

T  
R  
A  
B  
A  
J  
O

P  
R  
Á  
C  
T  
I  
C  
O

E  
V  
A  
L  
U  
A  
D  
O  
R



Herrera Vilar, Aida

Desarrollador Web Full-Stack

**Introducción:** Diseñe este esquema de Base de Datos, de manera que las tablas “producto”, “cliente” y “compra” contengan la información necesaria para gestionar los productos, los clientes y las transacciones de compras en una tienda online.

**Descripción de la tarea:** Trabajé en la creación de tablas, inserción de datos, consultas, modifiqué la estructura de la Base de Datos, etc.

A continuación explico por qué lo resolví de esta manera:

En la tabla “**producto**” usé INT para el campo “producto\_id” como clave primaria para identificar cada producto de manera única. El campo “nombre” lo identifiqué como VARCHAR(45) para almacenar el nombre del producto. El campo “precio” lo he definido como DECIMAL(5,2), de ésta manera puedo almacenar valores con 5 dígitos en total y 2 dígitos decimales, pudiendo almacenar precios con centavos. En el campo “stock” también usé INT para almacenar la cantidad de unidades que se encuentran disponibles para cada producto.

En la tabla “**cliente**” usé INT para identificar a cada cliente. VARCHAR(45) lo usé para almacenar el nombre del cliente y también lo usé en el campo correo para almacenar el correo electrónico del cliente y en el campo dirección para almacenar la dirección del cliente.

En la tabla “**compra**” utilicé INT como clave primaria para identificar cada compra. Cantidad la definí como INT para almacenar la cantidad de productos comprados en cada compra. Fecha la definí como DATE para almacenar la fecha de cada compra realizada.

“producto\_id” y “cliente\_id” los definí como claves externas, los mismos hacen referencia a la tabla “producto” y “cliente”, ésto me permite vincular cada compra con el producto y el cliente correspondiente y las claves primarias utilizadas permiten que no haya duplicados.

A Continuación muestro la creación del esquema correspondiente a ésta Base de Datos.



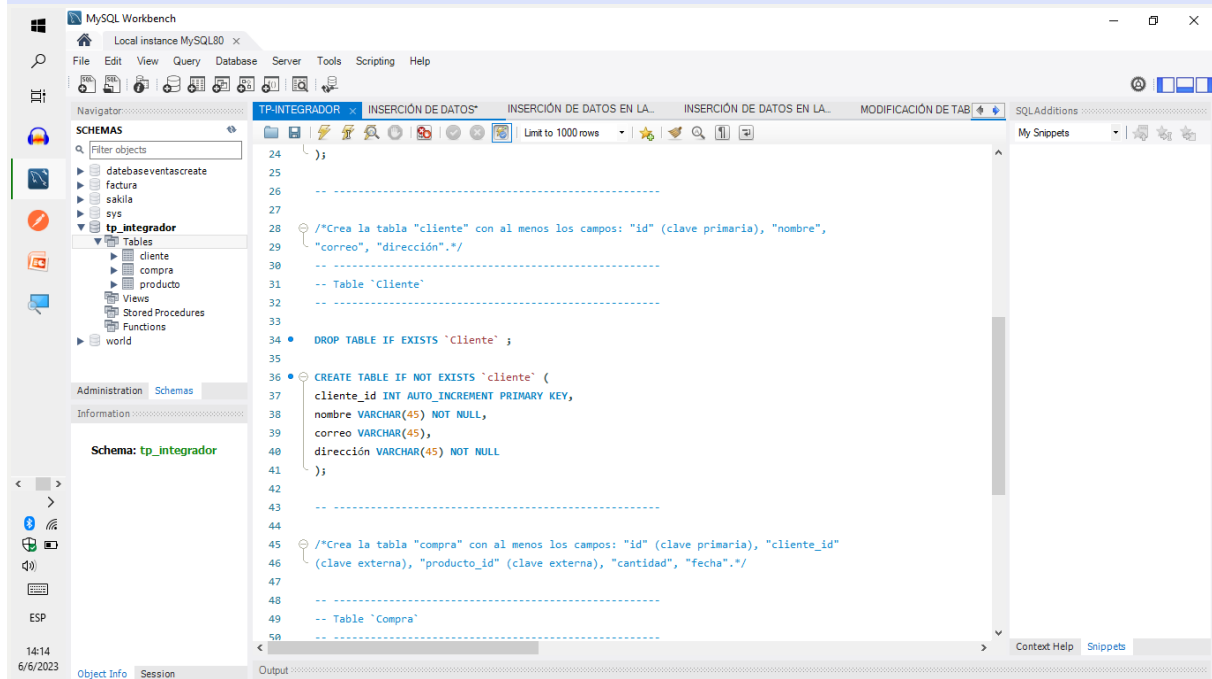
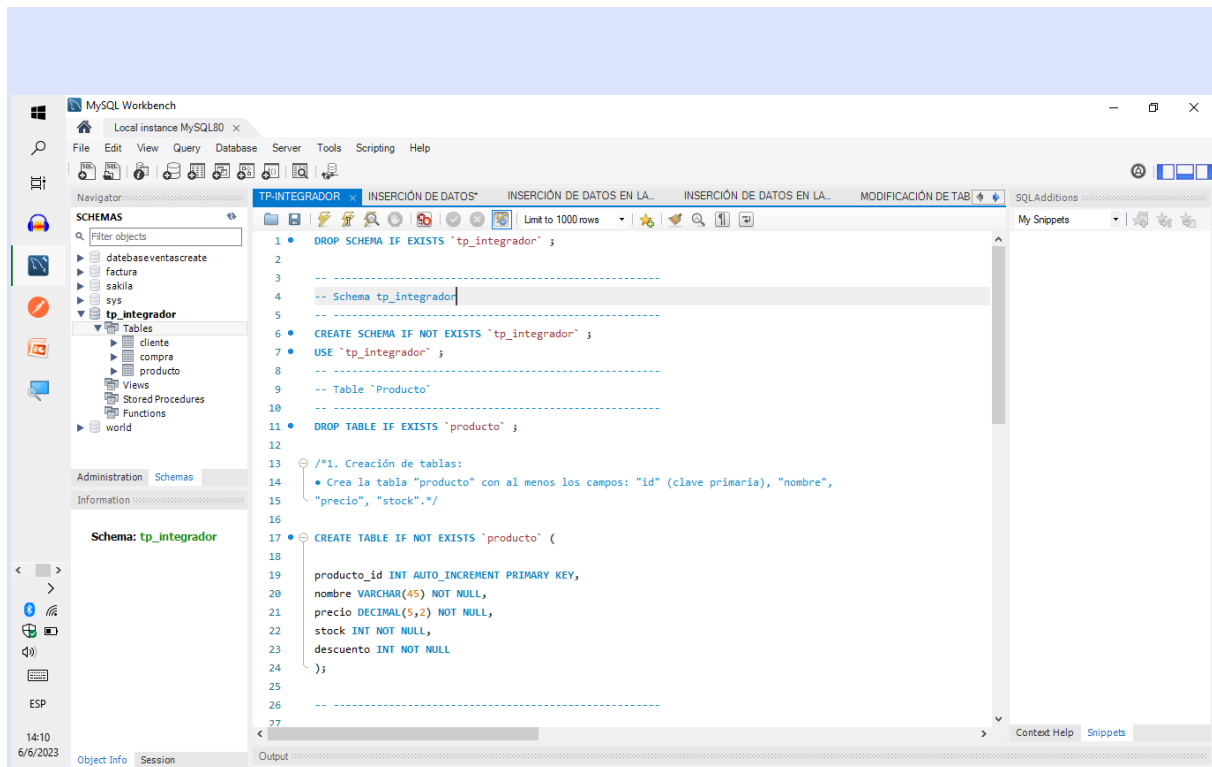
## Actividad

### 1. Creación de tablas

Crea la tabla "Productos" con al menos los campos: "id" (clave primaria), "nombre", "precio", "stock".

