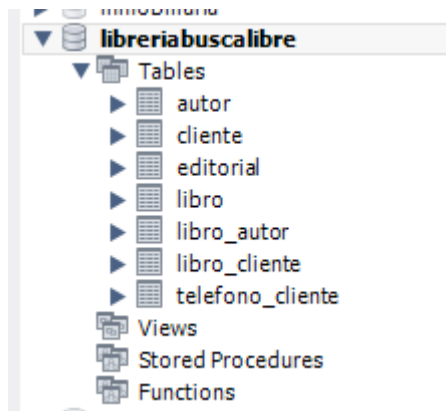
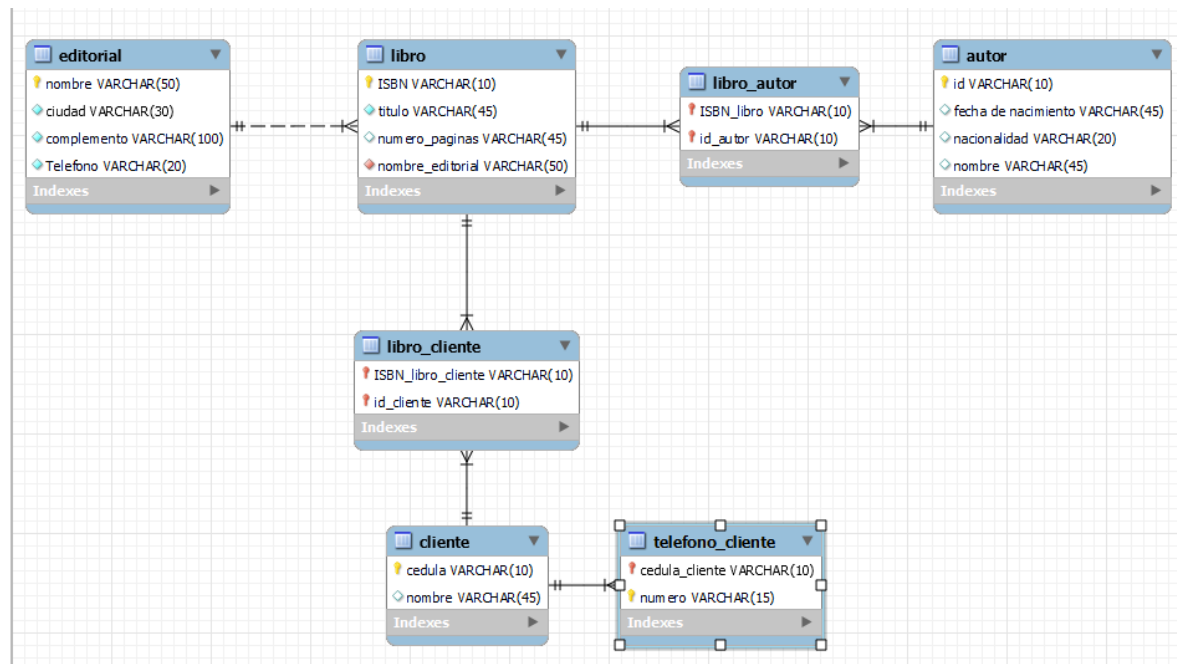


PRIMERA ACTIVIDAD

Genero el schema con el script que se encuentra en el repositorio:



Le aplico ingeniería en reversa para tener mejor visibilidad de las tablas, el tipo de datos de los atributos y como se relacionan entre ellas, y así facilitarme la creación de los registros.



Analizando el diagrama se puede observar:

- Las tablas que no tienen llaves foráneas son: editorial, autor y cliente, por ende, en estas se deben crear los primeros registros para evitar problemas con las relaciones.
- Para la tabla libro_autor, se debe tener registros en las tablas libro y autor.
- Para la tabla libro, se debe tener registros en la tabla editorial.
- Para la tabla libro_cliente, se debe tener registros en las tablas libro y cliente.
- Para la tabla teléfono_cliente, se debe tener registros en la tabla cliente.

