



**MÓDULO IV:**

**DESARROLLA SOFTWARE DE APLICACIÓN WEB CON ALMACENAMIENTO PERSISTENTE DE DATOS.**

**SUBMÓDULO 1:**

**CONSTRUYE BASES DE DATOS PARA APLICACIONES WEB.**

Unidad 3 Administra la información de la base de datos

# DESARROLLA SOFTWARE DE APLICACIÓN WEB CON

# ALMACENAMIENTO PERSISTENTE DE DATOS.

Actividad 2-Unidad 3

**Programación**

Elaborado por: Abraham Castañeda Quintero

Docente: Alejandra Serrano Luna

Semestre y grupo: 5° “B”

CBTis#42

# Índice.

Contenido

[DESARROLLA SOFTWARE DE APLICACIÓN WEB CON 0](file:///C:\Users\Apple\Music\S1U3A1_Abraham_Castañeda_Quintero.docx#_Toc57391778)

[ALMACENAMIENTO PERSISTENTE DE DATOS. 0](file:///C:\Users\Apple\Music\S1U3A1_Abraham_Castañeda_Quintero.docx#_Toc57391779)

[Índice. 1](#_Toc57391780)

[Introducción. 2](#_Toc57391781)

[Desarrollo. 3](#_Toc57391782)

[1. ¿Qué es un procedimiento almacenado en bases de datos? 3](#_Toc57391783)

[1.1. Características de los procedimientos almacenados 4](#_Toc57391784)

[1.2. Ventajas y desventajas de los procedimientos almacenados 4](#_Toc57391785)

[1.3. Elementos de los procedimientos almacenados 5](#_Toc57391786)

[**Elementos de los procedimientos almacenados** 5](#_Toc57391787)

[**Sintaxis** 6](#_Toc57391788)

[2. ¿Qué es una función en MySql? 7](#_Toc57391789)

[3. ¿Qué es un trigger en bases de datos? 8](#_Toc57391790)

[3.1. ¿Cuáles son las características de los triggers? 8](#_Toc57391791)

[3.2. Menciona los elementos de la estructura de un trigger 9](#_Toc57391792)

[4. ¿Qué es ODBC en bases de datos y para que se utiliza? 10](#_Toc57391793)

[5. ¿Qué es ADO.NET en bases de datos y para que se utiliza? 11](#_Toc57391794)

[6. ¿Qué es JDBC en bases de datos y para que se utiliza? 11](#_Toc57391795)

[Conclusión. 13](#_Toc57391796)

[Referencias. 14](#_Toc57391797)

# Introducción.

En esta la actividad No. 2 unidad 3 del submódulo 1 de la carrera de programación, se investigará sobre como se almacenan los datos en una base de datos además sobre las funciones y los triggers en MySQL mas aparte conceptos como ODBC, ADO.NET y JDBC con la intención de lograr gestionarla de una manera más fácil, rápida y eficiente.

# Desarrollo.

## ¿Qué es un procedimiento almacenado en bases de datos?

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.

Los procedimientos pueden ser ventajosos: 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.

Podemos ver un claro ejemplo de estos procedimientos cuando requerimos realizar una misma operación en un servidor dentro de algunas o todas las bases de datos y a la vez dentro de todas o algunas de las tablas de las bases de datos del mismo. Para ello podemos utilizar a los Procedimientos almacenados auto creables que es una forma de generar ciclos redundantes a través de los procedimientos almacenados.

Un procedimiento almacenado es un conjunto de instrucciones de T-SQL que SQL Server compila, en un único plan de ejecución, los llamados "store procedures" se encuentran almacenados en la base de datos, los cuales pueden ser ejecutados en cualquier momento.

Se pueden emplear cuatro métodos para crear procedimientos almacenados, utilizando:

El Transact-SQL (T-SQL).

El Administrador Corporativo de SQL Server.

El asistente para La Creación de Procedimientos Almacenados (Create Store Procedure Wizard).

Los objetos de Administración distribuida de SQL (SQL Distributed Management Objects, SQL-DMO).

¿Para que se utilizan?

Los procedimientos almacenados se utilizan para agrupar las instrucciones de T-SQL y cualquier lógica asociada necesaria para llevar a cabo una tarea.

¿Cómo se ejecutan?

Cuando un procedimiento almacenado es ejecutado por primera vez se compila, se crea y se guarda en memoria su plan de ejecución., luego SQL utiliza ese plan de ejecución cuando se vuelve a llamar al procedimiento sin volver a compilarlo nuevamente.

