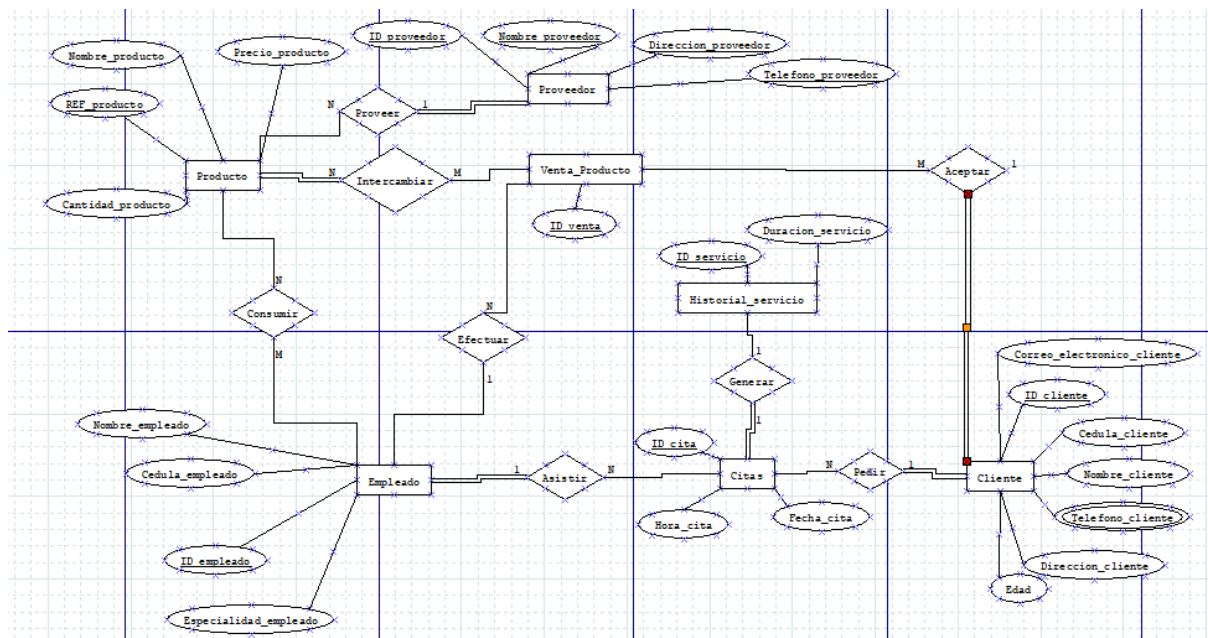


El ejercicio asignado fue el A (Barberia)

1. Se realiza el modelo E/R y queda de esta forma:

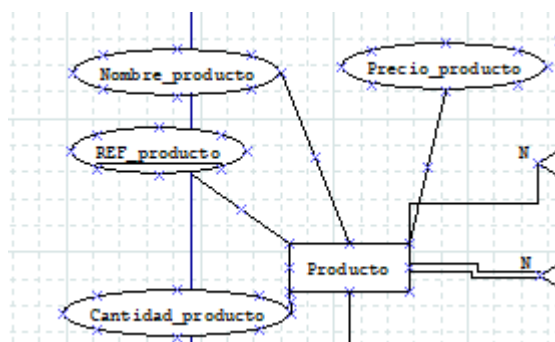
En el análisis para la creación del modelo, se observan dos condiciones muy importantes:

- Los clientes serán atendidos por medio de una cita, no importa si llegan directamente al local y piden un servicio, se les da la cita dependiendo la disponibilidad de los empleados.
- Cuando el empleado atiende una cita, consume x cantidad de productos que están tomados en cuenta en el costo del servicio.

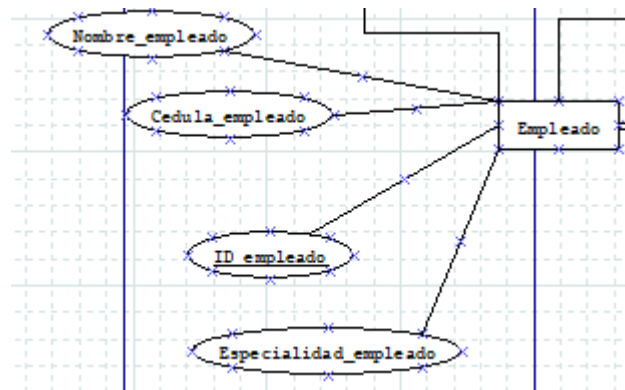


Las entidades involucradas son:

