

# Reto BD

Barbería (Ejercicio A)

Entregado A

Juanes Pineda

Presentado

Yeison Ferney Osorio Buitrago

SOFKAU

SAN JOSE DE CUCUTA

## Contenido

Indicar que ejercicio fue asignado .....	3
Realizar el modelo E-R.....	5
Realizar el modelo relacional .....	7
Normalizar correctamente .....	8
Escribir con sentencias SQL toda la definición de la base de datos.....	9
Escribir consultas que me permitan ver la información de cada tabla o de varias tablas (10).....	11
Generar de 4 a 6 vistas donde se evidencie lo más importante de cada ejercicio (haga una selección muy responsable de la información realmente importante según el contexto). ....	17
Generar al menos 4 procedimientos almacenados. ....	19
Generar al menos 4 Triggers .....	20
Poblar la base de datos (50 registros por tabla) utilizando una conexión desde Java. ....	21
Al terminar el ejercicio responda ¿ Está conforme con el resultado obtenido según el contexto o cree que hubiera obtenido un mejor resultado con una base de datos no relacional? .....	30

Indicar que ejercicio fue asignado

## Barbería (Ejercicio A)

### Enunciado



BARBER YEIO es una barbería que ofrece la posibilidad de realizar reservas para sus clientes. Cuando un cliente realiza una reserva, se genera una cita con identificador único en la que se indica la fecha y hora en la que será atendido por un empleado de la barbería. Cada cita está asociada a un servicio en el que se indica el tipo de servicio a realizar y un identificador único.

Cada servicio genera un gasto de insumos, que son los productos necesarios para realizar el servicio, y que están disponibles en la barbería. Los insumos son facturados junto con el servicio prestado. Además, los clientes también tienen la posibilidad de comprar productos disponibles en la barbería, que son distribuidos por diferentes proveedores. Estos productos son vendidos por un empleado de la barbería el cual gana una pequeña comisión por venta realizada que en la factura se refleja el empleado que vendió dicho producto

La barbería cuenta con un registro de clientes en el que se almacenan sus datos personales, como su identificación, nombre, profesión, teléfono y correo. También se guarda un historial de los servicios prestados a cada cliente, para llevar un control detallado de las citas y servicios realizados.

En resumen, BARBER YEIO cuenta con diferentes entidades como reserva, cita, servicio, insumos, factura, productos, proveedor, cliente e historial de servicio, que están relacionadas entre sí de diferentes maneras para llevar un control detallado de los servicios prestados y los productos disponibles en la barbería.

Se desea almacenar la siguiente información:

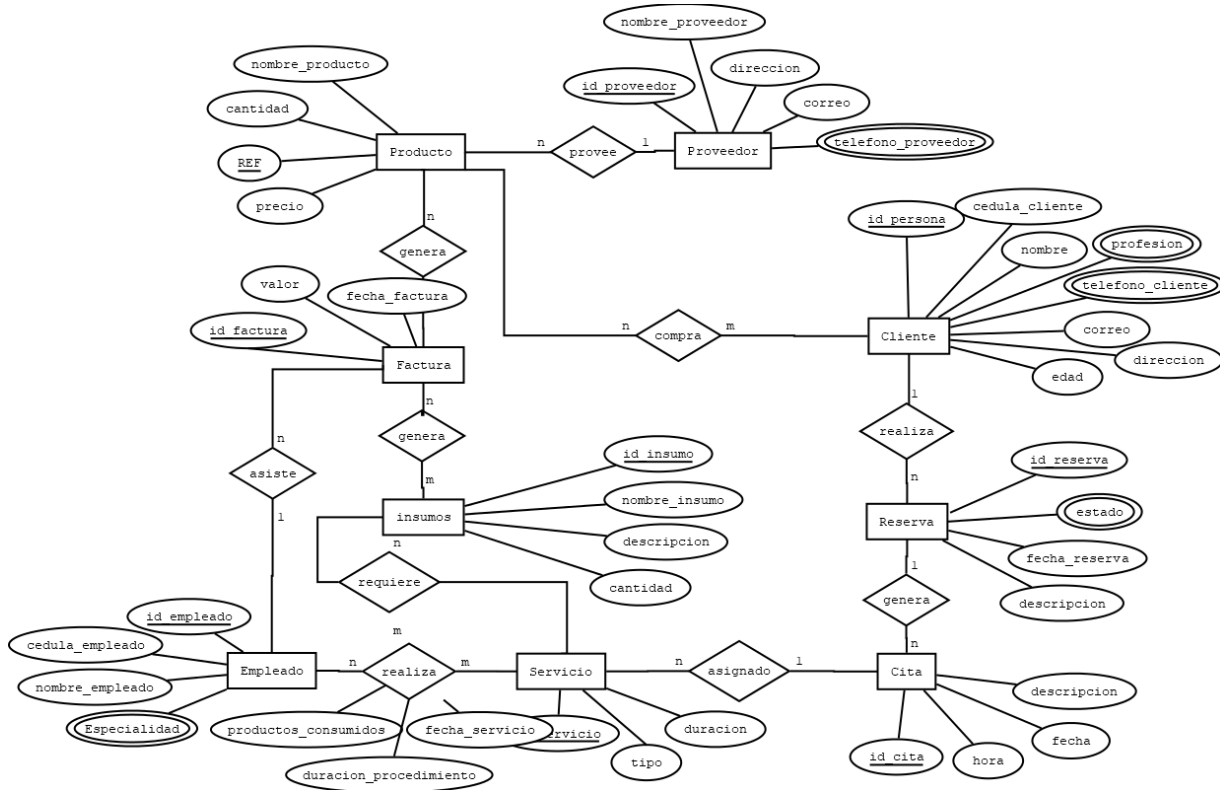
- Empleados: ID, cedula, Nombre, Especialidad (Masaje, Corte, Cejas, etc.)
- Clientes: Datos personales (ID, cedula, Nombre, Profesión, Teléfono, correo, edad y Dirección).
- Historial de Servicios prestados por la barbería: Un registro para saber información del servicio prestado por un empleado a un cliente, productos consumidos, duración del procedimiento y fecha.
- Citas: ID, Fecha y Hora y una descripción en la que se cita al cliente y el barbero que realizará el servicio.
- Productos vendidos por la barbería: REF, Nombre, Cantidad y Precio.
- Proveedor: los productos vendidos deben tener una fuente.
- Registro de Ventas: Si un barbero vende un producto a un cliente, termina obteniendo una "liga" ganancia ocasional.
- Insumo: ID, nombre, descripción, cantidad
- Reserva: ID, estado (confirmada, pendiente, cancelada.), fechas reserva, descripción
- Factura: ID, valor, fecha

Se seba que:

- Los clientes pueden realizar una reserva para programar una cita, si no realiza la reserva no se le atenderá, las políticas de la compañía son estrictas. Además, los clientes tienen la opción de comprar productos, lo cual le genera una factura.
- Un empleado de la barbería realiza un servicio para el cual se le asigna una cita con un cliente en particular. Durante el servicio, se pueden utilizar insumos que generan un costo que se cargará a la factura. El historial del servicio se guarda para llevar registro del empleado que atendió al cliente.
- Si un empleado vende un producto a un cliente, se le otorgará una ganancia o "liga".
- Los productos son provistos por un proveedor externo.

## Realizar el modelo E-R

### Modelo Entidad Relación Barbería



#### Relación Cliente – Reserva (1-N)

Cardinalidad: Un cliente puede realizar una o muchas reserva, una reserva puede tener muchas reservas

#### Relación Reserva – Cita (1-N)

Cardinalidad: Una Reserva puede apartar una o muchas citas y una cita pertenece a una reserva

#### Relación Cita – Servicio(1-N)

Cardinalidad: Una cita puede tener uno o muchos servicios y estos servicios están asociados a una o muchas facturas

**Relación Servicio – Insumo(N-M)**

Cardinalidad: un servicio puede tener uno o muchos insumos y un insumo puede ser utilizado por uno o muchos servicios

**Relación Insumo – Factura(N-M)**

Cardinalidad: una factura tiene uno o muchos insumos y un insumo puede estar en uno o muchas facturas

**Relación Empleado -Servicio(N-M)**

Cardinalidad: El Empleado puede realizar uno a muchos servicios y un servicio puede tener muchos empleados por ejemplo un corte de cabello y un corte de uñas

**Relación Empleado – Factura(1-N)**

Cardinalidad: un empleado puede asesorar uno o mucha compra y una compra es asesorada por un empleado

**Relación Factura – Producto(N-M)**

Cardinalidad: una factura tiene uno o muchos productos y un producto puede pertenecer a una o muchas facturas una relación n:m

