	SERVICIO NACIONAL DE APRENDIZAJE SENA **SISTEMA INTEGRADO DE GESTIÓN** **Procedimiento Ejecución de la Formación Profesional Integral** **GUÍA DE APRENDIZAJE**	Versión: 02 Código: GFPI-F-019

GUÍA DE APRENDIZAJE Nº 15

1. IDENTIFICACIÓN DE LA GUIA DE APRENDIZAJE

<table>
<tr><td>Programa de Formación: Técnico en programación de software</td><td>Código:
Versión:</td><td colspan="3">228120
100</td></tr>
<tr><td>Nombre del Proyecto:
Sistema de información para la gestión
Empresarial V.1.3</td><td>Código:</td><td colspan="3">704330</td></tr>
<tr><td colspan="2">Fase del proyecto:
No. 2 Desarrollo del proyecto</td><td colspan="3"></td></tr>
<tr><td rowspan="2">Actividad (es) del Proyecto:
Desarrollo de la base de datos.</td><td rowspan="2">Actividad (es) de Aprendizaje:
Relacionar las tablas construidas teniendo en cuenta las restricciones (primary key, foreign key) en el motor de base de datos.</td><td rowspan="2">Ambiente de formación

Aula de informática dotada con computadores, conexión a internet, videobeam, tablero, salida de emergencia, extintor.</td><td colspan="2">MATERIALES DE FORMACIÓN</td></tr>
<tr><td>DEVOLUTIVO

Computadores, videobeam, tablero.</td><td>CONSUMIBLE

Marcadores, hojas tamaño carta</td></tr>
<tr><td>Resultados de Aprendizaje:

22050100701
Construir las tablas que hacen parte del diseño del diagrama relacional en el motor de base de datos empleando las cuatro formas de normalización</td><td colspan="4">Competencia:

220501007
Desarrollar el sistema que cumpla con los requerimientos de la solución informática</td></tr>
<tr><td>22050100702
Relacionar las tablas construidas para presentar la información solicitada en el diseño.</td><td colspan="4">220501007
Desarrollar el sistema que cumpla con los requerimientos de la solución informática</td></tr>
<tr><td>22050100704
Construir la matriz CRUD en el lenguaje de programación seleccionado para verificar la funcionalidad del sistema de acuerdo con el diseño entregado.</td><td colspan="4">220501007
Desarrollar el sistema que cumpla con los requerimientos de la solución informática</td></tr>
<tr><td>22050103204
Interpreta el diagrama relacional para identificar el modelo de datos.</td><td colspan="4">220501007
Desarrollar el sistema que cumpla con los requerimientos de la solución informática</td></tr>
<tr><td>Duración de la guía (en horas):</td><td colspan="4">Presenciales: 20</td></tr>
</table>

2. INTRODUCCIÓN

Un procedimiento almacenado (stored procedure en inglés) es un programa (o procedimiento) almacenado físicamente en una base de datos. Su implementación varía de un gestor de bases de datos a otro. La ventaja de un procedimiento almacenado es que al ser ejecutado, en respuesta a una petición de usuario, es ejecutado directamente en el motor de bases de datos, el cual usualmente corre en un servidor separado. Como tal, posee acceso directo a los datos que necesita manipular y sólo necesita enviar sus resultados de regreso al usuario, deshaciéndose de la sobrecarga resultante de comunicar grandes cantidades de datos salientes y entrantes.

Usos típicos para procedimientos almacenados incluyen la validación de datos siendo integrados a la estructura de base de datos (los procedimientos almacenados utilizados para este propósito a menudo son llamados disparadores; triggers en inglés), o encapsular un proceso grande y complejo. El último ejemplo generalmente ejecutará más rápido como un procedimiento almacenado que de haber sido implementado como, por ejemplo, un programa corriendo en el sistema cliente y comunicándose con la base de datos mediante el envío de consultas SQL y recibiendo sus resultados.

Ventajas

Cuando una base de datos es manipulada desde muchos programas externos. Al incluir la lógica de la aplicación en la base de datos utilizando procedimientos almacenados, la necesidad de embeber la misma lógica en todos los programas que acceden a los datos es reducida. Esto puede simplificar la creación y, particularmente, el mantenimiento de los programas involucrados.

Los procedimientos almacenados pueden recibir y devolver información; para ello se emplean parámetros, de entrada y salida, respectivamente.