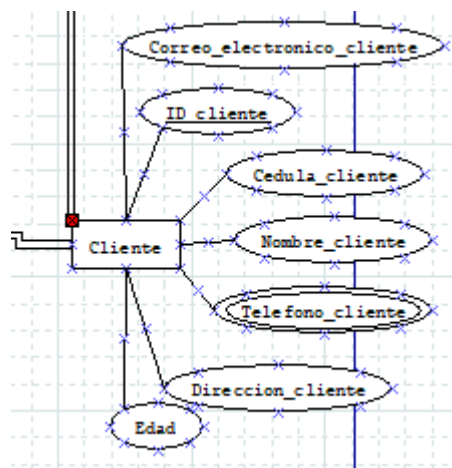
- Producto



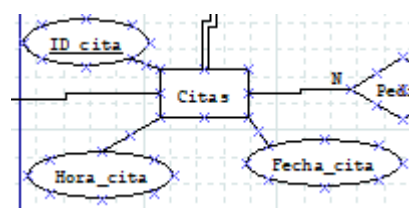
- Empleado



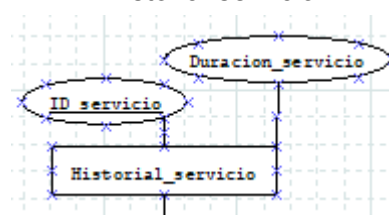
- Cliente



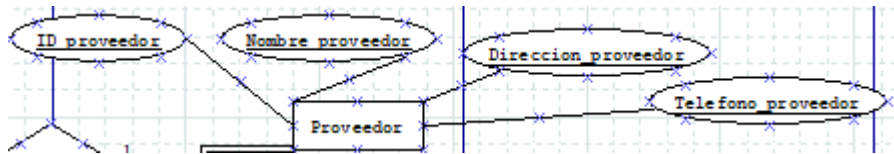
- Cita



- Historial servicio



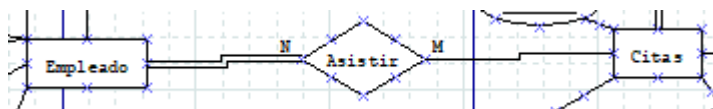
- Proveedor



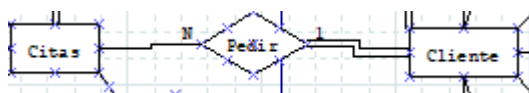
- Venta producto



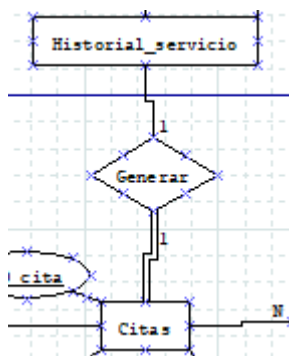
Las relaciones que se pueden evidenciar en el modelo E/R son:



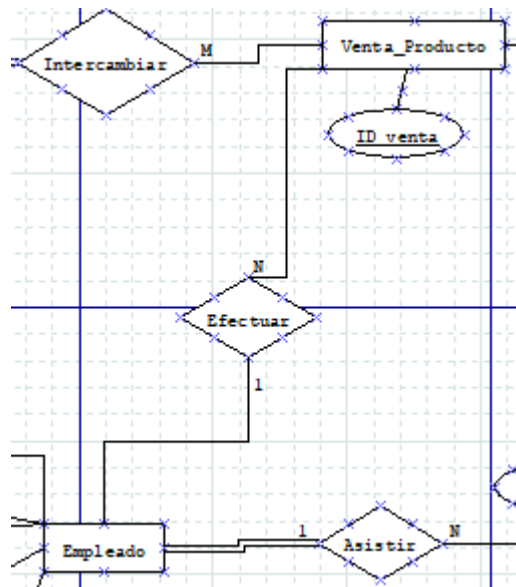
- Un empleado puede asistir a una o varias citas.
- Una cita puede ser asistida por uno empleado o varios empleados.



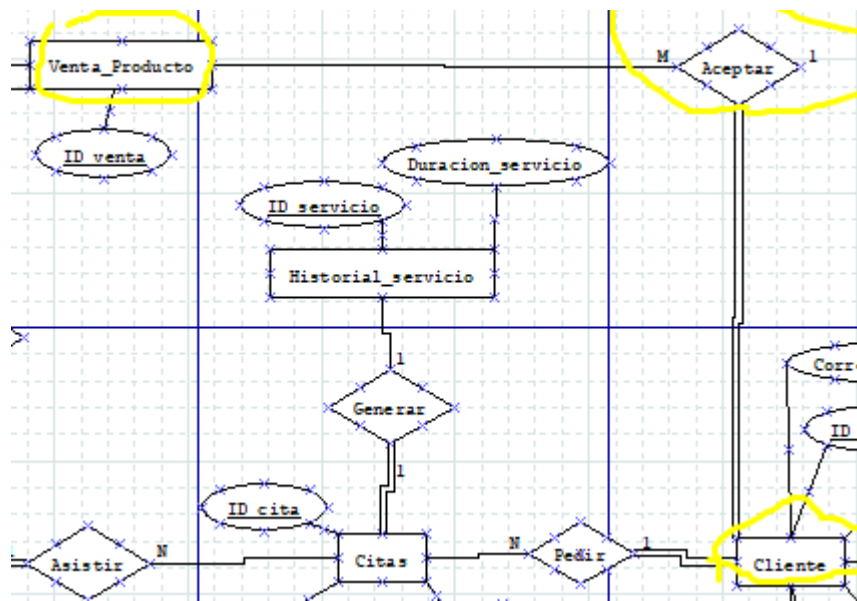
- Una cita puede ser pedida por un cliente.
- Un cliente puede pedir una o varias citas.



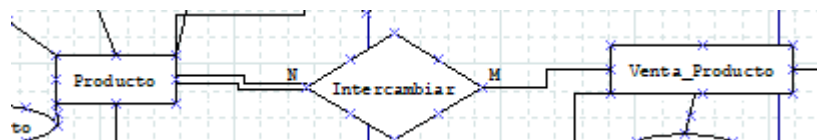
- Una cita puede generar un historial.
- Un historial puede ser generado por una cita.