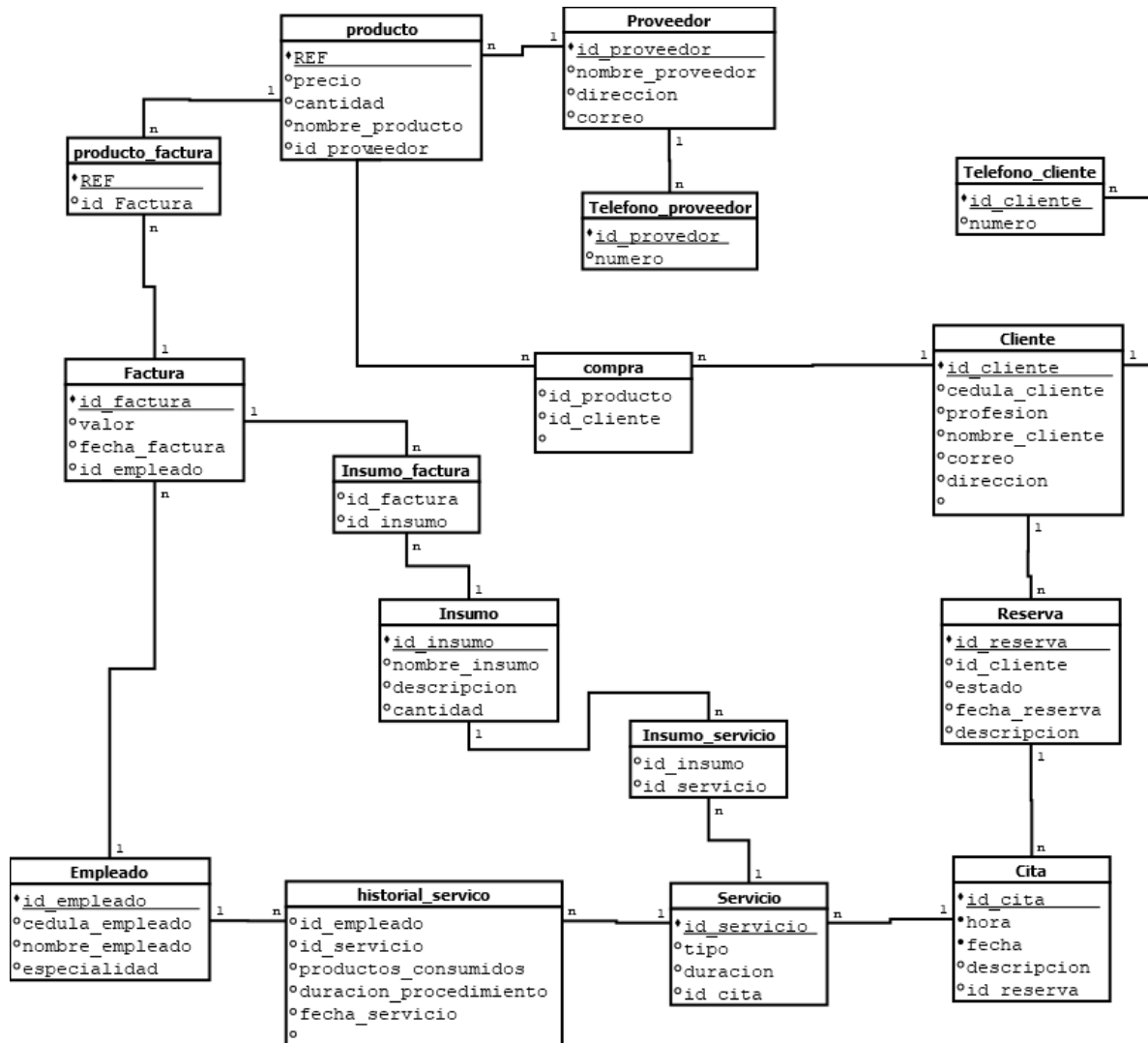
**Relación Cliente – Producto(N-M)**

Cardinalidad: un cliente puede comprar uno o muchos productos y un producto puede ser comprado por uno o muchos clientes

**Relación Proveedor- Producto(N-M)**

Cardinalidad: un proveedor puede proveer uno a muchos productos y un producto es proveído por un proveedor

## Realizar el modelo relacional



En total tenemos 9 Tablas principales y 7 tablas intermedias

Entre ellas tenemos

**Cliente:** Se encarga de guardar la información del cliente

**Reserva:** Guardar la información de la reserva ya que un cliente es atendido por una reserva

**Cita:** La reserva le genera una cita al cliente

**Servicio:** El servicio es el medio por el cual el cliente es atendido

**Empleado:** El empleado es el que atiende al cliente

**Factura:** La factura del servicio incluido los productos consumidos y el empleado que lo atendió

**Producto:** Producto consumido por el cliente

**Proveedor:** Proveedor aquel que provee la materia prima

**Telefono cliente:** Guarda el teléfono del cliente

**Telefono\_proveedor:** Guarda el teléfono del proveedor

**Historial:** Guarda el historial

**Insumo\_factura:** relación entre el insumo y la factura

**Insumo\_servicio** Relación entre el insumo y servicio

**Compra:** relación entre el cliente y producto

## Normalizar correctamente

### Normalización

#### Primera Forma Normal 1FN

- Para cumplir con la función debimos crear tabla para los valores multivaluados y de esta manera cumplir con los requerimientos.
- No deben existir registros duplicados
- Todas las tablas tienen llave primaria

Se cumple con la 2FN.

- Cumple con 1FN
- Todos los valores de las columnas dependen únicamente de la llave primaria de la tabla.
- Las tablas tienen una única llave primaria que identifica a la tabla y sus atributos dependen de ella. Esto significa que, si se conoce la llave primaria, se pueden determinar todos los demás valores en la tabla.

Se cumple con la 3FN.

- Estar en 2FN
- Cada atributo que no está incluido en la clave primaria no depende de la clave primaria. Esto significa que un atributo no debe depender de otro atributo que no forme parte de la clave primaria.



Escribir con sentencias SQL toda la definición de la base de datos.

Pantallazo del script de la base de datos, archivo anexo en el repo

```

1  ● CREATE DATABASE barber;
2
3  -- Crear tabla Cliente
4  ● CREATE TABLE Cliente (
5      id_cliente INT PRIMARY KEY,
6      nombre_cliente VARCHAR(50),
7      cedula_cliente VARCHAR(20),
8      profesion VARCHAR(50),
9      correo VARCHAR(50),
10     edad INT,
11     direccion VARCHAR(50)
12 );
13
14 ● CREATE TABLE Compra(
15     REF VARCHAR(10),
16     id_cliente int,
17     PRIMARY KEY (id_cliente, REF),
18     FOREIGN KEY(REF) REFERENCES Producto(REF),
19     FOREIGN KEY(id_cliente) REFERENCES Cliente(id_cliente)
20 );
21 ● CREATE TABLE Telefono_cliente(
22     id_cliente INT PRIMARY KEY,
23     numero VARCHAR(12),
24     FOREIGN KEY(id_cliente) REFERENCES Cliente(id_cliente)
25 );
26
27
28 -- Crear tabla Cita
29 ● CREATE TABLE Cita (
30     id_cita INT PRIMARY KEY,
31     id_reserva INT,
32     fecha DATE,
33     hora TIME,
34     descripcion VARCHAR(255),
35
36     FOREIGN KEY (id_reserva) REFERENCES Reserva(id_reserva)
37 );
38
39 -- Crear tabla Servicio
40 ● CREATE TABLE Servicio (
41     id_servicio INT PRIMARY KEY,
42     id_cita INT,
43     tipo VARCHAR(50),
44     duracion INT,
45
46     FOREIGN KEY (id_cita) REFERENCES Cita(id_cita)
47 );

```

```

59      -- Crear tabla Historial
60      ● ○ CREATE TABLE Historial (
61          id_empleado INT,
62          id_servicio INT,
63          productos_consumidos VARCHAR(100),
64          duracion_procedimiento INT,
65          fecha_servicio DATE,
66          PRIMARY KEY (id_empleado, id_servicio),
67          FOREIGN KEY (id_empleado) REFERENCES Empleado(id_empleado),
68          FOREIGN KEY (id_servicio) REFERENCES Servicio(id_servicio)
69      );
70
71      -- Crear tabla Empleado
72      ● ○ CREATE TABLE Empleado (
73          id_empleado INT PRIMARY KEY,
74          cedula VARCHAR(20),
75          nombre_empleado VARCHAR(50),
76          especialidad VARCHAR(50)
77      );
78
79      -- Crear tabla Factura
80      ● ○ CREATE TABLE Factura (
81          id_factura INT PRIMARY KEY,
82          id_empleado INT,
83          valor DOUBLE,
84          fecha_factura DATE,
85          FOREIGN KEY (id_empleado) REFERENCES Empleado(id_empleado)
86      );
87
88      -- Crear tabla Producto
89      ● ○ CREATE TABLE Producto (
90          REF varchar(10) PRIMARY KEY,
91          precio DOUBLE,
92          cantidad INT,
93          nombre_producto VARCHAR(50),
94          id_proveedor INT,
95          FOREIGN KEY (id_proveedor) REFERENCES Proveedor(id_proveedor)
96      );
97      -- Crear tabla Proveedor
98      ● ○ CREATE TABLE Proveedor (
99          id_proveedor INT PRIMARY KEY,
100         nombre_proveedor VARCHAR(50),
101         direccion VARCHAR(50),
102         correo VARCHAR(50)
103     );
104     ● ○ CREATE TABLE Telefono_proveedor(
105         id_proveedor INT PRIMARY KEY,
106         numero VARCHAR(12),
107         FOREIGN KEY(id_proveedor) REFERENCES proveedor(id_proveedor)
108     );
109     ● ○ CREATE TABLE Insumo(
110         id_insumo int,
111         nombre_insumo VARCHAR(50),
112         descripcion VARCHAR(255),
113         cantidad int
114     );

```



```

1
2 -- Mostrar todas las reservas y sus citas asociadas:
3 • SELECT r.id_reserva, c.id_cita, c.fecha, c.hora, c.descripcion
4 FROM Reserva r
5 LEFT JOIN Cita c ON r.id_reserva = c.id_reserva;
6

```

id_reserva	id_cita	fecha	hora	descripcion
1	1	2023-02-21	10:00:00	Cita para limpieza dental
2	2	2023-02-23	14:00:00	Cita para consulta médica
3	3	2023-02-25	11:00:00	Cita para asesoría legal
4	4	2023-02-20	16:00:00	Cita para diseño de interiores
5	5	2023-02-26	09:00:00	Cita para asesoría contable