Con esta información se decide ingresar los registros en el siguiente orden:

Tabla autor:

Se generan 5 registros

```
5
6 • select * from autor;
7 • INSERT INTO autor ('id', 'fecha de nacimiento', 'nacionalidad', 'nombre') VALUES
8 ('1001', '01/01/90', 'Colombiano', 'Juan Garcia'),
9 ('1002', '12/06/95', 'Mexicano', 'Pedro Mancha'),
10 ('1003', '13/10/86', 'Venezolano', 'Jhon Cena'),
11 ('1004', '06/05/70', 'Colombiano', 'Mario Bross'),
12 ('1005', '25/01/78', 'Peruano', 'Pablo Escobar');
```

id	fecha de nacimiento	nacionalidad	nombre
1001	01/01/90	Colombiano	Juan Garcia
1002	12/06/95	Mexicano	Pedro Mancha
1003	13/10/86	Venezolano	Jhon Cena
1004	06/05/70	Colombiano	Mario Bross
1005	25/01/78	Peruano	Pablo Escobar

Tabla editorial:

Se generan 4 registros

```
Query 1
1 • select * from editorial;
2 • INSERT INTO editorial ('nombre', 'ciudad', 'complemento', 'Telefono') VALUES
3 ('Editorial1', 'Buga', 'Calle 11 #60-27 segundo piso', '2213300'),
4 ('Editorial2', 'Cali', 'Calle 20 #50-30', '2236699'),
5 ('Editorial3', 'Medellin', 'Calle 45 #50-30 primer piso', '2315060'),
6 ('Editorial4', 'Bogota', 'Calle 50 #30-30', '2315060');
```

nombre	ciudad	complemento	Telefono
Editorial1	Buga	Calle 11 #60-27 segundo piso	2213300
Editorial2	Cali	Calle 20 #50-30	2236699
Editorial3	Medellin	Calle 45 #50-30 primer piso	2315060
Editorial4	Bogota	Calle 50 #30-30	2315060

Tabla libro:

Se generan 20 registros

```
16 • select * from libro;
17 • INSERT INTO libro ('ISBN', 'titulo', 'numero_paginas', 'nombre_editorial') VALUES
18 ('ABC123', 'El amanecer', '100', 'Editorial4'),
19 ('ABC124', 'Programacion', '560', 'Editorial2'),
20 ('ABC125', 'Cocina', '60', 'Editorial1'),
21 ('ABC126', 'El anochecer', '56', 'Editorial2'),
22 ('ABC127', 'La puerta', '50', 'Editorial3'),
23 ('ABC128', 'Pokemon', '60', 'Editorial4'),
24 ('ABC129', 'Zelda', '3600', 'Editorial1'),
25 ('ABC130', 'Aprenda ingles', '1000', 'Editorial3'),
26 ('ABC131', 'Aprenda frances', '1200', 'Editorial1'),
27 ('ABC132', 'Aprenda programar', '80', 'Editorial4'),
28 ('ABC133', 'Matematica financiera', '54', 'Editorial2'),
29 ('ABC134', 'Calculo diferencial', '453', 'Editorial1'),
30 ('ABC135', 'Calculo vectorial', '415', 'Editorial3');
```

ISBN	titulo	numero_paginas	nombre_editorial
ABC124	Programacion	560	Editorial2
ABC125	Cocina	60	Editorial1
ABC126	El anochecer	56	Editorial2
ABC127	La puerta	50	Editorial3
ABC128	Pokemon	60	Editorial4
ABC129	Zelda	3600	Editorial1
ABC130	Aprenda ingles	1000	Editorial3
ABC131	Aprenda frances	1200	Editorial1
ABC132	Aprenda programar	80	Editorial4
ABC133	Matematica financi...	54	Editorial2
ABC134	Calculo diferencial	453	Editorial1
ABC135	Calculo vectorial	415	Editorial3
ABC136	Fisica	57	Editorial1
ABC137	Leyes de newton	756	Editorial4
ABC138	La biblia	745	Editorial4
ABC139	El senior de los anillos	57	Editorial1
ABC140	The last of us	213	Editorial1
ABC141	Batman	214	Editorial3
ABC142	La caída de Edgar	456	Editorial2

Tabla libro_autor
Se generan 10 registros

```

39 • select * from libro_autor;
40 • INSERT INTO libro_autor ('ISBN_libro', 'id_autor') VALUES
41   ('ABC124', '1002'),
42   ('ABC125', '1003'),
43   ('ABC129', '1005'),
44   ('ABC132', '1001'),
45   ('ABC140', '1001'),
46   ('ABC126', '1002'),
47   ('ABC127', '1004'),

