Universidad Estatal a Distancia

Vicerrectoría Académica

Escuela de Ciencias Exactas y Naturales

Carrera de Diplomado en Ingeniería Informática

Proyecto Programado 3

Generación de procedimientos y funciones en una base de datos.

Asignatura: 00826 Base de Datos

Grupo: 07

Profesor: IRVIN FABIAN SEQUEIRA GAMBOA

Estudiante: Francisco Campos Sandi

Cedula: 114750560

Centro Universitario: San Vito

Primer cuatrimestre, 2024

Contenido

Introducción:

En el contexto del lenguaje SQL avanzado y el manejo de bases de datos, la creación de procedimientos almacenados y funciones juega un papel fundamental para automatizar tareas y realizar operaciones complejas dentro del entorno de la base de datos. El presente proyecto tiene como objetivo principal poner en práctica los conocimientos adquiridos en el curso, centrándose en la implementación de código SQL que permita ejecutar diversas operaciones a nivel de la base de datos utilizando Microsoft SQL Express como DBMS.

Objetivo del Proyecto

El propósito de este proyecto es desarrollar un procedimiento almacenado y una función que aborden necesidades específicas de una aplicación de gestión, basándose en un diseño entidad-relación previamente establecido. El procedimiento almacenado se centrará en la inserción de nuevos servicios en la base de datos, validando la existencia de empleados antes de la inserción. Por otro lado, la función permitirá obtener información detallada de un cliente y su vehículo para una orden de trabajo determinada, asegurando que la orden esté activa y no cancelada.

Metodología y Desarrollo

Para la implementación de este proyecto, se utilizarán como punto de partida el diseño entidad-relación, el script de creación de tablas y los datos de ejemplo previamente publicados en la plataforma del curso. Asimismo, se hará uso de la carga de datos realizada en el proyecto anterior para garantizar coherencia en el entorno de pruebas.

Estructura del Documento y Entregables

El documento final del proyecto estará organizado de acuerdo con las reglas generales para la presentación de trabajos académicos, incluyendo portada, índice, introducción, desarrollo, conclusión y bibliografía en formato APA. En la sección de desarrollo, se incluirán las sentencias SQL utilizadas para la creación del procedimiento almacenado y la función. Además, se presentará la evidencia de los resultados obtenidos, mostrando la ejecución exitosa del procedimiento y la función en el motor de base de datos Microsoft SQL Express.