2. La consulta SQL que me permitiría obtener el cliente con sus datos personales, la fecha y hora de su cita, la descripción del servicio y su tipo, y el nombre del empleado que lo atendió:

```

SELECT cliente.nombre_cliente, cliente.cedula_cliente, cliente.correo, cita.fecha, cita.hora,
cita.descripcion, servicio.tipo, empleado.nombre_empleado
FROM servicio
INNER JOIN cita ON servicio.id_cita = cita.id_cita
INNER JOIN reserva ON cita.id_reserva = reserva.id_reserva
INNER JOIN cliente ON reserva.id_cliente = cliente.id_cliente
INNER JOIN historial ON servicio.id_servicio = historial.id_servicio
INNER JOIN empleado ON historial.id_empleado = empleado.id_empleado;

```

```

8 -- consulta que me permite saber ver el cliente con sus datos personales el día de su cita hora y la descripción del
9 -- servicio y su tipo de servicio y el empleado que lo atendió
10 • SELECT cliente.nombre_cliente, cliente.cedula_cliente, cliente.correo, cita.fecha, cita.hora,
11 cita.descripcion, servicio.tipo, empleado.nombre_empleado
12 FROM servicio
13 INNER JOIN cita ON servicio.id_cita = cita.id_cita
14 INNER JOIN reserva ON cita.id_reserva = reserva.id_reserva
15 INNER JOIN cliente ON reserva.id_cliente = cliente.id_cliente
16 INNER JOIN historial ON servicio.id_servicio = historial.id_servicio
17 INNER JOIN empleado ON historial.id_empleado = empleado.id_empleado;
18

```

nombre_cliente	cedula_cliente	correo	fecha	hora	descripcion	tipo	nombre_empleado
Juan Perez	1234567890	juanperez@gmail.com	2023-02-21	10:00:00	Cita para limpieza dental	Limpieza	Juan Pérez
Laura Hernandez	3216549870	laurahernandez@yahoo.com	2023-02-20	16:00:00	Cita para diseño de interiores	tinte pelo	Pedro Rodríguez
Andres Torres	9876543210	andrestorres@gmail.com	2023-02-26	09:00:00	Cita para asesoría contable	corte de cabello	María González

- Esta consulta te permitirá obtener información del cliente (nombre, cédula, correo), fecha y hora de la cita, la descripción del servicio y su tipo.

```
SELECT cliente.nombre_cliente, cliente.cedula_cliente, cliente.correo, cita.fecha, cita.hora, cita.descripcion,
servicio.tipo
FROM servicio
INNER JOIN cita ON servicio.id_cita = cita.id_cita
INNER JOIN reserva ON cita.id_reserva = reserva.id_reserva
INNER JOIN cliente ON reserva.id_cliente = cliente.id_cliente;
```

```

23  -- consulta que me permite ver el cliente con sus datos personales el día de su cita hora
24  -- y la descripción del servicio y su tipo de servicio
25  • SELECT cliente.nombre_cliente, cliente.cedula_cliente, cliente.correo, cita.fecha, cita.hora, cita.descripcion, servicio.tipo
26  FROM servicio
27  INNER JOIN cita ON servicio.id_cita = cita.id_cita
28  INNER JOIN reserva ON cita.id_reserva = reserva.id_reserva
29  INNER JOIN cliente ON reserva.id_cliente = cliente.id_cliente;
30

```

nombre_cliente	cedula_cliente	correo	fecha	hora	descripcion	tipo
Juan Perez	1234567890	juanperez@gmail.com	2023-02-21	10:00:00	Cita para limpieza dental	Limpieza
Maria Rodriguez	0987654321	maria.rodriguez@hotmail.com	2023-02-23	14:00:00	Cita para consulta médica	Consulta
Pedro Gomez	4567890123	pedrogomez@gmail.com	2023-02-25	11:00:00	Cita para asesoría legal	corte de cabello
Laura Hernandez	3216549870	laurahernandez@yahoo.com	2023-02-20	16:00:00	Cita para diseño de interiores	tinte pelo
Andres Torres	9876543210	andrestorres@gmail.com	2023-02-26	09:00:00	Cita para asesoría contable	corte de cabello

- consulta que permite conocer la cantidad de productos comprados por cada cliente, incluyendo su nombre:

```
SELECT COUNT(*) AS cantidad_productos, cliente.nombre_cliente, factura.valor, factura.fecha_factura,
producto.nombre_producto
FROM producto
INNER JOIN compra ON producto.REF = compra.REF
INNER JOIN cliente ON compra.id_cliente = cliente.id_cliente
INNER JOIN producto_factura ON producto.REF = producto_factura.REF
INNER JOIN factura ON producto_factura.id_factura = factura.id_factura
GROUP BY cliente.nombre_cliente, factura.valor, factura.fecha_factura, producto.nombre_producto;
```

```

33  -- Esta consulta contará la cantidad de productos comprados por cada cliente
34  • SELECT COUNT(*) AS cantidad_productos, cliente.nombre_cliente, factura.valor, factura.fecha_factura, producto.nombre_producto
35  FROM producto
36  INNER JOIN compra ON producto.REF = compra.REF
37  INNER JOIN cliente ON compra.id_cliente = cliente.id_cliente
38  INNER JOIN producto_factura ON producto.REF = producto_factura.REF
39  INNER JOIN factura ON producto_factura.id_factura = factura.id_factura
40  GROUP BY cliente.nombre_cliente, factura.valor, factura.fecha_factura, producto.nombre_producto;
41
42

```

	cantidad_productos	nombre_cliente	valor	fecha_factura	nombre_producto
1	1	Juan Perez	75.25	2021-02-18	Producto 1
1	1	Maria Rodriguez	100.5	2021-02-17	Producto 2
1	1	Pedro Gomez	200	2021-02-18	Producto 3

5. Esta consulta permite visualizar los productos comprados por un cliente y el empleado que lo asesoró. Se presenta el nombre del cliente, el nombre del empleado, el nombre del producto, la fecha de la compra y su precio.

```
SELECT cliente.nombre_cliente, factura.valor, factura.fecha_factura, producto.nombre_producto,
empleado.nombre_empleado
FROM producto
INNER JOIN compra ON producto.REF = compra.REF
INNER JOIN cliente ON compra.id_cliente = cliente.id_cliente
INNER JOIN producto_factura ON producto.REF = producto_factura.REF
INNER JOIN factura ON producto_factura.id_factura = factura.id_factura
INNER JOIN empleado ON factura.id_empleado = empleado.id_empleado;
```

```
44 -- Esta consulta me permite los productos comprados por un cliente y el empleado que lo asesoro
45 • SELECT cliente.nombre_cliente, factura.valor, factura.fecha_factura, producto.nombre_producto, empleado.nombre_empleado
46 FROM producto
47 INNER JOIN compra ON producto.REF = compra.REF
48 INNER JOIN cliente ON compra.id_cliente = cliente.id_cliente
49 INNER JOIN producto_factura ON producto.REF = producto_factura.REF
50 INNER JOIN factura ON producto_factura.id_factura = factura.id_factura
51 INNER JOIN empleado ON factura.id_empleado = empleado.id_empleado;
52
53
```

Result Grid   Filter Rows:   Export:   Wrap Cell Content:				
nombre_cliente	valor	fecha_factura	nombre_producto	nombre_empleado
Juan Perez	75.25	2021-02-18	Producto 1	María González
Maria Rodriguez	100.5	2021-02-17	Producto 2	Juan Pérez
Pedro Gomez	200	2021-02-18	Producto 3	Pedro Rodríguez

6. Consulta que me permite obtener información sobre las reservas de los clientes, incluyendo la fecha y hora de las citas, los empleados asignados a los servicios, los insumos utilizados y las facturas correspondientes. En particular, se mostrará la reserva que tiene el cliente, con detalles sobre la fecha y hora de la cita, el empleado asignado para realizar el servicio, los insumos que se utilizaron en dicho servicio, y la factura correspondiente.

```
SELECT cliente.nombre_cliente, cliente.cedula_cliente, cliente.correo, cita.fecha, cita.hora,
cita.descripcion, servicio.tipo, empleado.nombre_empleado, factura.valor, factura.fecha_factura
FROM servicio
INNER JOIN cita ON servicio.id_cita = cita.id_cita
INNER JOIN reserva ON cita.id_reserva = reserva.id_reserva
INNER JOIN cliente ON reserva.id_cliente = cliente.id_cliente
INNER JOIN historial ON servicio.id_servicio = historial.id_servicio
INNER JOIN empleado ON historial.id_empleado = empleado.id_empleado
INNER JOIN insumo_servicio ON servicio.id_servicio = insumo_servicio.id_servicio
INNER JOIN insumo ON insumo_servicio.id_insumo = insumo.id_insumo
INNER JOIN insumo_factura ON insumo.id_insumo = insumo_factura.id_insumo
INNER JOIN factura ON insumo_factura.id_factura = factura.id_factura;
```

```

55 -- consulta que me permite conocer la reserva que tiene el cliente, el dia de la cita el empleado que le va a
56 -- realizar el servicio los insumos gastados y la factura del servicio
57 • SELECT cliente.nombre_cliente, cliente.cedula_cliente, cliente.correo, cita.fecha, cita.hora,
58 cita.descripcion, servicio.tipo, empleado.nombre_empleado, factura.valor, factura.fecha_factura
59 FROM servicio
60 INNER JOIN cita ON servicio.id_cita = cita.id_cita
61 INNER JOIN reserva ON cita.id_reserva = reserva.id_reserva
62 INNER JOIN cliente ON reserva.id_cliente = cliente.id_cliente
63 INNER JOIN historial ON servicio.id_servicio = historial.id_servicio
64 INNER JOIN empleado ON historial.id_empleado = empleado.id_empleado
65 INNER JOIN insumo_servicio ON servicio.id_servicio = insumo_servicio.id_servicio
66 INNER JOIN insumo ON insumo_servicio.id_insumo = insumo.id_insumo
67 INNER JOIN insumo_factura ON insumo.id_insumo = insumo_factura.id_insumo
68 INNER JOIN factura ON insumo_factura.id_factura = factura.id_factura

