PROCESO DE GESTIÓN DE FORMACIÓN PROFESIONAL INTEGRAL

FORMATO GUÍA DE APRENDIZAJE

1. IDENTIFICACIÓN DE LA GUIA DE APRENDIZAJE

- Denominación del Programa de Formación: Tecnólogo en Análisis y Desarrollo de Software
- Código del Programa de Formación: 2560979
- Nombre del Proyecto (si es formación Titulada): Construcción de Software Orientado a Diferentes Sectores Productivos del Área Metropolitana de Cúcuta, Norte de Santander
- Fase del Proyecto (si es formación Titulada): CONSTRUCCIÓN
- Actividad de Proyecto(si es formación Titulada)
- Competencia: 220501096 - DESARROLLAR LA SOLUCIÓN DE SOFTWARE DE ACUERDO CON EL DISEÑO Y METODOLOGÍAS DE DESARROLLO.
- Resultados de Aprendizaje Alcanzar:
  CONSTRUIR LA BASE DE DATOS PARA EL SOFTWARE A PARTIR DEL MODELO DE DATOS.
- Duración de la Guía

2. PRESENTACIÓN

- Esta guía de aprendizaje se presenta como complemento al taller realizado en la última formación: Creación de una base de datos.

3. FORMULACIÓN DE LAS ACTIVIDADES DE APRENDIZAJE

- CREA LA BASE DE DATOS INTEGRANDO LOS OBJETOS DE ACUERDO CON LA FUNCIONALIDAD DEL SOFTWARE.
- IMPLEMENTA RESTRICCIONES EN LA BASE DE DATOS DE ACUERDO CON LAS REGLAS DE DISEÑO.
- DEFINE ESQUEMAS DE SEGURIDAD EN LA BASE DE DATOS PARA MANTENER LA INTEGRIDAD DE LA INFORMACIÓN.

TEMAS A DESARROLLAR

3.1. INTRODUCCIÓN

¿Qué es MySQL?

Mysql es un sistema de gestión de **bases de datos relacionales y de código abierto**, que utiliza SQL (*Structured Query Language*). En la actualidad, MySQL se ha convertido en un destacado sistema de gestión de bases de datos gratuito desarrollado y respaldado por Oracle y Microsoft SQL Server. Este modelo de gestión de datos es utilizado por millones de sitios web y aplicaciones en todo el mundo.

Dentro de sus características, MySQL destaca por ser **rápido, fiable y fácil de usar**. Es una opción ideal para empresas de todos los tamaños. La funcionalidad de MySQL se basa en un modelo relacional, lo que significa que los datos se organizan en tablas y pueden vincularse fácilmente entre sí. Esto facilita el almacenamiento y la consulta de datos, así como la creación de aplicaciones complejas.

MySQL también ofrece una amplia gama de características como: el **soporte para transacciones, procedimientos almacenados y triggers**. Como resultado, es muy adecuado para el desarrollo de aplicaciones basadas en la web. MySQL es un sistema de gestión de bases de datos potente.

Sin embargo, para entender **cómo usar MySQL** tenemos que ir un paso atrás y asociar dos importantes conceptos: Base de datos relacional y el modelo cliente-servidor.

3.1.1. **Base de datos relacional**

Una base de datos relacional es un tipo de gestión de datos que contiene y recupera toda la información basándose en enlaces entre tablas. Para utilizarla, necesitas un lenguaje como SQL (*Structured Query Language*).

Además el sistema de datos relacional te ayuda a agrupar toda tu información en múltiples formas y de manera separada. Es decir, los almacena en diferentes tablas, en lugar de mantener todo en una sola unidad.

3.1.2. **Modelo cliente-servidor**

En un modelo cliente-servidor, cada computador o proceso de la red es un cliente o un servidor. Los clientes inician las peticiones a los servidores y éstos las procesan y devuelven los resultados a los clientes.

Este modelo se utiliza para todo, desde el acceso a páginas web hasta el envío de correo electrónico. **MySQL es un sistema de bases de datos relacionales** que también emplea una arquitectura cliente-servidor. Al operar con SQL, el cliente transmite una petición al servidor de la database solicitando los datos que necesita.

¿Por qué usar MySQL?

