



INFORME DE GUÍA PRÁCTICA

I. PORTADA

Tema:	Ejercicios Subconsultas
Unidad de Organización Curricular:	PROFESIONAL
Nivel y Paralelo:	5 ^{to} – ‘A’
Alumnos participantes:	Ases Tiban Jeremy Damian Palate Moreta Kevin Damian Poveda Gomez William Alberto Pullupaxi Chango Daniel
Asignatura:	Sistemas de Bases de Datos Distribuidas
Docente:	Ing. Ruben Caiza Mg.

II. INFORME DE GUÍA PRÁCTICA

2.1 Objetivos

General:

Diseñar una base de datos denominada Tienda en PostgreSQL para aplicar consultas y subconsultas.

Específicos:

- Crear una base de datos Tienda en PostgreSQL.
- Crear las tablas de la base de datos Tienda.
- Elaborar consultas y subconsultas sobre los ejercicios planteados

2.2 Modalidad

Presencial.

2.3 Tiempo de duración

Presenciales: 3 horas.

No presenciales: N/A.

2.4 Instrucciones

- Crear la base de datos Tienda en PostgreSQL.
- Definir y crear las tablas necesarias.
- Insertar datos de prueba en las tablas.
- Ejecutar consultas y subconsultas según los ejercicios planteados.
- Verificar los resultados obtenidos en cada consulta.

2.5 Listado de equipos, materiales y recursos

- Computadora.
- PostgreSQL

TAC (Tecnologías para el Aprendizaje y Conocimiento) empleados en la guía práctica:

- ☐ Plataformas educativas
- ☐ Simuladores y laboratorios virtuales
- ☒ Aplicaciones educativas
- ☐ Recursos audiovisuales
- ☐ Gamificación
- ☒ Inteligencia Artificial

Otros (Especifique): _____



2.6 Actividades desarrolladas

Creación de la Base de Datos: tienda_almacen

```
-- DROP DATABASE IF EXISTS tienda_almacen;
```

```
CREATE DATABASE tienda_almacen  
WITH  
OWNER = postgres  
ENCODING = 'UTF8'  
LC_COLLATE = 'Spanish_Ecuador.1252'  
LC_CTYPE = 'Spanish_Ecuador.1252'  
LOCALE_PROVIDER = 'libc'  
TABLESPACE = pg_default  
CONNECTION LIMIT = -1  
IS_TEMPLATE = False;
```

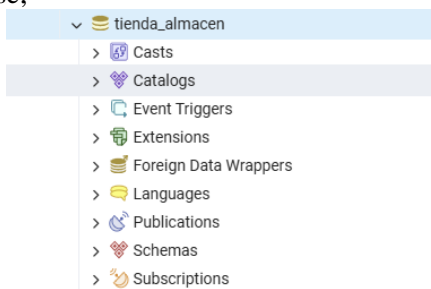


Fig. 1: Base de Datos en PostgreSQL

Creación de las tablas:

```
create table category (  
category_id int primary key,  
name varchar(60) not null,  
description text,  
parent_category_id int references category(category_id)  
);
```

```
create table product (  
product_id int primary key,  
product_name varchar(40) not null,  
category_id int not null references category(category_id),  
quantity_per_unit int,  
unit_price decimal(10,2),  
units_in_stock int,  
discontinued boolean  
);
```

```
create table employee (  
employee_id int primary key,  
last_name varchar(40) not null,  
first_name varchar(20) not null,  
birth_date date,  
hire_date date,  
address varchar(128),  
city varchar(30),  
country varchar(30),  
reports_to int references employee(employee_id)  
);
```



```
create table customer (  
customer_id int primary key,  
contact_name varchar(30) not null,  
company_name varchar(40),  
contact_email varchar(128) not null,  
address varchar(120),  
city varchar(30),  
country varchar(30)  
);
```

```
create table purchase (  
purchase_id int primary key,  
customer_id int not null references customer(customer_id),  
employee_id int not null references employee(employee_id),  
total_price decimal(10,2) not null,  
purchase_date timestamp,  
shipped_date timestamp,  
ship_address varchar(60),  
ship_city varchar(15),  
ship_country varchar(15)  
);
```

```
create table purchase_item (  
purchase_id int not null references purchase(purchase_id),  
product_id int not null references product(product_id),  
unit_price decimal(10,2) not null,  
quantity int not null  
);
```