Veamos los primeros. Los parámetros de entrada posibilitan pasar información a un procedimiento.
Para que un procedimiento almacenado admita parámetros de entrada se deben declarar variables como parámetros al crearlo. La sintaxis es:

```
create proc NOMBREPROCEDIMIENTO
  @NOMBREPARAMETRO TIPO =VALORPORDEFECTO
  as SENTENCIAS;
```

Los parámetros se definen luego del nombre del procedimiento, comenzando el nombre con un signo arroba (@). Los parámetros son locales al procedimiento, es decir, existen solamente dentro del mismo. Pueden declararse varios parámetros por procedimiento, se separan por comas.

Cuando el procedimiento es ejecutado, deben explicitarse valores para cada uno de los parámetros (en el orden que fueron definidos), a menos que se haya definido un valor por defecto, en tal caso, pueden omitirse. Pueden ser de cualquier tipo de dato (excepto cursor).

Luego de definir un parámetro y su tipo, opcionalmente, se puede especificar un valor por defecto; tal valor es el que asume el procedimiento al ser ejecutado si no recibe parámetros. Si no se coloca valor por defecto, un procedimiento definido con parámetros no puede ejecutarse sin valores para ellos. El valor por defecto puede ser "null" o una constante, también puede incluir comodines si el procedimiento emplea "like".

Creamos un procedimiento que recibe el nombre de un autor como parámetro para mostrar todos los libros del autor solicitado:

```
create procedure pa_libros_autor
 @autor varchar(30)
as
 select titulo, editorial,precio
  from libros
  where autor= @autor;
```

El procedimiento se ejecuta colocando "execute" (o "exec") seguido del nombre del procedimiento y un valor para el parámetro:

```
exec pa_libros_autor 'Borges';
```

Los valores de un parámetro pueden pasarse al procedimiento mediante el nombre del parámetro o por su posición. La sintaxis anterior ejecuta el procedimiento pasando valores a los parámetros por posición. También podemos emplear la otra sintaxis en la cual pasamos valores a los parámetros por su nombre:

```
exec pa_libros_autor_editorial @editorial='Planeta', @autor='Richard Bach';
```

Cuando pasamos valores con el nombre del parámetro, el orden en que se colocan puede alterarse.

No podríamos ejecutar el procedimiento anterior sin valores para los parámetros. Si queremos ejecutar un procedimiento que permita omitir los valores para los parámetros debemos, al crear el procedimiento, definir valores por defecto para cada parámetro:

```
create procedure pa_libros_autor_editorial2
 @autor varchar(30)='Richard Bach',
 @editorial varchar(20)='Planeta'
as
 select titulo, autor,editorial,precio
  from libros
  where autor= @autor and
  editorial=@editorial;
```

Podemos ejecutar el procedimiento anterior sin enviarle valores, usará los predeterminados.

Si enviamos un solo parámetro a un procedimiento que tiene definido más de un parámetro sin especificar a qué parámetro corresponde (valor por posición), asume que es el primero. Es decir, SQL Server asume que los valores se dan en el orden que fueron definidos, no se puede interrumpir la secuencia.

Si queremos especificar solamente el segundo parámetro, debemos emplear la sintaxis de paso de valores a parámetros por nombre:

```
exec pa_libros_autor_editorial2 @editorial='Paidos';
```

Podemos emplear patrones de búsqueda en la consulta que define el procedimiento almacenado y utilizar comodines como valores por defecto:

```
create proc pa_libros_autor_editorial3
 @autor varchar(30) = '%',
 @editorial varchar(30) = '%'
as
 select titulo,autor,editorial,precio
  from libros
  where autor like @autor and
  editorial like @editorial;
```

La sentencia siguiente ejecuta el procedimiento almacenado "pa_libros_autor_editorial3" enviando un valor por posición, se asume que es el primero.

```
exec pa_libros_autor_editorial3 'P%';
```

La sentencia siguiente ejecuta el procedimiento almacenado "pa_libros_autor_editorial3" enviando un valor para el segundo parámetro, para el primer parámetro toma el valor por defecto:

```
exec pa_libros_autor_editorial3 @editorial='P%';
```

También podríamos haber tipeado:

```
exec pa_libros_autor_editorial3 default, 'P%';
```

Ejemplos:

Creamos un procedimiento que recibe 2 parámetros, el nombre de un autor y el de una editorial:

```
create procedure pa_libros_autor_editorial
 @autor varchar(30),
 @editorial varchar(20)
as
 select titulo, precio
  from libros
  where autor= @autor and
  editorial=@editorial;
```

El procedimiento se ejecuta colocando "execute" (o "exec") seguido del nombre del procedimiento y los valores para los parámetros separados por comas:

```
exec pa_libros_autor_editorial 'Richard Bach','Planeta';
```

3. ESTRUCTURACION DIDACTICA DE LAS ACTIVIDADES DE APRENDIZAJE

3.1 Actividades de Reflexión inicial.

Existen operaciones que se deben ejecutar múltiples veces (insert, select, update) dentro de una aplicación. ¿Cree usted que con la implementación de procedimientos almacenados es más fácil repetir estos procesos?

