

Universidad Tecnológica de Panamá

Facultad de Sistemas Computacionales

Asignatura: Programación I

Taller Práctico3

Profesor: Napoleón Ibarra

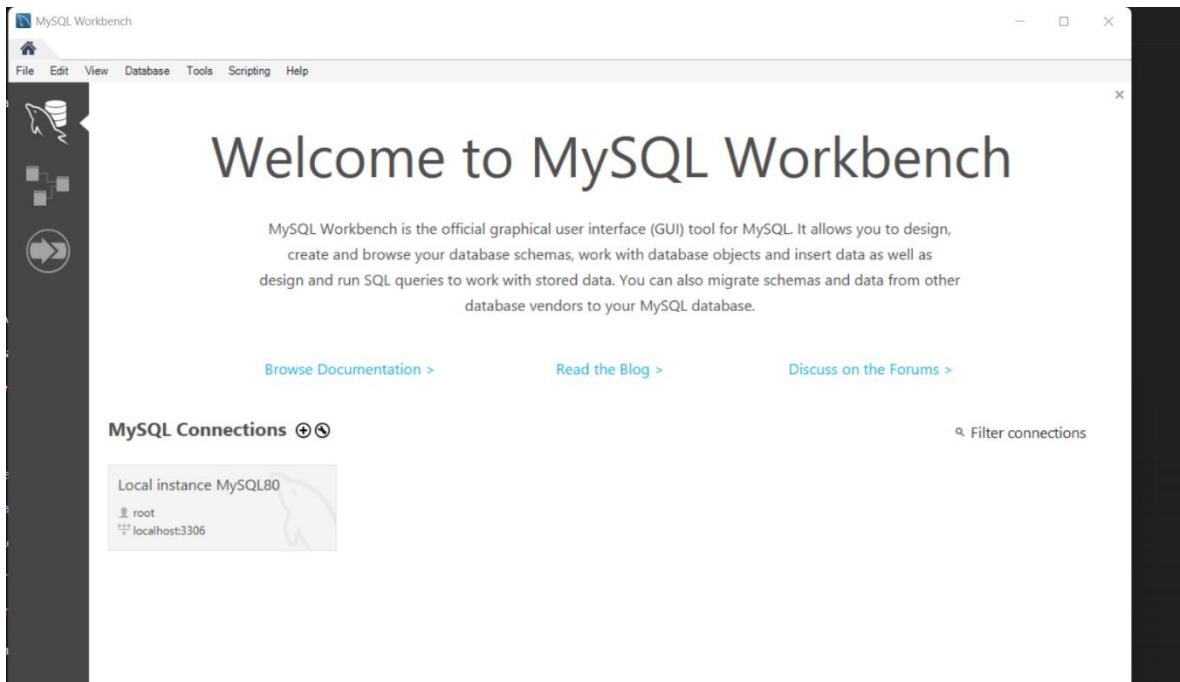
Valor: 100 puntos

Estudiante: Angeline Urriola & Daniela Sanchez

Procedimiento:

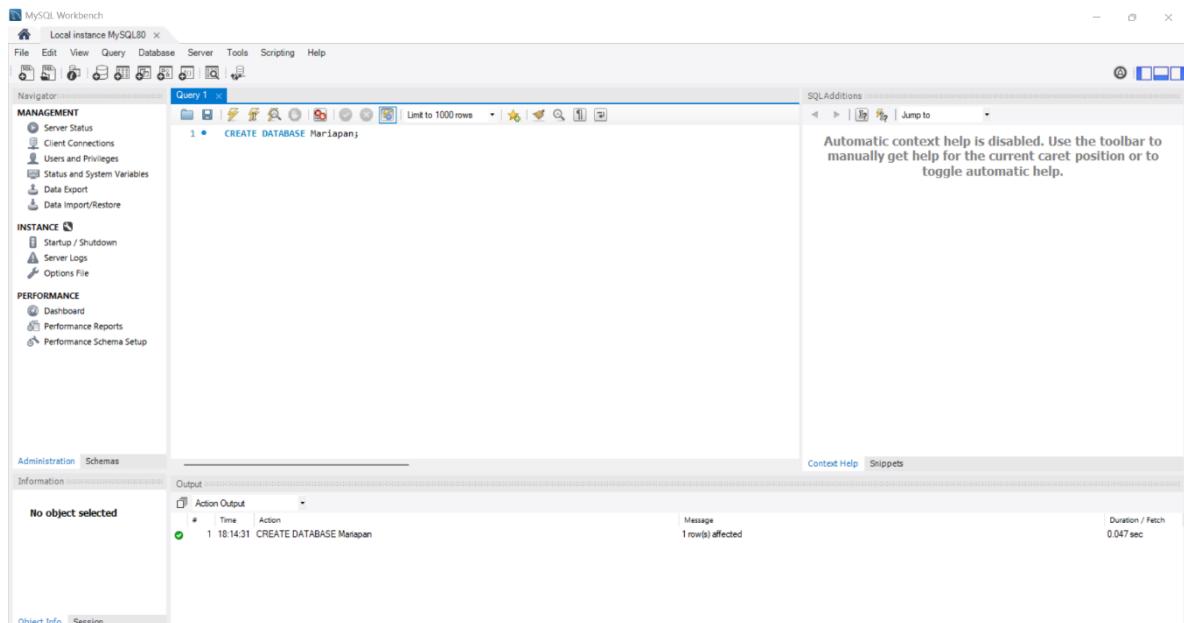
De manera individual o en grupo de 2 personas confeccione la asignación. Utilice la herramienta Internet como apoyo para realizar su proceso de investigación.

