UNIDAD 2: PHP – MySQL

1. UNIDAD 2: Introducción

#### Bienvenido a **Interactuando con Bases de Datos**

Ahora que sabes lo que es una base de datos y todas las maravillosas herramientas disponibles para almacenar información, es necesario que conozcas la arquitectura de un proyecto Web que hace uso de una base de datos. De eso se trata esta Unidad. Por eso, verás: cómo interactúa PHP con una base de datos, crearás una base de datos MySQL, actualizarás y eliminarás registros y realizarás consultas en bases de datos.

 Puntos de aprendizaje

**Unidad 2: PHP – MySQL**  
¡Aquí comienza la Unidad 2: PHP – MySQL!

* Los objetivos del Programa que corresponden a esta Unidad son:
  + **Usar metodologías sincrónicas y asincrónicas para gestionar datos desde el servidor.**
  + **Crear ambientes de programación y a gestionar recursos.**
* Los objetivos del Curso 7 que corresponden a esta Unidad son:
  + **Asimilar, diseñar y crear bases de datos SQL.**
  + **Gestionar información con bases de datos SQL.**

1. Lección 1: ¿Cómo interactúa PHP con una base de datos?

2.1. Interacción con una base de datos

Arquitectura de un proyecto web + Base de datos

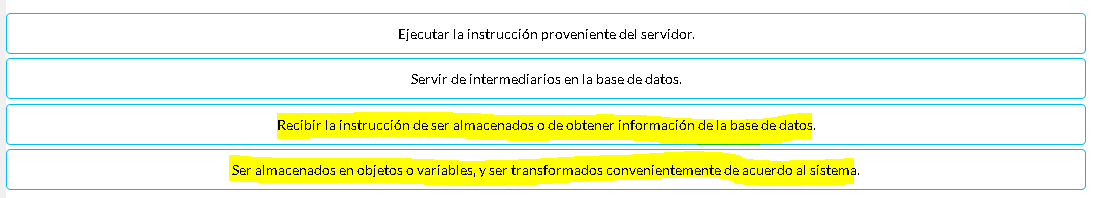
Controlador, mediador entre servidor y base de datos

2.2. Actividad 1

Tiempo de pensar

Hemos estudiado el viaje que hacen los datos en un sistema que hace uso de una base de datos, el cual corresponde a la arquitectura completa de un sistema de información Web que interactúa con una base de datos.  
  
Instrucciones:  
Escoge las opciones correctas que completan el enunciado que ves a continuación.

Enunciado:  
Es posible afirmar que los datos que son recibidos y verificados en el servidor, pueden:



2.3. Lectura: ¿Qué es un controlador?

¿Qué es un controlador?



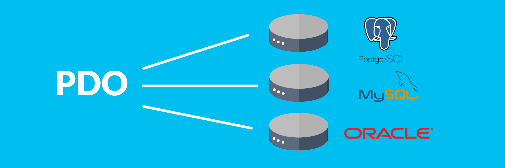
Un controlador o driver, como es mejor conocido en el mundo del desarrollo Web, es una herramienta de software que hace las veces de intérprete entre una base de datos y un sistema Web. Su funcionamiento se basa en el uso de bibliotecas propias de los Sistemas Gestores de Bases de Datos que alojan la información que interactúa con el sistema a enlazar, otorgando una conexión entre el servidor del sistema web y la base de datos correspondiente.  
  
Como cada Sistema Gestor de Base de Datos (SGBD) posee características particulares, así como cada entorno de back-end, o tecnología, que pueda consumir la base de datos, también puede ser muy diferente una de otra; para cada una de ellas, existen controladores específicos.  
  
En nuestro caso usaremos uno de los SGBD más usados en el desarrollo Web: MySQL. También usaremos el lenguaje PHP para los procesos del servidor que interactuarán con la base de datos. Consecuentemente, usaremos un driver específico para crear conexiones desde PHP a MySQL, el cual nos permitirá acceder desde nuestro proyecto de desarrollo a una base de datos a la cual podremos agregar y consultar datos para mostrarlos en una página Web.  
  
Para PHP, específicamente, existen una gran cantidad de controladores y de sus respectivas extensiones para cumplir las funciones de conexión a bases de datos. Las extensiones más comunes, que están incluidas en el core de PHP 7, son PDO y MySQLi, de las cuales hablaremos con mayor profundidad en esta lección.