3.2 Actividades de contextualización e identificación de conocimientos necesarios para el aprendizaje

Tarea de Aprendizaje

1. Qué es un procedimiento almacenado?
2. Cuál es el objetivo de los procedimientos almacenados?
3. En qué casos se pueden usar los procedimientos almacenados?
4. Definir y crear los procedimientos almacenados para su proyecto formativo

3.3 Actividades de apropiación del conocimiento (Conceptualización y Teorización).

1. Adjunte su base de datos en SQL server.
2. Utilice el editor de consultas de SQL Server para implementar los ejemplos propuestos en la guía.

3.4 Actividades de transferencia del conocimiento.

Actividad 1: Realice un listado de procedimientos almacenados para su proyecto y diséñelos.

Actividad 2: Implemente los procedimientos almacenados para su proyecto.

CONCLUSIONES

Un procedimiento almacenado es un grupo de una o varias instrucciones Transact-SQL o una referencia a un método de Common Runtime Language (CLR) de Microsoft .NET Framework. Los procedimientos se asemejan a las construcciones de otros lenguajes de programación, porque pueden:

- Aceptar parámetros de entrada y devolver varios valores en forma de parámetros de salida al programa que realiza la llamada.
- Contener instrucciones de programación que realicen operaciones en la base de datos. Entre otras, pueden Contener llamadas a otros procedimientos.
- Devolver un valor de estado a un programa que realiza una llamada para indicar si la operación se ha realizado correctamente o se han producido errores, y el motivo de estos.

En la siguiente lista se describen algunas de las ventajas que brinda el uso de procedimientos.
Tráfico de red reducido entre el cliente y el servidor

Los comandos de un procedimiento se ejecutan en un único lote de código. Esto puede reducir significativamente el tráfico de red entre el servidor y el cliente porque únicamente se envía a través de la red la llamada que va a ejecutar el procedimiento. Sin la encapsulación de código que proporciona un procedimiento, cada una de las líneas de código tendría que enviarse a través de la red.

Mayor seguridad

Varios usuarios y programas cliente pueden realizar operaciones en los objetos de base de datos

subyacentes a través de un procedimiento, aunque los usuarios y los programas no tengan permisos directos sobre esos objetos subyacentes. El procedimiento controla qué procesos y actividades se llevan a cabo y protege los objetos de base de datos subyacentes. Esto elimina la necesidad de conceder permisos en cada nivel de objetos y simplifica los niveles de seguridad.

La cláusula EXECUTE AS puede especificarse en la instrucción CREATE PROCEDURE para habilitar la suplantación de otro usuario o para permitir que los usuarios o las aplicaciones puedan realizar ciertas actividades en la base de datos sin necesidad de contar con permisos directos sobre los objetos y comandos subyacentes. Por ejemplo, algunas acciones como TRUNCATE TABLE no tienen permisos que se puedan conceder. Para poder ejecutar TRUNCATE TABLE, el usuario debe tener permisos ALTER en la tabla especificada. Puede que la concesión de permisos ALTER a un usuario en una tabla no sea lo ideal, pues en realidad el usuario tendrá permisos muy superiores a la posibilidad de truncar una tabla. Si se incorpora la instrucción TRUNCATE TABLE en un módulo y se especifica la ejecución del módulo como un usuario con permisos para modificar la tabla, se pueden ampliar los permisos para truncar la tabla al usuario al que se concedan permisos EXECUTE para el módulo.

Al llamar a un procedimiento a través de la red, solo está visible la llamada que va a ejecutar el procedimiento. Por lo tanto, los usuarios malintencionados no pueden ver los nombres de los objetos de base de datos y tabla, incrustados en sus propias instrucciones Transact-SQL, ni buscar datos críticos.

El uso de parámetros de procedimientos ayuda a protegerse contra ataques por inyección de código SQL. Dado que la entrada de parámetros se trata como un valor literal y no como código ejecutable, resulta más difícil para un atacante insertar un comando en la instrucción Transact-SQL del procedimiento y comprometer la seguridad.