Una vez culminada la actividad, subirla a la plataforma Moodle (PDF) y presentar su debida sustentación.



1. Creación de la conexión local en MySQL Workbench

En la imagen se muestra la pantalla principal de MySQL Workbench, donde se configuró la conexión local “Local instance MySQL80” utilizando el usuario root y el puerto 3306. Esta conexión permitió acceder al servidor MySQL instalado en la máquina local para la creación de la base de datos del proyecto.

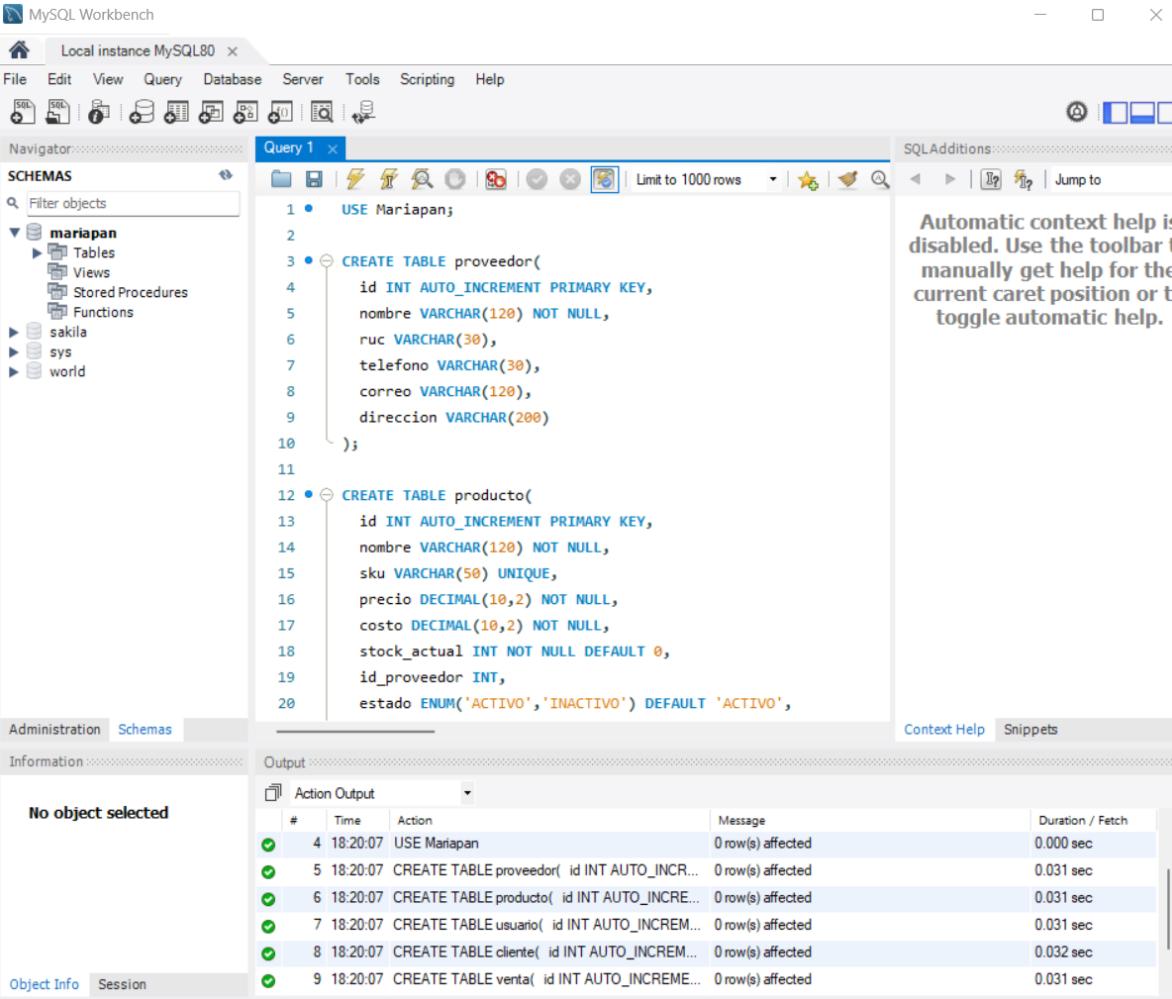


2. Creación de la base de datos “Mariapan”

Una vez establecida la conexión, se ejecutó el comando: CREATE DATABASE mariapan;

Con este paso se creó la base de datos principal que servirá para almacenar toda la información del sistema de facturación.