2.4. Lectura: PDO y MySQLi

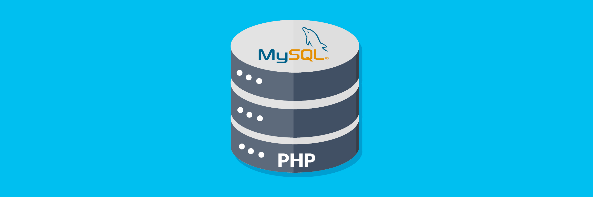
PDO y MySQLi



PDO  
PDO (Objetos de Datos de PHP) es una extensión que define una interfaz para establecer una comunicación con bases de datos, es un intermediario que permite que un script de PHP pueda lanzar sentencias que se ejecuten en una base de datos. Su característica particular consta de su flexibilidad, ya que puede ser usado para una gran variedad de SGBD sólo cambiando algunos parámetros en sus métodos de acuerdo al sistema.



MySQLi  
MySQLi es una extensión incluida en PHP para realizar conexiones a bases de datos MySQL. Sus siglas significan MySql improved o MySql mejorado, ya que nace como la mejora a una versión antigua de la extensión para PHP y MySQL por defecto llamada MySQL. Esta versión anterior ya se encuentra descontinuada y la documentación oficial de PHP sugiere el uso de MySQLi, la cual corresponde a una extensión poderosa y rica en métodos para que el desarrollador pueda sacar todo el provecho de la interacción con la base de datos MySql.



2.5. PHP MyAdmin

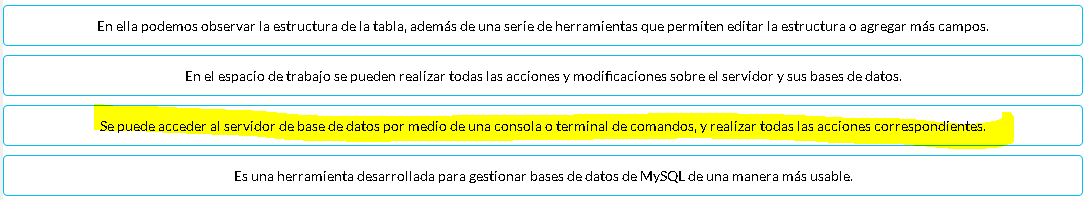
Con xampp

<http://localhost:PORT/phpmyqdmin>

2.6. Actividad 2

Tiempo de pensar

Hemos estudiado una herramienta que puede ser de gran ayuda: PHP MyAdmin. Te invitamos a que realices esta actividad.  
  
Instrucciones:  
  
Lee con atención el enunciado y escoge la respuesta correcta.  
  
Enunciado:  
  
De las opciones que ves listadas a continuación, escoge aquella que no hace referencia a la herramienta PHP MyAdmin:

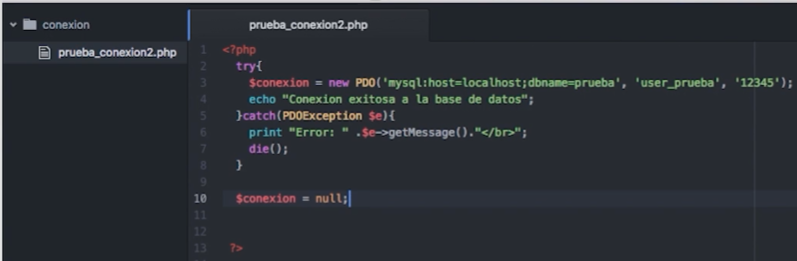


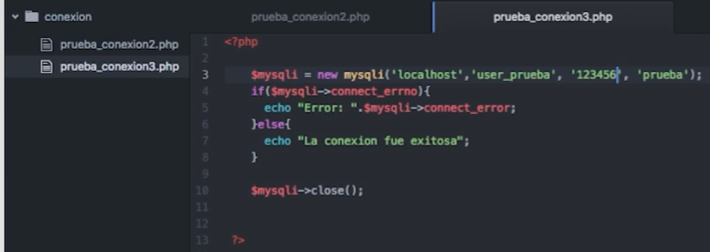
1. Lección 2: Crear una base de datos MySQL