¿Qué nos proporcionan?

Nos proporcionan a nosotros los usuarios un acceso fácil a la base de datos, se puede tener acceso a la base de datos sin tener que conocer los detalles de la arquitectura de tablas -simplemente se ejecutan los procedimientos almacenados que llevan a cabo las tareas solicitadas.

Los procedimientos almacenados pueden aceptar datos de entrada, utilizar variables locales y devolver datos.

### Características de los procedimientos almacenados

Los procedimientos almacenados MySQL se han diseñado para aligerar a las aplicaciones clientes, pudiendo ejecutar directamente en el servidos aquellas tareas pesadas y que necesitan muchos recursos.

Estas son las características principales de los procedimientos almacenados MySQL:

Pueden recibir y devolver parámetros.

Pueden manejar tablas, ejecutando operaciones e iteraciones de lectura/escritura.

Pueden devolver una tabla como resultado.

Se almacenan en la base de datos en la cual se crean.

No dependen de ninguna tabla en particular.

Pueden aceptar recursividad.

### Ventajas y desventajas de los procedimientos almacenados

Entre las principales ventajas de [usar los procedimientos almacenados](https://blog.mdcloud.es/sistemas-de-almacenamiento-como-elegir-el-mejor-para-mi-empresa/) MySQL, destacan:

* **Compatibilidad**: los procedimientos almacenados MySQL permiten ejecutar la misma operación sobre la base de datos también en el caso de que las aplicaciones cliente estén implementadas en distintas plataformas y programadas en distintos lenguajes.
* **Integridad**: gracias a los procedimientos almacenados MySQL es posible centralizar el acceso a la información.
* **Seguridad**: cuando necesitamos evitar el acceso directo a la base de datos, los procedimientos almacenados MySQL permiten establecer un entorno seguro, otorgando permisos y privilegios para la ejecución.
* **Rendimiento**: con los procedimientos almacenados MySQL todo el trabajo se ejecuta en el servidor, por lo cual se minimiza de forma importante la cantidad de información intercambiada con las aplicaciones cliente y se reduce el tráfico de acceso a la base de datos además del numero de accesos.
* **Centralización:**los procedimientos almacenados MySQL permiten centralizar la lógica comercial ofreciendo a todas las aplicaciones clientes la misma versión actualizada, lo que hace el mantenimiento más sencillo.
* **Reutilización del código**: al escribir un mismo código que se ejecuta por toda las aplicaciones, es posible reducir las inconsistencias.
* **Sencillez**: los procedimientos almacenados permiten la creación de bibliotecas o funciones, lo que facilita mucho el trabajo del programador ya que se trata de características compartidas por los principales lenguajes de programación modernos.

Entre las principales desventajas de los procedimientos almacenados MySQL se encuentran:

* **Saturación**: si creamos demasiados procedimientos almacenados MySQL, podríamos saturar el servidor.
* **Seguridad**: si están mal programados, los procedimientos almacenados MySQL puede suponer un riesgo para la seguridad y convertirse en un objetivo para ataques cibernéticos.
* **Complejidad**: conviene no utilizar los procedimientos almacenados MySQL para implementar la lógica de negocio, sino limitarse a las operaciones sobre datos que pueden ayudarnos a mejorar el rendimiento

Hemos visto como gracias a los procedimientos almacenados MySQL podemos aumentar el rendimiento de nuestras aplicaciones, trabajar con grandes conjuntos de datos y [Big Data](https://blog.mdcloud.es/que-es-big-data-y-para-que-sirve/) y, sobre todo, hacer frente al [gran reto de la ciberseguridad](https://blog.mdcloud.es/por-que-un-ataque-cibernetico-puede-hacer-desaparecer-tu-empresa/), evitando ataques que puedan comprometer nuestros datos.

### Elementos de los procedimientos almacenados

## **Elementos de los procedimientos almacenados**

Los procedimientos almacenados están compuestos por algunos de estos elementos:

* Parámetros de entrada (pueden esperar parámetros)
* Parámetros de salida (pueden devolver resultados)
* Declaración de variables (puede usarse variables en su cuerpo)
* Cuerpo del procedimiento (en su cuerpo se indican las acciones a realizar)

Tanto los parámetros de entrada como los de salida son opcionales. Podemos tener un procedimiento que no tenga parámetros de entrada, pero sí de salida y viceversa. Como todo buen [programador](https://www.ecured.cu/Programador) sabe, los parámetros de entrada sirven para pasarle datos al procedimiento que necesita para llevar a cabo una acción determinada. Los parámetros de salida devuelven al programa cliente que ejecutó el procedimiento el resultado del mismo.