- Crea la tabla "Clientes" con al menos los campos: "id" (clave primaria), "nombre", "correo", "dirección".

- Crea la tabla "Compras" con al menos los campos: "id" (clave primaria), "cliente\_id" (clave externa), "producto\_id" (clave externa), "cantidad", "fecha".



MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator: LOCALIZACIÓN DE TABLAS PROCEDIMIENTOS ALMACEN... INSERCIÓN CLIENTES INSERCIÓN PRODUCTOS TP-INTEGRADOR

Schemas: factura sakila sys tp\_integrador

Tables: cliente compra producto

Views: productos\_con\_desc

Stored Procedures: cliente\_orden

Administration Schemas

Information

No object selected

22:21 8/6/2023

```

42
43
44 -- Table 'Compra'
45
46
47 CREATE TABLE IF NOT EXISTS `compra` (
48   compra_id INT AUTO_INCREMENT PRIMARY KEY,
49   cantidad INT NOT NULL,
50   fecha DATE NOT NULL,
51   producto_id INT NOT NULL,
52   cliente_id INT NOT NULL,
53   FOREIGN KEY (producto_id)
54     REFERENCES producto (producto_id)
55     ON DELETE NO ACTION ON UPDATE NO ACTION,
56   FOREIGN KEY (cliente_id)
57     REFERENCES cliente (cliente_id)
58     ON DELETE NO ACTION ON UPDATE NO ACTION
59 );
60
61
62

```

Output

#	Time	Action	Message	Duration / Fetch
24	22:19:11	CREATE TABLE IF NOT EXISTS `compra` ( compra_id INT AUTO_INCREMENT PRI...	0 row(s) affected	0.140 sec
25	22:19:39	INSERT INTO compra (cantidad, fecha, producto_id, cliente_id) VALUES (1, 2023...	54 row(s) affected Records: 54 Duplicates: 0 Warnings: 0	0.031 sec
26	22:20:07	CALL cliente_orden('Ibora', 'yoh_aho_10@hotmail.com', '619 Saavedra')	1 row(s) returned	0.015 sec / 0.000 sec

## 2. Inserción de datos (mínimo 100)

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator: PROCEDIMIENTOS ALMACEN... INSERCIÓN CLIENTES INSERCIÓN PRODUCTOS TP-INTEGRADOR INSERCIÓN COMPRA

Schemas: databaseventascreate factura sakila sys tp\_integrador

Tables: cliente compra producto

Views: productos\_con\_desc

Stored Procedures: cliente\_orden

Administration Schemas

Information

No object selected

00:45 8/6/2023

```

79 (77, 'Sophia Miller', 'sophiamiller@example.com', '456 Maple St, CityV2, CountryV2'),
80 (78, 'Oliver Anderson', 'oliveranderson@example.com', '789 Cedar St, CityZ2, CountryZ2'),
81 (79, 'Ava Thomas', 'avathomas@example.com', '321 Birch St, CityA3, CountryA3'),
82 (80, 'Isabella Garcia', 'isabellagarcia@example.com', '654 Spruce St, CityB3, CountryB3'),
83 (81, 'Mason Davis', 'masondavis@example.com', '987 Ash St, CityC3, CountryC3'),
84 (82, 'Charlotte Wilson', 'charlottewilson@example.com', '654 Elm St, CityD3, CountryD3'),
85 (83, 'James Taylor', 'jamestaylor@example.com', '321 Cedar St, CityE3, CountryE3'),
86 (84, 'Ava Martinez', 'avamartinez@example.com', '789 Birch St, CityF3, CountryF3'),
87 (85, 'Liam Davis', 'liamdavis@example.com', '123 Spruce St, CityG3, CountryG3'),
88 (86, 'Emma Anderson', 'emmaanderson@example.com', '456 Ash St, CityH3, CountryH3'),
89 (87, 'Aiden Garcia', 'aidengarcia@example.com', '789 Oak St, CityI3, CountryI3'),
90 (88, 'Olivia Johnson', 'oliviajohnson@example.com', '123 Pine St, CityJ3, CountryJ3'),
91 (89, 'Lucas Smith', 'lucassmith@example.com', '456 Maple St, CityK3, CountryK3'),
92 (90, 'Sophia Brown', 'sophiabrown@example.com', '789 Cedar St, CityL3, CountryL3'),
93 (91, 'Jackson Martinez', 'jacksonmartinez@example.com', '321 Birch St, CityM3, CountryM3'),
94 (92, 'Ava Davis', 'avadavis@example.com', '654 Spruce St, CityN3, CountryN3'),
95 (93, 'Liam Wilson', 'liamwilson@example.com', '987 Ash St, CityO3, CountryO3'),
96 (94, 'Olivia Johnson', 'oliviajohnson@example.com', '654 Elm St, CityP3, CountryP3'),
97 (95, 'Noah Smith', 'noahsmith@example.com', '321 Cedar St, CityQ3, CountryQ3'),
98 (96, 'Emma Garcia', 'emmagarcia@example.com', '789 Birch St, CityR3, CountryR3'),
99 (97, 'Sophia Miller', 'sophiamiller@example.com', '123 Spruce St, CityS3, CountryS3'),
100 (98, 'Oliver Anderson', 'oliveranderson@example.com', '456 Ash St, CityT3, CountryT3'),
101 (99, 'Ava Thomas', 'avathomas@example.com', '789 Oak St, CityU3, CountryU3'),
102 (100, 'Isabella Garcia', 'isabellagarcia@example.com', '123 Pine St, CityV3, CountryV3');
103
104

```

Output

Action Output

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

databaseventascreate  
factura  
sakila  
sys  
tp\_integrador

Tables

cliente  
compra  
producto  
Views  
Stored Procedures  
Functions  
world

Administration Schemas

Information

Table: cliente

Columns:

cliente\_id int AI PK  
nombre varchar(45)  
correo varchar(45)  
dirección varchar(45)

CLIENTES PRODUCTOS ALTER TABLE... cliente cliente cliente cliente cliente INSECCIÓN PRODUCTOS

1 • SELECT \* FROM tp\_integrador.cliente;