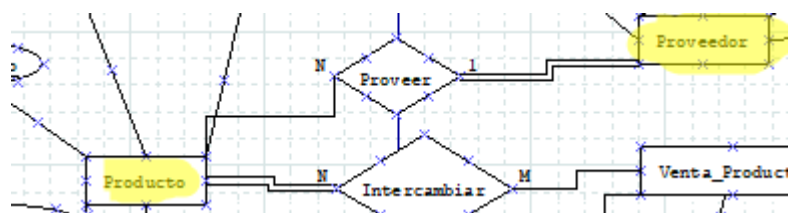
- Un empleado puede efectuar una o varias ventas.
- Una venta puede ser efectuada por un empleado.



- Un cliente puede aceptar una o varias ventas.
- Una venta puede ser aceptada por un cliente.

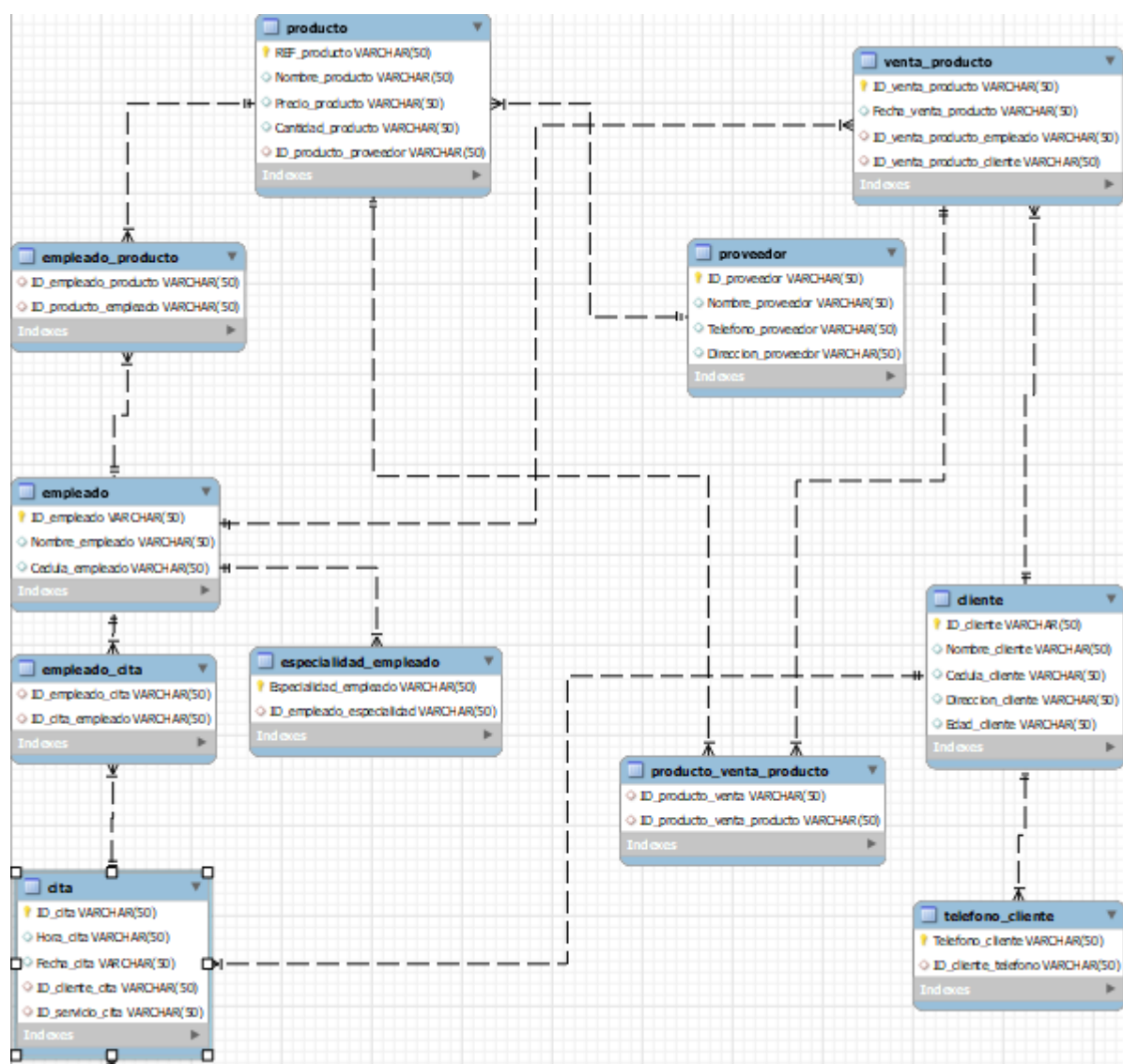


- Un producto puede ser intercambiado en una o varias ventas.
- Una venta puede intercambiar uno o varios productos.



- Un producto puede ser proveído por un proveedor.
- Un proveedor puede proveer uno o varios productos.