Tables (6)	
>	category
>	customer
>	employee
>	product
>	purchase
>	purchase_item

Fig. 2: Tablas de la base de datos tienda_almacen

CONSULTAS Y SUBCONSULTAS

Ejercicio 10: Calcular el porcentaje que gasta el cliente en cada compra

Para cada cliente que realizó al menos una compra, muestre el ID de cada compra y el porcentaje del dinero gastado en esa compra con respecto al total. Redondee los porcentajes a números enteros. Muestre tres columnas: contact_name, purchase_id y percentage.

```
select c.contact_name, p.purchase_id,  
round(p.total_price*100/(select sum(total_price)  
from purchase),0) as percentage  
from customer c  
join purchase p on c.customer_id = p.customer_id;
```



Query

Query History

```
1 -- Ejercicio 10
2 select c.contact_name, p.purchase_id,
3 round(p.total_price*100/(select sum(total_price)
4 from purchase),0) as percentage
5 from customer c
6 join purchase p on c.customer_id = p.customer_id;
7
```

Data Output

Messages

Notifications

SQL

	contact_name character varying (30)	purchase_id integer	percentage numeric
1	Martín Ruiz	1	23
2	Elena Torres	2	16
3	Jorge Vidal	3	1
4	Sofía Castro	4	0
5	David Ramos	5	23
6	Lorena Gómez	6	3
7	Hugo Salazar	7	12
8	Carla Ibáñez	8	1
9	Felipe Mendoza	9	0
10	Andrea Ríos	10	0
11	Pablo Giménez	11	0
12	Laura Soto	12	0
13	Roberto Pinto	13	4
14	Diana Castillo	14	6
15	Gabriel Orozco	15	0
16	Mónica López	16	0
17	Ricardo Paz	17	7
18	Ana Pérez	18	2
19	Juan Molina	19	2
20	Isabel Ramos	20	0

Total rows: 20

Query complete 00:00:00.113

Fig. 3: Resultado Subconsulta Ej. 10

Discusión

En esta consulta se combina columnas de distintas tablas para generar un resultado. La tabla muestra solo algunas columnas (contact_name, purchase_id, porcentaje), y se relaciona con la fragmentación vertical, ya que se dividen los atributos de cada tabla para enfocarse en los datos importantes.

Ejercicio 11: Encuentra el número de productos caros en cada categoría

Muestra los nombres de las categorías y la cantidad de productos de esta categoría cuyo precio unitario es mayor que el precio promedio de un producto de esta categoría.

Muestra solo las categorías que contienen dichos productos. Muestra dos columnas: nombre (el nombre de la categoría) y expensive_productscantidad de productos que cuestan más que el precio promedio de un producto de esta categoría.

```
SELECT
    c.name,
    COUNT(p.product_id) AS expensive_products_cantidad
FROM
    product AS p
JOIN
    category AS c ON p.category_id = c.category_id
WHERE
    p.unit_price > (
        SELECT AVG(unit_price)
        FROM product AS p2
        WHERE p2.category_id = p.category_id
    )
GROUP BY
    c.name
HAVING
    COUNT(p.product_id) > 0;
```



Query











Query History

```
-- Ejercicio 11
SELECT
  c.name,
  COUNT(p.product_id) AS expensive_products_cantidad
FROM
  product AS p
JOIN
  category AS c ON p.category_id = c.category_id
WHERE
  p.unit_price > (
    SELECT AVG(unit_price)
    FROM product AS p2
    WHERE p2.category_id = p.category_id
  )
GROUP BY
  c.name
HAVING
  COUNT(p.product_id) > 0;
```