Result Grid

cliente_id	nombre	correo	dirección
1	John Doe	john.doe@example.com	123 Main St, CityA, CountryA
2	Jane Smith	jane.smith@example.com	456 Elm St, CityB, CountryB
3	Michael Johnson	michael.johnson@example.com	789 Oak St, CityC, CountryC
4	Emily Williams	emily.williams@example.com	321 Pine St, CityD, CountryD
5	David Brown	david.brown@example.com	654 Maple St, CityE, CountryE
6	Sarah Davis	sarah.davis@example.com	987 Cedar St, CityF, CountryF
7	Christopher Martinez	christopher.martinez@example.com	654 Birch St, CityG, CountryG
8	Jessica Thompson	jessica.thompson@example.com	321 Spruce St, CityH, CountryH
9	Daniel Rodriguez	daniel.rodriguez@example.com	789 Ash St, CityI, CountryI
10	Jennifer Lee	jennifer.lee@example.com	123 Oak St, CityJ, CountryJ
11	Matthew Wilson	matthew.wilson@example.com	456 Elm St, CityK, CountryK
12	Amanda Anderson	amanda.anderson@example.com	789 Pine St, CityL, CountryL
13	Andrew Thomas	andrew.thomas@example.com	321 Maple St, CityM, CountryM
14	Jessica Garcia	jessica.garcia@example.com	654 Cedar St, CityN, CountryN
15	James Martinez	james.martinez@example.com	987 Birch St, CityO, CountryO

cliente 2 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
✓ 1	23:03:58	SELECT * FROM tp_integrador.cliente LIMIT 0, 1000	100 row(s) returned	0.015 sec / 0.000 sec

Object Info Session

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

databaseventascreate  
factura  
sakila  
sys  
tp\_integrador

Tables

cliente  
compra  
producto  
Views  
Stored Procedures  
productos\_con\_desc

Administration Schemas

Information

No object selected

TOS TP-INTEGRADOR INSECCIÓN COMPRA CONSULTAS (M) PRODUCTOS ALTER TABLE... FUNCIONES cliente

1 • SELECT \* FROM tp\_integrador.cliente;

Result Grid

cliente_id	nombre	correo	dirección
84	Ava Martinez	avamartinez@example.com	789 Birch St, CityF3, CountryF3
85	Liam Davis	liam.davis@example.com	123 Spruce St, CityG3, CountryG3
86	Emma Anderson	emma.anderson@example.com	456 Ash St, CityH3, CountryH3
87	Aiden Garcia	aiden.garcia@example.com	789 Oak St, CityI3, CountryI3
88	Olivia Johnson	olivia.johnson@example.com	123 Pine St, CityJ3, CountryJ3
89	Lucas Smith	lucas.smith@example.com	456 Maple St, CityK3, CountryK3
90	Sophia Brown	sophia.brown@example.com	789 Cedar St, CityL3, CountryL3
91	Jackson Martinez	jackson.martinez@example.com	321 Birch St, CityM3, CountryM3
92	Ava Davis	ava.davis@example.com	654 Spruce St, CityN3, CountryN3
93	Liam Wilson	liam.wilson@example.com	987 Ash St, CityO3, CountryO3
94	Olivia Johnson	olivia.johnson@example.com	654 Elm St, CityP3, CountryP3
95	Noah Smith	noah.smith@example.com	321 Cedar St, CityQ3, CountryQ3
96	Emma Garcia	emma.garcia@example.com	789 Birch St, CityR3, CountryR3
97	Sophia Miller	sophia.miller@example.com	123 Spruce St, CityS3, CountryS3
98	Oliver Anderson	oliver.anderson@example.com	456 Ash St, CityT3, CountryT3
99	Ava Thomas	ava.thomas@example.com	789 Oak St, CityU3, CountryU3
100	Isabella Garcia	isabella.garcia@example.com	123 Pine St, CityV3, CountryV3

cliente 1 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
✓ 1	00:46:06	SELECT * FROM tp_integrador.cliente LIMIT 0, 1000	100 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

The screenshot shows the MySQL Workbench interface. The 'Schemas' pane on the left lists the database structure, including the 'tp\_integrador' database with tables 'cliente', 'compra', and 'producto'. The 'Query' tab is active, showing a SQL query: `SELECT * FROM tp_integrador.producto;`. The 'Result Grid' displays the data from the 'producto' table, limited to 1000 rows. The output shows columns: producto\_id, nombre, precio, stock, and descuento.

producto_id	nombre	precio	stock	descuent
91	Monitor de gaming ultra ancho	499.99	5	0.00
92	Teclado mecánico retroriluminado	99.99	8	0.00
93	Mouse inalámbrico óptico	29.99	12	3.00
94	Proyector HD	199.99	5	0.00
95	Reproductor de CD	49.99	10	0.00
96	Cámara de video profesional	999.99	4	0.00
97	Dispositivo de streaming 4K	99.99	10	0.00
98	Altavoces Bluetooth portátiles	79.99	10	0.00
99	Impresora multifuncional láser	199.99	6	0.00
100	Monitor de alta resolución	299.99	5	0.00

The 'Output' pane at the bottom shows the execution details: `SELECT * FROM tp_integrador.producto LIMIT 0, 1000` returned 100 row(s) in 0.016 seconds.

The screenshot shows the MySQL Workbench interface. The 'Schemas' pane on the left lists the database structure, including the 'tp\_integrador' database with tables 'cliente', 'compra', and 'producto'. The 'Query' tab is active, showing a SQL query: `SELECT * FROM tp_integrador.compra;`. The 'Result Grid' displays the data from the 'compra' table, limited to 1000 rows. The output shows columns: compra\_id, cantidad, fecha, producto\_id, and cliente\_id.

compra_id	cantidad	fecha	producto_id	cliente_id
99	1	2023-07-21	50	65
100	2	2023-07-22	5	85
101	3	2023-07-23	75	30
102	4	2023-07-24	15	60
103	1	2023-07-25	35	95
104	2	2023-07-26	90	55
105	3	2023-07-27	25	10
106	4	2023-07-28	70	80
107	1	2023-07-29	40	20
108	2	2023-07-30	5	75

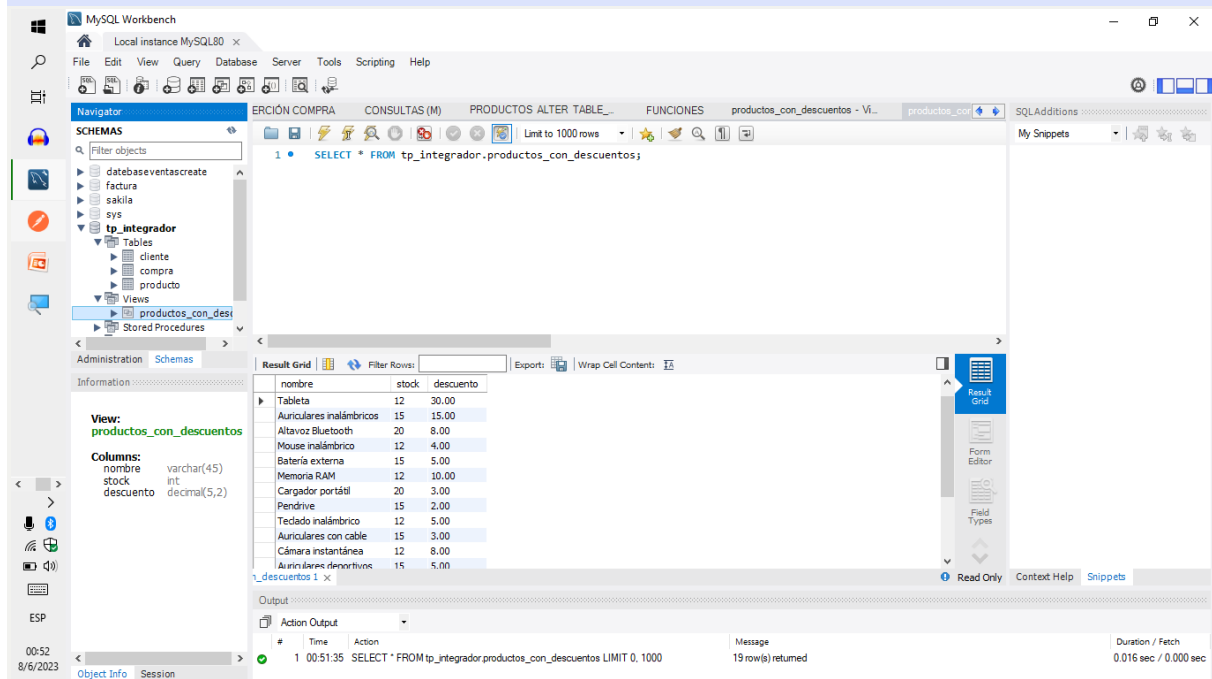
The 'Output' pane at the bottom shows the execution details: `SELECT * FROM tp_integrador.compra LIMIT 0, 1000` returned 108 row(s) in 0.000 seconds.