3.1 [Conexión a un servidor de Base de Datos](https://learn.nextu.com/mod/lesson/view.php?id=3562&pid=P_WEBDEV_V2)

PHP conexión a DB 🡪 Driver

PDO o MySQLi

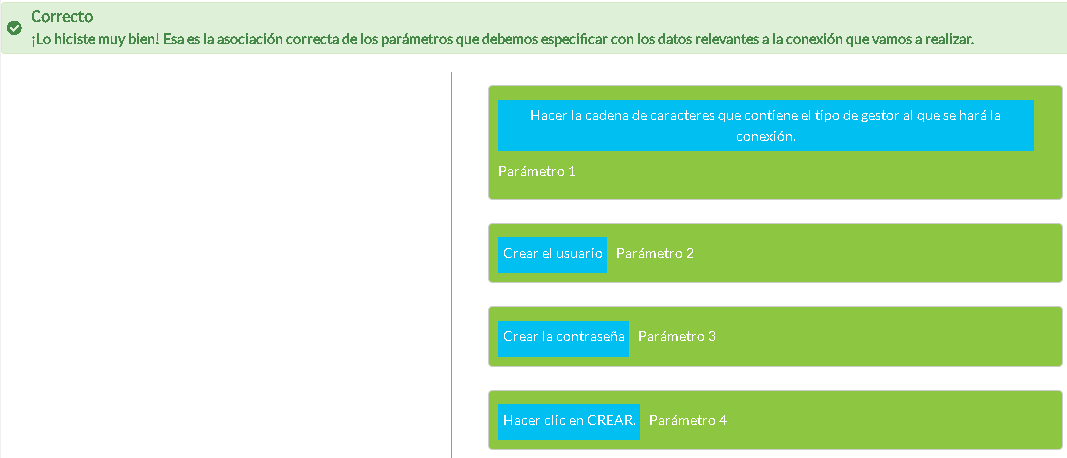




3.2. Actividad 1

Tiempo de pensar

En esta lección hemos aprendido a conectar el servidor en el que se ejecuta nuestro proyecto Web al servidor de bases de datos. Realiza la siguiente actividad para que valides tus conocimientos.  
  
Instrucciones:  
  
Tienes una misión en esta actividad: debes asociar cada parámetro con el dato para realizar la conexión. ¡Comienza!



3.3. Creación de una Base de Datos

Se puede utilizar tanto PDO como MySQLi drivers

3.4. Creación de una tabla

MySQLi y PDO 🡪 La ejecución de sentencias es muy similar

3.5. Creación de relaciones y restricciones

ALTER TABLE usuarios ADD FOREIGN KEY fk\_ciudad REFERENCES ciudades (id);

Para poder crear una llave foránea la llave de la tabla debe ser primaria, en otro caso genera el error:

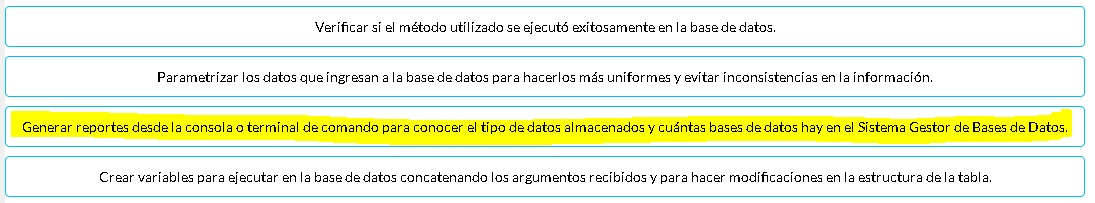
MariaDB [inventario\_db]> ALTER TABLE usuarios ADD FOREIGN KEY (fk\_ciudad) REFERENCES ciudades (id);

ERROR 1005 (HY000): Can't create table `inventario\_db`.`usuarios` (errno: 150 "Foreign key constraint is incorrectly formed")

3.6. Actividad 2

Tiempo de pensar

En el video anterior vimos cómo hacer uso de PHP para crear restricciones y relaciones entre tablas. Te invitamos a que realices esta actividad para validar los conocimientos que adquiriste sobre esta herramienta.  
  
Instrucciones:  
  
Lee cuidadosamente el enunciado y escoge la respuesta correcta.  
  
Enunciado:  
  
¿Cuál de las siguientes opciones no corresponden a las características de las restricciones y relaciones entre tablas?