Data Output

Messages

Notifications



	name character varying (60)	expensive_products_cantidad bigint
1	Audio	1
2	Computadoras	1
3	Frutas y Verduras	1
4	Herramientas	1
5	Lácteos	1
6	Limpieza	1
7	Muebles	1

Fig. 4: Resultado Subconsulta Ej. 11

Discusión

En esta consulta se identifican las categorías que tienen productos con precios superiores al promedio de su propia categoría. Se usa una subconsulta para calcular ese promedio y luego se filtran los productos que lo superan. Esto se relaciona con la fragmentación horizontal, ya que se seleccionan únicamente las filas que cumplen con el criterio definido.

Ejercicio 12: Mostrar los productos comprados con su cantidad máxima comprada

Para cada producto adquirido, muestre su nombre, la cantidad máxima comprada y el número de compras realizadas. Muestre tres columnas: product_name, quantity, y purchases_number.

```
SELECT
  p.product_name,
  MAX(pi.quantity) AS quantity,
  COUNT(DISTINCT pi.purchase_id) AS purchases_number
FROM product p
JOIN purchase_item pi ON p.product_id = pi.product_id
GROUP BY p.product_name;
```



Query

Query History

1

-- Ejercicio 12

2

SELECT

3

p.product_name,

4

MAX(pi.quantity) AS quantity,

5

COUNT(DISTINCT pi.purchase_id) AS purchases_number

6

FROM product p

7

JOIN purchase_item pi ON p.product_id = pi.product_id

8

GROUP BY p.product_name;

Data Output

Messages

Notifications

SQL

	product_name character varying (40)	quantity integer	purchases_number bigint
1	Altavoz Bluetooth	1	1
2	Aspiradora robot	1	1
3	Auriculares con cancelación de ruido	1	1
4	Brócoli fresco	1	1
5	Cafetera Espresso	1	1
6	Filete de salmón	1	1
7	Laptop UltraPro	1	2
8	Leche entera 1L	1	1
9	Limpiador Multiusos	1	1
10	Manzanas Red	1	1
11	Mesa de centro	1	1
12	Monitor curvo 27"	1	1
13	Mouse inalámbrico	1	1
14	Set de destornilladores	1	1
15	Sierra eléctrica	1	1
16	Silla Ergonómica de Oficina	1	1
17	Smartphone X1	1	1
18	Snack de avena	1	1
19	Televisor SmartTV 4K	1	1
20	Yogur natural	1	1

Total rows: 20

Query complete 00:00:00.122

Fig. 5: Resultado Subconsulta Ej. 12

Discusión

Esta consulta aplica fragmentación vertical, ya que de las tablas product y purchase_item solo se usan atributos específicos (product_name, quantity, purchase_id) en lugar de todos los campos. Esto optimiza el acceso y concentra la información relevante.

Ejercicio 13: Enumere los productos discontinuados, continuados y totales en cada categoría

Para cada categoría, mostrar:

- Su nombre.
- La cantidad de productos discontinuados (es decir, que ya no están disponibles) en esta categoría (nombre esta columna discontinued_products).
- El número de productos continuados (es decir, actualmente disponibles) en esta categoría (nombre esta columna continued_products).
- El número de todos los productos en esta categoría (nombre esta columna all_products).

```
SELECT
    c.name AS category_name,
    (SELECT COUNT(*)
     FROM product p
     WHERE p.category_id = c.category_id
          AND p.discontinued = TRUE) AS discontinued_products,
    (SELECT COUNT(*)
     FROM product p)
```



```
WHERE p.category_id = c.category_id
AND p.discontinued = FALSE) AS continued_products,
(SELECT COUNT(*)
FROM product p
WHERE p.category_id = c.category_id) AS all_products
FROM category c
ORDER BY c.name;
```