### 3. Modificación de tablas

- Agrega una columna "descuento" a la tabla "Productos" utilizando ALTER TABLE.

- Modifica el tipo de datos de la columna "precio" en la tabla "Productos" utilizando ALTER COLUMN.

Usé ALTER TABLE para agregar una columna nueva (“descuento”), en este caso a la tabla “producto” mediante ADD COLUMN, lo que me permite ingresar nuevos datos sin modificar los datos existentes.

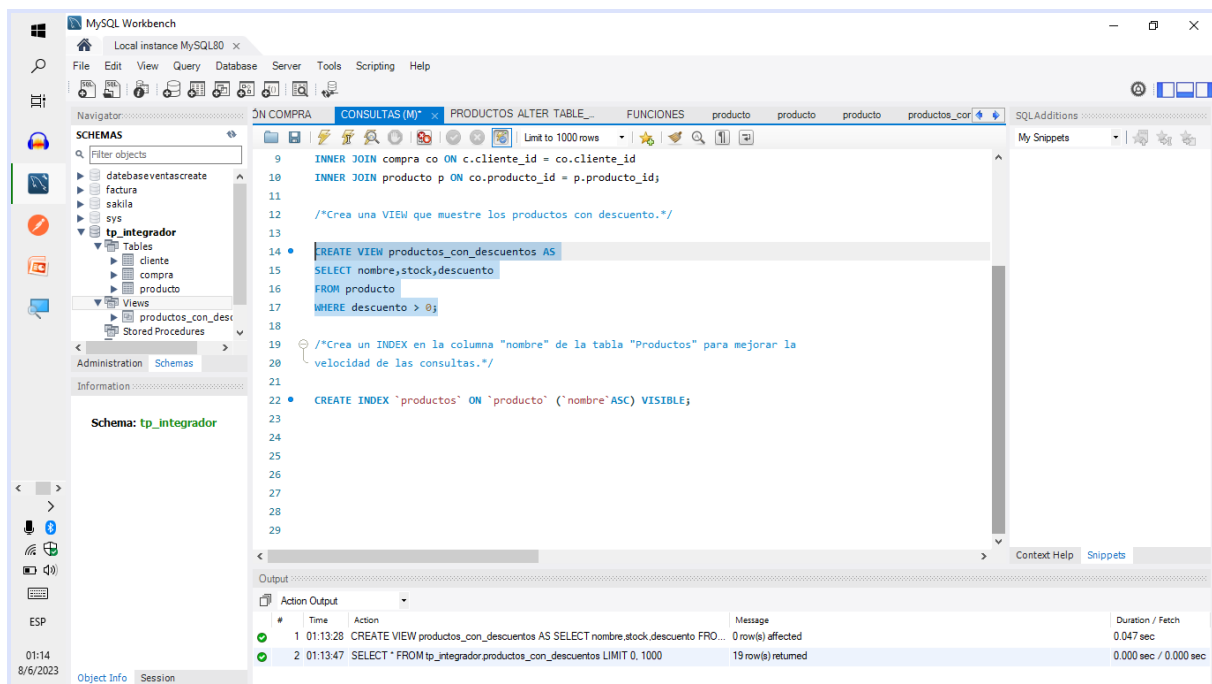


## 4. Consultas

- Realiza una consulta utilizando SELECT JOIN para obtener la información de los productos comprados por cada cliente.

Acá usé JOIN para combinar las tablas “compra”, “cliente” y “producto” utilizando las columnas “cliente\_id” y “producto\_id”. JOIN combina las filas de las tablas relacionadas. Lo usé en las tablas “cliente” y “producto” para obtener el nombre del cliente correspondiente con cada producto en la tabla compra.

Cuando realice una operación va a mostrar la información de los productos comprados por cada cliente, mostrando el nombre del cliente y del producto, la cantidad y la fecha de compra.

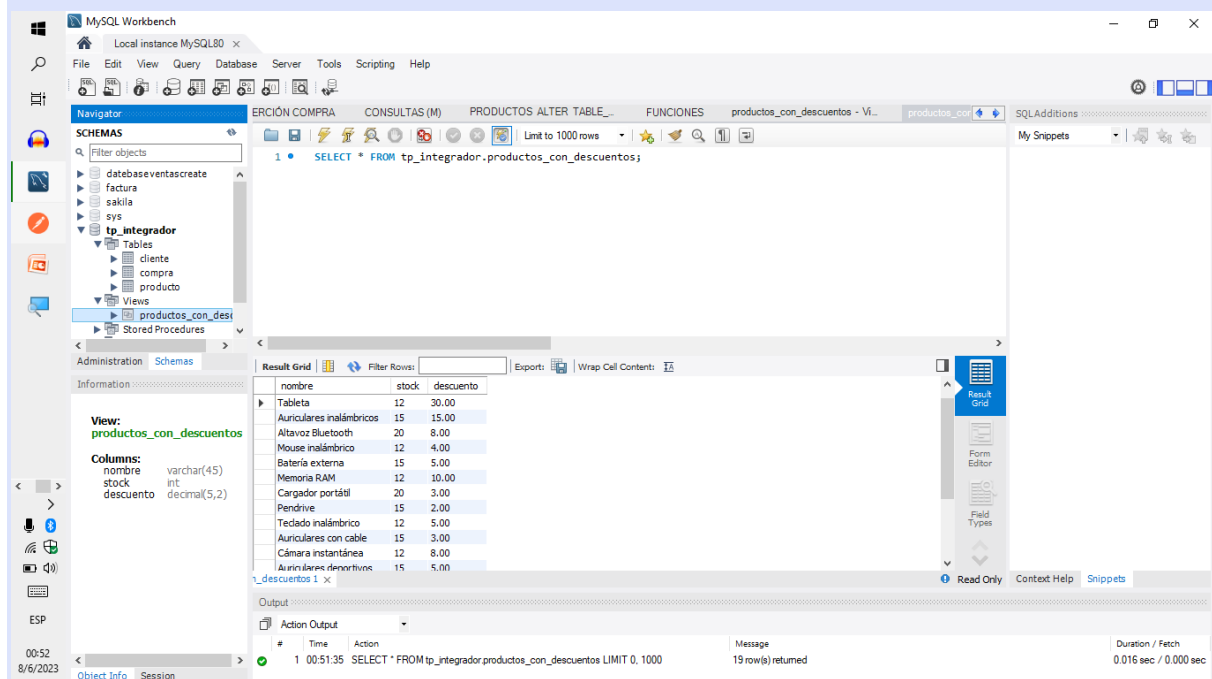


## 4. Consultas:

Crea una VIEW que muestre los productos con descuento.