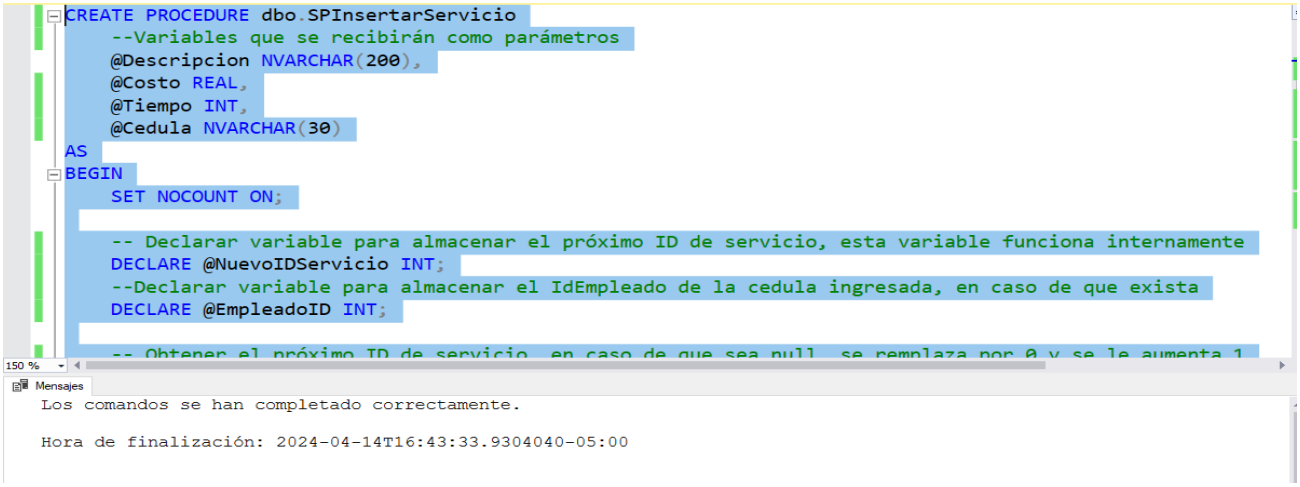
Script completo de la parte1:

```
/*
1. Crear un procedimiento almacenado que incluya un nuevo servicio en la base de
datos.
La lógica del procedimiento almacenado es la siguiente:
se recibe como parámetros la descripción del servicio, el costo de la mano de
obra,
el tiempo estimado del servicio y la cédula del empleado que presta el servicio.
Antes de insertar  debe  validar  que  el  empleado  exista  en  la  tabla
Empleado.
Si  las validaciones son exitosas, inserta los valores en la tabla Servicios
*/

USE AUTOTECH
CREATE PROCEDURE dbo.SPInsertarServicio
       --Variables que se recibirán como parámetros
    @Descripcion NVARCHAR(200),
    @Costo REAL,
    @Tiempo INT,
    @Cedula NVARCHAR(30)
AS
BEGIN
    SET NOCOUNT ON;

    -- Declarar variable para almacenar el próximo ID de servicio, esta variable
funciona internamente
    DECLARE @NuevoIDServicio INT;
       --Declarar variable para almacenar el IdEmpleado de la cedula ingresada,
en caso de que exista
       DECLARE @EmpleadoID INT;

    -- Obtener el próximo ID de servicio, en caso de que sea null, se remplaza
por 0 y se le aumenta 1
    SELECT @NuevoIDServicio = ISNULL(MAX(IDServicio), 0) + 1 FROM dbo.SERVICIOS;

    -- Verificar si el empleado existe
       --Si no existe 1 registro de la tabla empleados con la cedula que se está
ingresando
    IF NOT EXISTS (SELECT 1 FROM dbo.EMPLEADOS WHERE Cedula = @Cedula)
    BEGIN
             --se muestra en pantalla que el empleado no existe
        PRINT 'El Empleado Con La Cédula '+@Cedula+' no existe';
        RETURN;
    END;

       SELECT @EmpleadoID=(SELECT IDEmpleado FROM dbo.EMPLEADOS WHERE Cedula =
@Cedula);
    -- Insertar el nuevo servicio
       --Se indica los campos en el orden que se van a insertar
    INSERT INTO dbo.SERVICIOS (IDServicio, Descripcion, CostoManoObra,
TiempoEjecucion, IDEmpleado)
    VALUES (@NuevoIDServicio, @Descripcion, @Costo, @Tiempo, @EmpleadoID);

       --Si los errores son 0
    IF @@ERROR = 0
```