	category_name character varying (60)	discontinued_products bigint	continued_products bigint	all_products bigint
1	Accesorios de Cocina	0	0	0
2	Alimentos	0	1	1
3	Audio	0	2	2
4	Calzado	0	0	0
5	Carnes	0	1	1
6	Cereales	0	0	0
7	Computadoras	0	3	3
8	Electrónicos	0	0	0
9	Frutas y Verduras	0	2	2
10	Herramientas	0	2	2
11	Hogar	0	1	1
12	Jardinería	0	0	0
13	Juguetes	0	0	0
14	Lácteos	0	2	2
15	Libros	0	0	0
16	Limpieza	0	2	2
17	Muebles	0	2	2
18	Ropa	0	0	0
19	Smartphones	0	1	1
20	Video	0	1	1
Total rows: 20 Query complete 00:00:00.135				

Fig. 6: Resultado Subconsulta Ej. 13

Discusión

El caso de esta consulta se relaciona con la fragmentación horizontal, porque se está trabajando con diferentes campos de filas de la tabla product según la condición discontinued = TRUE o FALSE.

Ejercicio 14: Contar las compras gestionadas por cada empleado en Houston

Muestra el ID del empleado y el número total de compras que gestionó. Utiliza una subconsulta para obtener información sobre el número de pedidos que gestionó cada empleado por cliente y haz que la consulta principal seleccione FROM esta subconsulta.

```
select e.employee_id, (select count(purchase_id)
from purchase where employee_id = e.employee_id) as cant_comp_gest
from employee e
where city = 'houston';
```

employee_id [PK] integer	cant_comp_gest bigint
-----------------------------	--------------------------

Fig. 7: Resultado Subconsulta Ej. 14

Discusión



En este ejercicio la condición WHERE e.city = 'Houston' selecciona únicamente las filas de empleados que trabajan en esa ciudad. Esto corresponde a una fragmentación horizontal, porque la tabla employees se divide en filas según su ubicación.

Ejercicio 15: Encuentra el mayor número de categorías de productos en una compra

Utilice una subconsulta para seleccionar el ID de compra y el número de categorías distintas que contiene. En la consulta principal, seleccione el número máximo de categorías de esta subconsulta.

```
select max(num_cat) as max_categoria
from ( select pu.purchase_id, count( distinct p.category_id) as num_cat
from purchase_item pi
join purchase pu on pi.purchase_id = pu.purchase_id
join product p on pi.product_id = p.product_id
group by pu.purchase_id);
```



Fig. 8: Resultado Subconsulta Ej. 15

Discusión

En este ejercicio se usa la fragmentación vertical, porque se están seleccionando y utilizando únicamente ciertos atributos (purchase_id y category_id) para construir en la primera subconsulta, sin necesidad de trabajar con todas las columnas de las tablas originales.

2.7 Resultados obtenidos

- Ejercicio 10: Se obtuvo el porcentaje del gasto de cada cliente en relación con el total de sus compras, mostrando correctamente las columnas contact_name, purchase_id y percentage.
- Ejercicio 11: Se identificaron las categorías que contienen productos con precios superiores al promedio de su categoría, mostrando el número de productos más caros por cada una.
- Ejercicio 12: Se listaron los productos adquiridos junto con la cantidad máxima comprada y el número total de compras, optimizando la visualización de datos relevantes.
- Ejercicio 13: Se mostraron, por categoría, los productos discontinuados, continuados y el total general de productos, diferenciando cada tipo según su estado de disponibilidad.
- Ejercicio 14: Se determinó el número de compras gestionadas por cada empleado en la ciudad de Houston, evidenciando la fragmentación horizontal aplicada.
- Ejercicio 15: Se identificó la compra con el mayor número de categorías de productos distintas, utilizando subconsultas para resumir la información de manera eficiente.