```

nombre_cliente	cedula_cliente	correo	fecha	hora	descripcion	tipo	nombre_empleado	valor	fecha_factura
Juan Perez	1234567890	juanperez@gmail.com	2023-02-21	10:00:00	Cita para limpieza dental	Limpieza	Juan Pérez	100.5	2021-02-17

## 7. Consulta para saber cuál es el empleado que ha realizado más servicios:

- ✓ Selecciona el nombre del empleado y cuenta el número de servicios que ha realizado.
- ✓ La tabla "historial" se une con la tabla "empleado" usando el ID del empleado.
- ✓ Los resultados se agrupan por el nombre del empleado.
- ✓ Los resultados se ordenan de manera descendente por la cantidad de servicios realizados.
- ✓ Se limita la consulta a mostrar solo el primer resultado, que sería el empleado con mayor cantidad de servicios realizados.

```

SELECT empleado.nombre_empleado, COUNT(*) as cantidad_servicios
FROM historial
INNER JOIN empleado ON historial.id_empleado = empleado.id_empleado
GROUP BY empleado.nombre_empleado
ORDER BY cantidad_servicios DESC
LIMIT 1;

```

```

72 • SELECT empleado.nombre_empleado, COUNT(*) as cantidad_servicios
73 FROM historial
74 INNER JOIN empleado ON historial.id_empleado = empleado.id_empleado
75 GROUP BY empleado.nombre_empleado
76 ORDER BY cantidad_servicios DESC
77 LIMIT 1;
78
79
80

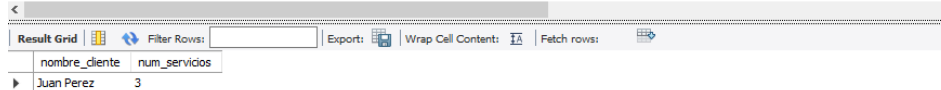
```

nombre_empleado	cantidad_servicios
Juan Pérez	2

8. Cliente que más ha asistido a servicios, podemos contar el número de servicios que ha tenido cada cliente y luego ordenarlos en orden descendente.

```
SELECT cliente.nombre_cliente, COUNT(*) AS num_servicios
FROM cliente
INNER JOIN reserva ON cliente.id_cliente = reserva.id_cliente
INNER JOIN cita ON reserva.id_reserva = cita.id_reserva
INNER JOIN servicio ON cita.id_cita = servicio.id_cita
GROUP BY cliente.id_cliente
ORDER BY num_servicios DESC
LIMIT 1;
```

```
80 -- consulta que me permite saber cual cliente es el que más ha asistido a servicios, podemos contar el número de
81 -- servicios que ha tenido cada cliente y luego ordenarlos en orden descendente.
82 • SELECT cliente.nombre_cliente, COUNT(*) AS num_servicios
83 FROM cliente
84 INNER JOIN reserva ON cliente.id_cliente = reserva.id_cliente
85 INNER JOIN cita ON reserva.id_reserva = cita.id_reserva
86 INNER JOIN servicio ON cita.id_cita = servicio.id_cita
87 GROUP BY cliente.id_cliente
88 ORDER BY num_servicios DESC
89 LIMIT 1;
```

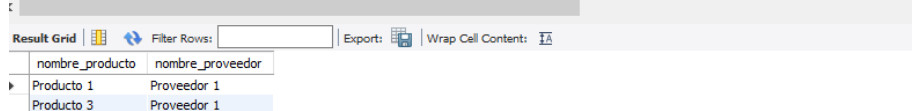


nombre_cliente	num_servicios
Juan Perez	3

9. En esta consulta, ver las tablas de productos y proveedores, utilizando la columna id\_proveedor como clave de unión. Luego, se especifica en la cláusula WHERE el nombre del proveedor del cual se desean los productos.

```
SELECT producto.nombre_producto, proveedor.nombre_proveedor
FROM producto
INNER JOIN proveedor ON producto.id_proveedor = proveedor.id_proveedor
WHERE proveedor.nombre_proveedor = 'Proveedor 1';
```

```
93 -- En esta consulta, ver las tablas de productos y proveedores,
94 -- utilizando la columna id_proveedor como clave de unión.
95 -- Luego, se especifica en la cláusula WHERE el nombre del proveedor del cual se desean los productos.
96 • SELECT producto.nombre_producto, proveedor.nombre_proveedor
97 FROM producto
98 INNER JOIN proveedor ON producto.id_proveedor = proveedor.id_proveedor
99 WHERE proveedor.nombre_proveedor = 'Proveedor 1';
```

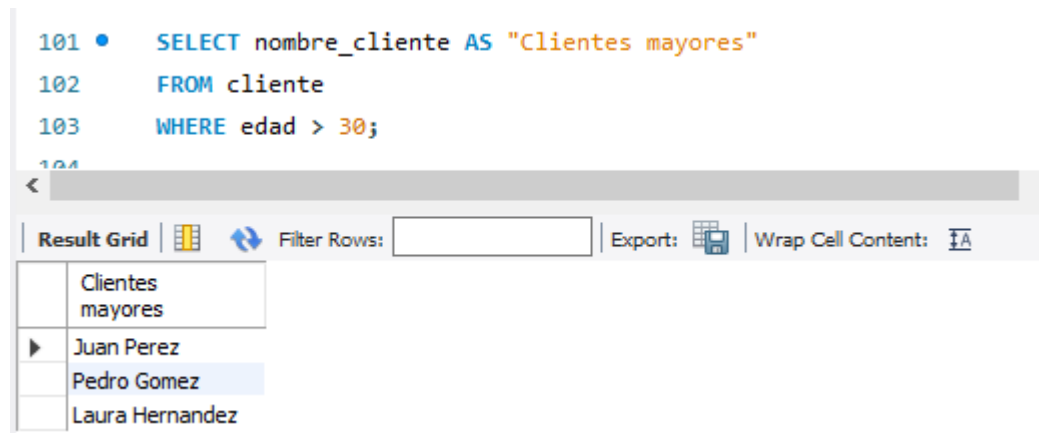


nombre_producto	nombre_proveedor
Producto 1	Proveedor 1
Producto 3	Proveedor 1



10. Consulta que me permite obtener los nombres de los clientes mayores de 30 años podrías hacer

```
SELECT nombre_cliente AS "Clientes mayores"
FROM cliente
WHERE edad > 30;
```



Generar de 4 a 6 vistas donde se evidencie lo más importante de cada ejercicio (haga una selección muy responsable de la información realmente importante según el contexto).

1. Vista que me permite ver el empleado que atendió un cliente

```
1 • CREATE VIEW VISTA_CLIENTE_EMPLEADO_SERVICIO AS
2 SELECT cliente.nombre_cliente, cliente.cedula_cliente, cliente.correo, cita.fecha, cita.hora,
3 cita.descripcion, servicio.tipo, empleado.nombre_empleado
4 FROM servicio
5 INNER JOIN cita ON servicio.id_cita = cita.id_cita
6 INNER JOIN reserva ON cita.id_reserva = reserva.id_reserva
7 INNER JOIN cliente ON reserva.id_cliente = cliente.id_cliente
8 INNER JOIN historial ON servicio.id_servicio = historial.id_servicio
9 INNER JOIN empleado ON historial.id_empleado = empleado.id_empleado;
```

2. Vista que me permite ver todo el ciclo de cita, reserva, servicio, insumo, factura me muestra los datos correspondientes

```

12 • CREATE VIEW VISTA_CLIENTE_SERVICIO_FACTURA AS
13 SELECT cliente.nombre_cliente, cliente.cedula_cliente, cliente.correo, cita.fecha, cita.hora,
14 cita.descripcion, servicio.tipo, empleado.nombre_empleado, factura.valor, factura.fecha_factura
15 FROM servicio
16 INNER JOIN cita ON servicio.id_cita = cita.id_cita
17 INNER JOIN reserva ON cita.id_reserva = reserva.id_reserva
18 INNER JOIN cliente ON reserva.id_cliente = cliente.id_cliente
19 INNER JOIN historial ON servicio.id_servicio = historial.id_servicio
20 INNER JOIN empleado ON historial.id_empleado = empleado.id_empleado
21 INNER JOIN insumo_servicio ON servicio.id_servicio = insumo_servicio.id_servicio
22 INNER JOIN insumo ON insumo_servicio.id_insumo = insumo.id_insumo
23 INNER JOIN insumo_factura ON insumo.id_insumo = insumo_factura.id_insumo
24 INNER JOIN factura ON insumo_factura.id_factura = factura.id_factura;

```

3. Vista que me permite ver el cliente que mas veces me ha visita y cuantas veces

```

27 • CREATE VIEW CLIENTE_MAS_VISITAS AS
28 SELECT cliente.nombre_cliente, COUNT(*) AS num_servicios
29 FROM cliente
30 INNER JOIN reserva ON cliente.id_cliente = reserva.id_cliente
31 INNER JOIN cita ON reserva.id_reserva = cita.id_reserva
32 INNER JOIN servicio ON cita.id_cita = servicio.id_cita
33 GROUP BY cliente.id_cliente
34 ORDER BY num_servicios DESC
35 LIMIT 1;

```

4. Vista que me permite ver que empleado asesora a un cliente en una compra de un producto

```

38 • CREATE VIEW CLIENTE_EMPLEADO_ASESORA_VENTA AS
39 SELECT cliente.nombre_cliente, factura.valor, factura.fecha_factura, producto.nombre_producto, empleado.nombre_empleado
40 FROM producto
41 INNER JOIN compra ON producto.REF = compra.REF
42 INNER JOIN cliente ON compra.id_cliente = cliente.id_cliente
43 INNER JOIN producto_factura ON producto.REF = producto_factura.REF
44 INNER JOIN factura ON producto_factura.id_factura = factura.id_factura
45 INNER JOIN empleado ON factura.id_empleado = empleado.id_empleado;
46

```

5. Vista que me permite ver los proveedores según mi criterio de búsqueda en este caso por nombre del proveedor

```

48 • CREATE VIEW PRODUCTOS_PROVEEDOR AS
49 SELECT producto.nombre_producto, proveedor.nombre_proveedor
50 FROM producto
51 INNER JOIN proveedor ON producto.id_proveedor = proveedor.id_proveedor
52 WHERE proveedor.nombre_proveedor = 'Proveedor 1';
53

```

6. Vista que me permite ver la cantidad de productos que compro un cliente y su factura asociada a dicha compra.

```

55 • CREATE VIEW PRODUCTOS_COMPRADO_CLIENTE AS
56 SELECT COUNT(*) AS cantidad_productos, cliente.nombre_cliente, factura.valor, factura.fecha_factura, producto.nombre_producto
57 FROM producto
58 INNER JOIN compra ON producto.REF = compra.REF
59 INNER JOIN cliente ON compra.id_cliente = cliente.id_cliente
60 INNER JOIN producto_factura ON producto.REF = producto_factura.REF
61 INNER JOIN factura ON producto_factura.id_factura = factura.id_factura
62 GROUP BY cliente.nombre_cliente, factura.valor, factura.fecha_factura, producto.nombre_producto;

```

Generar al menos 4 procedimientos almacenados.