En la parte inferior del panel “Output” se confirma que la acción se ejecutó correctamente.



The screenshot shows the MySQL Workbench interface. In the top-left, the Navigator pane displays the schema 'mariapan' with its tables, views, stored procedures, and functions. The central Query Editor pane contains the SQL code for creating two tables:

```
1 USE Mariapan;
2
3 CREATE TABLE proveedor(
4     id INT AUTO_INCREMENT PRIMARY KEY,
5     nombre VARCHAR(120) NOT NULL,
6     ruc VARCHAR(30),
7     telefono VARCHAR(30),
8     correo VARCHAR(120),
9     direccion VARCHAR(200)
10 );
11
12 CREATE TABLE producto(
13     id INT AUTO_INCREMENT PRIMARY KEY,
14     nombre VARCHAR(120) NOT NULL,
15     sku VARCHAR(50) UNIQUE,
16     precio DECIMAL(10,2) NOT NULL,
17     costo DECIMAL(10,2) NOT NULL,
18     stock_actual INT NOT NULL DEFAULT 0,
19     id_proveedor INT,
20     estado ENUM('ACTIVO','INACTIVO') DEFAULT 'ACTIVO',
```

The right side of the interface has a status bar message: "Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help." Below the Query Editor is the Output pane, which shows the execution log:

Action	Time	Action	Message	Duration / Fetch
4	18:20:07	USE Mariapan	0 row(s) affected	0.000 sec
5	18:20:07	CREATE TABLE proveedor(id INT AUTO_INCR...	0 row(s) affected	0.031 sec
6	18:20:07	CREATE TABLE producto(id INT AUTO_INCRE...	0 row(s) affected	0.031 sec
7	18:20:07	CREATE TABLE usuario(id INT AUTO_INCREMENT...	0 row(s) affected	0.031 sec
8	18:20:07	CREATE TABLE cliente(id INT AUTO_INCREMENT...	0 row(s) affected	0.032 sec
9	18:20:07	CREATE TABLE venta(id INT AUTO_INCREMENT...	0 row(s) affected	0.031 sec

3. Creación de las tablas principales

Posteriormente, se crearon las tablas que conforman el modelo relacional:

proveedor, producto, usuario, cliente, venta, detalle_venta y factura.

Cada tabla fue diseñada con sus claves primarias, tipos de datos adecuados y relaciones con otras tablas mediante llaves foráneas.

En la parte inferior se observa el registro de ejecución correcta de cada instrucción CREATE TABLE, sin errores.

The screenshot shows the MySQL Workbench interface with the following details:

- Navigator:** Shows the database schema with the 'mariapan' schema expanded, containing Tables, Views, Stored Procedures, and Functions.
- Query Editor:** Contains the following SQL script:


```

1 • USE mariapan;
2
3 • INSERT INTO proveedor (nombre, ruc, telefono, correo, direccion)
4     VALUES ('Harinas Doña María', 'RUC-123', '6000-0000', 'compras@loc...
5
6 • INSERT INTO usuario (nombre, rol, usuario, hash_password)
7     VALUES ('Angeline', 'ADMIN', 'admin', 'hash-demo');
8
9 • INSERT INTO cliente (nombre, cedula_ruc, telefono, correo)
10    VALUES ('Cliente Mostrador', 'N/A', '0000-0000', 'mostrador@loc...
11
12 • INSERT INTO producto (nombre, sku, precio, costo, stock_actual,
13    VALUES ('Pan de molde', 'SKU-001', 2.50, 1.20, 100, 1, 'ACTIVO');
14
      
```
- Output Window:** Displays the log of actions taken by the script:

#	Time	Action	Message	Duration / Fetch
4	18:20:07	USE Mariapan	0 row(s) affected	0.000 sec
5	18:20:07	CREATE TABLE proveedor(id INT AUTO_INCREMENT, nombre VARCHAR(100), ruc VARCHAR(50), telefono VARCHAR(15), correo VARCHAR(100), direccion VARCHAR(200))	0 row(s) affected	0.031 sec
6	18:20:07	CREATE TABLE producto(id INT AUTO_INCREMENT, nombre VARCHAR(100), sku VARCHAR(50), precio DECIMAL(10,2), costo DECIMAL(10,2), stock_actual INT, stock_minimo INT, status VARCHAR(50))	0 row(s) affected	0.031 sec
7	18:20:07	CREATE TABLE usuario(id INT AUTO_INCREMENT, nombre VARCHAR(100), rol VARCHAR(50), usuario VARCHAR(50), hash_password VARCHAR(255))	0 row(s) affected	0.031 sec
8	18:20:07	CREATE TABLE cliente(id INT AUTO_INCREMENT, nombre VARCHAR(100), cedula_ruc VARCHAR(50), telefono VARCHAR(15), correo VARCHAR(100))	0 row(s) affected	0.032 sec
- Right Panel:** A message box states: "Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help."