2.8 Habilidades blandas empleadas en la práctica

- ☐ Liderazgo
- ☒ Trabajo en equipo
- ☐ Comunicación asertiva
- ☐ La empatía



- ☐ Pensamiento crítico
- ☐ Flexibilidad
- ☐ La resolución de conflictos
- ☐ Adaptabilidad
- ☒ Responsabilidad

2.9 Conclusiones

En conclusión, las consultas realizadas permitieron analizar y relacionar datos de distintas tablas de la base tienda_almacen, aplicando subconsultas y fragmentaciones para obtener información precisa y optimizada. Además, se comprobó la eficacia de PostgreSQL en el manejo y análisis de datos complejos.

2.10 Recomendaciones

- Optimizar las consultas utilizando índices y condiciones adecuadas para mejorar el rendimiento y facilitar el análisis de grandes volúmenes de datos en PostgreSQL.

2.11 Referencias bibliográficas

- [1] Subqueries in PostgreSQL. (s/f). Google Cloud. Recuperado el 7 de octubre de 2025, de <https://cloud.google.com/spanner/docs/reference/postgresql/subqueries>

2.12 Anexos

Inserciones de Ejemplo:

INSERT

INTO employee (employee_id, last_name, first_name, birth_date, hire_date, reports_to)

VALUES

(1, 'García', 'Juan', '1985-05-15', '2010-03-01', NULL),
(2, 'Pérez', 'Ana', '1990-08-20', '2015-06-10', 1),
(3, 'López', 'Carlos', '1988-11-25', '2012-09-15', 1),
(4, 'Rodríguez', 'María', '1992-02-12', '2018-01-22', 2),
(5, 'Fernández', 'Luis', '1987-07-30', '2014-04-05', 3),
(6, 'Díaz', 'Paola', '1995-03-01', '2020-05-20', 1),
(7, 'Sánchez', 'Andrés', '1993-09-19', '2019-11-11', 2),
(8, 'Romero', 'Julia', '1989-12-05', '2016-08-28', 3),
(9, 'Suárez', 'Pedro', '1991-04-18', '2017-07-07', 4),
(10, 'Castro', 'Gabriela', '1996-06-22', '2021-02-14', 5),
(11, 'Navarro', 'Jorge', '1984-01-30', '2013-09-01', 1),
(12, 'Jiménez', 'Sofía', '1997-07-03', '2022-03-10', 6),
(13, 'Ruiz', 'Daniel', '1994-11-14', '2019-06-25', 7),
(14, 'Ortiz', 'Valeria', '1998-08-08', '2023-01-05', 8),
(15, 'Mendoza', 'Ricardo', '1990-10-21', '2018-04-12', 9),
(16, 'Guerrero', 'Natalia', '1992-05-09', '2017-10-30', 10),
(17, 'Silva', 'Alejandro', '1995-02-28', '2020-11-19', 11),
(18, 'Vargas', 'Lucía', '1999-09-11', '2023-08-21', 12),
(19, 'Morales', 'Eduardo', '1994-04-04', '2019-03-03', 13),
(20, 'Torres', 'Cristina', '1996-12-16', '2021-06-06', 14);



UNIVERSIDAD TÉCNICA DE AMBATO
FACULTAD DE INGENIERÍA EN SISTEMAS ELECTRÓNICA E INDUSTRIAL
CARRERA DE TECNOLOGÍAS DE LA INFORMACIÓN
CICLO ACADÉMICO: AGOSTO 2025 – ENERO 2026