### Procedimiento insertar un cliente

```

1 • use barber;
2 DELIMITER //
3 CREATE PROCEDURE insertar_cliente(IN id INT, IN nombre varchar(50), IN cedula varchar(20), IN profesion varchar(50), IN correo varchar(50),
4 IN edad int, IN direccion varchar(50)
5 )
6 BEGIN
7     INSERT INTO cliente (id_cliente,nombre_cliente, cedula_cliente, profesion, correo, edad, direccion)
8     VALUES (id, nombre, cedula, profesion, correo, edad, direccion);
9 END;
10
11 call insertar_cliente('58','YEISON', '1010110', 'INGENIERO', 'YEISON@GMAIL.COM', 12, 'INJNCD');
12

```

### Procedimiento leer cliente

```

14 DELIMITER //
15 CREATE PROCEDURE leer_cliente(
16     IN id int
17 )
18 BEGIN
19     SELECT * FROM cliente WHERE id_cliente = id;
20 END;
--

```

### Procedimiento borrar un cliente

```

24 DELIMITER //
25 • CREATE PROCEDURE borrar_cliente(
26     IN id int
27 )
28 BEGIN
29     DELETE FROM cliente WHERE id_cliente = id;
30 END //
31 DELIMITER ;
32
33 • call borrar_cliente(58);
--

```

### Procedimiento para actualizar cliente

```

37 DELIMITER //
38 • CREATE PROCEDURE actualizar_cliente(
39     IN id int,
40     IN nombre varchar(50),
41     IN cedula varchar(20),
42     IN profesion varchar(50),
43     IN correo varchar(50),
44     IN edad int,
45     IN direccion varchar(50)
46 )
47 BEGIN
48     UPDATE cliente SET
49         nombre_cliente = nombre,
50         cedula_cliente = cedula,
51         profesion = profesion,
52         correo = correo,
53         edad = edad,
54         direccion = direccion
55     WHERE id_cliente = id;
56 END; //
57
58 • call actualizar_cliente(58, 'YEISON FERNEY', 'OSORIO BUITRAGO', 'QA', 'JEISON@GMAIL.COM', 10, 'CALLE VIRIGLIO');
--

```

## Generar al menos 4 Triggers

### Trigger para insert en la tabla cliente

```

3 • create table control_de_cambios_barber(
4     nombre varchar(50),
5     accion varchar(30),
6     fecha datetime default current_timestamp
7 );
8
9 -- trigger insert
10 DELIMITER //
11 • CREATE TRIGGER control_insert
12 AFTER INSERT ON cliente
13 FOR EACH ROW
14 BEGIN
15     INSERT INTO control_de_cambios_barber (usuario, accion, fecha)
16     VALUES (user(), 'insert', NOW());
17 END;
18 //
--

```

**Trigger para eliminar en la tabla cliente**

```

26  -- eliminar trigger
27
28  DELIMITER //
29 • CREATE TRIGGER control_delete
30  AFTER DELETE ON cliente
31  FOR EACH ROW
32  BEGIN
33      INSERT INTO control_de_cambios_barber (usuario, accion, fecha)
34      VALUES (user(), 'delete', NOW());
35  END;
36  //
37
38
39 • call borrar_cliente(58);

```

**Trigger para actualizar en la tabla paciente**

```

43  -- trigger actualizar
44  DELIMITER //
45 • CREATE TRIGGER control_update
46  AFTER UPDATE ON cliente
47  FOR EACH ROW
48  BEGIN
49      INSERT INTO control_de_cambios_barber (usuario, accion, fecha)
50      VALUES (user(), 'update', NOW());
51  END;
52  //
53  DELIMITER ;
54
55
56 • call actualizar_cliente(70, 'YEISON FERNEY', 'OSORIO BUITRAGO', 'QA', 'JEISON@GMAIL.COM', 10, 'CALLE VIRIGLIO');
57

```

Poblar la base de datos (50 registros por tabla) utilizando una conexión desde Java.

**Representación de las entidades**

- La Inserción de Datos mediante la creación de los modelos de cada entidad, como se ve en la imagen tenemos la primera inserción aleatoria donde usamos la librería Java Faker, la cual nos genera una gran cantidad de datos aleatorios para poder poblar nuestra base de datos, usamos los datos de conexión de nuestra base de datos

```

package com.softkau;

import ...

public class Main {
    1 usage
    private static final String SERVER = "localhost";
    1 usage
    private static final String DATA_BASE_NAME = "barber";
    1 usage
    private static final String USER = "root";
    1 usage
    private static final String PASSWORD = "root";

    7 usages
    private static final MySQLoperation mySqlOperation = new MySQLoperation();

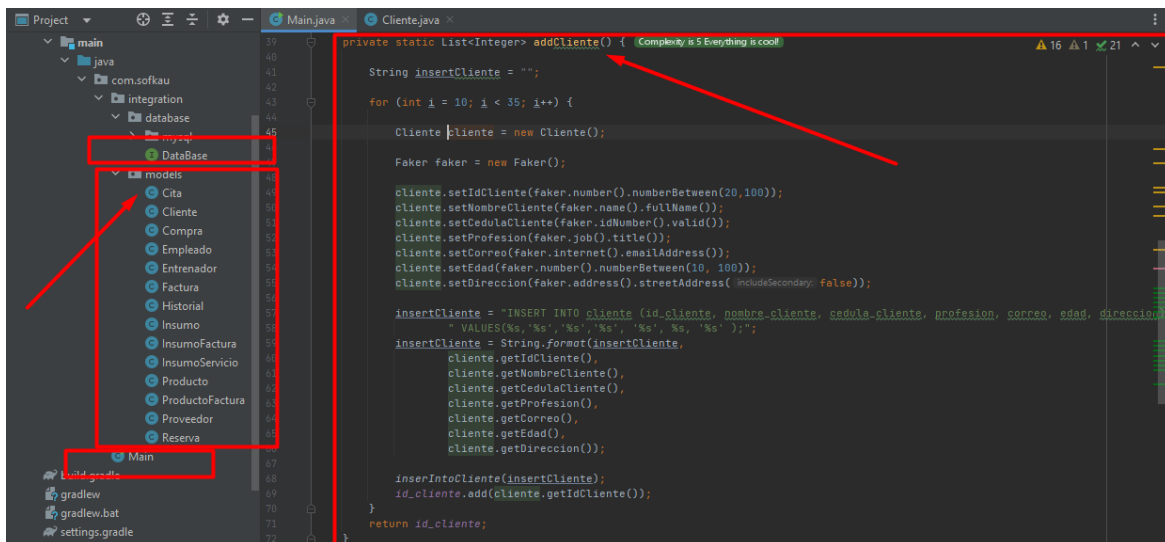
    2 usages
    public static List<Integer> id_cliente = new ArrayList<>();

    public static void main(String[] args) throws SQLException {

```

El método que nos permite hacer la inserción de datos a la base de datos, donde accedemos a sus propiedades por medio de los métodos Get y Set de cada atributo y realizamos la query para insertar los datos.