1. Lección 3: DML desde PHP
   1. Insertar registros

4.2. Prepare

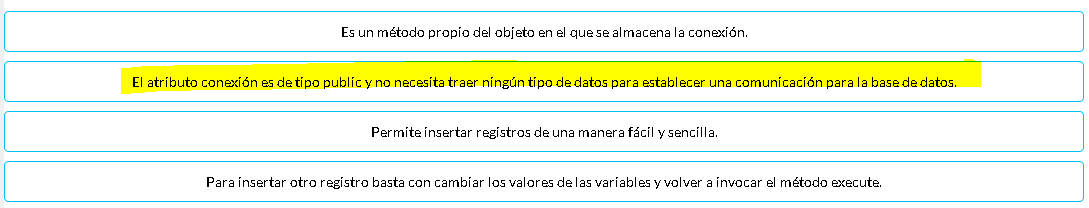
Método más sencillo para la carga de datos insert

INSERT INTO usuarios (id, nombre, email, telefono, fk\_ciudad) VALUES (?,?,?,?,?)

4.3. Actividad 1

Tiempo de pensar

Ya conocemos el método prepare y aprendimos a usarlo, te invitamos a que afiances tus conocimientos realizando esta actividad.  
  
Instrucciones:  
  
Determina cuál de las siguientes opciones responde correctamente a la pregunta planteada.  
  
Enunciado:  
  
¿Cuál de las siguientes opciones no corresponden a las ventajas del método prepare?



4.4. Actualizar registros

UPDATE Personas SET nombre='Luis' WHERE id=4;

Los valores de las variables string deben de colocarse dentro de comillas dobles y simples:

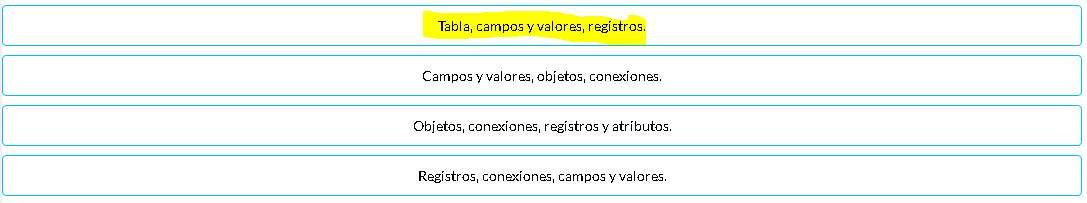
$datos['nombre'] = "'Pablo Garcia'";

$datos['telefono'] = "'4328754'";

4.5. Actividad 2

Tiempo de pensar

Hemos estudiado cómo se actualizan los registros en una base de datos desde un script de PHP. Te proponemos la siguiente actividad para que pongas en práctica lo que aprendiste.  
  
Instrucciones:  
Lee atentamente el enunciado y escoge la respuesta que consideras correcta.  
  
Enunciado:  
En la actualización de registros en una base de datos existen 3 partes importantes a determinar:



4.6. Eliminar registros

Sintaxis:

DELETE FROM Personas WHERE genero = Masculino;

1. Lección 4: Consultas a la base de datos
   1. Realizar Consultas

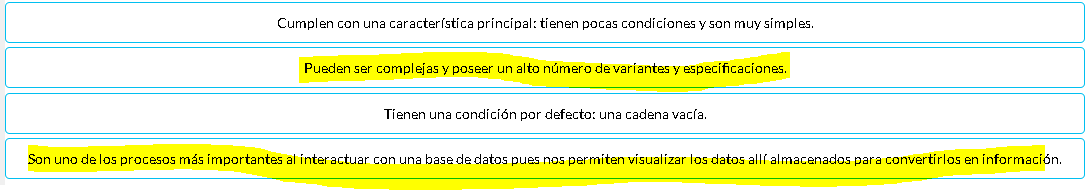
Se crea una función con la consulta a ejecutar, esta deberá llevar un arreglo con las tablas, los campos y la condición.

consultar($tablas, $campos, $condicion = "")