Una VIEW me permite mostrar los datos de una o varias tablas. En éste caso la cree para que muestre únicamente los productos con descuento.

Lo resolví utilizando SELECT CREATE VIEW para seleccionar las columnas de la tabla "producto" y WHERE lo utilicé para filtrar sólo los productos que tengan un descuento mayor a 0, de ésta manera con VIEW puedo mostrar únicamente los productos con descuentos.



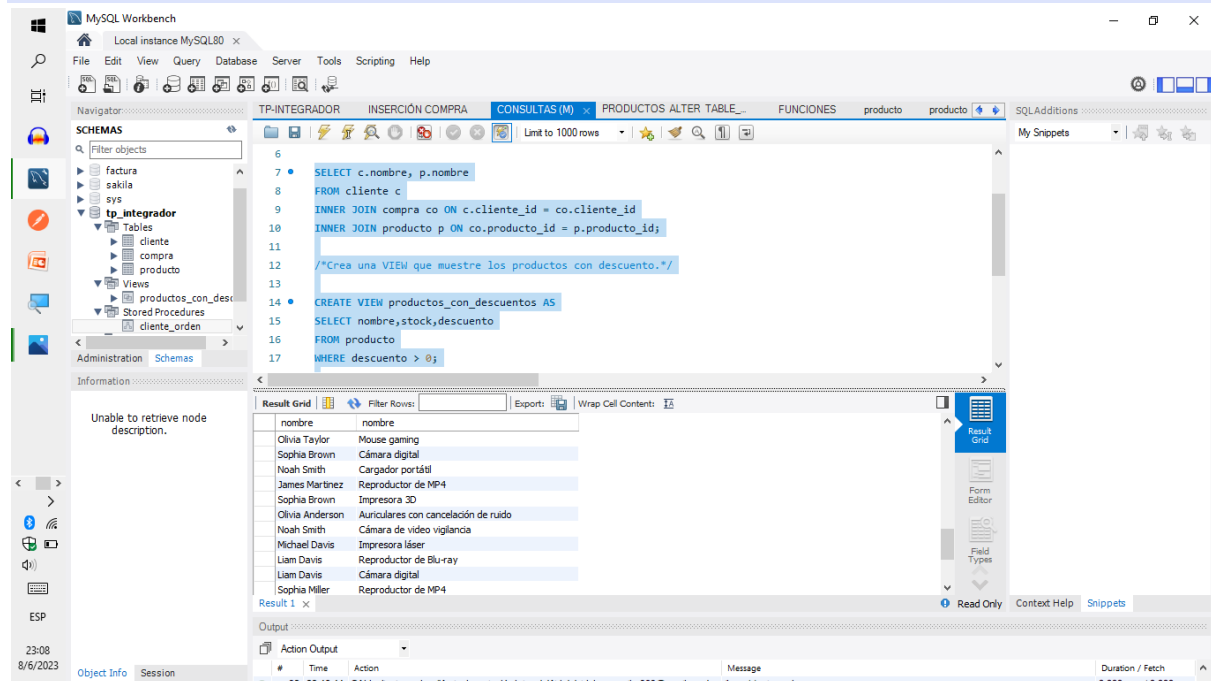
## 4. Consultas



Crea un INDEX en la columna "nombre" de la tabla "Productos" para mejorar la velocidad de las consultas.

Lo resolví de la siguiente manera: `CREATE INDEX `productos` ON `producto` (`nombre`ASC) VISIBLE;`

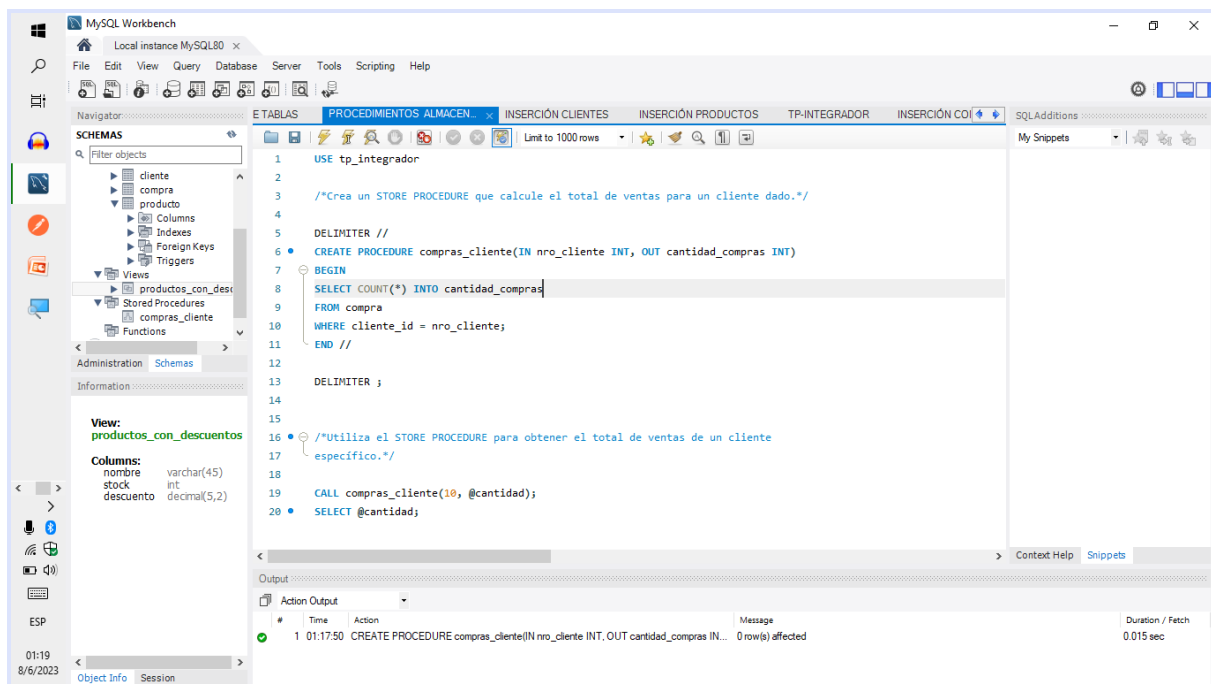
INDEX es el nombre del índice, productos es el nombre de la tabla donde voy a crear el índice y nombre es la columna donde se va a crear el índice. El índice se creó en la columna nombre de la tabla "producto", el cual permitirá mejorar la velocidad de las consultas.