### Método para insertar datos a la entidad empleado



The screenshot shows an IDE with the project structure on the left and the `addCliente()` method in `Cliente.java` on the right. The project structure includes a `models` package with various entities like `Cita`, `Cliente`, `Compra`, `Empleado`, `Entrenador`, `Factura`, `Historial`, `Insumo`, `InsumoFactura`, `InsumoServicio`, `Producto`, `ProductoFactura`, `Proveedor`, `Reserva`, and `Main`. The `addCliente()` method in `Cliente.java` is highlighted with a red box and a red arrow pointing to it. The method is annotated with "Complexity is 5 Everything is cool!".

```

private static List<Integer> addCliente() {
    String insertCliente = "";
    for (int i = 10; i < 35; i++) {
        Cliente cliente = new Cliente();
        Faker faker = new Faker();

        cliente.setIdCliente(faker.number().numberBetween(20, 100));
        cliente.setNombreCliente(faker.name().fullName());
        cliente.setCedulaCliente(faker.idNumber().valid());
        cliente.setProfesion(faker.job().title());
        cliente.setCorreo(faker.internet().emailAddress());
        cliente.setEdad(faker.number().numberBetween(10, 100));
        cliente.setDireccion(faker.address().streetAddress(includeSecondary: false));

        insertCliente = "INSERT INTO cliente (id_cliente, nombre_cliente, cedula_cliente, profesion, correo, edad, direccion) "
            + "VALUES(%s, '%s', '%s', '%s', '%s', %s, '%s')";
        insertCliente = String.format(insertCliente,
            cliente.getIdCliente(),
            cliente.getNombreCliente(),
            cliente.getCedulaCliente(),
            cliente.getProfesion(),
            cliente.getCorreo(),
            cliente.getEdad(),
            cliente.getDireccion());

        insertIntoCliente(insertCliente);
        id_cliente.add(cliente.getIdCliente());
    }
    return id_cliente;
}

```

**Representación de la entidad cliente** en java donde tenemos sus atributos correspondientes, id cliente, nombre, correo etc. Donde vemos sus constructores correspondientes

```

public class Cliente { //Complexity is 3 Everything is cool!
    //3 usages
    private int idCliente;
    //3 usages
    private String nombreCliente;
    //3 usages
    private String cedulaCliente;
    //3 usages
    private String profesion;
    //3 usages
    private String correo;
    //3 usages
    private int edad;
    //3 usages
    private String direccion;

    public Cliente(int idCliente, String nombreCliente, String cedulaCliente, String profesion, String correo, int edad, String direccion) {
        this.idCliente = idCliente;
        this.nombreCliente = nombreCliente;
        this.cedulaCliente = cedulaCliente;
        this.profesion = profesion;
        this.correo = correo;
        this.edad = edad;
        this.direccion = direccion;
    }

    public Cliente() {}
}

```

**Inserción de los primeros datos a la base de datos** algo que me parece muy útil al momento de realizar gran cantidad de datos y de practicar la conexión a la base de datos

Navigator

SCHEMAS

Filter objects

animales

barber

cliente

compra

control\_de\_cambios\_barber

empleado

factura

historial

insumo

insumo\_factura

insumo\_servicio

producto

producto\_factura

proveedor

Reserva

Administration

Schemas

Information

Table: cliente

Columns:

id\_cliente int PK

nombre\_cliente varchar(50)

cedula\_cliente varchar(20)

profesion varchar(50)

correo varchar(50)

1 \* SELECT \* FROM barber.cliente;

Datos generados aleatoriamente por medio de la libreria

id_cliente	nombre_cliente	cedula_cliente	profesion	correo	edad	direccion
4	Laura Hernandez	3216549870	Arquitecta	laura.hernandez@yahoo.com	31	Calle 789
5	Andres Torres	9876543210	Contador	andrestorres@gmail.com	29	Calle 456
10	Allan Renda	Brandon Abernathy	Joshua	51	Gottlieb	
15	Amado Lizette	Shira Glover	Tyree	42	Rempel	
21	Joey Quizon	098-84-2344	Future Retail Administrator	permi.waters@hotmail.com	77	9688 Marquardt River
23	Otto Boehm	844-99-7627	Manufacturing Planner	hilda.schinner@yahoo.com	78	6062 Xenia Drive
25	Elna Bosco	710-96-5024	Marketing Strategist	ferne.kunde@gmail.com	92	5054 Glover Corners
26	Amado Nenow	020-16-6180	Farming Planner	mel.herman@yahoo.com	25	6833 Stronan Wall
28	Mr. Josef Stolte	494-06-9916	Advertising Analyst	leandra.bogisch@hotmail.com	62	78231 Long Islands
32	Lavonne Hintz III	056-16-5707	Future Consulting Associate	alva.vundt@yahoo.com	15	Apt. 618 50918 Suzy...
33	Brenton Nella	Kristan Schimmel	Michael	14	Flatley	
34	Astrid Flatley	063-97-3982	Construction Executive	adalberto.luelwitz@yahoo.com	44	Apt. 980 783 Boyle R...
37	Louanne Mohr	651-60-6137	Corporate Construction C...	sol.stehr@hotmail.com	79	005 Predovic Unions,...
38	Evon Nietzsche Sr.	739-10-7474	Central Facilitator	brooks.hayes@gmail.com	80	922 Ferry Branch
45	Chrylle Alden	Stefanie Hannelore	Mel Maczajovic	Scotty	75	Nikolaus Senger

## Método para insertar datos a la entidad empleado

Project

Main.java

Empleado.java

Cliente.java

main

java

com.softau

integration

database

mysql

DataBase

models

Cita

Cliente

Compra

Empleado

Entrenador

Factura

Historial

Insumo

InsumoFactura

InsumoServicio

Producto

ProductoFactura

Proveedor

Reserva

79

private static List<Integer> addEmpleado() { //Complexity is 5 Everything is cool!

String insertEmpleado = "";

for (int i = 10; i < 80; i++) {

Empleado empleado = new Empleado();

Faker faker = new Faker();

empleado.setIdEmpleado(faker.number().numberBetween(20, 100));

empleado.setCedula(faker.idNumber().valid());

empleado.setNombreEmpleado(faker.name().fullName());

empleado.setEspecialidad(faker.job().title());

insertEmpleado = "INSERT INTO empleado (id\_empleado, cedula, nombre\_empleado, especialidad)" +

" VALUES(%, '%', '%', '%')";

insertEmpleado = String.format(insertEmpleado,

empleado.getIdEmpleado(),

empleado.getCedula(),

empleado.getNombreEmpleado(),

empleado.getEspecialidad());

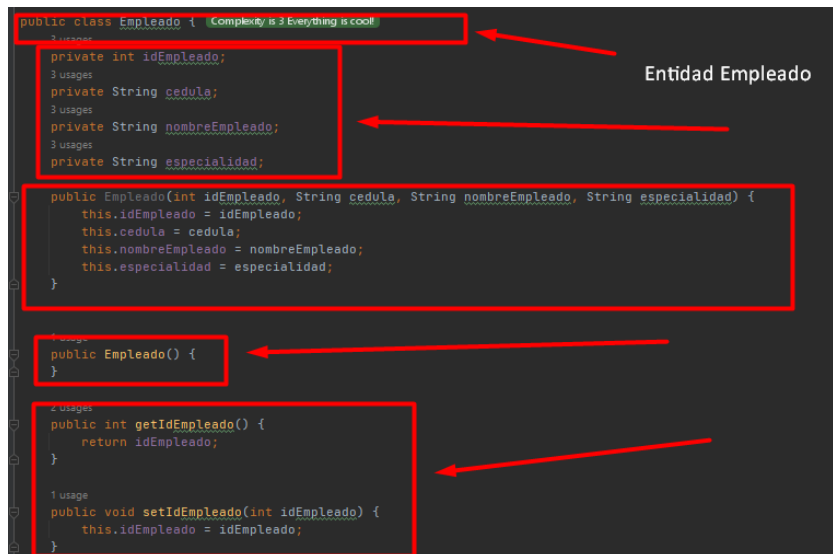
insertIntoEmpleado(insertEmpleado);

id\_empleado.add(empleado.getIdEmpleado());

}

return id\_empleado;

**Representación de la entidad Empleado** en java donde tenemos sus atributos correspondientes, idempleado, nombre, profrsion, etc. Donde vemos sus constructores correspondientes



The screenshot shows the Java code for the `Empleado` class. Red boxes and arrows highlight specific parts of the code:

- Entity Declaration:** A red box around `public class Empleado {` is labeled "Entidad Empleado".
- Attributes:** A red box around the private attributes `private int idEmpleado;`, `private String cedula;`, `private String nombreEmpleado;`, and `private String especialidad;` is highlighted.
- Constructor:** A red box around the constructor `public Empleado(int idEmpleado, String cedula, String nombreEmpleado, String especialidad) {` is highlighted.
- Getter and Setter:** A red box around the `getIdEmpleado()` and `setIdEmpleado()` methods is highlighted.

**Inserción de los primeros datos a la base de datos:** algo que me parece muy útil al momento de realizar gran cantidad de datos y de practicar la conexión a la base de datos

```
1 • SELECT * FROM barber.empleado;
```

<

Result Grid

Filter Rows:

Edit:

Export/Import:

Wrap Cell Contents:

	id_empleado	cedula	nombre_empleado	especialidad
22	089-40-5694	Mr. Leandra Ernser	Central Construction Agent	
23	193-96-0994	Jared Hettinger	Community-Services Coordinator	
24	455-76-2871	Garth Crist	Manufacturing Strategist	
26	565-04-9295	Harry McClure	Human IT Technician	
28	690-76-7238	Dorinda Bahringer	Direct Construction Director	
32	279-45-6764	Elna Gaylord	Future Legal Planner	
33	891-26-6879	Mignon Collier	Advertising Planner	
34	694-87-4908	Edgar Anderson MD	Manufacturing Coordinator	
35	698-30-5471	Mr. Kourtney Saw...	Hospitality Technician	
36	302-18-1178	Danilo Schimmel	Forward Sales Manager	
38	769-59-2150	Alsa Bergstrom	IT Planner	
40	184-91-0977	Conception Kutch ...	Lead Associate	
41	362-81-2068	Elana Monahan	Construction Facilitator	
42	563-62-6285	Glynis Rutherford	Construction Executive	
43	337-72-2330	Cherise Watsica	District Supervisor	
46	722-92-9759	Ms. Majorie Lemke	Investor Design Facilitator	
47	597-56-8032	Joseph Berge	Regional Coordinator	
48	597-07-6746	Zachariah Rowe	Senior Legal Manager	
49	429-02-0468	Demetrius Mertz	Education Consultant	
50	629-55-6767	Branden Conroy	Mining Planner	
53	782-38-7217	Wilbert Hintz Sr.	Legacy Supervisor	
55	475-43-6426	Zetta Rosenbaum	Legacy Engineer	
56	434-67-8551	Julio Wisoky	Legal Technician	
59	199-16-7882	Kory Harvey	Senior Supervisor	