INSERT

```
INTO customer (customer_id, contact_name, company_name, contact_email, address, city,
country) VALUES
(101, 'Martín Ruiz', 'Innovación Digital S.A.', 'martin.r@innova.com', 'Calle Mayor 123',
'Madrid', 'Spain'),
(102, 'Elena Torres', 'Tech Solutions Ltda.', 'elena.t@techsol.cl', 'Av. Libertador 456',
'Santiago', 'Chile'),
(103, 'Jorge Vidal', 'Grupo Empresarial S.A.', 'jorge.v@ge.com', 'Avenida Central 789',
'Bogotá', 'Colombia'),
(104, 'Sofía Castro', 'Logística Global S.L.', 'sofia.c@logistica.es', 'Paseo del Sol 10',
'Barcelona', 'Spain'),
(105, 'David Ramos', 'Negocios Unidos', 'david.r@negociosunidos.com', 'Calle de la Luna 5',
'CDM', 'Mexico'),
(106, 'Lorena Gómez', 'Comercializadora del Sur', 'l.gomez@cosur.com', 'Carrera 7, 34-56',
'Medellín', 'Colombia'),
(107, 'Hugo Salazar', 'Distribuciones Norte', 'hugo.s@distribunorte.cl', 'Av. Providencia 200',
'Valparaíso', 'Chile'),
(108, 'Carla Ibáñez', 'Market Place', 'carla.i@market.es', 'Plaza de Cataluña 1', 'Sevilla',
'Spain'),
(109, 'Felipe Mendoza', 'Soluciones Tecnológicas MX', 'felipe.m@solutec.mx', 'Insurgentes
Sur 300', 'Guadalajara', 'Mexico'),
(110, 'Andrea Ríos', 'Todo para la Oficina', 'andrea.r@todoofi.com', 'Calle 50, 2-10', 'Bogotá',
'Colombia'),
(111, 'Pablo Giménez', 'Futuro Digital', 'pablo.g@futuro.es', 'Gran Vía 5', 'Madrid', 'Spain'),
(112, 'Laura Soto', 'Global Express', 'laura.s@globalexp.com', 'Calle Larga 10-20', 'Quito',
'Ecuador'),
(113, 'Roberto Pinto', 'Innovación en Productos', 'roberto.p@innovaprod.com', 'Avenida
Brasil 500', 'Lima', 'Peru'),
(114, 'Diana Castillo', 'Compu Mundo', 'diana.c@compumundo.com', '25 de Mayo 1000',
'Buenos Aires', 'Argentina'),
(115, 'Gabriel Orozco', 'Materiales y Suministros', 'gabriel.o@mym.com', 'Libertad 15',
'Montevideo', 'Uruguay'),
(116, 'Mónica López', 'Electro Hogar', 'monica.l@electro.com', 'Avenida Central 345', 'San
José', 'Costa Rica'),
(117, 'Ricardo Paz', 'El Descuento', 'ricardo.p@eldescuento.com', 'Calle del Águila 88',
'Panamá', 'Panama'),
(118, 'Ana Pérez', 'Tecno Store', 'ana.p@tecnostore.com', 'Calle 5ta, 3-21', 'Guatemala',
'Guatemala'),
(119, 'Juan Molina', 'Súper Ofertas', 'juan.m@superofertas.com', 'Avenida Las Américas 123',
'Asunción', 'Paraguay'),
(120, 'Isabel Ramos', 'Servicios Integrales', 'isabel.r@servicios.com', 'Rua da Paz, 99',
'Lisboa', 'Portugal');
```

INSERT INTO category (category_id, name, description, parent_category_id) VALUES

```
(1, 'Electrónicos', 'Dispositivos y gadgets electrónicos.', NULL),
(2, 'Computadoras', 'Portátiles, de escritorio y accesorios.', 1),
(3, 'Smartphones', 'Teléfonos inteligentes y sus accesorios.', 1),
```



UNIVERSIDAD TÉCNICA DE AMBATO
FACULTAD DE INGENIERÍA EN SISTEMAS ELECTRÓNICA E INDUSTRIAL
CARRERA DE TECNOLOGÍAS DE LA INFORMACIÓN
CICLO ACADÉMICO: AGOSTO 2025 – ENERO 2026