2. Se crea modelo relacional y queda de esta forma:



Las tablas del modelo relacional son:

- Producto

| producto | |
|-----------------------|-------------|
| REF_producto | VARCHAR(50) |
| Nombre_producto | VARCHAR(50) |
| Precio_producto | VARCHAR(50) |
| Cantidad_producto | VARCHAR(50) |
| ID_producto_proveedor | VARCHAR(50) |
| Ind. axes | |

- Venta producto

| venta_producto | |
|----------------------------|-------------|
| ID_venta_producto | VARCHAR(50) |
| Fecha_venta_producto | VARCHAR(50) |
| ID_venta_producto_empleado | VARCHAR(50) |
| ID_venta_producto_cliente | VARCHAR(50) |
| Ind. axes | |

- Empleado

| empleado | |
|-----------------|-------------|
| ID_empleado | VARCHAR(50) |
| Nombre_empleado | VARCHAR(50) |
| Cedula_empleado | VARCHAR(50) |
| Ind. axes | |

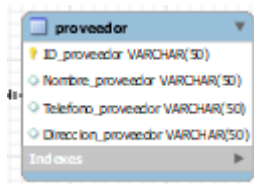
- Cita

| cita | |
|------------------|-------------|
| ID_cita | VARCHAR(50) |
| Hora_cita | VARCHAR(50) |
| Fecha_cita | VARCHAR(50) |
| ID_cliente_cita | VARCHAR(50) |
| ID_servicio_cita | VARCHAR(50) |
| Ind. axes | |

- Cliente

| cliente | |
|-------------------|-------------|
| ID_cliente | VARCHAR(50) |
| Nombre_cliente | VARCHAR(50) |
| Cedula_cliente | VARCHAR(50) |
| Direccion_cliente | VARCHAR(50) |
| Edad_cliente | VARCHAR(50) |
| Ind. axes | |

- Proveedor

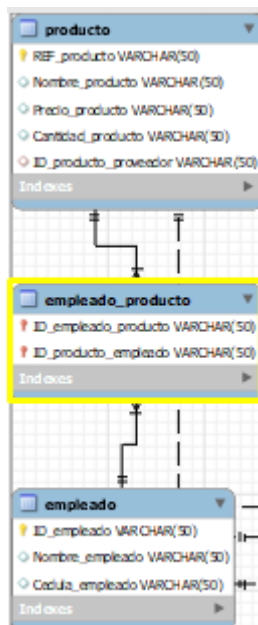


- Historial servicio

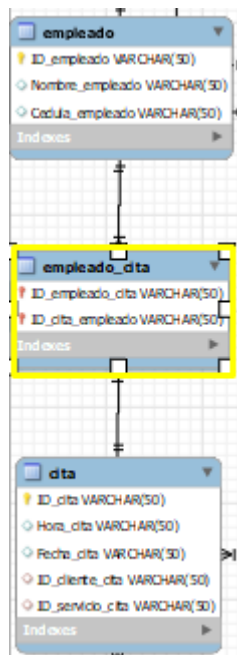


Tomando en cuenta las relaciones entre entidades y los atributos multivaluados se crean nuevas tablas y nuevos atributos en las tablas:

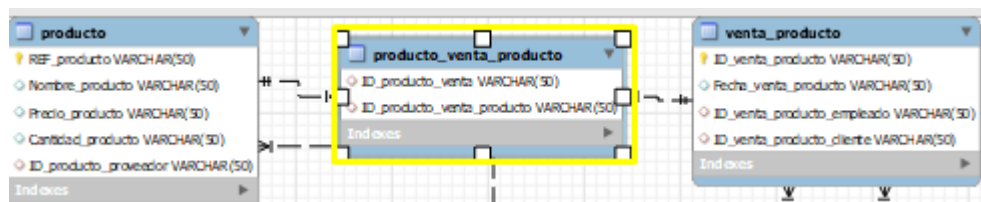
- Tenemos la entidad empleado y producto con una cardinalidad de m/n. Por esta razón se crea otra tabla con sus respectivas llaves primarias:



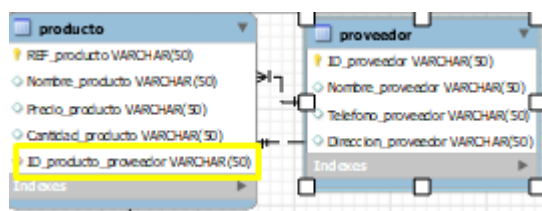
- Tenemos la entidad empleado y cita con una cardinalidad de n/m. Por esta razón se crea otra tabla con las llaves primarias de cada una:



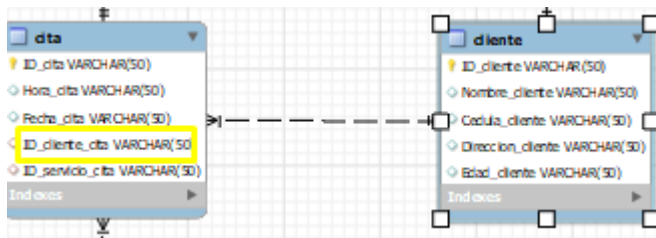
- Tenemos las entidades producto y venta_producto con una cardinalidad n/m. Por esta razón se crea otra tabla con las llaves primarias de cada entidad:



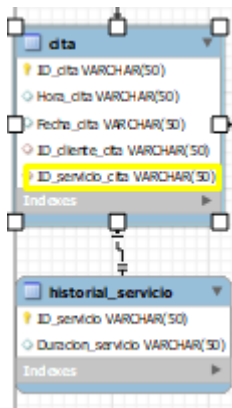
- Tenemos la entidad proveedor y producto con una cardinalidad 1/n. Por esta razón se crea un llave foránea en la entidad producto, referenciada a la llave primaria de la entidad proveedor:



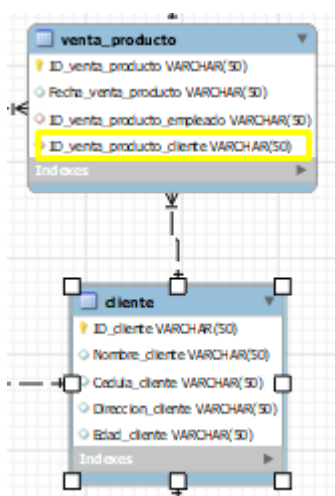
- Tenemos las entidades cita y cliente con una cardinalidad de n/1. Por esta razón se crea una llave foránea en la entidad producto, referenciada a la llave primaria de la entidad cliente:



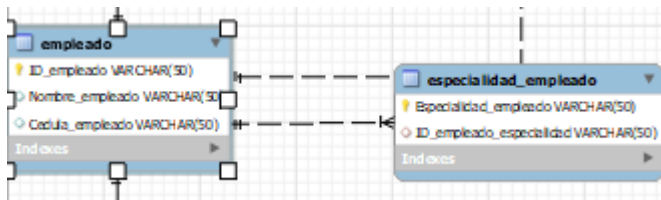
- Tenemos las entidades cita e historial servicio con una cardinalidad de 1/n. Por esta razón se crea una llave foránea en la entidad historial_servicio, referenciada a la llave primaria de la entidad cita:



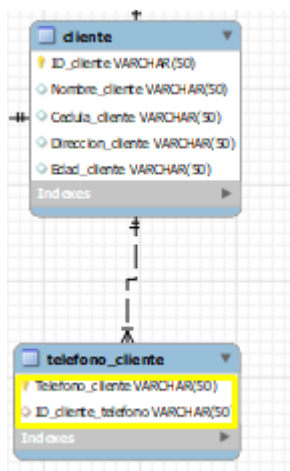
- Tenemos las entidades cliente y venta producto con una cardinalidad de 1/n. Por esta razón se crea una llave foránea en la entidad venta producto, referenciada a la llave primaria de la entidad cliente servicio:



- Tenemos la entidad empleado con un atributo multivaluado. Por esta razón se crea otra tabla con una llave foránea referenciada a la llave primaria de la entidad empleado y se crea el atributo multivaluado como llave primaria:



- Tenemos una entidad cliente con un atributo multivaluado. Por esta razón se crea otra tabla con una llave foránea referenciada a la llave primaria de la entidad cliente y se crea el atributo multivaluado como llave primaria:



Normalización:

1. NF:
 - Todos los atributos tienen valores atómicos.
 - No hay atributos multivaluados.
 - Se eliminaron los registros y columnas duplicadas .
 - Se definen claves primarias.
2. NF:
 - Está en 1NF.
 - Todos los valores de las columnas dependen de la llave primaria de la tabla.
 - Las tablas dependen de una única llave primaria.
3. NF:
 - Está en 2NF.
 - Los atributos que no están incluidos en la llave primaria no dependen de la clave.

Se crea BD(Barbería) con sentencias SQL y queda de esta forma:

- Tabla producto:

```
create table producto(  
  REF_producto varchar(50) primary key,  
  Nombre_producto varchar(50),  
  Precio_producto varchar(50),  
  Cantidad_producto varchar(50),  
  ID_producto_proveedor varchar(50),  
  foreign key (ID_producto_proveedor) references proveedor(ID_proveedor)  
);
```

- Tabla empleado:

```
create table empleado(  
  ID_empleado varchar(50) primary key,  
  Nombre_empleado varchar(50),  
  Cedula_empleado varchar(50)  
);
```

- Tabla producto_venta_producto: Se crea tabla por la cardinalidad n/m entre la entidad producto y venta_producto.

```
create table producto_venta_producto(  
  ID_producto_venta varchar(50),  
  ID_producto_venta_producto varchar(50),  
  foreign key (ID_producto_venta) references producto(REF_producto),  
  foreign key (ID_producto_venta_producto) references venta_producto(ID_venta_producto)  
);
```

- Tabla empleado_producto: Se crea tabla por la cardinalidad n/m entre la entidad producto y empleado.

```
create table empleado_producto(
ID_empleado_producto varchar(50),
ID_producto_empleado varchar(50),
foreign key (ID_empleado_producto) references empleado(ID_empleado),
foreign key (ID_producto_empleado) references producto(REF_producto)
);
```

- Tabla especialidad_cliente : Se crea por el atributo multivaluado especialidad en la entidad cliente.

```
create table especialidad_empleado(
Especialidad_empleado varchar(50) primary key,
ID_empleado_especialidad varchar(50),
foreign key (ID_empleado_especialidad) references empleado(ID_empleado)
);
```

- Tabla cliente:

```
create table cliente(
ID_cliente varchar(50) primary key,
Nombre_cliente varchar(50),
Cedula_cliente varchar(50),
Direccion_cliente varchar(50),
Edad_cliente varchar(50)
);
```