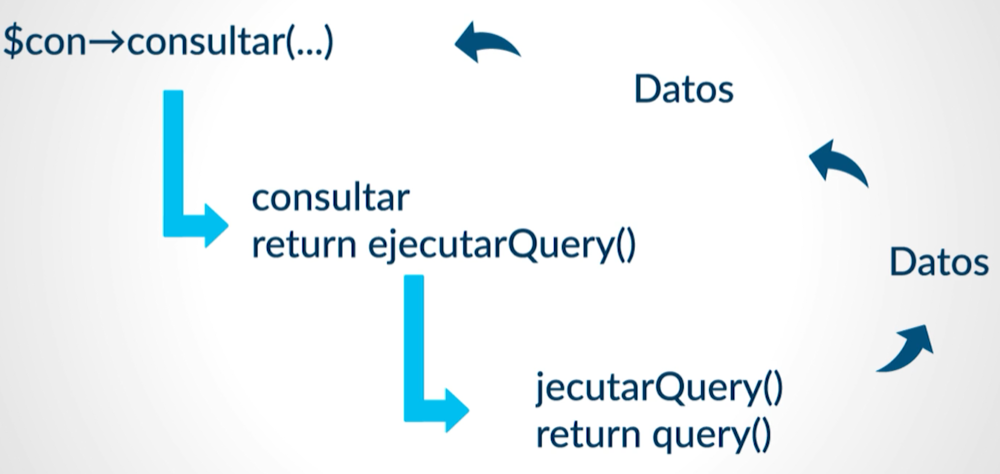
5.2. Actividad 1

Tiempo de pensar

Hemos realizado consultas a una base de datos desde un script de PHP, así que afianza el conocimiento que adquiriste a través de esta actividad.  
  
Instrucciones:  
Escoge las opciones correctas que completan el enunciado que ves a continuación.  
  
Enunciado:  
De acuerdo al contenido que acabamos de ver, podemos concluir que las consultas:

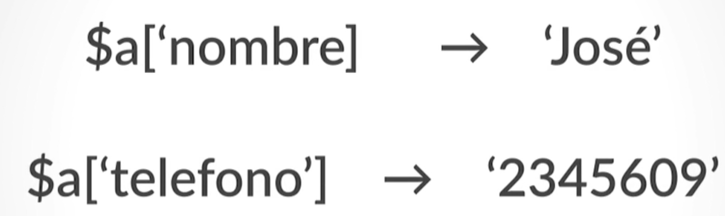


5.3. Recepción de resultados



Fetch\_assoc propio de MySQLi

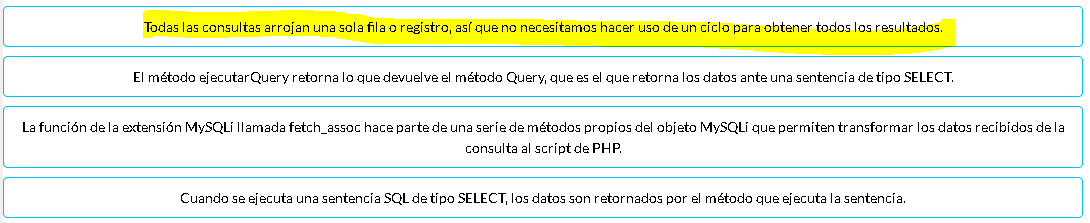




5.4. Actividad 2

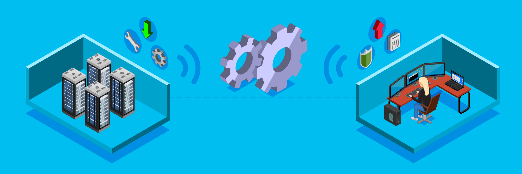
Tiempo de pensar

Hemos construido un método que hace una consulta SQL y la ejecuta en la base de datos designada, y cómo obtener en PHP los datos almacenados en MySQL. Para afianzar tu conocimiento en este tema, te invitamos a realizar la siguiente actividad.  
  
Instrucciones:  
Lee el enunciado con mucha atención y escoge la respuesta correcta.  
  
Enunciado:  
¿Cuál de las siguientes opciones no hacen parte de las características que podemos concluir sobre la recepción de resultados?



5.5. Lectura: Invocación de Procedimientos almacenados

Invocación de procedimientos almacenados



Como ya lo viste en la unidad anterior, los procedimientos almacenados son estructuras internas de la base de datos que permiten automatizar procesos de la misma para aumentar la seguridad y proveer soluciones rápidas ante las necesidades de los servicios que deben ser alimentados por dichos datos.  
La invocación de un procedimiento almacenado, una vez definido en una base de datos MySQL, se realiza mediante el comando CALL, seguido del nombre del procedimiento a invocar y los parámetros correspondientes.