## 5. Procedimientos almacenados

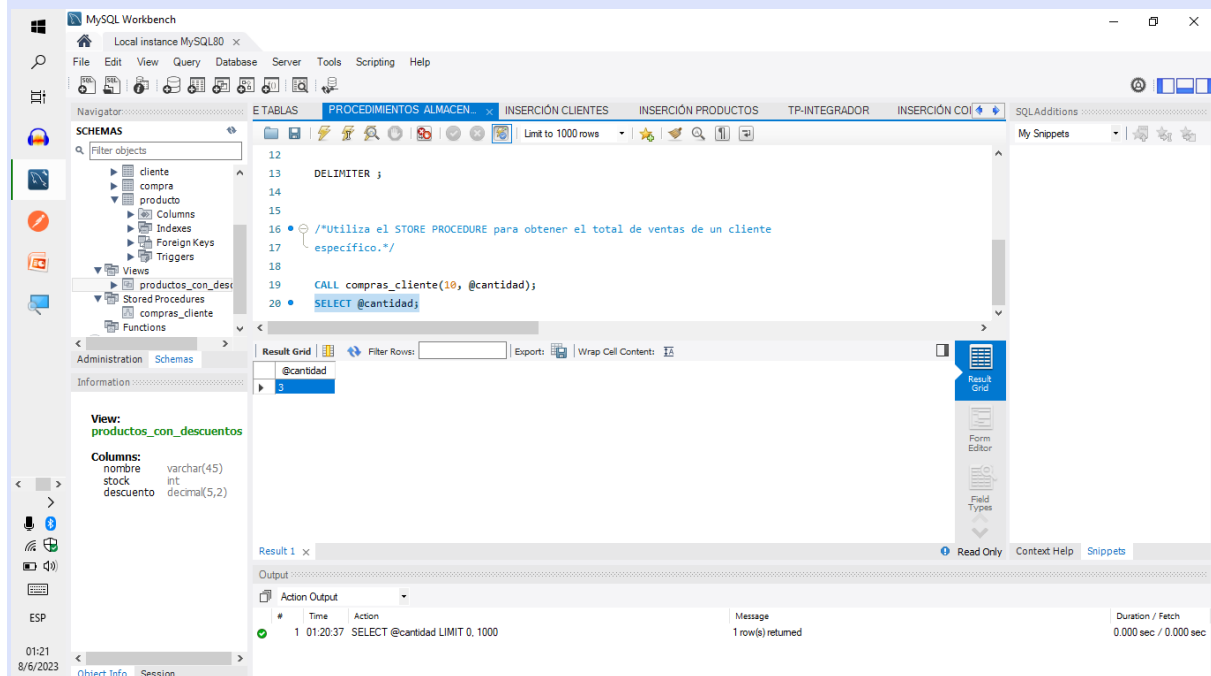
Crea un STORE PROCEDURE que calcule el total de ventas para un cliente dado.

Para resolverlo utilicé un parámetro de entrada "nro\_cliente" para identificar el número de clientes de los cuales voy a obtener la cantidad de compras realizadas. Usé la variable de salida "cantidad\_compras" para almacenar la cuenta total de compras. Al utilizar esta variable, el valor que calculo puede ser recuperado luego de llamar al stored procedure. El `SELECT COUNT (*) INTO cantidad_compras` para contar los registros en la tabla "compra" que cumplen con la condición "cliente\_id". Con esto voy a obtener la cantidad de compras realizadas por un cliente específico.



Utiliza el STORE PROCEDURE para obtener el total de ventas de un cliente específico.

Acá llamo al STORED PROCEDURE CALL y le asigno un número de cliente, en éste caso se lo asigné al cliente 10 y el resultado lo veré reflejado en “cantidad\_compras” (variable). El SELECT lo uso para recuperar el resultado de “cantidad\_compras”. En éste caso va a tener el total de ventas del cliente 10, que fue el seleccionado.

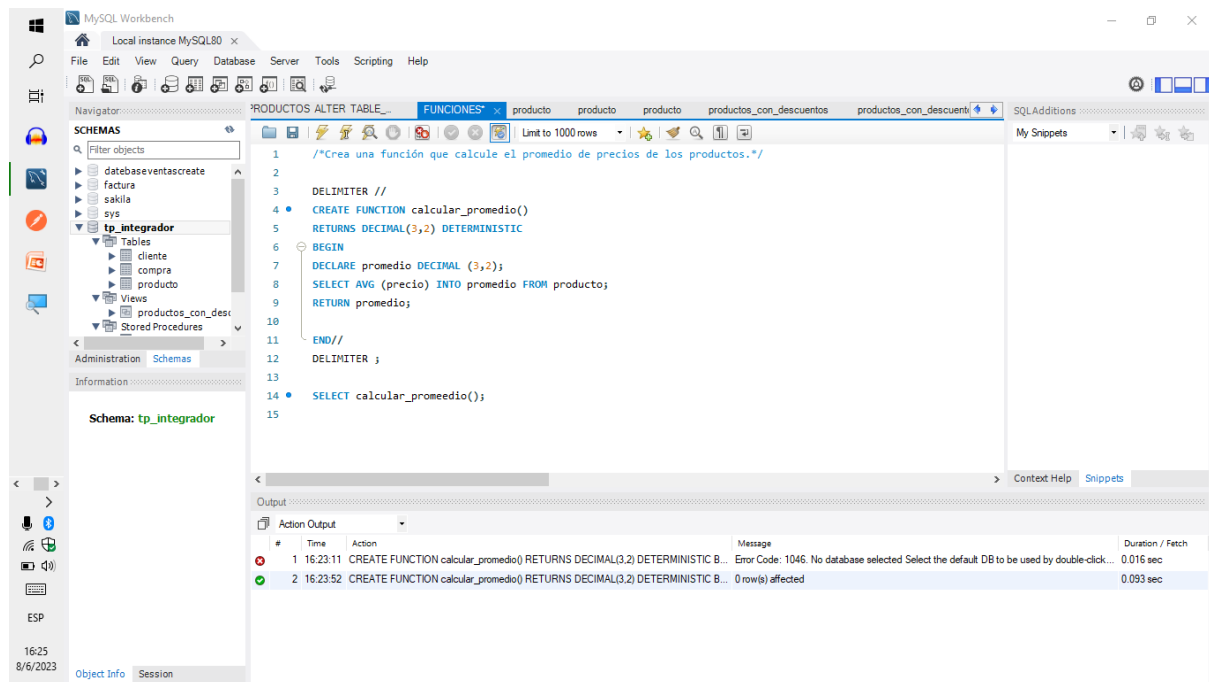


## 6. Funciones:

- Crea una función que calcule el promedio de precios de los productos.

El nombre de la función es “calcular\_promedio”, usé una declaración RETURNS DECIMAL (5,2), tengo la variable promedio de tipo DECIMAL (5,2), la cual va a almacenar el resultado del cálculo del promedio. El SELECT AVG(precio) INTO promedio lo usé para calcular el promedio de los precios de la columna precios. El RETURN me devuelve el valor del promedio.

DETERMINISTIC hace que la función devuelva siempre el mismo resultado y por último llamo a la función de la siguiente manera: SELECT calcular\_promedio(); al ejecutarla me va a mostrar el resultado del promedio de precios de los productos.