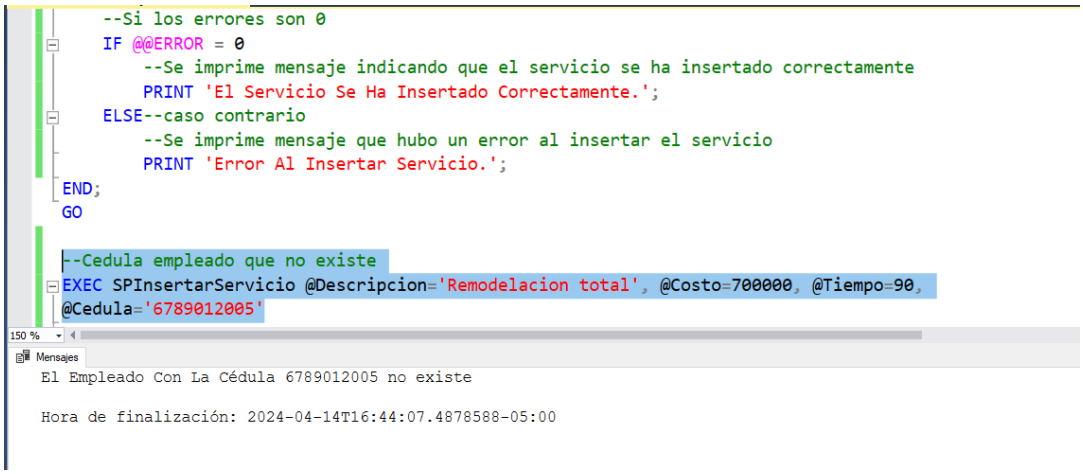
```
            --Se imprime mensaje indicando que el servicio se ha insertado
correctamente
        PRINT 'El Servicio Se Ha Insertado Correctamente.';
    ELSE--caso contrario
            --Se imprime mensaje que hubo un error al insertar el servicio
        PRINT 'Error Al Insertar Servicio.';
END;
GO

--Cedula empleado que no existe
EXEC SPInsertarServicio @Descripcion='Remodelacion total', @Costo=700000,
@Tiempo=90,
@Cedula='6789012005'

--Cedula empleado que si existe
EXEC SPInsertarServicio @Descripcion='Cambio de pintura', @Costo=200000,
@Tiempo=25,
@Cedula='6789012345'



--ver todos los servicios
SELECT * FROM SERVICIOS
```

1- Crear Procedimiento Almacenado que incluya un nuevo servicio en la base de datos.

Procedimiento almacenado creado correctamente



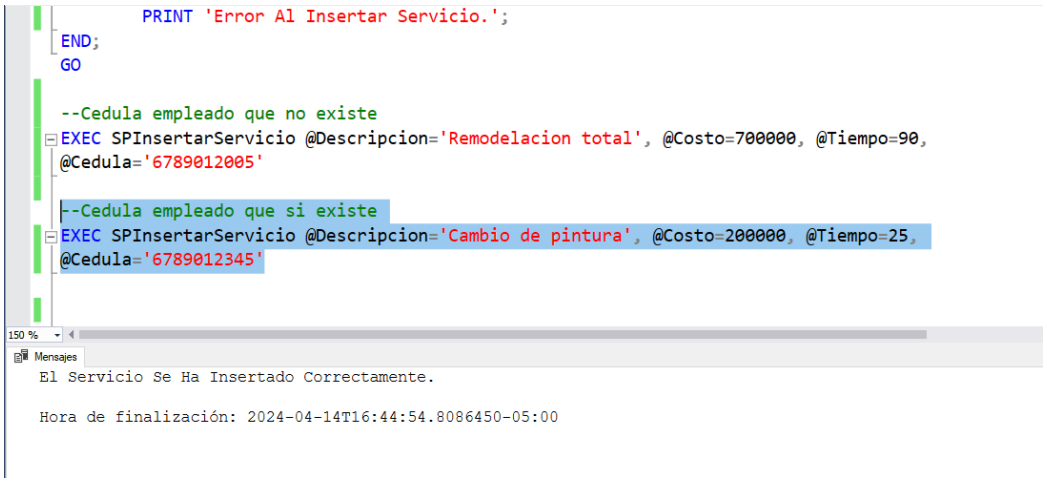
Ejecutando procedimiento con una cédula que no existe



```
    --Si los errores son 0
    IF @@ERROR = 0
        --Se imprime mensaje indicando que el servicio se ha insertado correctamente
        PRINT 'El Servicio Se Ha Insertado Correctamente.';
    ELSE--caso contrario
        --Se imprime mensaje que hubo un error al insertar el servicio
        PRINT 'Error Al Insertar Servicio.';
END;
GO

--Cedula empleado que no existe
EXEC SPInsertarServicio @Descripcion='Remodelacion total', @Costo=700000, @Tiempo=90,
@Cedula='6789012005'
```

```
Mensajes
El Empleado Con La Cédula 6789012005 no existe

Hora de finalización: 2024-04-14T16:44:07.4878588-05:00
```