4. Inserción de datos de ejemplo

Luego se insertaron registros de prueba en las tablas para validar su funcionamiento.

Por ejemplo, se añadió un proveedor (“Harinas Doña María”), un usuario administrador (“Angeline”), un cliente genérico (“Cliente Mostrador”) y un producto inicial (“Pan de molde”).

Esto permite verificar la integridad de las relaciones y facilita la demostración práctica del sistema.

The screenshot shows the MySQL Workbench interface with the following details:

- Navigator:** Shows the database schema with the 'mariapan' schema expanded, containing Tables (productos, proveedor), Views, Stored Procedures, and Functions.
- Query Editor:** Contains the following SQL query:


```
1 • SELECT * FROM mariapan.proveedor;
```
- Result Grid:** Displays the data from the proveedor table:

	id	nombre	ruc	telefono	correo	direccion
1	NULL	Harinas Doña María	RUC-123	6000-0000	compras@gmail.com	Via Princ...
*	NULL	NULL	NULL	NULL	NULL	NULL
- Right Panel:** A message box states: "Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help."

5. Consulta de datos

Se ejecutó la instrucción: SELECT * FROM mariapan.proveedor;

La consulta devolvió los registros insertados, confirmando que la tabla almacena y muestra correctamente la información ingresada.

```
GNU nano 7.2                               /etc/mysql/mysql.conf.d/mysqld.cnf
#
# For explanations see
# http://dev.mysql.com/doc/mysql/en/server-system-variables.html
#
# Here is entries for some specific programs
# The following values assume you have at least 32M ram
#
[mysqld]
#
# * Basic Settings
#
user          = mysql
# pid-file     = /var/run/mysqld/mysqld.pid
# socket       = /var/run/mysqld/mysqld.sock
# port         = 3306
# datadir      = /var/lib/mysql

# If MySQL is running as a replication slave, this should be
# changed. Ref https://dev.mysql.com/doc/refman/8.0/en/server-system-variables.html#sysvar_tmpdir
# tmpdir        = /tmp
#
# Instead of skip-networking the default is now to listen only on
# localhost which is more compatible and is not less secure.
bind-address    = 127.0.0.1
mysqlx-bind-address = 127.0.0.1
#
# * Fine Tuning
#
key_buffer_size     = 16M
# max_allowed_packet = 64M
# thread_stack       = 256K
#
# thread_cache_size   = -1

# This replaces the startup script and checks MyISAM tables if needed
# the first time they are touched
myisam-recover-options = BACKUP

# max_connections     = 151
# table_open_cache     = 4000
#
# * Logging and Replication
#
```

6. Configuración del servidor MySQL en Ubuntu Server

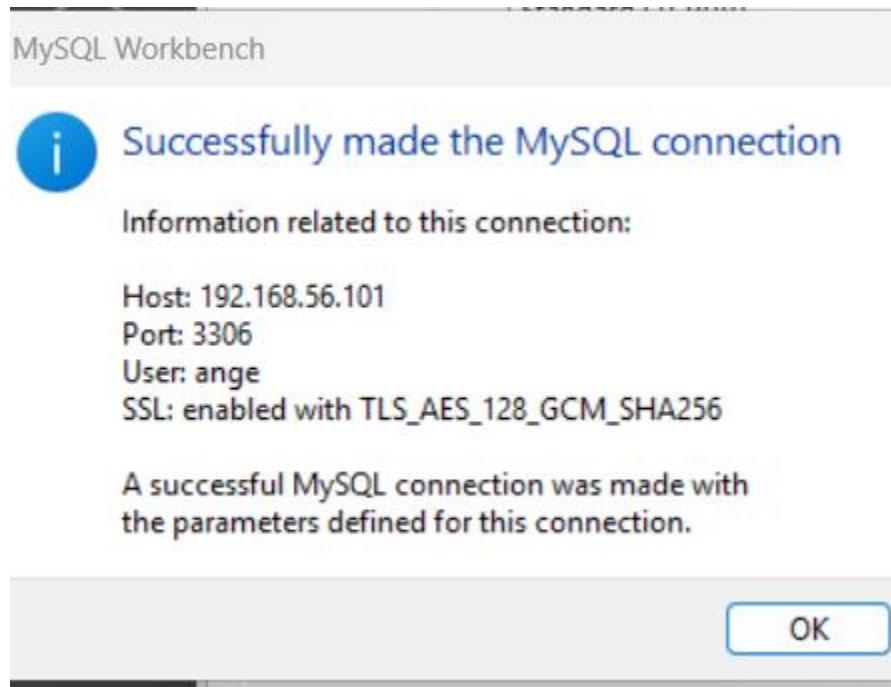
En el entorno del servidor Ubuntu, se configuró el archivo /etc/mysql/mysql.conf.d/mysqld.cnf para definir parámetros de conexión.