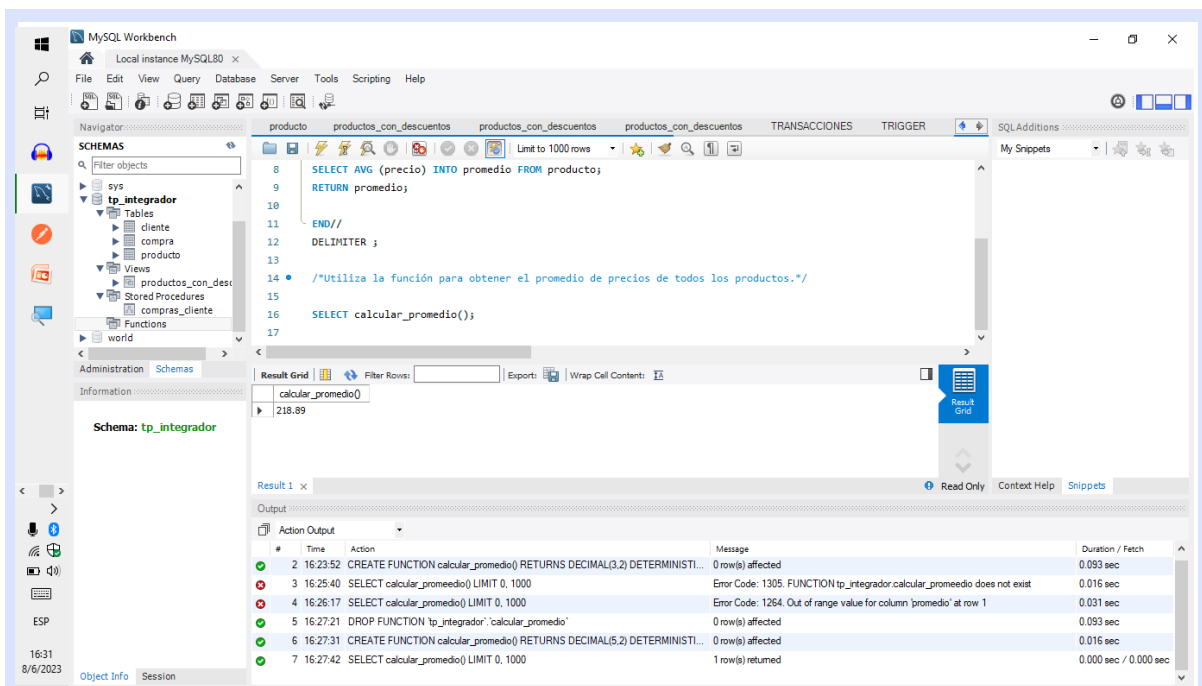
The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with the 'tp\_integrador' schema selected. The main editor window shows the following SQL code:

```
1 /*Crea una función que calcule el promedio de precios de los productos.*/
2
3 DELIMITER //
4 CREATE FUNCTION calcular_promedio()
5 RETURNS DECIMAL(3,2) DETERMINISTIC
6 BEGIN
7 DECLARE promedio DECIMAL (3,2);
8 SELECT AVG (precio) INTO promedio FROM producto;
9 RETURN promedio;
10
11 END//
12 DELIMITER ;
13
14 SELECT calcular_promedio();
15
```

The 'Output' panel at the bottom shows the execution results:

#	Time	Action	Message	Duration / Fetch
1	16:23:11	CREATE FUNCTION calcular_promedio() RETURNS DECIMAL(3,2) DETERMINISTIC B...	Error Code: 1046. No database selected Select the default DB to be used by double click...	0.016 sec
2	16:23:52	CREATE FUNCTION calcular_promedio() RETURNS DECIMAL(3,2) DETERMINISTIC B...	0 row(s) affected	0.093 sec

- Utiliza la función para obtener el promedio de precios de todos los productos.



## 7. Transacciones:

- Crea una transacción que inserte un nuevo cliente y una nueva orden de compra al mismo tiempo.

En esta transacción puedo insertar un nuevo cliente y una nueva orden de compra al mismo tiempo.

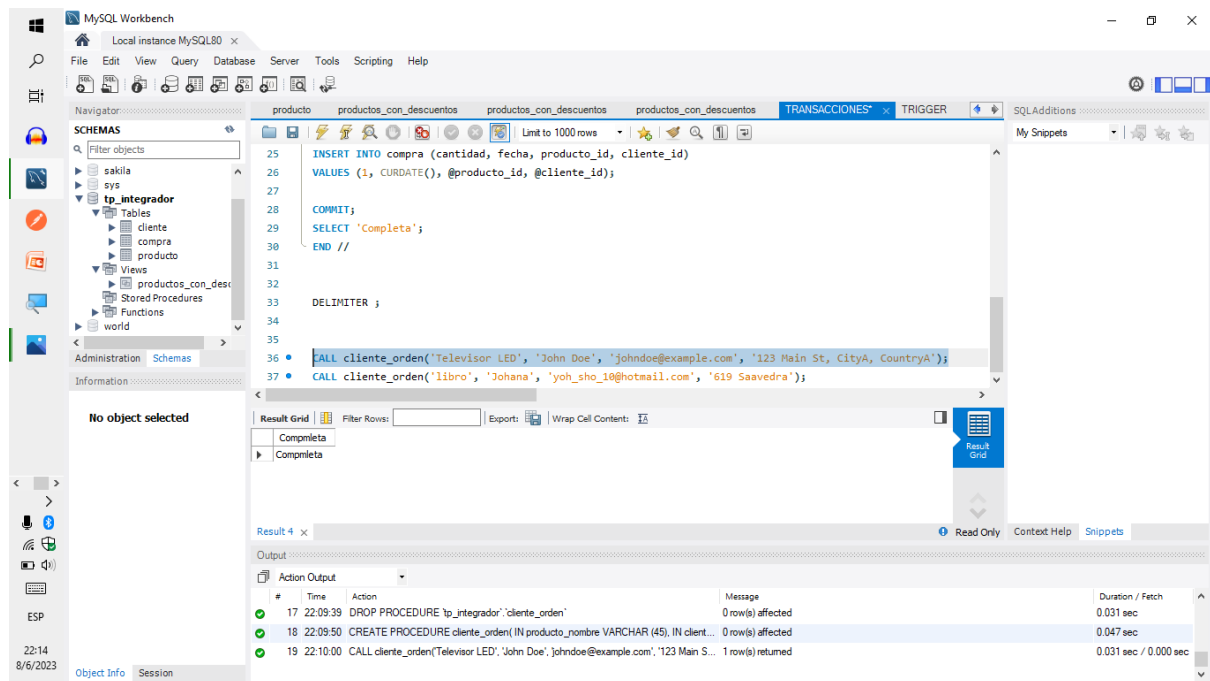
**START TRANSACTION** indica el inicio de la unidad de trabajo, con **INSERT INTO** inserto el nuevo cliente en la tabla “cliente”, **MAX** lo uso para obtener el id máximo en la tabla “cliente” y se incrementa en 1 para asignarle un nuevo id al cliente que estoy insertando, el cual lo almaceno en la variable “id\_cliente”.

Luego realizo una consulta para obtener el id del producto de la tabla “producto”, el cual tiene que coincidir con el nombre del producto pasado como parámetro. El id de producto se almacena en la variable “@producto\_id”, luego hago una inserción de una nueva orden de compra en la tabla “compra”, en este caso con un valor 1, la fecha actual **CURDATE()**, el id del producto obtenido anteriormente y el id del cliente que inserté.