- Tabla telefono_cliente: Se crea tabla por el atributo multivaluado teléfono en la entidad cliente.

```
create table telefono_cliente(
Telefono_cliente varchar(50) primary key,
ID_cliente_telefono varchar(50),
foreign key (ID_cliente_telefono) references cliente(ID_cliente)
);
```

- Tabla cita:

```
create table cita(  
ID_cita varchar(50) primary key,  
Hora_cita varchar(50),  
Fecha_cita varchar(50),  
ID_cliente_cita varchar(50),  
foreign key (ID_cliente_cita) references cliente(ID_cliente)  
);
```

- Tabla empleado_cita: Se crea tabla por la cardinalidad de n/m de la entidad empleado y cita:

```
create table empleado_cita(  
ID_empleado_cita varchar(50),  
ID_cita_empleado varchar(50),  
foreign key (ID_empleado_cita) references empleado(ID_empleado),  
foreign key (ID_cita_empleado) references cita(ID_cita)  
);
```

- Tabla historial_servicio:

```
create table historial_servicio(  
ID_servicio varchar(50) primary key,  
Duracion_servicio varchar(50),  
ID_servicio_cita varchar(50),  
foreign key (ID_servicio_cita) references cita(ID_cita)  
);
```

- Tabla proveedor:

```
create table proveedor(  
ID_proveedor varchar(50) primary key,  
Nombre_proveedor varchar(50),  
Telefono_proveedor varchar(50),  
Direccion_proveedor varchar(50)  
);
```

- Tabla venta producto:

```
create table venta_producto(
ID_venta_producto varchar(50) primary key,
Fecha_venta_producto varchar(50),
ID_venta_producto_empleado varchar(50),
ID_venta_producto_cliente varchar(50),
foreign key (ID_venta_producto_empleado) references empleado(ID_empleado),
foreign key (ID_venta_producto_cliente) references cliente(ID_cliente)
);
```

Se realizan 10 consultas a la BD barbería:

- Esta consulta me muestra los clientes de la barbería:

```
3 • select * from cliente;
4
5
```

| ID_cliente | Nombre_cliente | Cedula_cliente | Direccion_cliente | Edad_cliente |
|------------|----------------|----------------|-------------------|--------------|
| 1 | Efrain | 1144046696 | cr 50 | 31 |
| NULL | NULL | NULL | NULL | NULL |

- Esta consulta me muestra los empleados de la barbería:




```
9 #2
10 • select * from empleado;
11
```

| ID_empleado | Nombre_empleado | Cedula_empleado |
|-------------|-----------------|-----------------|
| 1 | Andres | 12345667 |
| NULL | NULL | NULL |

- Esta consulta me muestra los proveedores:

12 #3

13 • `select * from proveedor;`

| Result Grid | | | | |
|---|--------------|------------------|--------------------|---------------------|
| Filter Rows: <input type="text"/> | | | | |
| Edit:    Export/In | | | | |
| | ID_proveedor | Nombre_proveedor | Telefono_proveedor | Direccion_proveedor |
| ▶ | 1 | Ego | 4325454 | cr50 |
| * | NULL | NULL | NULL | NULL |

- Esta consulta me muestra las citas que estas pendientes:






14 #4

15 • `select * from cita;`

| Result Grid | | | | |
|-----------------------------------|---------|-----------|------------|-----------------|
| Filter Rows: <input type="text"/> | | | | |
| E | | | | |
| | ID_cita | Hora_cita | Fecha_cita | ID_cliente_cita |
| ▶ | 1 | 2:00 | 20/02/2023 | 1 |
| * | NULL | NULL | NULL | NULL |

- Esta consulta me muestra los productos disponibles:


17 • `select * from producto;`

| Result Grid | | | | | |
|--|--------------|-----------------|-----------------|-------------------|-----------------------|
| Filter Rows: <input type="text"/> | | | | | |
| Edit:    Export/Import:   W | | | | | |
| | REF_producto | Nombre_producto | Precio_producto | Cantidad_producto | ID_producto_proveedor |
| ▶ | 1 | gel | 20000 | 50 | 1 |
| * | NULL | NULL | NULL | NULL | NULL |

- En esta consulta me muestra la especialidad de los empleados:

18 #6

19 • `select * from especialidad_empleado;`

| Result Grid | |
|---|-----------------------|
| Filter Rows: <input type="text"/> | |
| Edit:  | |
| | Especialidad_empleado |
| ▶ | Barbero, peluquero |
| * | NULL |

- En esta consulta me muestra los datos de la cita y el nombre del empleado asignado para la cita:

```

20 #7
21 • select ID_cita,Hora_cita,Fecha_cita,ID_cliente_cita, Nombre_empleado from cita
22 join empleado_cita on ID_cita_empleado = ID_cita
23 join empleado on ID_empleado = ID_empleado_cita;

```

| Result Grid | | | | | |
|-----------------------------------|---------|-----------|------------|-----------------|-----------------|
| Filter Rows: <input type="text"/> | | | | | |
| Export: Wrap Cell Content: | | | | | |
| | ID_cita | Hora_cita | Fecha_cita | ID_cliente_cita | Nombre_empleado |
| ▶ | 1 | 2:00 | 20/02/2023 | 1 | Andres |

- En esta consulta me muestra el teléfono y el cliente que le pertenece:

```

24 #8
25 • select Telefono_cliente, Nombre_cliente from telefono_cliente
26 join cliente on ID_cliente_telefono = ID_cliente;