Hay muchas razones para usar MySQL, pero aquí están algunas de las más importantes:

- **Facilidad de uso:** Incluso si nunca has trabajado con una database relacional antes, podrás utilizar MySQL sin ningún problema.
- **Rapidez:** Esto es especialmente importante si se trabaja con una amplia información.

- **Versatilidad:** Puedes utilizar MySQL para distintas tareas, desde el simple almacenamiento de datos hasta complejas aplicaciones web.

¿Cómo trabajar con MySQL?

Principalmente la gestión de la BD en MySQL se puede hacer por consola o por la interfaz gráfica: phpMyAdmin; ambos incluidos con el XAMPP.

- A la interfaz gráfica se accede mediante un navegador internet en la dirección **http://localhost:puerto/phpmyadmin/**, donde el puerto generalmente no se escribe nada si es por defecto el 80**.**

- La base de datos se puede gestionar cargando la consola desde donde esta instalado el XAMPP (\xampp\mysql\bin>). Lo primero que tenemos que hacer para poder ejecutar comandos es ejecutar el programa '**mysql**' que si ya esta instalado en el PC, bastaría con ejecutar desde la consola el siguiente comando:

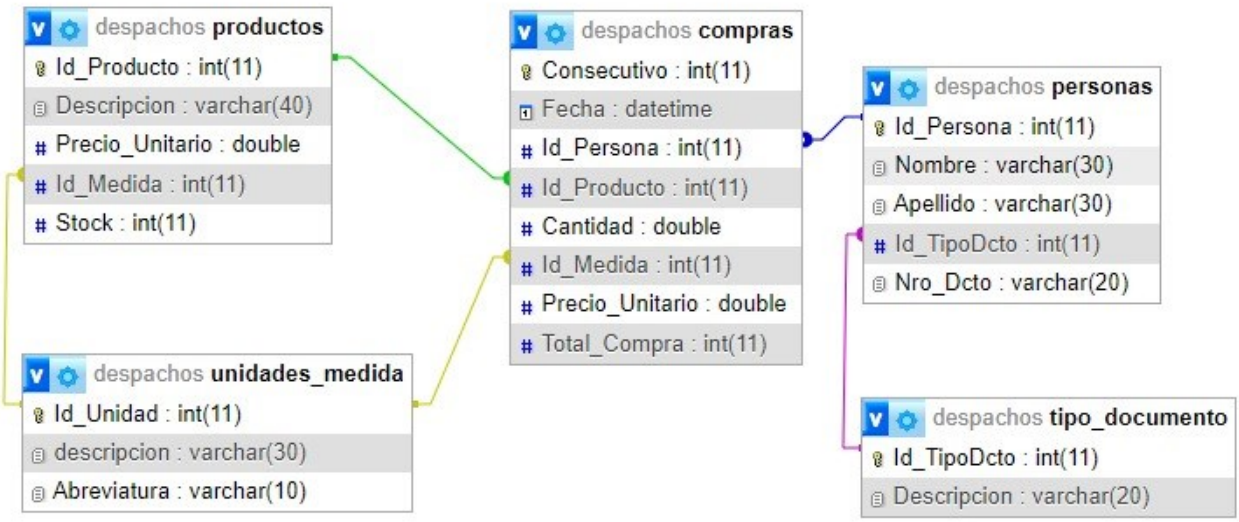
  >mysql -h servidor -u usuario -p clave

  Si el servidor es localhost y el usuario es root y no tiene clave, es suficiente con teclear lo siguiente:
  >mysql -u root

En esta guía los comandos serán explicados principalmente por consola, ya que ya se dio una explicación principalmente empleando la interfaz: phpmyadmin.

3.2. CREACIÓN DE UNA BASE DE DATOS EN MySQL

3.2.1. Crear la Base de Datos DESPACHOS como se muestra en el siguiente diagrama



Para consultar las bases de datos existentes en MySQL, escribe el siguiente comando:

> show databases;

Para crear una base de datos, escribe el siguiente comando:

>create database Nombre_Database;

Para este caso reemplaza "**Nombre_Database**" por el nombre de la base de datos a crear "**DESPACHOS**". Si deseas establecer un conjunto de caracteres específico utiliza el siguiente comando:

>create database despachos character set utf8mb4;