```

ISBN_libro	id_autor
ABC123	1001
ABC132	1001
ABC140	1001
ABC141	1001
ABC124	1002
ABC126	1002
ABC125	1003
ABC127	1004
ABC129	1005
ABC131	1005
NULL	NULL

Tabla cliente
Se crean 7 registros

```

52 • select * from cliente;
53 • INSERT INTO cliente ('cedula', 'nombre') VALUES
54   ('1115069123', 'Juan Valdez'),
55   ('1115069456', 'Ash Ketshup'),
56   ('1115069789', 'Gon Freecs'),
57   ('1115069741', 'Naruto Uzumaki'),
58   ('1115069369', 'Walter White'),
59   ('1115069258', 'Jessie Pinkman'),
60   ('1115069852', 'Jorge El Curioso');
61

```

cedula	nombre
1115069123	Juan Valdez
1115069258	Jessie Pinkman
1115069369	Walter White
1115069456	Ash Ketshup
1115069741	Naruto Uzumaki
1115069789	Gon Freecs
1115069852	Jorge El Curioso
NULL	NULL

Tabla teléfono_cliente
Se crean 12 registros

```

62 • select * from telefono_cliente;
63 • INSERT INTO telefono_cliente ('cedula_cliente', 'numero') VALUES
64   ('1115069123', '3135643231'),
65   ('1115069123', '3135643232'),
66   ('1115069456', '3135643233'),
67   ('1115069456', '3135643234'),
68   ('1115069456', '3135643235'),
69   ('1115069789', '3135643236'),
70   ('1115069741', '3135643237'),
71   ('1115069369', '3135643238'),
72   ('1115069258', '3135643239'),
73   ('1115069852', '3135643210'),

```

cedula_cliente	numero
1115069123	3135643211
1115069123	3135643231
1115069123	3135643232
1115069258	3135643239
1115069369	3135643238
1115069456	3135643233
1115069456	3135643234
1115069456	3135643235
1115069741	3135643237
1115069789	3135643236
1115069852	3135643210

telefono_cliente 9 x

Tabla libro_cliente

Se generan 10 registros

```
77 • select * from libro_cliente;
78 • INSERT INTO libro_cliente (`ISBN_libro_cliente`, `id_cliente`) VALUES
79   ('ABC123','1115069123'),
80   ('ABC124','1115069123'),
81   ('ABC125','1115069123'),
82   ('ABC126','1115069456'),
83   ('ABC127','1115069456'),
84   ('ABC128','1115069789'),
85   ('ABC129','1115069741'),
86   ('ABC130','1115069369'),
87   ('ABC131','1115069258'),
88   ('ABC132','1115069852');
```

ISBN_libro_cliente	id_cliente
ABC123	1115069123
ABC124	1115069123
ABC125	1115069123
ABC131	1115069258
ABC130	1115069369
ABC126	1115069456
ABC127	1115069456
ABC129	1115069741
ABC128	1115069789
ABC132	1115069852

Consultas:

Nombre y fecha de nacimiento de los escritores

select nombre, `fecha de nacimiento` from autor;

```
89
90 • select nombre, `fecha de nacimiento` from autor;
91
```

nombre	fecha de nacimiento
Juan Garcia	01/01/90
Pedro Mancha	12/06/95
Jhon Cena	13/10/86
Mario Bross	06/05/70
Pablo Escobar	25/01/78

Cantidad de libros vendidos diferentes

select count(distinct ISBN_libro_cliente) as "Compras totales" from libro_cliente;

Aplicamos **distinct** en el ISBN del libro para que no cuente los libros repetidos.

```
91
92 • select count(distinct ISBN_libro_cliente) as "Compras totales" from libro_cliente;
```

Compras totales
10

Cliente y número de teléfono

select libro.titulo as "libro", cliente.nombre as "cliente", telefono_cliente.numero as "telefono"

from libro inner join libro_cliente on libro.ISBN = libro_cliente.ISBN_libro_cliente

inner join cliente on cedula = libro_cliente.id_cliente

inner join telefono_cliente on cedula_cliente=cliente.cedula;

Un cliente al poder tener más de un numero se generan registros repetidos en la búsqueda.

```

93
94 • select libro.titulo as "libro", cliente.nombre as "cliente", telefono_cliente.numero as "telefono"
95 from libro inner join libro_cliente on libro.ISBN = libro_cliente.ISBN_libro_cliente
96 inner join cliente on cedula = libro_cliente.id_cliente
97 inner join telefono_cliente on cedula_cliente=cliente.cedula;