Registro de empleados  
Aleatorios





1. `SELECT * FROM barber.insumo;`

id_insumo	nombre_insumo	descripcion	cantidad
1	Rustic Plastic Computer	Ut aperiam est omnis deserunt et tempora corp...	20
2	Small Steel Bench	Exercitationem voluptatem minus consequatur l...	2
3	Intelligent Steel Knife	Aut iusto architecto enim explicabo consequunt...	45
4	Incredible Bronze Gloves	Deserunt et voluptatem est eos omnis tenetur q...	54
5	Awesome Marble Hat	Nel corrupti voluptas doloribus sed est molestias...	46
6	Aerodynamic Rubber Car	Esse ipsa enim dolore rem provident laudantbu...	73
7	Small Rubber Keyboard	At sed esse et voluptatibus odio molestiae et et...	84
8	Mediocre Wool Bench	Animi facere dignissimos ullam sit doloreque no...	70
9	Mediocre Cotton Pants	Sequi cumque illo sed similique omnis ipsa nulla f...	35
10	Gorgeous Wool Gloves	Quia in qui mollitia consequatur et maiores volup...	77
11	Synergistic Aluminum Bo...	Tenetur nam qui exercitationem aut distinctio e...	3
12	Rustic Iron Computer	Quam qui dolorum quo architecto fugiat mollitia ...	57
13	Mediocre Bronze Bottle	Dolore magnam dolores cum laborum tempora p...	44
14	Lightweight Wool Gloves	Ipsa distinctio similique voluptatem ipsam porro ...	55
15	Heavy Duty Plastic Knife	Inventore officia fugiat soluta error quas odio a...	39
16	Heavy Duty Copper Chair	Fuga magni similique consectetur aliquam repudi...	49
17	Fantastic Cotton Gloves	Quia sunt similique ea itaque eos totam aut ut e...	64
18	Ergonomic Aluminum Knife	Qui et cumque consectetur dolores dolorum quo...	12

Output

#	Time	Action	Message
716	19:31:29	SELECT * FROM barber.insumo LI...	10 row(s) returned
717	19:32:30	SELECT * FROM barber.insumo LI...	50 row(s) returned

## Método para insertar datos a la entidad Reserva

```

1 usage
2 private static List<Integer> addReserva() { (Complexity: 15.5 Everything is cool)
3
4     String insertReserva = "";
5     Faker faker = new Faker();
6     for (int i = 0; i <= 20; i++) {
7         Reserva reserva = new Reserva();
8         reserva.setId_reserva(i);
9         reserva.setId_cliente(i);
10        reserva.setEstado(faker.options().option(...options: "Pendiente", "Confirmada", "Cancelada"));
11        java.sql.Date fecha = new java.sql.Date(faker.date().past(...atMost: 1, TimeUnit.DAYS).getTime());
12        reserva.setFecha_reserva(fecha);
13        reserva.setDescripcion(faker.lorem().sentence(...wordCount: 5));
14
15        insertReserva = "INSERT INTO reserva (id_reserva, id_cliente, estado, fecha_reserva, descripcion)" +
16            " VALUES('" + reserva.getId_reserva() + "','" + reserva.getId_cliente() + "','" + reserva.getEstado() + "','" +
17            reserva.getFecha_reserva() + "','" + reserva.getDescripcion() + "');";
18        insertReserva = String.format(insertReserva,
19            reserva.getId_reserva(),
20            reserva.getId_cliente(),
21            reserva.getEstado(),
22            reserva.getFecha_reserva(),
23            reserva.getDescripcion());
24
25        insertIntoReserva(insertReserva);
26    }
27    return id_Reserva;
28 }

```

**Representación de la entidad Reserva** en java donde tenemos sus atributos correspondientes, idCita, idReserva, fecha, etc. Donde vemos sus constructores correspondientes.

```

public class Reserva { Complexity is 3 Everything is cool.
    private int id_reserva;
    private int id_cliente;
    private String estado;
    private Date fecha_reserva;
    private String descripcion;

    public Reserva() {
    }

    public Reserva(int id_reserva, int id_cliente, String estado, Date fecha_reserva, String descripcion) {
        this.id_reserva = id_reserva;
        this.id_cliente = id_cliente;
        this.estado = estado;
        this.fecha_reserva = fecha_reserva;
        this.descripcion = descripcion;
    }

    public int getId_reserva() {
        return id_reserva;
    }

    public void setId_reserva(int id_reserva) {
        this.id_reserva = id_reserva;
    }
}

```

Entidad Reserva

**Inserción de los primeros datos a la base de datos** algo que me parece muy útil al momento de realizar gran cantidad de datos y de practicar la conexión a la base de datos

**SCHEMAS**

Filter objects

- animales
- barber**
  - cita
  - cliente
  - compra
  - control\_de\_cambios\_barber
  - empleado
  - factura
  - historial
  - insumo
  - insumo\_factura
  - insumo\_servido
  - producto
  - producto\_factura
  - proveedor
  - reserva**
  - servicio
  - telefono\_cliente
  - telefono\_proveedor

Administration Schemas

Information

Table: **reserva**

Columns:

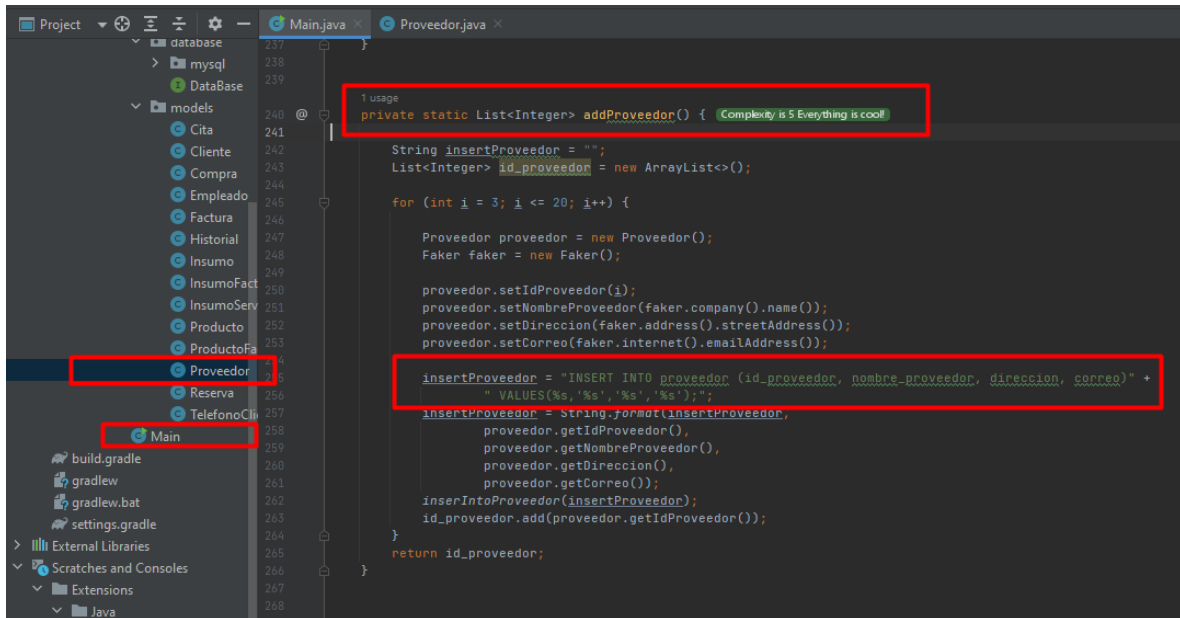
- id\_reserva int PK
- id\_cliente int
- estado varchar(50)
- fecha\_reserva date
- descripcion varchar(255)