De lo contrario, MySQL utilizará el juego de caracteres por defecto. Para cambiar el conjunto de caracteres especifico se utiliza el comendo Alter Database:

Para crear una base de datos, averiguando primero si no existe:

>CREATE DATABASE IF NOT EXISTS Nombre_Database DEFAULT CHARACTER SET = 'utf8' DEFAULT COLLATE 'utf8_general_ci';

Para cambiar el conjunto de caracteres de una base de datos:

ALTER DATABASE `Nombre_D`atabase CHARACTER SET = utf8mb4 COLLATE = utf8mb4_unicode_ci;

Para borrar una base de datos:

>drop database `Nombre_D`atabase;

Para trabajar o conectar o colocar en uso una base de datos, escribe el siguiente comando:

>Use Nombre_Database;

Por ejemplo: >Use Despachos;

Para ver todas las tablas de una base de datos:

>show tables;

3.2.2. Crear Tablas

Para crear tablas se escribe el siguiente comando:

CREATE TABLE **nombre_tabla** (**columna1** tipo_dato, **columna2** tipo_dato, **columna3** tipo_dato, ...);

- Donde **nombre_tabla** es el nombre de la tabla, por ejemplo: Unidades_medida, Tipo_documento, personas, compras y productos.

- Donde **Columna1, columna2, columna3,….** son los nombres de los campos de las tablas, por ejemplo: **Id_TipoDcto** y **descripcion** son los campos para la tabla **Tipo_documento**.

- Donde **tipo_dato** es el tipo del campo que se esta declarando para las Columna1, Columna2, Columna3, …, por ejemplo: **Int(11), varchar(30) y varchar(10)** son los tipos de dato para **Id_Unidad, Descripcion y Abreviatura** de la tabla **Unidades_medida**.

Tipos de datos en MySQL

MySQL admite una amplia variedad de tipos de datos que puedes utilizar al crear tablas. Algunos de los tipos de datos más comunes son:

- **INT o INTEGER**: para almacenar números enteros. Int e INTEGER, 4 bytes
- **TinyInt:** Es un número entero con o sin signo. Con signo el rango de valores válidos va desde -128 a 127. Sin signo, el rango de valores es de 0 a 255
- **SmallInt:** Número entero con o sin signo. Con signo el rango de valores va desde -32768 a 32767. Sin signo, el rango de valores es de 0 a 65535.
- **MediumInt:** Número entero con o sin signo. Con signo el rango de valores va desde -8.388.608 a 8.388.607. Sin signo el rango va desde 0 a 16777215.
- **VARCHAR**: para almacenar cadenas de texto de longitud variable.
- **CHAR**: para almacenar cadenas de texto de longitud fija.
- **DECIMAL**: para almacenar números decimales, por ejemplo Decimal(3,2): 3 digitos para 2 decimales.
- **DATE**: para almacenar fechas.
- **TIME**: para almacenar horas.
- **DATETIME**: para almacenar fechas y horas combinadas.
- **FLOAT:** Número pequeño en coma flotante de precisión simple. Los valores válidos van desde -3.402823466E+38 a -1.175494351E-38, 0 y desde 1.175494351E-38 a 3.402823466E+38. (4 bytes)
- **xReal, Double:** Número en coma flotante de precisión doble. Los valores permitidos van desde -1.7976931348623157E+308 a -2.2250738585072014E-308, 0 y desde 2.2250738585072014E-308 a 1.7976931348623157E+308. (8 bytes)

❖ Para crear la tabla **Tipo_Documento**, desde la consola se escribe:

>Create Table **Tipo_Documento** (**Id_TipoDcto** Int Null Auto_Increment Primary Key, **Descripcion** varchar(20));

Para consultar las características de la tabla creada, desde la consola se escribe lo siguiente:

>Desc **Tipo_Documento;**

Para modificar alguna propiedad de la tabla, se utiliza el comando “Alter Table”, por ejemplo para cambiar el nombre del campo “Descrpcion” por “Descripción” de la tabla “Tipo_Documento”

>Alter Table **Tipo_Documento** change Descrpcion Descripcion varchar(20);

Para adicionar un campo **Temporal** a la tabla Tipo_Documento:

>Alter Table **Tipo_Documento** add Temporal **decimal(3,2);**

Para modificar el tipo de campo **Temporal** a real de la tabla Tipo_Documento:

>Alter Table **Tipo_Documento** Modify Temporal **real;**

Para borrar el campo **Temporal** de la tabla Tipo_Documento:

>Alter Table **Tipo_Documento** drop Temporal**;**

Para cambiar el nombre de una tabla, por ejemplo Tipo_Documento a TipoDocumento:

>Rename Table **Tipo_Documento** to TipoDocumento**;**

También se puede utilizar el siguiente comando:

>Alter Table **TipoDocumento** Rename to Tipo_Documento**;**

Para ver el listado de las tablas de la base de datos:

>show tables;

- Para crear la tabla **Unidades_Medida**, desde la consola se escribe:

>Create Table **Unidades_Medida** (**Id_Unidad** Int Null Auto_Increment Primary Key, **Descripcion** varchar(30), **Abreviatura** varchar(10));

- Para crear la tabla **Personas**, desde la consola se escribe:

>Create Table **Personas** (**Id_Persona** Int Null Auto_Increment Primary Key, **Nombre** varchar(30), **Apellido** varchar(30), **Id_TipoDcto** Int, **Nro_Dcto** varchar(20));

- Para crear la tabla **Productos**, desde la consola se escribe:

>Create Table **Productos** (**Id_Producto** Int Null Auto_Increment Primary Key, **Descripcion** varchar(40), **Precio_Unitario** Double, **Id_Medida** Int, **Stock** Int);

- Para crear la tabla **Productos**, desde la consola se escribe:

>Create Table **Compras** (**Consecutivo** Int Null Auto_Increment Primary Key, **Fecha** datetime, **Id_Persona** Int, **Id_Producto** int, Cantidad int, Id_Medida Int, **Precio_Unitario** Double, **Total_Compra** Real);

3.2.3. Crear Llaves primarias

- Para eliminar un Primary Key de la tabla **Tipo_Documento**, desde la consola se escribe:

```
> ALTER TABLE Tipo_Documento DROP PRIMARY KEY;
```

- Para agregar un Primary Key de la tabla **Tipo_Documento**, desde la consola se escribe:

```
> ALTER TABLE Tipo_Documento Add PRIMARY KEY(id_TipoDcto);
```

- Para agregar la propiedad Auto_Increment al campo `id_TipoDcto` de la tabla **Tipo_Documento**, desde la consola se escribe:

```
> ALTER TABLE Tipo_Documento MODIFY id_TipoDcto INT NOT NULL Auto_Increment;
```

3.2.4. Crear Llaves foráneas (Relaciones entre Tablas)

- Para crear la llave foránea en la tabla **personas** con el campo **Id_TipoDcto** y la llave primaria **Id_TipoDcto** de la tabla **Tipo_Dcumento**, se escribe de la siguiente forma:

```
>ALTER TABLE Personas Add FOREIGN KEY (Id_TipoDcto) REFERENCES Tipo_Documento (Id_TipoDcto)
>ALTER TABLE Personas ADD INDEX (Id_TipoDcto);
```

- Para crear la llave foránea en la tabla **Productos** con el campo **Id_Medida** y la llave primaria **Id_Unidad** de la tabla **Unidades_Medida**, se escribe de la siguiente forma:

```
>ALTER TABLE Productos ADD INDEX (Id_Medida);
>ALTER TABLE Productos Add FOREIGN KEY (Id_Medida) REFERENCES Unidades_Medida (Id_Unidad)
```

- Para crear la llave foránea en la tabla **Productos** con el campo **Id_Medida** y la llave primaria **Id_Unidad** de la tabla **Unidades_Medida**, se escribe de la siguiente forma:

```
>ALTER TABLE Compras ADD INDEX (Id_Persona);
>ALTER TABLE Compras ADD INDEX (Id_Medida);
>ALTER TABLE Compras ADD INDEX (Id_Producto);
>ALTER TABLE Compras Add FOREIGN KEY (Id_Persona) REFERENCES Personas (Id_Persona);
>ALTER TABLE Compras Add FOREIGN KEY (Id_Producto) REFERENCES Productos (Id_Producto);
>ALTER TABLE Compras Add FOREIGN KEY (Id_Medida) REFERENCES Unidades_Medida (Id_Unidad);
```