Los procedimientos pueden cifrarse, lo que ayuda a ofuscar el código fuente. Para obtener más información, vea Cifrado de SQL Server.

Reutilización del código

El código de cualquier operación de base de datos redundante resulta un candidato perfecto para la encapsulación de procedimientos. De este modo, se elimina la necesidad de escribir de nuevo el mismo código, se reducen las inconsistencias de código y se permite que cualquier usuario o aplicación que cuente con los permisos necesarios pueda acceder al código y ejecutarlo.

Mantenimiento más sencillo

Cuando las aplicaciones cliente llaman a procedimientos y mantienen las operaciones de base de datos en la capa de datos, solo deben actualizarse los cambios de los procesos en la base de datos subyacente.El nivel de aplicación permanece independiente y no tiene que tener conocimiento sobre los cambios realizados en los diseños, las relaciones o los procesos de la base de datos.

Rendimiento mejorado

De forma predeterminada, un procedimiento se compila la primera vez que se ejecuta y crea un plan de ejecución que vuelve a usarse en posteriores ejecuciones. Como el procesador de consultas no tiene que crear un nuevo plan, normalmente necesita menos tiempo para procesar el procedimiento. Si ha habido cambios importantes en las tablas o datos a los que se hace referencia en el procedimiento, el plan precompilado podría hacer que el procedimiento se ejecutara con mayor lentitud. En este caso, volver a crear el procedimiento y forzar un nuevo plan de ejecución puede mejorar el rendimiento.

3.5 Actividades de evaluación.

Evidencias de Aprendizaje	Criterios de Evaluación	Técnicas e Instrumentos de Evaluación
Evidencias de Conocimiento : ✓ Investigación **Evidencias de Desempeño:** ✓ Implementación de los triggers. **Evidencias de Producto:** ✓ Implementación de triggers propuestos para su proyecto.	• Plantea la necesidad de utilización de los procedimientos almacenados según los requerimientos del cliente. • Diseña las expresiones para implementar los triggers. • Genera ideas para automatizar procesos desde base de datos.	Socialización de la investigación. Inspección resultados de la inyección de código. Juego de Roles – Verificación ejecución y verificación de triggers. Script creación de triggers. Lista de chequeo.

4. RECURSOS PARA EL APRENDIZAJE

ACTIVIDADES DEL PROYECTO	DURACIÓN (Horas)	Materiales de formación devolutivos: (Equipos/Herramientas)		Materiales de formación (consumibles)		Talento Humano (Instructores)		AMBIENTES DE APRENDIZAJE TIPIFICADOS
		Descripción	Cantidad	Descripción	Cantidad	Especialidad	Cantidad	ESCENARIO (Aula, Laboratorio, taller, unidad productiva) y elementos y condiciones de seguridad industrial, salud ocupacional y medio ambiente
Diseñar consultas.(Create,Read, Update, Delete) para ser ejecutadas desde la interfaz.	15	Computadores, conexión a internet, Videobeam, tablero. Plataforma Blackboard	Según cantidad de aprendices	Marcadores	2	Ing. De Sistemas o Tecnólogo en Desarrollo de sistemas de información	1	Aula de informática dotada con salida de emergencia, extintor.

También cuenta con el material de apoyo documentos :

- ✓ Triggers en Transact SQL. docx

5. BIBLIOGRAFÍA/ WEBGRAFÍA

- ✓ Bases de Datos: Enfoque práctico
  McGrawHill KORTH, F. Henry y SILBERCHATZ, Abraham

- ✓ FUNDAMENTO DE BASES DE DATOS
  Edit. Mc Graw Hill TechNet Microsoft

- ✓ MSDN SQL Server CREATE TRIGGER (Transact-SQL)
  http://msdn.microsoft.com/es-es/library/ms189799.aspx

- ✓ MyGNET Triggers
  http://mygnet.net/articulos/sql/triggers.774

- ✓ Blog Edison García Tecnologías Microsoft Sql Server
  http://mredison.wordpress.com/2008/10/26/sql-server-qu-es-un-procedimientoalmacenado/

- ✓ Blog dedicado a SQL Server Maximiliano Damián Accotto
  http://blog.maxiaccotto.com/category/TSQL.aspx

También cuenta con el material de apoyo documentos :

- ✓ Triggers en Transact SQL. Docx

6. CONTROL DEL DOCUMENTO (ELABORADA POR)

OSCAR JAVIER ORTEGON REYES	Instructora Ingeniero de Sistemas	Integración con la media técnica	Marzo 15 de 2015	Distrito Capital