCRYPTBYPASSPHRASE(@frase, @contraseña));  -- Encriptamos la contraseña
```

3. Visualizar los Datos Encriptados:

Una vez que los datos están encriptados, podemos visualizar los registros en la tabla USUARIOS mediante una consulta *SELECT*. En este caso, la contraseña se almacenará en formato binario en el campo *password_encriptado*.

```
-- Consultamos todos los datos de la tabla para visualizar los registros
SELECT * FROM USUARIOS;  -- Muestra todos los registros de la tabla, incluyendo la contraseña encriptada
```

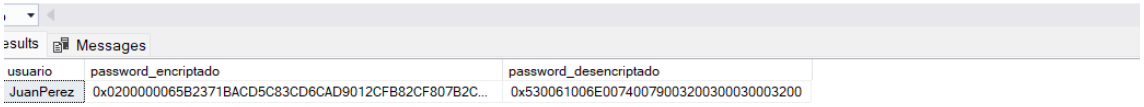


4. Agregar Más Usuarios a la Tabla:

A continuación, agregamos más usuarios a la tabla, encriptando sus contraseñas con la misma frase de paso. Esto asegura que las contraseñas se encripten de manera coherente.

```
-- Declaramos la variable para la frase de paso
DECLARE @frase NVARCHAR(50) = 'FraseSecreta';  -- Frase utilizada para encriptar y desencriptar las contraseñas

-- Consultamos todos los registros de la tabla 'USUARIOS' y desencriptamos las contraseñas
SELECT
    usuario,  -- Nombre del usuario
    password_encriptado,  -- Contraseña encriptada
    -- Desencriptamos la contraseña usando la frase de paso
    DECRYPTBYPASSPHRASE(@frase, password_encriptado) AS password_desencriptado
FROM
    USUARIOS;  -- Tabla donde están almacenadas las contraseñas
```



esults | Messages

usuario	password_encriptado	password_desencriptado
JuanPerez	0x0200000065B2371BACD5C83CD6CAD9012CFB82CF807B2C...	0x530061006E0074007900320030003200

Desencriptado con Frase Diferente:

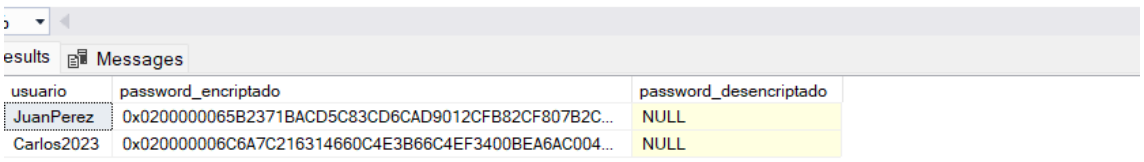
Si intentamos desencriptar con una frase de paso diferente a la utilizada para encriptar, no obtendremos la contraseña original. Por ejemplo, si encriptamos un valor con una frase y luego intentamos desencriptarlo con una frase distinta, los datos no serán correctos.

```
    -- Insertamos un nuevo usuario con una contraseña diferente
INSERT INTO USUARIOS (usuario, password)
VALUES ('Carlos2023', 'OtraContraseña456');  -- Contraseña sin encriptar

-- Encriptamos la nueva contraseña con una frase diferente
UPDATE USUARIOS
SET password_encriptado = ENCRYPTBYPASSPHRASE('OtraFraseSecreta', password)
WHERE usuario = 'Carlos2023';

-- Intentamos desencriptar los datos con una frase diferente
DECLARE @frase_diferente NVARCHAR(50) = 'FraseIncorrecta';

SELECT
    usuario,  -- Nombre del usuario
    password_encriptado,  -- Contraseña encriptada
    -- Intentamos desencriptar con una frase diferente
    DECRYPTBYPASSPHRASE(@frase_diferente, password_encriptado) AS password_desencriptado
FROM
    USUARIOS;
```



esults | Messages

usuario	password_encriptado	password_desencriptado
JuanPerez	0x0200000065B2371BACD5C83CD6CAD9012CFB82CF807B2C...	NULL
Carlos2023	0x020000006C6A7C216314660C4E3B66C4EF3400BEA6AC004...	NULL