## **Sintaxis**

Para crear un procedimiento almacenado debemos emplear la sentencia CREATE PROCEDURE.

**CREATE** **PROCEDURE** <nombre\_procedure> [@param1 <tipo>, ...]

**AS**

*-- Sentencias del procedure*

Para modificar un procedimiento almacenado debemos emplear la sentencia ALTER PROCEDURE.

**ALTER** **PROCEDURE**<nombre\_procedure> [@param1 <tipo>, ...]

**AS**

*-- Sentencias del procedure*

El siguiente ejemplo muestra un procedimiento almacenado, llamado spu\_addCliente que añade clientes, o sea, inserta un registro en la tabla ‘Clientes’:

**CREATE** **PROCEDURE** spu\_addCliente @nombre varchar(100),

@apellidos varchar(100),

@f\_Naciento datetime,

@email varchar(50)

**AS**

NSERT **INTO** CLIENTES (nombre, apellidos, f\_nacimiento, email)

**VALUES** (@nombre, @apellidos, @f\_naciento, @email)

## ¿Qué es una función en MySql?

MySQL puede hacer mucho más que simplemente almacenar y recuperar datos . También podemos realizar manipulaciones en los datos antes de recuperarlos o guardarlos. Ahí es donde entran las Funciones de MySQL. Las funciones son simplemente piezas de código que realizan algunas operaciones y luego devuelven un resultado. Algunas funciones aceptan parámetros mientras que otras funciones no aceptan parámetros.

Veamos brevemente un ejemplo de función MySQL. Por defecto, MySQL guarda los tipos de datos de fecha en el formato “AAAA-MM-DD”. Supongamos que hemos creado una aplicación y nuestros usuarios desean que la fecha se devuelva en el formato “DD-MM-YYYY”, podemos usar la función incorporada MySQL DATE\_FORMAT para lograr esto. DATE\_FORMAT es una de las funciones más utilizadas en MySQL. Lo veremos con más detalles a medida que desarrollemos la lección.

Una función en MySQL es una rutina creada para tomar unos parámetros, procesarlos y retornar

en un salida.

Se diferencian de los procedimientos en las siguientes características:

 Solamente pueden tener parámetros de entrada IN y no parámetros de salida OUT o

INOUT

 Deben retornar en un valor con algún tipo de dato definido

 Pueden usarse en el contexto de una sentencia SQL

 Solo retornan un valor individual, no un conjunto de registros

CREATE FUNCTION factorial(x INT) RETURNS INT(11)

BEGIN

DECLARE factorial INT;

-- Guardamos el valor de x

SET factorial = x ;

-- Caso en que x sea menor o igual a 0

IF x <= 0 THEN

RETURN 1;

END IF;

-- Iteramos para obtener multiplicaciones

consecutivas

bucle: LOOP

SET x = x - 1 ;

## ¿Qué es un trigger en bases de datos?

Un trigger o disparador es un objeto que se asocia con tablas y se almacena en la base de datos. Su nombre se deriva por el comportamiento que presentan en su funcionamiento, ya que se ejecutan cuando sucede algún evento sobre las tablas a las que se encuentra asociado. Los eventos que hacen que se ejecute un trigger son las operaciones de inserción (INSERT), borrado (DELETE) o actualización (UPDATE), ya que modifican los datos de una tabla.

La utilidad principal de un trigger es mejorar la gestión de la base de datos, ya que no requieren que un usuario los ejecute. Por lo tanto, son empleados para implementar las REGLAS DE NEGOCIO (tipo especial de integridad) de una base de datos. Una Regla de Negocio es cualquier restricción, requerimiento, necesidad o actividad especial que debe ser verificada al momento de intentar agregar, borrar o actualizar la información de una base de datos. Los triggers pueden prevenir errores en los datos, modificar valores de una vista, sincronizar tablas, entre otros.

La acción del trigger,siempre que no se viole la restricción del trigger se ejecuta dependiendo de la combinación de tipos de trigger:

- Before statement: Antes de ejecutar la sentencia de disparo.

-Before row: Antes de modificar cada fila afectada por la sentencia de disparo,y antes de chequear las restricciones de integridad apropiadas .

-After statement: Después de ejecutar la sentencia de disparo,y después de chequear las restricciones de integridad apropiadas.

-After row: Después de modificar cada fila afectada por la sentencia de disparo y posiblemente aplicando las restricciones de integridad apropiadas .