Dentro de este archivo se habilitó el puerto 3306 y se verificó la dirección de enlace (bind-address), lo que permite establecer comunicación con el servicio MySQL desde otros equipos o entornos virtuales.

```
sudo apt install netcat-client
ange@ubuntuserver:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 scope host lo
            valid_lft forever preferred_lft forever
        inet6 ::1/128 scope host noprefixroute
            valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:9b:4d:76 brd ff:ff:ff:ff:ff:ff
        inet 10.0.2.15/24 metric 100 brd 10.0.2.255 scope global dynamic enp0s3
            valid_lft 85647sec preferred_lft 85647sec
        inet6 fd17:625c:f087:2:a00:27ff:fe9b:4d76/64 scope global dynamic mngtmpaddr noprefixroute
            valid_lft 86160sec preferred_lft 14160sec
        inet6 fe80::a00:27ff:fe9b:4d76/64 scope link
            valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:e6:e9:52 brd ff:ff:ff:ff:ff:ff
        inet 192.168.56.101/24 scope global enp0s8
            valid_lft forever preferred_lft forever
        inet6 fe80::a00:27ff:fee6:e952/64 scope link
            valid_lft forever preferred_lft forever
ange@ubuntuserver:~$ _
```

7. Verificación de interfaces de red

Mediante el comando ip a se comprobaron las direcciones IP asignadas al servidor Ubuntu. La interfaz enp0s8 mostró la dirección 192.168.56.101, utilizada para establecer la conexión desde MySQL Workbench y desde las pruebas externas de conexión (cliente–servidor).



8. Prueba de conexión exitosa desde MySQL Workbench (Windows)

Finalmente, se configuró una nueva conexión en Workbench utilizando:

- **Host:** 192.168.56.101
- **Puerto:** 3306

- **Usuario:** ange

El mensaje “Successfully made the MySQL connection” confirma que la conexión entre el cliente (Windows) y el servidor MySQL (Ubuntu) fue establecida correctamente.

```
Database changed
mysql> SHOW TABLES;
+-----+
| Tables_in_mariapan |
+-----+
| cliente
| detalle_venta
| factura
| producto
| proveedor
| usuario
| venta
+-----+
7 rows in set (0.01 sec)

mysql> SELECT * FROM proveedor;
Empty set (0.00 sec)

mysql> _
```

9. Verificación de tablas en el servidor

Desde la consola de Ubuntu se ejecutaron los comandos: SHOW TABLES; SELECT * FROM proveedor;

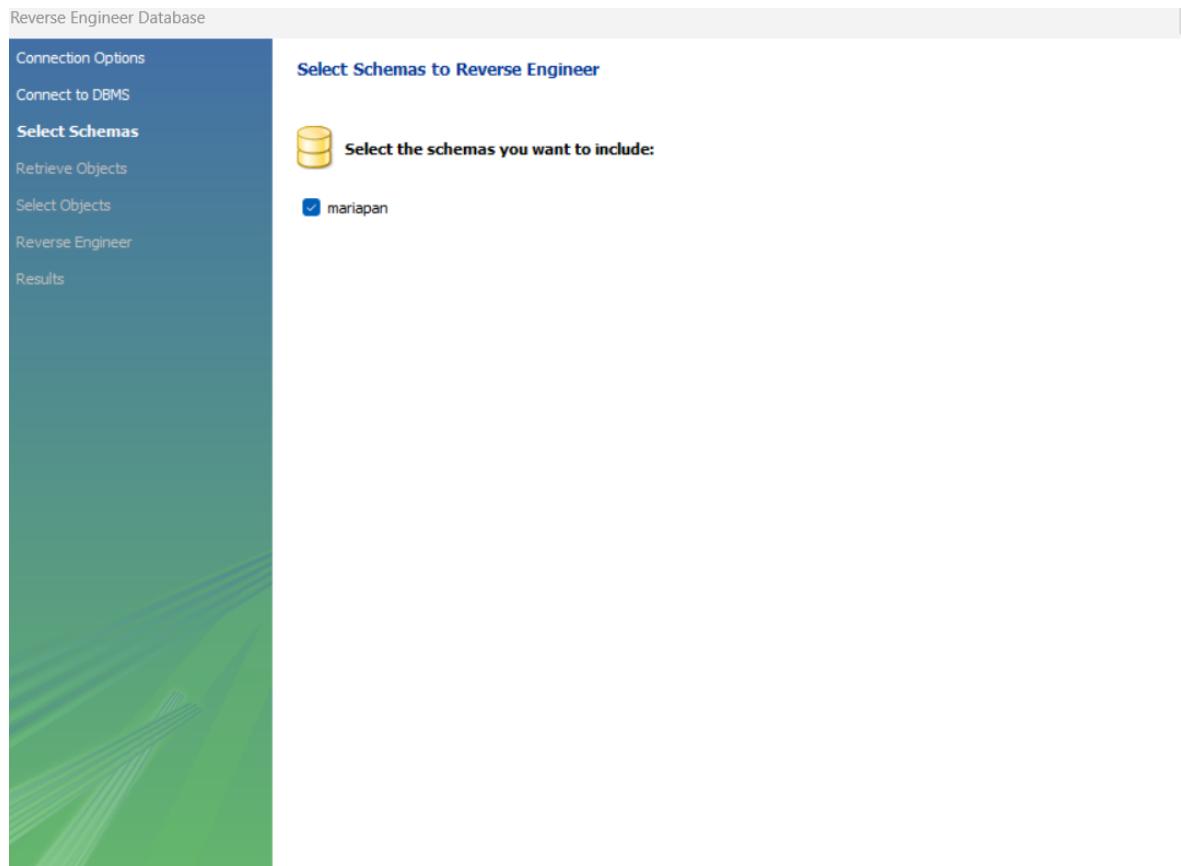
Estos comandos permitieron comprobar que la base de datos “mariapan” y sus tablas existen y están sincronizadas correctamente en el servidor.