3.2.5. Ingresar Registros a las tablas

- Para Ingresar registros en la tabla Tipo_Documento, se escribe de la siguiente forma:

>INSERT INTO Tipo_Documento (Id_TipoDcto, Descripcion) Values (Null, ‘Cedula de Ciudadania`’), (Null,‘Tarjeta de Identidad’);

- Para Ingresar registros en la tabla Unidades_Medida, se escribe de la siguiente forma:

>INSERT INTO Unidades_Medida (Id_Unidad, Descripcion,Abreviatura) Values (Null, ‘Toneladas’,’Tn’), (Null,‘Kilogramos’,’Kgs’);

- Para Ingresar registros en la tabla Personas, se escribe de la siguiente forma:

>INSERT INTO Personas (Id_Persona, Nombre, Apellido, Id_TipoDcto, Nro_Dcto) Values (Null,’Juan’,’Rodriguez’,1, ‘1234343’), (Null,’Pedro’,’Suarez’,2, ‘1243343’), (Null,’Arnulfo’,’Briseño’,1, ‘1242343’);

- Para Ingresar registros en la tabla Productos, se escribe de la siguiente forma:

>INSERT INTO Productos (Id_Producto, Descripcion, Precio_Unitario, Id_Medida, Stock) Values (Null,’Purina’,3000.50,1, 10000), (Null,’Melasa’,4000.80,2,2000), (Null,’Sal Industrial’,1000,1,5000);

- Para Ingresar registros en la tabla Compras, se escribe de la siguiente forma:

>INSERT INTO Compras (Consecutivo, Fecha, Id_Persona, Id_Producto, Cantidad, Id_Medida, Precio_Unitario, Total_Compra) Values (Null,Now(),1,1,400,1,3000.50,400*3000.50), (Null,Now(),2,2,200,1,4000.80,200*4000.80), (Null,Now(),1,1,100,2,3000.50,100*3000.50);

3.2.6. Actualizar Registros en las tablas

- Para actualizar la abreviatura de ‘TN’ a ‘Tns’ en la tabla Unidades_Medida para la unidad ‘Toneladas’, se escribe de la siguiente forma:

>Update Unidades_Medida set Abreviatura=’Tns’ where Id_Unidad=1;

- Para actualizar el Stock de 10000 a 12000 en la tabla Productos para el producto ‘Purina’, se escribe de la siguiente forma:

>Update Productos set Stock=12000 where Id_Producto=1;

- Para actualizar el Nombre de ‘Arnulfo’ a ‘Luis Arnulfo’ en la tabla Personas para el Id_Persona 3, se escribe de la siguiente forma:

>Update Personas set Nombre=’Luis Arnulfo’ where Id_Persona=3;

3.2.7. Borrar Registros en las tablas

- Para anular (borrar) el producto ‘Sal Industrial’ en la tabla Productos, se escribe de la siguiente forma:

>Delete From Productos where Descripcion=’Sal Industrial’;

- Para anular la compra 3 (borrar) de 100 kgs de ‘Purina’ para ‘Juan’ en la tabla Compras, se escribe de la siguiente forma:

>Delete From Compras where Consecutivo=3;

3.3. SALIR DE LA CONSOLA DE MySQL

Para salir de la línea de comandos de MySQL, una vez de terminado de trabajar, cerramos la conexión con el servidor, escribiendo “quit” desde el prompt de MySQL:

> quit

4. OTRAS FUENTES

https://www.youtube.com/watch?v=hB-iaxIIIss
https://www.youtube.com/watch?v=JKMEDTfUZ58&list=PLg9145ptuAig5eqBYvOuMdpTP2ORUNXbf
https://www.youtube.com/watch?v=Q6sn6UMgR54&list=PLg9145ptuAihPxpM3YfQJYwPHv-Vnh_bV
Cómo crear una base de datos en MYSQL [PASO a PASO] (tekla.io)
Comandos básicos para MySQL - BAEHOST.com | Nuestro Blog

5. CONTROL DEL DOCUMENTO

	Nombre	Cargo	Dependencia	Fecha
Autor (es)				

6. CONTROL DE CAMBIOS (diligenciar únicamente si realiza ajustes a la guía)

	Nombre	Cargo	Dependencia	Fecha	Razón del Cambio
Autor (es)					