### ¿Cuáles son las características de los triggers?

No aceptan parámetros o argumentos (pero podrían almacenar los datos afectados en tablas temporales)

No pueden ejecutar las operaciones COMMIT o ROLLBACK porque estas son parte de la sentencia SQL del disparador (únicamente a través de transacciones autónomas)

Pueden causar errores de mutaciones en las tablas, si se han escrito de manera deficiente.

Un sencillo ejemplo (para SQL Server) sería crear un Trigger para insertar un pedido de algún producto cuando la cantidad de éste, en nuestro almacén, sea inferior a un valor dado.

**CREATE** **TRIGGER** TR\_ARTICULO

**ON** ARTICULOS

**AFTER** **UPDATE**

**AS**

**BEGIN**

**INSERT** **INTO** HCO\_ARTICULO

(IDARTICULO, STOCK, FECHA)

**SELECT** ID\_ARTICULO, STOCK, GETDATE()

**FROM** INSERTED

**END**

**INSERT** **INTO** ARTICULOS **VALUES** (1, 'MEMORIA', 12, '12/03/2014')

**SELECT** \* **FROM** ARTICULOS

**UPDATE** ARTICULOS

**SET** STOCK = STOCK - 20

**WHERE** ID\_ARTICULO = 1

**SELECT** \* **FROM** HCO\_ARTICULO

### Menciona los elementos de la estructura de un trigger

Estructura básica de un "trigger" (disparador):

Llamada de activación: es la sentencia que permite "disparar" el código a ejecutar.

Restricción: es la condición necesaria para realizar el código. Esta restricción puede ser de tipo condicional o de tipo nulidad.

Acción a ejecutar: es la secuencia de instrucciones a ejecutar una vez que se han cumplido las condiciones iniciales.

## ¿Qué es ODBC en bases de datos y para que se utiliza?

Open DataBase Connectivity (ODBC) es un estándar de acceso a las bases de datos desarrollado por SQL Access Group (SAG) en 1992. El objetivo de ODBC es hacer posible el acceder a cualquier dato desde cualquier aplicación, sin importar qué sistema de gestión de bases de datos (DBMS) almacene los datos. ODBC logra esto al insertar una capa intermedia denominada nivel de Interfaz de Cliente SQL (CLI), entre la aplicación y el DBMS. El propósito de esta capa es traducir las consultas de datos de la aplicación en comandos que el DBMS entienda. Para que esto funcione tanto la aplicación como el DBMS deben ser compatibles con ODBC, esto es que la aplicación debe ser capaz de producir comandos ODBC y el DBMS debe ser capaz de responder a ellos. Desde la versión 2.0 el estándar soporta SAG (SQL Access Group) y SQL.

El software funciona de dos modos, con un software manejador en el cliente, o una filosofía cliente-servidor. En el primer modo, el driver interpreta las conexiones y llamadas SQL y las traduce desde el API ODBC hacia el DBMS. En el segundo modo para conectarse a la base de datos se crea una DSN dentro del ODBC que define los parámetros, ruta y características de la conexión según los datos que solicite el creador o fabricante.

Puede utilizar ODBC Server como:

•Una fuente de datos. Puede compartir archivos de base de datos de FileMaker Pro Advanced alojados con otras aplicaciones en el mismo equipo o a través de una red mediante ODBC (Open Database Connectivity) y JDBC (Java Database Connectivity). Por ejemplo, utilice los datos de FileMaker Pro Advanced en otras aplicaciones para crear gráficos, analizar números y generar informes.

•Una aplicación cliente ODBC. Las bases de datos alojadas de FileMaker Pro Advanced pueden interactuar con fuentes de datos en el mismo equipo o a través de una red. Por ejemplo, las bases de datos alojadas de FileMaker Pro Advanced pueden trabajar de forma interactiva con una fuente de datos Oracle, como se define en el gráfico de relaciones de FileMaker Pro Advanced y en las presentaciones.

## ¿Qué es ADO.NET en bases de datos y para que se utiliza?

ADO.NET es un conjunto de componentes del software que pueden ser usados por los programadores para acceder a datos y a servicios de datos. Es parte de la biblioteca de clases base que están incluidas en el Microsoft .NET Framework. Es comúnmente usado por los programadores para acceder y para modificar los datos almacenados en un Sistema Gestor de Bases de Datos Relacionales, aunque también puede ser usado para acceder a datos en fuentes no relacionales. ADO.NET es a veces considerado como una evolución de la tecnología ActiveX Data Objects (ADO), pero fue cambiado tan extensivamente que puede ser concebido como un producto enteramente nuevo.