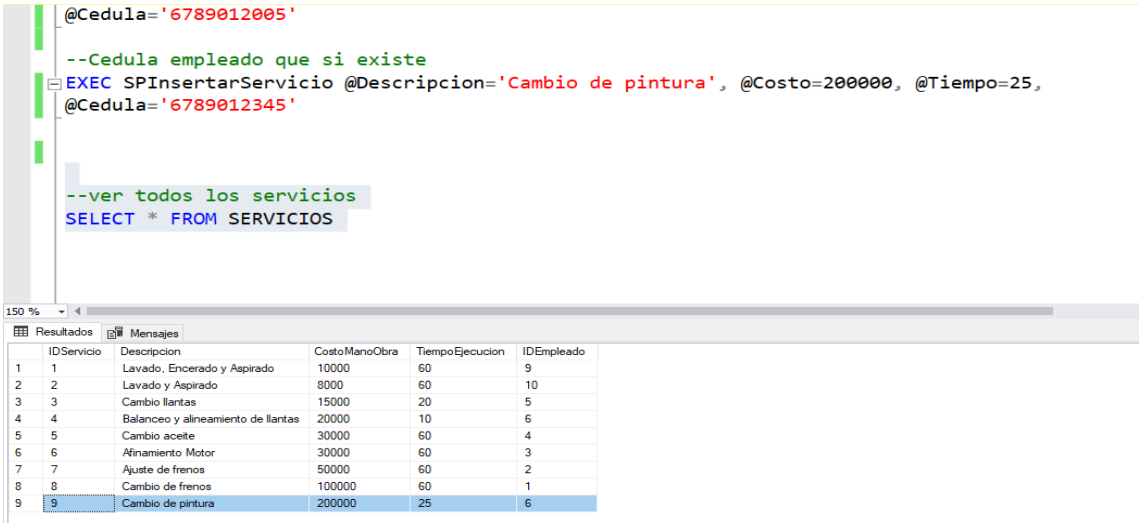
Ejecutando procedimiento con una cédula que si existe:



```
        PRINT 'Error Al Insertar Servicio.';
END;
GO

--Cedula empleado que no existe
EXEC SPInsertarServicio @Descripcion='Remodelacion total', @Costo=700000, @Tiempo=90,
@Cedula='6789012005'

--Cedula empleado que si existe
EXEC SPInsertarServicio @Descripcion='Cambio de pintura', @Costo=200000, @Tiempo=25,
@Cedula='6789012345'
```

```
Mensajes
El Servicio Se Ha Insertado Correctamente.

Hora de finalización: 2024-04-14T16:44:54.8086450-05:00
```

Al ejecutar select para ver todos los servicios, observamos el nuevo servicio que se ha insertado correctamente.



```
@Cedula='6789012005'

--Cedula empleado que si existe
EXEC SPInsertarServicio @Descripcion='Cambio de pintura', @Costo=200000, @Tiempo=25,
@Cedula='6789012345'


--ver todos los servicios
SELECT * FROM SERVICIOS
```