```

| Result Grid | | |
|-----------------------------------|------------------|----------------|
| Filter Rows: <input type="text"/> | | |
| Export: Wrap Cell Content: | | |
| | Telefono_cliente | Nombre_cliente |
| ▶ | 23242453 | Efrain |

- En esta consulta me muestra el nombre del producto y la fecha en que se vendió:

```

27 #9
28 • select Nombre_producto, Fecha_venta_producto from producto
29 join producto_venta_producto on ID_producto_venta = REF_producto
30 join venta_producto on ID_venta_producto = ID_producto_venta_producto;

```

| Result Grid | | |
|-----------------------------------|-----------------|----------------------|
| Filter Rows: <input type="text"/> | | |
| Export: Wrap Cell Content: | | |
| | Nombre_producto | Fecha_venta_producto |
| ▶ | gel | 20/2/2023 |

- En esta consulta me muestra el historial de servicio:

```

31      #10
32 •    select * from historial_servicio;

```

| ID_servicio | Duracion_servicio | ID_servicio_cita |
|-------------|-------------------|------------------|
| 1 | 2:00 | 1 |
| 2 | 2:00 | 1 |
| NULL | NULL | NULL |

Se realizan las 4 vistas en la BD barbería:

- En esta vista se visualiza el nombre del empleado, el producto que vendió y el cliente que compró el producto.

```

1      #1
2 •    create view productos_vendidos_empleado as
3      select Nombre_empleado, Nombre_cliente, Nombre_producto from venta_producto
4      join empleado on ID_empleado = ID_venta_producto
5      join cliente on ID_cliente = ID_venta_producto_cliente
6      join producto on ID_venta_producto = REF_producto;
7 •    select * from productos_vendidos_empleado;
8      #2

```

| Nombre_empleado | Nombre_cliente | Nombre_producto |
|-----------------|----------------|-----------------|
| Andres | Efrain | gel |

- En esta vista se visualiza una tabla con el nombre del cliente, nombre del empleado, fecha y hora de la cita.

```

8      #2
9      • create view cita_servicio as
10     select nombre_cliente, Nombre_empleado, Fecha_cita, Hora_cita from cliente
11     join cita on ID_cliente_cita = ID_cliente
12     join empleado_cita on ID_cita_empleado = ID_cita
13     join empleado on ID_empleado_cita = ID_empleado;
14     • select * from cita_servicio;
15     #3

```

| nombre_cliente | Nombre_empleado | Fecha_cita | Hora_cita |
|----------------|-----------------|------------|-----------|
| Efrain | Andres | 20/02/2023 | 2:00 |

- En esta vista se visualiza el nombre del proveedor y el producto que vende.

```

5      #3
6      • create view Proveedor_producto as
7      select Nombre_proveedor, Nombre_producto from proveedor
8      join producto on ID_producto_proveedor = ID_proveedor;
9      • select * from Proveedor_producto;
10     #4

```

| Nombre_proveedor | Nombre_producto |
|------------------|-----------------|
| Ego | gel |

- En esta vista se visualiza el nombre del empleado y el producto que gatos en una cita.

```

20     #4
21     • create view empleado_producto_utilizado as
22     select Nombre_empleado, Nombre_producto from empleado
23     join empleado_producto on ID_empleado_producto = ID_empleado
24     join producto on REF_producto = ID_producto_empleado;
25     • select * from empleado_producto_utilizado;

```

| Nombre_empleado | Nombre_producto |
|-----------------|-----------------|
| Andres | gel |

Se realizan 4 procedimientos almacenados:

- Este procedimiento permite crear un registro de un cliente en la tabla cliente con todos sus atributos. Se hace un llamado del procedimiento y se agregan los datos pertinentes para agregar un nuevo cliente a la tabla.

```
#1
delimiter //
> create procedure agregar_registro_cliente(in ID_cliente_nuevo varchar(15),in Nombre_cliente_nuevo varchar(
in Cedula_cliente_nuevo varchar(50),
in Direccion_cliente_nuevo varchar(50),
in Edad_cliente_nuevo varchar(50))
> begin
select * from cliente;
> insert into cliente values (ID_cliente_nuevo, Nombre_cliente_nuevo,Cedula_cliente_nuevo,
Direccion_cliente_nuevo,
Edad_cliente_nuevo);
end//
delimiter ;

15 • call barberia.agregar_registro_cliente(5, 'Juna', '123456', 'cr60', '45');
```

| Result Grid Filter Rows: Export: Wrap Cell Content: | | | | | |
|---|------------|----------------|----------------|-------------------|--------------|
| | ID_cliente | Nombre_cliente | Cedula_cliente | Direccion_cliente | Edad_cliente |
| 1 | 1 | cristian | 1234567 | cr67 | 45 |

- Este procedimiento permite eliminar un registro de la tabla cliente. Se hace un llamado del procedimiento pasandole un id para eliminar el registro.

```
8 • create procedure eliminar_registro_cliente(in ID_cliente_nuevo varchar(15))
9 begin
0 select * from cliente;
1 delete from cliente where ID_cliente = ID_cliente_nuevo;
2 end//
3 delimiter ;
4 call eliminar_registro_cliente(5)
5 #2
```

| Result Grid Filter Rows: Export: Wrap Cell Content: | | | | | |
|---|------------|----------------|----------------|-------------------|--------------|
| | ID_cliente | Nombre_cliente | Cedula_cliente | Direccion_cliente | Edad_cliente |
| 1 | 1 | cristian | 1234567 | cr67 | 45 |
| 5 | 5 | Juna | 123456 | cr60 | 45 |