Su uso principalmente es para combinar las capas ADO y OLE DB en una única capa de proveedores (managed providers). Cada proveedor contiene un conjunto de clases que implementan interfaces comunes para permitir el acceso uniforme a distintas fuentes de datos. Ejemplos: ADO Managed Provider (da acceso a cualquier fuente de datos OLE DB), SQL Server Managed Provider (específico para el DBMS de Microsoft), Exchange Managed Provider (datos almacenados con Microsoft Exchange)

## ¿Qué es JDBC en bases de datos y para que se utiliza?

Java Database Connectivity (en español: Conectividad a bases de datos de Java), más conocida por sus siglas JDBC,1​2​ es una API que permite la ejecución de operaciones sobre bases de datos desde el lenguaje de programación Java, independientemente del sistema operativo donde se ejecute o de la base de datos a la cual se accede, utilizando el dialecto SQL del modelo de base de datos que se utilice.

El API JDBC se presenta como una colección de interfaces Java y métodos de gestión de manejadores de conexión hacia cada modelo específico de base de datos. Un manejador de conexiones hacia un modelo de base de datos en particular es un conjunto de clases que implementan las interfaces Java y que utilizan los métodos de registro para declarar los tipos de localizadores a base de datos (URL) que pueden manejar. Para utilizar una base de datos particular, el usuario ejecuta su programa junto con la biblioteca de conexión apropiada al modelo de su base de datos, y accede a ella estableciendo una conexión; para ello provee el localizador a la base de datos y los parámetros de conexión específicos. A partir de allí puede realizar cualquier tipo de tarea con la base de datos a la que tenga permiso: consulta, actualización, creación, modificación y borrado de tablas, ejecución de procedimientos almacenados en la base de datos, etc.

Su utilidad es la API JDBC define clases Java para representar conexiones a bases de datos, sentencias SQL, conjuntos de resultados, metadatos de bases de datos, etc. Además permite al programador escribir en el lenguaje de programación Java y que resultará en sentencias SQL y procesar los resultados. JDBC es la API primaria para acceso a bases de datos en el lenguaje de programación Java.

# Conclusión.

En conclusión, en esta actividad aprendí sobre la diferencia entre procedimientos, funciones, triggers y como se usan en las bases de datos como MySQL mas aparte los conceptos de ODBC, ADO.NET Y JDBC y como es que se utilizan es necesario aprenderlo por que puede que estos conocimientos los necesitemos mas adelante.

# Referencias.

Carisio, E. (2019, 21 junio). *Procedimientos almacenados MySQL: qué son, cómo crearlos y ventajas*. #ADNCLOUD. <https://blog.mdcloud.es/procedimientos-almacenados-mysql-que-son-como-crearlos-y-ventajas/>

colaboradores de Wikipedia. (2019a, septiembre 18). *Procedimiento almacenado*. Wikipedia, la enciclopedia libre. <https://es.wikipedia.org/wiki/Procedimiento_almacenado>

colaboradores de Wikipedia. (2019b, noviembre 13). *ADO.NET*. Wikipedia, la enciclopedia libre. <https://es.wikipedia.org/wiki/ADO.NET>

colaboradores de Wikipedia. (2020a, mayo 6). *Java Database Connectivity*. Wikipedia, la enciclopedia libre. <https://es.wikipedia.org/wiki/Java_Database_Connectivity#:%7E:text=Java%20Database%20Connectivity%20(en%20espa%C3%B1ol,datos%20a%20la%20cual%20se>

colaboradores de Wikipedia. (2020b, septiembre 29). *Trigger (base de datos)*. Wikipedia, la enciclopedia libre. <https://es.wikipedia.org/wiki/Trigger_(base_de_datos)>

EcuRed. (s. f.). *Procedimientos almacenados - EcuRed*. Recuperado 24 de noviembre de 2020, de https://www.ecured.cu/Procedimientos\_almacenados

*https://guru99.es/*. (s. f.). Guru99. Recuperado 24 de noviembre de 2020, de <https://guru99.es/functions/>

*Procedimientos Almacenados*. (s. f.). Procedimientos almacenados. Recuperado 24 de noviembre de 2020, de <https://www.usmp.edu.pe/publicaciones/boletin/fia/info41/procedimiento.html#:%7E:text=Un%20procedimiento%20almacenado%20es%20un,ser%20ejecutados%20en%20cualquier%20momento>.