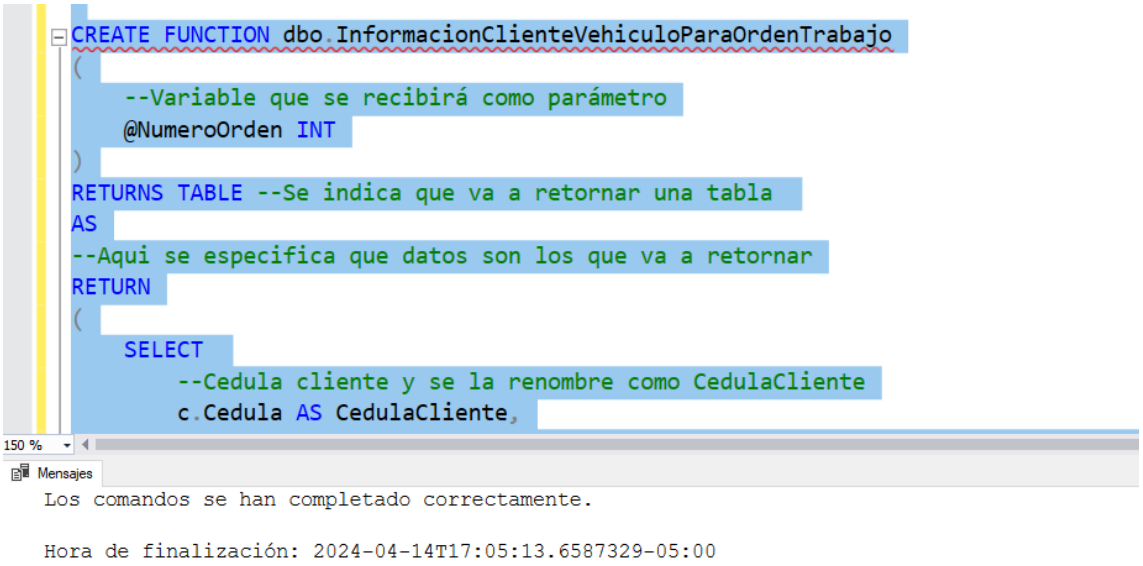
150 %

Resultados | Mensajes

	IDServicio	Descripcion	CostoManoObra	TiempoEjecucion	IDEmpleado
1	1	Lavado, Encerado y Aspirado	10000	60	9
2	2	Lavado y Aspirado	8000	60	10
3	3	Cambio llantas	15000	20	5
4	4	Balanceo y alineamiento de llantas	20000	10	6
5	5	Cambio aceite	30000	60	4
6	6	Afinamiento Motor	30000	60	3
7	7	Ajuste de frenos	50000	60	2
8	8	Cambio de frenos	100000	60	1
9	9	Cambio de pintura	200000	25	6

2- CREAR FUNCIÓN

Función creada correctamente



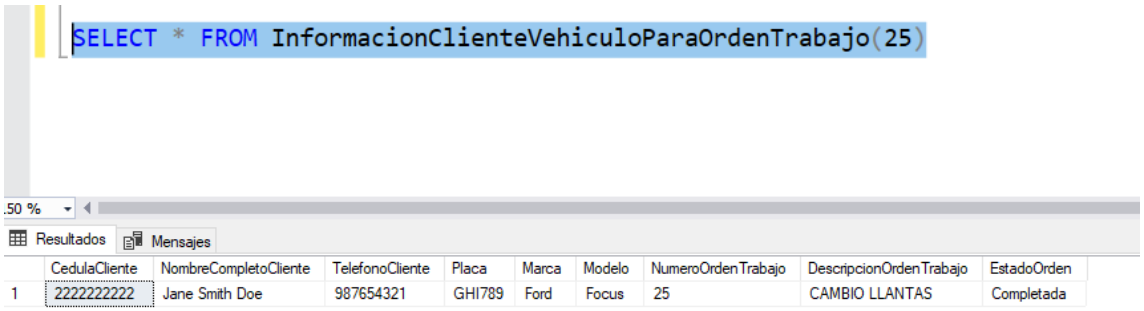
```
CREATE FUNCTION dbo.InformacionClienteVehiculoParaOrdenTrabajo
(
    --Variable que se recibirá como parámetro
    @NumeroOrden INT
)
RETURNS TABLE --Se indica que va a retornar una tabla
AS
--Aqui se especifica que datos son los que va a retornar
RETURN
(
    SELECT
        --Cedula cliente y se la renombre como CedulaCliente
        c.Cedula AS CedulaCliente,
```