```

libro	cliente	telefono
El amanecer	Juan Valdez	3135643211
El amanecer	Juan Valdez	3135643231
El amanecer	Juan Valdez	3135643232
Programacion	Juan Valdez	3135643211
Programacion	Juan Valdez	3135643231
Programacion	Juan Valdez	3135643232
Cocina	Juan Valdez	3135643211
Cocina	Juan Valdez	3135643231
Cocina	Juan Valdez	3135643232
Aprenda frances	Jessie Pinkman	3135643239
Aprenda ingles	Walter White	3135643238

Result 28 x

Nombre de libro y sus autores

select libro.titulo as "libro", autor.nombre as "autor"

from autor inner join libro_autor on autor.id = libro_autor.id_autor

inner join libro on ISBN = libro_autor.ISBN_libro;

```

98
99 • select libro.titulo as "libro", autor.nombre as "autor"
100 from autor inner join libro_autor on autor.id = libro_autor.id_autor
101 inner join libro on ISBN = libro_autor.ISBN_libro;
102

```

libro	autor
El amanecer	Juan Garcia
Aprenda programar	Juan Garcia
The last of us	Juan Garcia
Batman	Juan Garcia
Programacion	Pedro Mancha
El anocheceer	Pedro Mancha
Cocina	Jhon Cena
La puerta	Mario Bross
Zelda	Pablo Escobar
Aprenda frances	Pablo Escobar

El nombre de las editoriales que han vendido libros

select distinct libro.nombre_editorial as "editorial"

from libro inner join libro_cliente on libro.ISBN = libro_cliente.ISBN_libro_cliente;

```

103 • select distinct libro.nombre_editorial as "editorial"
104 from libro inner join libro_cliente on libro.ISBN = libro_cliente.ISBN_libro_cliente;
105

```

editorial
Editorial4
Editorial2
Editorial1
Editorial3

Vista1

Compras: Esta vista contiene la información importante de todas las compras que se han realizado en la librería, podría utilizarse para generar una factura.

create view Compras as

```
select cliente.cedula as "cedula", cliente.nombre as "nombre", libro.titulo as "libro",  
telefono_cliente.numero as "telefono"
```

```
from libro inner join libro_cliente on libro.ISBN = libro_cliente.ISBN_libro_cliente
```

```
inner join cliente on cedula = libro_cliente.id_cliente
```

```
inner join telefono_cliente on cedula_cliente=cliente.cedula;
```

```
108 • create view Compras as
109 select cliente.cedula as "cedula", cliente.nombre as "nombre", libro.titulo as "libro", telefono_cliente.numero as "telefono"
110 from libro inner join libro_cliente on libro.ISBN = libro_cliente.ISBN_libro_cliente
111 inner join cliente on cedula = libro_cliente.id_cliente
112 inner join telefono_cliente on cedula_cliente=cliente.cedula;
113 • select * from Compras;
```

cedula	nombre	libro	telefono
1115069123	Juan Valdez	Cocina	3135643232
1115069258	Jessie Pinkman	Aprenda frances	3135643239
1115069369	Walter White	Aprenda ingles	3135643238
1115069456	Ash Ketchup	El anochecer	3135643233
1115069456	Ash Ketchup	El anochecer	3135643234
1115069456	Ash Ketchup	El anochecer	3135643235
1115069456	Ash Ketchup	La puerta	3135643233
1115069456	Ash Ketchup	La puerta	3135643234
1115069456	Ash Ketchup	La puerta	3135643235
1115069741	Naruto Uzumaki	Zelda	3135643237
1115069789	Gon Freecs	Pokemon	3135643236