- Este procedimiento permite actualizar un registro de la tabla cliente. Se hace un llama del procedimiento. Se le pasa un id y los datos que necesita para hacer la actualización.

```
#3
delimiter //
create procedure actualizar_registro_cliente(in ID_cliente_nuevo varchar(15),in Nombre_cliente_nuevo varchar(50),
in Cedula_cliente_nuevo varchar(50),
in Direccion_cliente_nuevo varchar(50),
in Edad_cliente_nuevo varchar(50))
begin
select * from cliente;
update cliente set Nombre_cliente = Nombre_cliente_nuevo,
Cedula_cliente = Cedula_cliente_nuevo,
Direccion_cliente = Direccion_cliente_nuevo,
Edad_cliente = Edad_cliente_nuevo where ID_cliente = ID_cliente_nuevo;
end//
delimiter ;
```

```
39 • call actualizar_registro_cliente(1,'Roman', 1234567, 'cr67', 45);
```

| ID_cliente | Nombre_cliente | Cedula_cliente | Direccion_cliente | Edad_cliente |
|------------|----------------|----------------|-------------------|--------------|
| 1 | cristian | 1234567 | cr67 | 45 |

- Este procedimiento permite consultar la información de un registro. Haciendo el llamado del procedimiento se le pasa un id para que nos muestre el registro.

```
#1
delimiter //
create procedure consultar_registro_cliente(in ID_cliente_nuevo varchar(15))
begin
select * from cliente where ID_cliente = ID_cliente_nuevo;
end//
delimiter ;
#2
call consultar_registro_cliente(1);
```

| ID_cliente | Nombre_cliente | Cedula_cliente | Direccion_cliente | Edad_cliente |
|------------|----------------|----------------|-------------------|--------------|
| 1 | Roman | 1234567 | cr67 | 45 |

Se realizan los 4 trigger en la BD barbería pero antes que todo, se crea una tabla llamada registro_cambio donde se visualiza el nombre del usuario, un mensaje y la fecha en que se realizó un cambio, como agregar, eliminar o actualizar información en la tabla cliente.

```
create table registro_cambios(  
  Nombre_usuario varchar(50),  
  Respuesta varchar(50),  
  Fecha datetime default current_timestamp  
)
```

- Este trigger hace una alerta cuando un usuario crea un registro en la tabla cliente.

```
delimiter //  
create trigger registro_cliente after insert on cliente  
for each row  
begin  
  insert into registro_cambios values (user(),'se agrego un registro registro',now());  
end//  
delimiter ;
```

El registro se ve de esta forma.

| | Nombre_usuario | Respuesta | Fecha |
|---|----------------|--------------------------------|---------------------|
| ▶ | root@localhost | se agrego un registro registro | 2023-02-16 23:30:31 |

- Este trigger hace una alerta cuando un usuario elimina un registro en la tabla cliente.

```
delimiter //  
create trigger registro_cliente_eliminado after delete on cliente  
for each row  
begin  
  insert into registro_cambios values (user(),'se elimino un registro',now());  
end//  
delimiter //
```

El registro se ve de esta forma:

| | Nombre_usuario | Respuesta | Fecha |
|---|----------------|--------------------------------|---------------------|
| ▶ | root@localhost | se agrego un registro registro | 2023-02-16 23:30:31 |
| | root@localhost | se elimino un registro | 2023-02-16 23:32:28 |

- Este trigger hace una alerta cuando un usuario actualiza un registro en la tabla cliente.

```
create trigger registro_cliente_actualizado after update on cliente
for each row
begin
    insert into registro_cambios values (user(),'se actualizo un registro',now());
end//
delimiter ;
```

El registro se ve de esta forma:

| | Nombre_usuario | Respuesta | Fecha |
|---|----------------|--------------------------------|---------------------|
| ▶ | root@localhost | se agrego un registro registro | 2023-02-16 23:30:31 |
| | root@localhost | se elimino un registro | 2023-02-16 23:32:28 |
| | root@localhost | se actualizo un registro | 2023-02-16 23:34:40 |
| | root@localhost | se actualizo un registro | 2023-02-16 23:34:46 |

- Este trigger hace una alerta cuando un usuario crea un registro en la tabla empleado.

```
delimiter //
create trigger registro_empleado after insert on empleado
for each row
begin
    insert into registro_cambios values (user(),'se agrego un registro a empleado',now());
end//
delimiter ;
```

El registro se ve de este forma:

| | Nombre_usuario | Respuesta | Fecha |
|---|----------------|----------------------------------|---------------------|
| ▶ | root@localhost | se agrego un registro registro | 2023-02-16 23:30:31 |
| | root@localhost | se elimino un registro | 2023-02-16 23:32:28 |
| | root@localhost | se actualizo un registro | 2023-02-16 23:34:40 |
| | root@localhost | se actualizo un registro | 2023-02-16 23:34:46 |
| | root@localhost | se agrego un registro a empleado | 2023-02-16 23:42:22 |