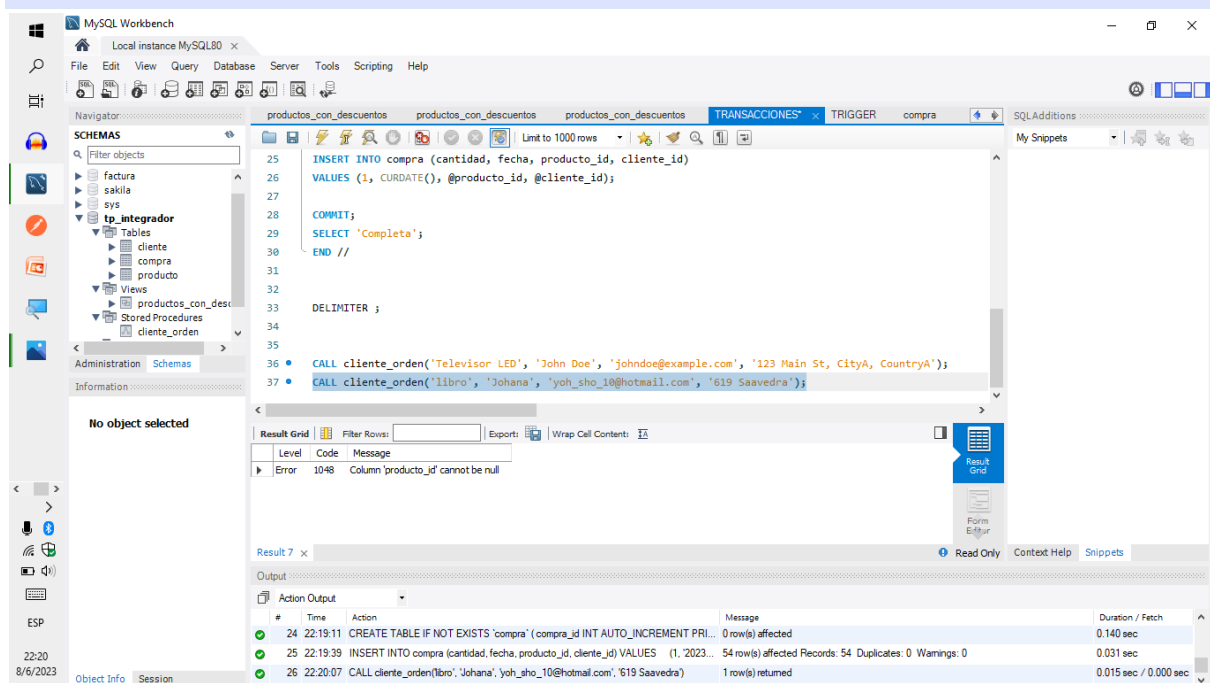
Si hay algún error utilizo **SQLEXCEPTION**, esto captura la excepción y muestra los errores con **SHOW ERRORS**. **ROLLBACK** deshace los cambios realizados.

**COMMIT** lo usé en el caso de que no haya errores y me va a mostrar el mensaje “completa”, lo cual indica que la transacción se realizó correctamente.

- Transacción ‘completa’



- Asegúrate de que la transacción se ejecute correctamente y se haga un rollback en caso de error.



## 8. Triggers:

- Crea un TRIGGER que actualice el stock de un producto después de realizar una orden de compra.

## Último cliente

MySQL Workbench interface showing the 'cliente' table in the 'tp\_integrador' database. The table structure is as follows:

cliente_id	nombre	correo	dirección
97	Sophia Miller	sophiamiller@example.com	123 Spruce St, City63, Countr...
98	Oliver Anderson	oliveranderson@example.com	456 Ash St, CityT3, CountryU3
99	Ava Thomas	avathomas@example.com	789 Oak St, CityU3, CountryU3
100	Isabella Garcia	isabellagarcia@example.com	123 Pine St, CityV3, CountryV3
103	John Doe	johndoe@example.com	123 Main St, CityA, CountryA
104	John Doe	johndoe@example.com	123 Main St, CityA, CountryA
105	Johana	yoh_sho_10@hotmail.com	619 Saavedra

## Última compra

MySQL Workbench interface showing the 'compra' table in the 'tp\_integrador' database. The table structure is as follows:

compra_id	cantidad	fecha	producto_id	cliente_id
48	4	2023-07-24	15	60
49	1	2023-07-25	35	95
50	2	2023-07-26	90	55
51	3	2023-07-27	25	10
52	4	2023-07-28	70	80
53	1	2023-07-29	40	20
54	2	2023-07-30	5	75

## •Stock actual

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHMAS

Filter objects

factura sakila sys tp\_integrador

Tables

cliente compra producto

Views

productos\_con\_descuentos

Stored Procedures

cliente\_order

Administration Schemas

Information

Table: producto

Columns:

producto\_id int PK

nombre varchar(45)

precio decimal(5,2)

stock int

descuento decimal(5,2)

Result Grid

producto_id	nombre	precio	stock	descuento
3	Teléfono inteligente	799.99	8	0.00
4	Tableta	299.99	12	30.00
5	Cámara digital	399.99	6	0.00
6	Auriculares inalámbricos	149.99	15	15.00
7	Altavoz Bluetooth	79.99	20	8.00
8	Smartwatch	199.99	10	0.00
9	Consola de videojuegos	499.99	3	0.00
10	Impresora láser	199.99	7	0.00

producto 1 x

Apply Revert Context Help Snippets

Output

Action Output

# Time Action Message Duration / Fetch

Verifica que el TRIGGER se dispare correctamente y actualice el stock de manera adecuada.

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHMAS

Filter objects

factura sakila sys tp\_integrador

Tables

cliente compra producto

Views

productos\_con\_descuentos

Stored Procedures

cliente\_order

Administration Schemas

Information

Table: producto

Columns:

producto\_id int PK

nombre varchar(45)

precio decimal(5,2)

stock int

descuento decimal(5,2)

Result Grid

Level	Code	Message
Error	1048	Column 'producto_id' cannot be null

Result 5 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
20	17:27:46	DROP TRIGGER actualizar_stock; CREATE TRIGGER actualizar_stock AFTER INS...	Error Code: 1360. Trigger does not exist	0.000 sec
21	17:28:35	CREATE TRIGGER actualizar_stock AFTER INSERT ON compra FOR EACH ROW ...	Error Code: 1415. Not allowed to return a result set from a trigger	0.000 sec
22	17:29:18	CREATE TRIGGER actualizar_stock AFTER INSERT ON compra FOR EACH ROW ...	0 row(s) affected	0.078 sec
23	17:29:27	CALL cliente_order ('Auriculares inalámbricos', 'Aida', 'aidaherreravilar383@gmail.com'...	1 row(s) returned	0.016 sec / 0.000 sec
24	17:29:34	SELECT * FROM tp_integrador.compra LIMIT 0, 1000	54 row(s) returned	0.000 sec / 0.000 sec
25	17:29:41	SELECT * FROM tp_integrador.producto LIMIT 0, 1000	100 row(s) returned	0.000 sec / 0.000 sec
26	17:36:30	DROP TRIGGER actualizar_stock; CREATE TRIGGER actualizar_stock AFTER INS...	0 row(s) affected	0.079 sec
27	17:36:45	CALL cliente_order ('Auriculares inalámbricos', 'Aida', 'aidaherreravilar383@gmail.com'...	1 row(s) returned	0.016 sec / 0.000 sec