	id	id_venta	numero	fecha_emision	metodo_pago	pdf_ruta_en
*	1	1	F000001	2025-10-22 00:47:11	EFFECTIVO	\\SANMATEO\\
*	HULL	HULL	HULL	HULL	HULL	HULL

10. Inserción de registros en la tabla “factura”

En esta imagen se observa la tabla **factura** con un registro insertado correctamente.

Esta inserción demuestra que la base de datos almacena correctamente la información de las facturas generadas, incluyendo la ruta del archivo PDF en el servidor, simulando el guardado en un recurso compartido.

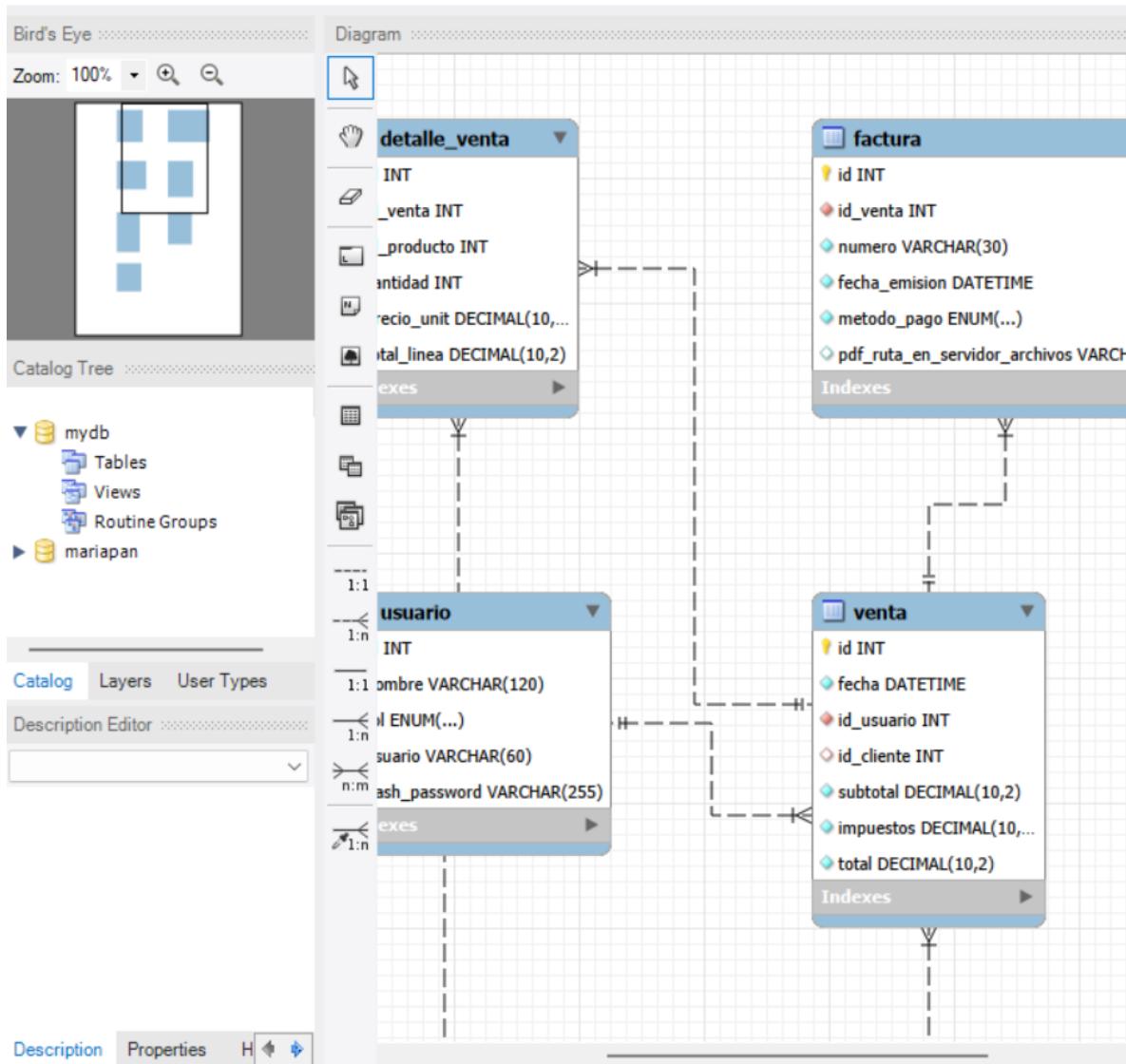


12. Selección del esquema para Reverse Engineer

En esta captura se aprecia la ventana del asistente de **Reverse Engineer Database**.