150 %

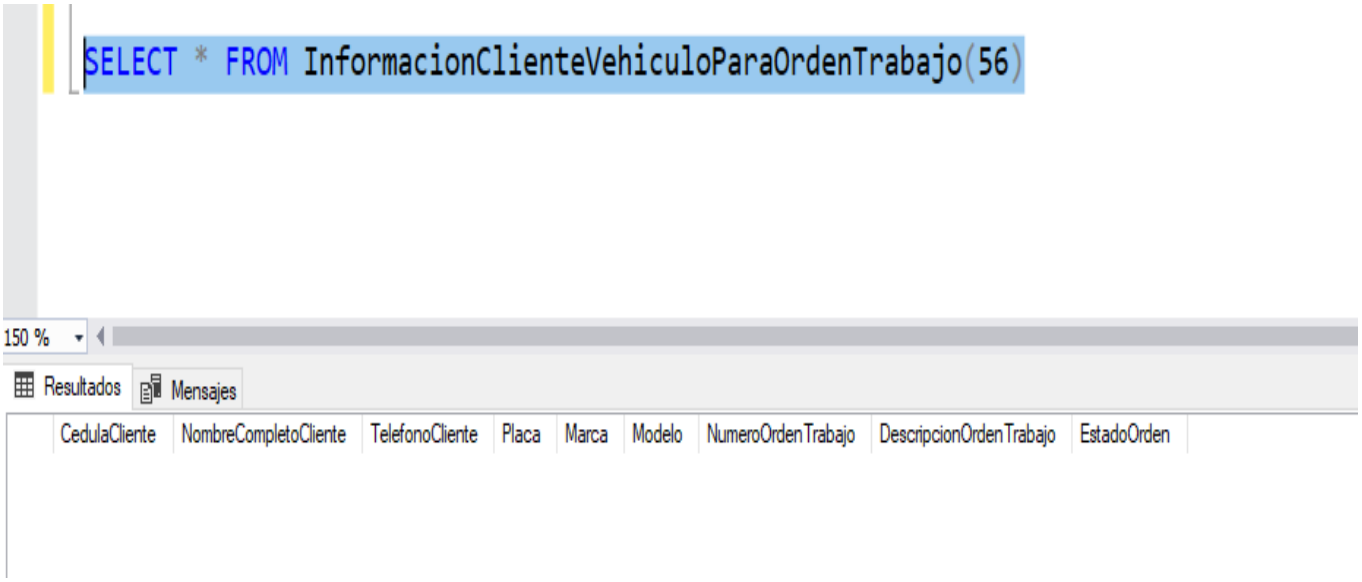
Mensajes

```
Los comandos se han completado correctamente.

Hora de finalización: 2024-04-14T17:05:13.6587329-05:00
```

Ejecutando función con un número de orden que si existe en la base de datos



Ejecutando la función con un número de orden que no existe en la base de datos



Script completo de la parte 2:

```
/*
Crear una función que permita obtener la información de un cliente y su vehículo
para una orden de trabajo específica. La lógica de la función es la siguiente:
se recibe como parámetro el número de orden y se verifica si esa orden existe y
no se encuentra en estado cancelada.
Si se cumple con las condiciones, se realiza una consulta en las tablas
correspondientes
y se retorna el número de cédula, nombre completo y teléfono del cliente.
También, el número de placa, marca y modelo del vehículo del cliente.
dicionalmente, el número de orden de trabajo, la descripción y el estado de la
orden.
*/

CREATE FUNCTION dbo.InformacionClienteVehiculoParaOrdenTrabajo
(
```

```
       --Variable que se recibirá como parámetro
    @NumeroOrden INT
)
RETURNS TABLE --Se indica que va a retornar una tabla
AS
--Aqui se especifica que datos son los que va a retornar
RETURN
(
    SELECT
              --Cedula cliente y se la renombre como CedulaCliente
        c.Cedula AS CedulaCliente,
              --Se concatena nombres y apellidos y se renombra como
NombreCompletoCliente
        CONCAT(c.Nombre, ' ', c.Apellido1, ' ', c.Apellido2) AS
NombreCompletoCliente,
              --el telefono se lo renombre como TelefonoCliente
        c.Telefono AS TelefonoCliente,
              --Placa, Marcar y Modelo del vehículo
              v.Placa, v.Marca, v.Modelo,
              --El numero de orden se lo renombra como NumeroOrdenTrabajo
        o.NumeroOrden AS NumeroOrdenTrabajo,
        --La descripcion de la orden se la renombre como DescripcionOrdenTrabajo
              o.Descripcion AS DescripcionOrdenTrabajo,
        --La descripcion de estado se la renombra como EstadoOrden
              e.Descripcion AS EstadoOrden
    FROM
        dbo.ORDENES o --Ordenes es o
       --Inner Join con la tabla clientes donde clientes es c
    INNER JOIN dbo.CLIENTES c ON o.CedulaCliente = c.Cedula
    --Inner Join con la tabla vehiculo donde vehiculos es v
       INNER JOIN dbo.VEHICULO v ON o.PlacaVehiculo = v.Placa
       --Inner Join con la tabla clientes donde ordenes estados es e
    INNER JOIN dbo.ORDENES_ESTADOS e ON o.EstadoOrden = e.IDEstado
       --La condición es que el número de orden debe ser igual al nro orden
ingresado
       --y el estado de la orden debe ser distinto a cancelada
    WHERE o.NumeroOrden = @NumeroOrden AND e.Descripcion <> 'Cancelada'
);

SELECT * FROM InformacionClienteVehiculoParaOrdenTrabajo(25)
```