- (4, 'Hogar', 'Artículos para el hogar y la cocina.', NULL),
- (5, 'Alimentos', 'Comida y bebidas.', NULL),
- (6, 'Muebles', 'Mobiliario para oficina y hogar.', 4),
- (7, 'Herramientas', 'Herramientas manuales y eléctricas.', NULL),
- (8, 'Jardinería', 'Productos para el cuidado del jardín.', 7),
- (9, 'Lácteos', 'Leche, quesos y yogures.', 5),
- (10, 'Frutas y Verduras', 'Productos frescos del campo.', 5),
- (11, 'Limpieza', 'Productos de limpieza para el hogar.', 4),
- (12, 'Audio', 'Auriculares, altavoces y sistemas de sonido.', 1),
- (13, 'Video', 'Televisores y proyectores.', 1),
- (14, 'Cereales', 'Granos y productos derivados.', 5),
- (15, 'Carnes', 'Productos cárnicos frescos y procesados.', 5),
- (16, 'Accesorios de Cocina', 'Utensilios y gadgets para la cocina.', 4),
- (17, 'Ropa', 'Prendas de vestir para todas las edades.', NULL),
- (18, 'Calzado', 'Todo tipo de zapatos y zapatillas.', NULL),
- (19, 'Libros', 'Libros de ficción, no ficción y académicos.', NULL),
- (20, 'Juguetes', 'Artículos para el entretenimiento de niños.', NULL);

INSERT

INTO product (product_id, product_name, category_id, quantity_per_unit, unit_price, units_in_stock, discontinued) VALUES

- (1001, 'Laptop UltraPro', 2, 1, 1200.00, 50, FALSE),
- (1002, 'Smartphone X1', 3, 1, 850.50, 150, FALSE),
- (1003, 'Mouse inalámbrico', 2, 1, 25.00, 300, FALSE),
- (1004, 'Cafetera Espresso', 4, 1, 75.99, 80, FALSE),
- (1005, 'Snack de avena', 5, 100, 2.50, 200, FALSE),
- (1006, 'Altavoz Bluetooth', 12, 1, 59.99, 120, FALSE),
- (1007, 'Televisor SmartTV 4K', 13, 1, 650.00, 30, FALSE),
- (1008, 'Silla Ergonómica de Oficina', 6, 1, 150.00, 45, FALSE),
- (1009, 'Set de destornilladores', 7, 6, 19.50, 200, FALSE),
- (1010, 'Leche entera 1L', 9, 1, 1.80, 500, FALSE),
- (1011, 'Manzanas Red', 10, 1, 3.20, 150, FALSE),
- (1012, 'Limpiador Multiusos', 11, 500, 4.10, 250, FALSE),
- (1013, 'Auriculares con cancelación de ruido', 12, 1, 199.99, 90, FALSE),
- (1014, 'Monitor curvo 27"', 2, 1, 299.00, 75, FALSE),
- (1015, 'Yogur natural', 9, 150, 1.10, 400, FALSE),
- (1016, 'Brócoli fresco', 10, 1, 2.30, 85, FALSE),
- (1017, 'Aspiradora robot', 11, 1, 350.00, 20, FALSE),
- (1018, 'Mesa de centro', 6, 1, 89.99, 60, FALSE),
- (1019, 'Sierra eléctrica', 7, 1, 120.50, 40, FALSE),
- (1020, 'Filete de salmón', 15, 200, 15.00, 100, FALSE);

INSERT

INTO purchase (purchase_id, customer_id, employee_id, total_price, purchase_date, shipped_date, ship_city, ship_country) VALUES

- (1, 101, 2, 1225.00, '2023-10-25 10:00:00', '2023-10-26 14:00:00', 'Madrid', 'Spain'),
- (2, 102, 3, 850.50, '2023-10-25 11:30:00', '2023-10-26 15:00:00', 'Santiago', 'Chile'),
- (3, 103, 4, 75.99, '2023-10-26 09:00:00', '2023-10-27 12:00:00', 'Bogotá', 'Colombia'),