1 • **SELECT \* FROM barber.reserva;**

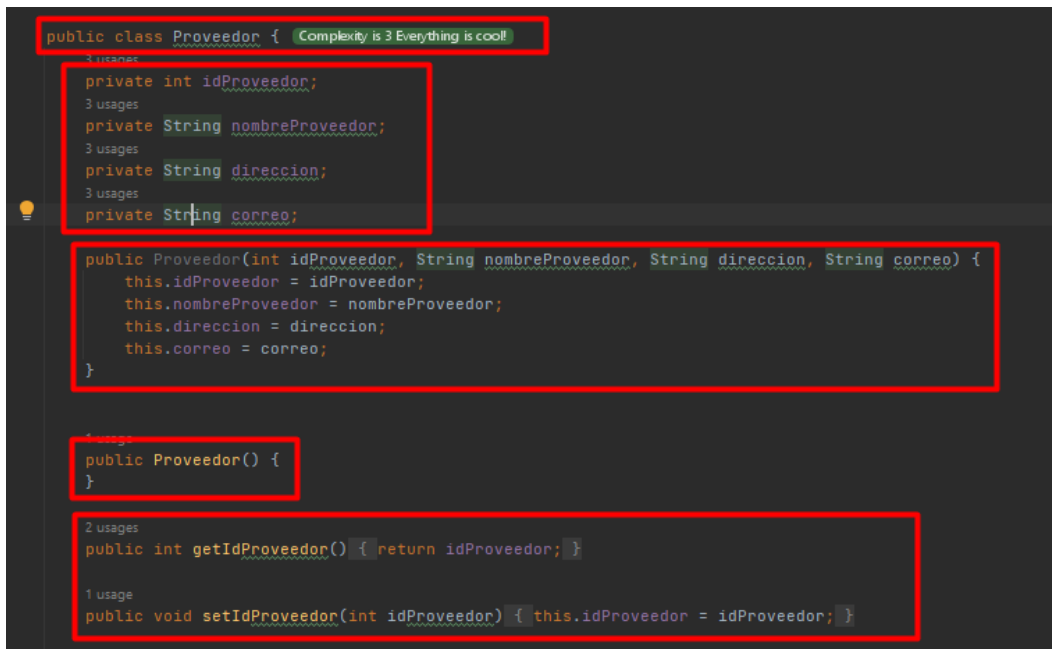
Result Grid

	id_reserva	id_cliente	estado	fecha_reserva	descripcion
1	1	1	Pendiente	2023-02-21	Reserva para limpieza dental
2	2	2	Confirmada	2023-02-23	Reserva para consulta médica
3	3	3	Pendiente	2023-02-25	Reserva para asesoría legal
4	4	4	Cancelada	2023-02-20	Reserva para diseño de interiores
5	5	5	Confirmada	2023-02-26	Reserva para asesoría contable
6	6	6	Confirmada	2023-02-17	Explicabo dolores voluptas beatae adipisci ea m...
7	7	7	Cancelada	2023-02-17	Facilis vitae cupiditate iure ut sunt tempora eos ...
8	8	8	Cancelada	2023-02-17	Architecto qui perferendis illo praesentium est e...
9	9	9	Cancelada	2023-02-18	Fuga eum dignissimos molestias vero aut eos re...
10	10	10	Confirmada	2023-02-18	Consequatur quidem aliquid saepe fugiat labore...
11	11	11	Confirmada	2023-02-18	Rem id voluptatem voluptas nihil consequuntur ...
12	12	12	Cancelada	2023-02-17	Autem quibusdam est dolores incidunt.
13	13	13	Cancelada	2023-02-17	Itaque quaeat et fugit quia.
14	14	14	Cancelada	2023-02-18	Ab aut est qui ab ut.
15	15	15	Pendiente	2023-02-17	Error et perferendis aut modi animi minima quia ...
16	16	16	Confirmada	2023-02-17	Dolores adipisci laborum perferendis voluptas vel.
17	17	17	Cancelada	2023-02-17	Dolor eum ea ea voluptatem deleniti laudantium ...
18	18	18	Cancelada	2023-02-17	Voluptates fuga quia illo quia ut.

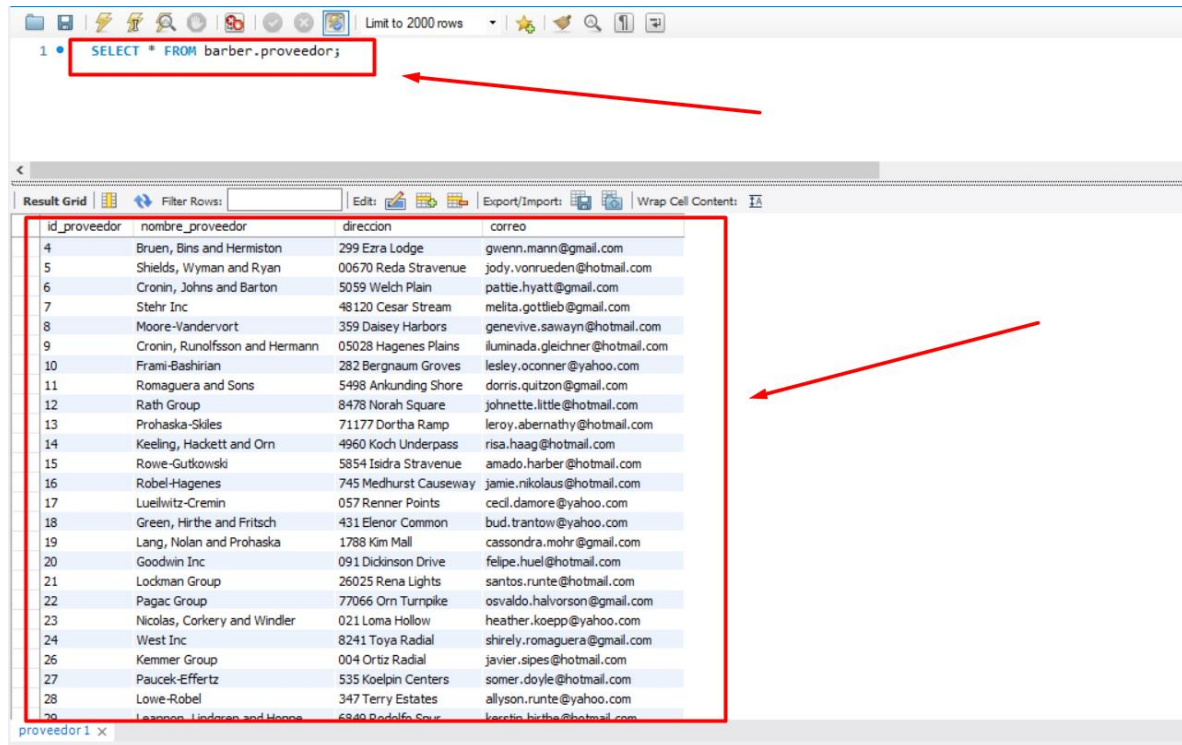
## Método para insertar datos a la entidad Proveedor



**Representación de la entidad Proveedor** en java donde tenemos sus atributos correspondientes, idProveedor, nombreProveedor, fecha, etc. Donde vemos sus constructores correspondientes.



**Inserción de los primeros datos a la base de datos** algo que me parece muy útil al momento de realizar gran cantidad de datos y de practicar la conexión a la base de datos



Limit to 2000 rows

1 • `SELECT * FROM barber.proveedor;`

id_proveedor	nombre_proveedor	direccion	correo
4	Bruen, Bins and Hermiston	299 Ezra Lodge	gwenn.mann@gmail.com
5	Shields, Wyman and Ryan	00670 Reda Stravenue	jody.vonrueden@hotmail.com
6	Cronin, Johns and Barton	5059 Welch Plain	pattie.hyatt@gmail.com
7	Stehr Inc	48120 Cesar Stream	melita.gottlieb@gmail.com
8	Moore-Vandervort	359 Daisey Harbors	genevive.sawayn@hotmail.com
9	Cronin, Runolfsson and Hermann	05028 Hagenes Plains	iluminada.gleichner@hotmail.com
10	Frami-Bashirian	282 Bergnaum Groves	lesley.oconner@yahoo.com
11	Romaguera and Sons	5498 Ankunding Shore	dorris.quitzon@gmail.com
12	Rath Group	8478 Norah Square	johnette.little@hotmail.com
13	Prohaska-Skiles	71177 Dortha Ramp	leroy.abernathy@hotmail.com
14	Keeling, Hackett and Orn	4960 Koch Underpass	risa.haag@hotmail.com
15	Rowe-Gutkowski	5854 Isidra Stravenue	amado.harber@hotmail.com
16	Robel-Hagenes	745 Medhurst Causeway	jamie.nikolaus@hotmail.com
17	Lueilwitz-Cremin	057 Renner Points	cecil.damore@yahoo.com
18	Green, Hirthe and Fritsch	431 Elenor Common	bud.trantow@yahoo.com
19	Lang, Nolan and Prohaska	1788 Kim Mall	cassandra.mohr@gmail.com
20	Goodwin Inc	091 Dickinson Drive	felipe.huel@hotmail.com
21	Lockman Group	26025 Rena Lights	santos.runte@hotmail.com
22	Pagac Group	77066 Orn Turnpike	osvaldo.halvorson@gmail.com
23	Nicolas, Corkery and Windler	021 Loma Hollow	heather.koepp@yahoo.com
24	West Inc	8241 Toya Radial	shirely.romaguera@gmail.com
26	Kemmer Group	004 Ortiz Radial	javier.sipes@hotmail.com
27	Pauczek-Effertz	535 Koelpin Centers	somer.doyle@hotmail.com
28	Lowe-Robel	347 Terry Estates	allyson.runte@yahoo.com
29	Leannon, Lindgren and Hoopes	6840 Redolfo Square	kerstin.hirthe@hotmail.com

veedor1 x

**Nota: Se muestran algunos pantallazos de la base de datos para no llenar el documento de solo img**

✓ **Java 11.0.15**

```
dependencies {
    testImplementation 'org.junit.jupiter:junit-jupiter-api:5.8.1'
    testRuntimeOnly 'org.junit.jupiter:junit-jupiter-engine:5.8.1'
    // https://mvnrepository.com/artifact/com.mysql/mysql-connector-j
    implementation 'com.mysql:mysql-connector-j:8.0.32'

    // https://mvnrepository.com/artifact/com.github.javafaker/javafaker
    implementation 'com.github.javafaker:javafaker:1.0.2'
}
```

Al terminar el ejercicio responda ¿ Está conforme con el resultado obtenido según el contexto o cree que hubiera obtenido un mejor resultado con una base de datos no relacional?

Si, estoy conforme con el modelo y creo que fui un poco más allá de la información especificada ya que el modelo me permite ver desde el cliente que realiza una reserva para una cita de un servicio y empleado que lo atendió en la Barberia y el precio de ese servicio y esos insumos y la factura que le genera, cubre todo el Core del negocio, lo pude observar mediante las consultas.

Crear la base de datos en un modelo no relacional sería más sencillo ya que no necesita modelar la base de datos, pero no sería escalable al largo plazo ya que las bases de datos no relaciones no son tan escalables y no importa muchos la consistencia de los datos al contrario de las bases de datos relacional ya que nos permite tener una integridad de la información y escalar de forma lógica la base de datos y crear unas consultas complejas en el modelo.

Además, una base de datos relacional nos permite establecer relaciones entre las diferentes tablas de la base de datos, lo que facilita el acceso a la información y nos permite realizar consultas complejas que nos ayudan a tomar decisiones más informadas. También nos proporciona la capacidad de hacer consultas a grandes conjuntos de datos y almacenar grandes cantidades de información de manera estructurada y organizada, lo que es importante para la gestión. En resumen, aunque el modelo no relacional puede ser más fácil de implementar en el corto plazo, el modelo relacional ofrece ventajas a largo plazo en términos de escalabilidad, integridad de los datos y capacidad de realizar consultas complejas.

documente muy bien su proceso (paso a paso) en un archivo PDF escriba todas las aclaraciones o especificaciones necesarias para realizar el ejercicio