Conclusiones

La implementación de procedimientos almacenados y funciones en una base de datos utilizando SQL avanzado ha demostrado ser una estrategia efectiva para optimizar procesos y automatizar tareas repetitivas. Mediante el uso de procedimientos almacenados, pudimos encapsular lógica compleja dentro de la base de datos, lo que facilita su reutilización y mantenimiento. Por ejemplo, el procedimiento diseñado para agregar nuevos servicios validó automáticamente la existencia de empleados antes de realizar la inserción en la tabla de servicios, lo que garantiza la integridad de los datos y simplifica el flujo de trabajo para futuras operaciones.

La aplicación de funciones SQL para obtener información detallada sobre clientes y vehículos en el contexto de órdenes de trabajo ha contribuido significativamente a mejorar la experiencia del usuario y la eficiencia operativa. Al permitir consultas rápidas y precisas con parámetros específicos, como el número de orden de trabajo, hemos facilitado el acceso a datos críticos de manera eficaz. Esta mejora en la accesibilidad de datos puede traducirse en una toma de decisiones más informada y una respuesta más rápida a las necesidades comerciales.

La centralización de la lógica de negocio dentro de la base de datos a través de procedimientos almacenados y funciones proporciona ventajas notables en términos de mantenimiento y coherencia. Al tener la lógica de procesamiento y validación directamente integrada en la base de datos, podemos asegurar la consistencia de los datos y simplificar la gestión del sistema en su conjunto. Además, la separación de las capas de aplicación y datos permite una mayor modularidad y facilita futuras actualizaciones o cambios en los requisitos comerciales sin afectar la integridad de la base de datos.

En resumen, la aplicación práctica de procedimientos almacenados y funciones en SQL no solo ha facilitado la implementación de lógica empresarial compleja, sino que también ha contribuido a mejorar la eficiencia operativa y la experiencia del usuario al simplificar el acceso a datos críticos y optimizar los flujos de trabajo.

Referencias:

learn.microsoft. 26 de 09 de 2022. Rescatado de:https://learn.microsoft.com/es-es/sql/relational-databases/user-defined-functions/create-user-defined-functions-database-engine?view=sql-server-ver16. 15 de 04 de 2024.

learn.microsoft. *CREAR FUNCIÓN (Transact-SQL)*. 18 de 11 de 2022. Rescatado de:https://learn.microsoft.com/en-us/sql/t-sql/statements/create-function-transact-sql?view=sql-server-ver16&viewFallbackFrom=sql-serverver15. 15 de 04 de 2024.

Tecno Zona. *youtube*. 10 de 10 de 2007. Rescatado de: https://www.youtube.com/watch?v=fe15wMKP0a0. 15 de 04 de 2024.

Tutoriales, Alexander. *youtube*. 30 de 07 de 2014. Rescatado de:https://www.youtube.com/@AlexanderTutoriales19/about. 15 de 04 de 2024.

UskoKruM2010. *youtube*. 11 de 10 de 2010. Rescatado de:https://www.youtube.com/@UskoKruM2010/about. 15 de 04 de 2024.

Valenzuela, Manuel. *youtube*. 13 de 05 de 2014. Rescatado de: https://www.youtube.com/watch?v=JHli2ea0ANk. 16 de 04 de 2024.