El esquema “**mariapan**” aparece seleccionado para ser incluido en el proceso.

Este paso permitió generar automáticamente el **modelo entidad-relación (EER)** a partir de las tablas creadas en la base de datos, facilitando la visualización de las relaciones y estructuras del sistema.



13. Diagrama entidad-relación (EER) final

La última imagen muestra el diagrama EER generado desde el proceso de Reverse Engineer. En él se representan las tablas principales: **detalle_venta**, **factura**, **usuario** y **venta**, junto con sus relaciones.

Las líneas entre tablas reflejan las claves foráneas establecidas, por ejemplo:

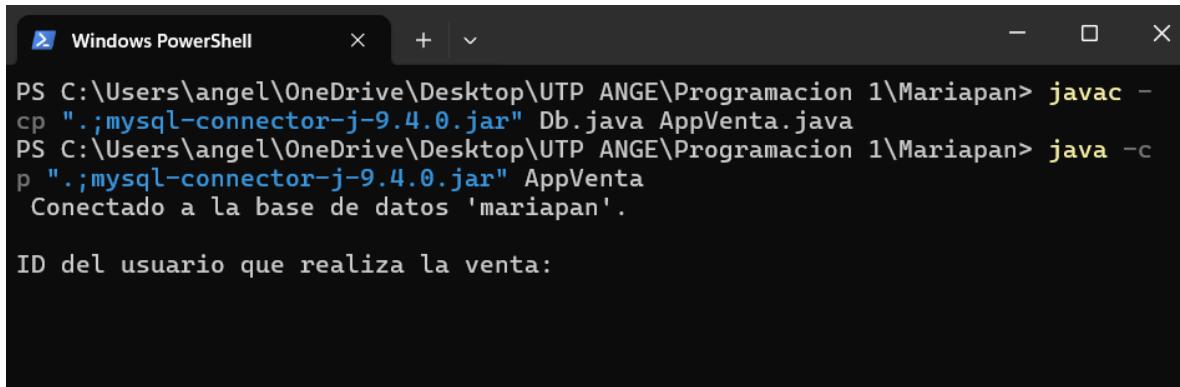
- **id_venta** conecta la tabla **factura** con **venta**.
 - **id_usuario** y **id_cliente** conectan **venta** con las tablas correspondientes.
- Este diagrama valida la estructura relacional del proyecto y resume gráficamente la organización de la base de datos **Mariapan**.

```
J Db.java > ...
1 import java.sql.Connection;
2 import java.sql.DriverManager;
3
4 public class Db {
5     private static final String URL = "jdbc:mysql://192.168.56.101:3306/mariapan?useSSL=false&allowPublicKey
6     private static final String USER = "ange";
7     private static final String PASS = "Angie@2025!";
8
9     public static Connection getConnection() throws Exception {
10         return DriverManager.getConnection(URL, USER, PASS);
11     }
12 }
13
```

En esta imagen se muestra el código fuente de la clase Db.java, encargada de establecer la conexión entre la aplicación Java y el servidor MySQL ubicado en el sistema Ubuntu. Se observa la cadena de conexión JDBC con la dirección IP del servidor (192.168.56.101), el nombre de la base de datos mariapan, y las credenciales de acceso. Este archivo centraliza la configuración de acceso para que la aplicación pueda comunicarse con la base de datos de manera remota y segura.

```
J AppVenta.java > AppVenta > main(String[])
1 import java.sql.Connection;
2 import java.sql.PreparedStatement;
3 import java.sql.ResultSet;
4 import java.sql.Statement;
5 import java.math.BigDecimal;
6 import java.util.Scanner;
7
8 public class AppVenta {
9     Run | Debug
10    public static void main(String[] args) {
11        Scanner sc = new Scanner(System.in);
12
13        try (Connection cn = Db.getConnection()) {
14            System.out.println("Conectado a la base de datos 'mariapan'.\n");
15
16            System.out.print("ID del usuario que realiza la venta: ");
17            int idUsuario = Integer.parseInt(sc.nextLine());
18
19            System.out.print("ID del cliente (o presiona Enter si no hay): ");
20            String clienteInput = sc.nextLine();
21            Integer idCliente = clienteInput.isEmpty() ? null : Integer.parseInt(clienteInput);
22
23            System.out.print("Cantidad de productos en esta venta: ");
24            int numProductos = Integer.parseInt(sc.nextLine());
25
26            BigDecimal subtotal = BigDecimal.ZERO;
27
28            String sqlVenta = "INSERT INTO venta (id_usuario, id_cliente, subtotal, impuestos, total) VALUES (?, ?, 0, 0, 0)";
29            PreparedStatement psVenta = cn.prepareStatement(sqlVenta, Statement.RETURN_GENERATED_KEYS);
30            psVenta.setInt(parameterIndex:1, idUsuario);
31            if (idCliente == null) psVenta.setNull(parameterIndex:2, java.sql.Types.INTEGER);
32            else psVenta.setInt(parameterIndex:2, idCliente);
33            psVenta.executeUpdate();
34
35            ResultSet rs = psVenta.getGeneratedKeys();
36            rs.next();
            int idVenta = rs.getInt(columnIndex:1);
```