CALL procedimiento1( param1, param2);

Para realizar este tipo de invocaciones desde un script de PHP se debe seguir el mismo procedimiento para ejecutar una sentencia SQL en la base de datos. De tal manera, se usa la extensión PDO ó MySQLi para crear un objeto conexión e invocar el método Query. En los parámetros de dicho método se envía la sentencia de invocación del procedimiento; si tiene un retorno específico, lo retornará como si fuese una consulta convencional. De este modo es posible aplicar el método fetch\_assoc sobre el resultado y obtener los datos resultado del procedimiento en un arreglo de PHP.

$resultado = $conexion->query(‘CALL nombre\_procedimiento(23, ‘Hombre’)’);

while ($fila = $resultado->fetch\_assoc()){

echo $fila[‘resultado\_procedimiento’];

}

En resumen, la invocación de un procedimiento almacenado desde un script de PHP a una base de datos MySQL, se realiza de la misma manera que se invoca cualquier tipo de sentencia y se gestionan sus resultados como una consulta de tipo SELECT.

5.6. Ejercicio

HORA DE  
**codificar**

La interacción de PHP con los servidores de bases de datos resulta sumamente sencilla en cuanto a que las extensiones que se encargan de establecer el puente entre PHP y los motores de bases de datos como PostgreSQL o MySQL, permiten realizar gran cantidad de acciones a través de funciones y métodos sencillos de implementar. Tal es el caso de mysqli para bases de datos en MySQL, o PDO para un uso generalizado ante cualquier motor que aplique al sistema.  
  
En este ejercicio crearás un script de PHP para codificar una clase que gestione todas las interacciones entre PHP y el motor de bases de datos. Podrás reutilizar este script para todos los proyectos en que los necesites.  
  
Para el desarrollo de este ejercicio debes seguir las instrucciones a continuación para codificar una clase conexión. Al finalizar, guarda este script para que lo puedas utilizar cada vez que realices un proyecto con PHP y una base de datos.

Instrucciones

1. Crea la estructura básica de una clase con el nombre que desees. Debes crear los siguientes atributos privados: host, usuario, contraseña y conexión.
2. Crea el constructor de la clase para que inicialice los atributos host, usuario y contraseña al instanciar un objeto.
3. Crea un método que establezca una conexión con una base de datos MySQL. Puedes usar PDO o mysqli. Dicha conexión debe asignarse al atributo conexión de la clase.
4. Crea un método que reciba una cadena de caracteres como parámetro, y la ejecute como sentencia SQL sobre la conexión. Este método debe retornar el resultado de esta ejecución.
5. Crea un método que cierre la conexión.
6. Crea un método que reciba el nombre de una tabla, y un arreglo asociativo con los nombres de los campos y su respectivo tipo de dato. Este método debe fabricar una sentencia SQL correcta y ejecutarla sobre la conexión de la clase para crear una nueva tabla.
7. Crea un método que cree una restricción sobre una tabla existente en la base de datos.
8. Crea un método que cree una relación entre dos tablas de la base de datos.
9. Crea un método que permita insertar un registro en una tabla de la base de datos.
10. Crea un método que retorne el objeto de la conexión de la clase.
11. Crea un método que actualice un registro en una tabla de la base de datos.
12. Crea un método que elmine un registro de una tabla en la base de datos.
13. Crea un método que realice una consulta sencilla a una tabla de la base de datos.
14. Crea un método que permita realizar consultas complejas a varias tablas relacionadas de la base de datos.
15. Implementa este script creando un objeto de la clase creada y prueba todos sus métodos sobre una base de datos para comprobar que realizaste cada método correctamente.

Buenas Prácticas

Desarrollar un script que te permita reutilizar código en futuras ocasiones es una práctica muy recomendada, ya que permite al desarrollador conformar un conjunto de librerías de autoría propia de fácil implementación en proyectos de software y hechas a la medida de las necesidades. Eventualmente también puedes compartir estas librerías para que otros desarrolladores hagan uso de ellas, pudiendo contribuir a la comunidad de desarrollo y ganar experiencia y prestigio en el mercado.

1. UNIDAD 2: Prueba