UNIVERSIDAD TÉCNICA DE AMBATO
FACULTAD DE INGENIERÍA EN SISTEMAS ELECTRÓNICA E INDUSTRIAL
CARRERA DE TECNOLOGÍAS DE LA INFORMACIÓN
CICLO ACADÉMICO: AGOSTO 2025 – ENERO 2026



(4, 104, 2, 2.50, '2023-10-26 14:00:00', '2023-10-27 10:00:00', 'Barcelona', 'Spain'),
(5, 105, 5, 1200.00, '2023-10-27 09:30:00', '2023-10-28 11:00:00', 'CDM', 'Mexico'),
(6, 106, 6, 150.00, '2023-10-28 10:30:00', '2023-10-29 11:00:00', 'Medellín', 'Colombia'),
(7, 107, 7, 650.00, '2023-10-28 12:45:00', '2023-10-29 13:00:00', 'Valparaíso', 'Chile'),
(8, 108, 8, 59.99, '2023-10-29 09:15:00', '2023-10-30 10:30:00', 'Sevilla', 'Spain'),
(9, 109, 9, 19.50, '2023-10-29 14:00:00', '2023-10-30 11:00:00', 'Guadalajara', 'Mexico'),
(10, 110, 10, 1.80, '2023-10-30 08:00:00', '2023-10-31 09:00:00', 'Bogotá', 'Colombia'),
(11, 111, 11, 3.20, '2023-10-30 11:00:00', '2023-10-31 10:00:00', 'Madrid', 'Spain'),
(12, 112, 12, 4.10, '2023-10-31 09:45:00', '2023-11-01 11:00:00', 'Quito', 'Ecuador'),
(13, 113, 13, 199.99, '2023-11-01 13:20:00', '2023-11-02 12:00:00', 'Lima', 'Peru'),
(14, 114, 14, 299.00, '2023-11-01 15:00:00', '2023-11-02 14:00:00', 'Buenos Aires',
'Argentina'),
(15, 115, 15, 1.10, '2023-11-02 10:00:00', '2023-11-03 10:30:00', 'Montevideo', 'Uruguay'),
(16, 116, 16, 2.30, '2023-11-02 14:30:00', '2023-11-03 12:00:00', 'San José', 'Costa Rica'),
(17, 117, 17, 350.00, '2023-11-03 09:00:00', '2023-11-04 11:00:00', 'Panamá', 'Panama'),
(18, 118, 18, 89.99, '2023-11-03 11:45:00', '2023-11-04 12:30:00', 'Guatemala', 'Guatemala'),
(19, 119, 19, 120.50, '2023-11-04 10:10:00', '2023-11-05 10:00:00', 'Asunción', 'Paraguay'),
(20, 120, 20, 15.00, '2023-11-04 15:00:00', '2023-11-05 14:00:00', 'Lisboa', 'Portugal');

INSERT INTO purchase_item (purchase_id, product_id, unit_price, quantity) VALUES

(1, 1001, 1200.00, 1),
(1, 1003, 25.00, 1),
(2, 1002, 850.50, 1),
(3, 1004, 75.99, 1),
(4, 1005, 2.50, 1),
(5, 1001, 1200.00, 1),
(6, 1008, 150.00, 1),
(7, 1007, 650.00, 1),
(8, 1006, 59.99, 1),
(9, 1009, 19.50, 1),
(10, 1010, 1.80, 1),
(11, 1011, 3.20, 1),
(12, 1012, 4.10, 1),
(13, 1013, 199.99, 1),
(14, 1014, 299.00, 1),
(15, 1015, 1.10, 1),
(16, 1016, 2.30, 1),
(17, 1017, 350.00, 1),
(18, 1018, 89.99, 1),
(19, 1019, 120.50, 1),
(20, 1020, 15.00, 1);