El código AppProveedor.java conecta Java con la base de datos mariapan y permite registrar proveedores y productos. El programa solicita al usuario datos como nombre del proveedor, producto, cantidad, precio y sucursal, y los guarda automáticamente en la tabla proveedor mediante sentencias SQL preparadas.



```

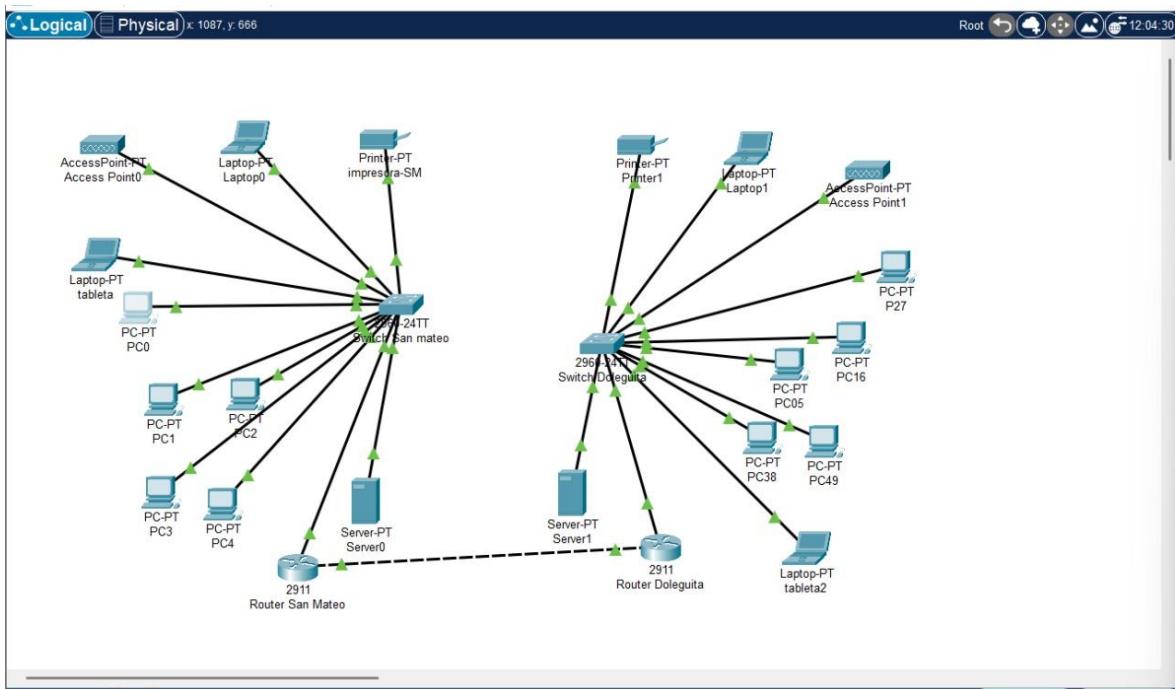
PS C:\Users\angel\OneDrive\Desktop\UTP ANGE\Programacion 1\Mariapan> javac -cp ".;mysql-connector-j-9.4.0.jar" Db.java AppVenta.java
PS C:\Users\angel\OneDrive\Desktop\UTP ANGE\Programacion 1\Mariapan> java -cp ".;mysql-connector-j-9.4.0.jar" AppVenta
Conectado a la base de datos 'mariapan'.

ID del usuario que realiza la venta:

```

El programa AppVenta.java establece la conexión con la base de datos *mariapan* y solicita los datos necesarios para registrar una venta. Inicia mostrando el mensaje de conexión exitosa y luego pide el ID del usuario que realiza la venta, preparando el proceso de ingreso de datos a la tabla correspondiente.

Diseño de red en Cisco Packet Tracer



La imagen muestra el diseño lógico de una red creada en Cisco Packet Tracer, conformada por dos sedes: San Mateo y Doleguita.

Cada sede cuenta con un router, un switch, varios equipos de cómputo, laptops, impresoras, puntos de acceso inalámbrico y un servidor.

Ambas redes están interconectadas mediante un enlace serial entre los routers, simulando la comunicación entre dos sucursales que comparten recursos y servicios dentro de una misma infraestructura.