Compras 35 x

Vista2

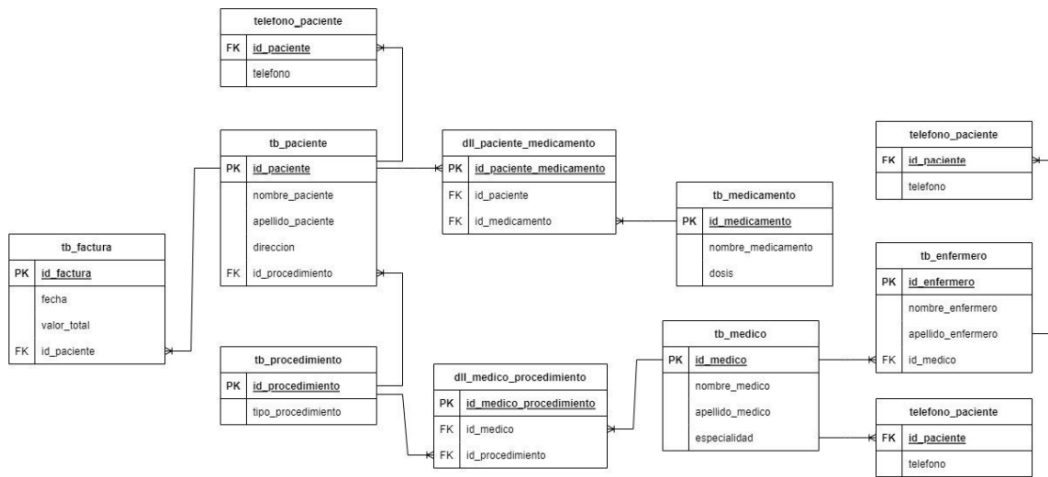
InfoLibro: Contiene toda la información necesaria de los libros, sus títulos, autores y editoriales, esta vista puede ser muy utilizada cuando alguien quiera saber donde puede conseguirse el libro buscado.

```
119 • create view InfoLibro as
120 select libro.titulo as "libro", libro.numero_paginas as "paginas", libro.nombre_editorial as "editorial", editorial.Telefono as "telefono", autor.nombre as autor
121 from editorial inner join libro on nombre=libro.nombre_editorial
122 inner join libro_autor on ISBN_libro=ISBN
123 inner join autor on id=libro_autor.id_autor;
124 • select * from InfoLibro;
```

libro	paginas	editorial	telefono	autor
Cocina	60	Editorial1	2213300	Jhon Cena
Zelda	3600	Editorial1	2213300	Pablo Escobar
Aprenda frances	1200	Editorial1	2213300	Pablo Escobar
The last of us	213	Editorial1	2213300	Juan Garcia
Programacion	560	Editorial2	2236699	Pedro Mancha
El anochecer	96	Editorial2	2236699	Pedro Mancha
La puerta	90	Editorial3	2315060	Mario Briss
Batman	214	Editorial3	2315060	Juan Garcia
El amanecer	100	Editorial4	2315060	Juan Garcia
Aprenda programar	80	Editorial4	2315060	Juan Garcia

SEGUNDA ACTIVIDAD

Descargo la imagen del modelo relacional para tener una guía con la creación del modelo en SQL (se ve un poco borrosa pero se alcanza a entender).



Analizando el diagrama se puede observar:

- Las tablas que no dependen de otras son: tb_medimento, tb_medico y tb_procedimiento.
- Las tablas teléfono_paciente y tb_factura dependen de tb_paciente.
- La tabla tb_paciente necesita de tb_procedimiento.
- La tabla dll_paciente_medimento necesita de tb_paciente y tb_medimento.
- La tabla dll_medico_procedimiento necesita de tb_procedimiento y tb_medico.
- Hay un error con la tabla que debería contener el teléfono del médico, este error será corregido en el diagrama creado en SQL, la tabla teléfono_medico y tb_enfermero dependen de tb_medico.
- Pasa el mismo error con la tabla de teléfonos de los enfermeros, el error será corregido en el diagrama creado en SQL, la tabla teléfono_enfermero depende de enfermero.

Se procede con la creación de tablas y el ingreso de registros al mismo tiempo.

Con este análisis se concluye que el orden de creación de las tablas y de ingresos será el siguiente:

Se inicia con la creación del schema:

```
1 • create schema hospital;  
2 |  
3 • use hospital;  
4
```

Tabla tb_medico

```
4
5 • create table tb_medico(
6     id_medico varchar(20) primary key,
7     nombre_medico varchar(50),
8     apellido_medico varchar(50),
9     especialidad varchar(60)
10 );
11
12 • select * from tb_medico;
13 • INSERT INTO tb_medico (`id_medico`,`nombre_medico`,`apellido_medico`,`especialidad`) values
14     ("1001","Raul","Restrepo","Cirugia"),
15     ("1002","Fabio","Lopez","Dermatologo"),
16     ("1003","Julian","Ramirez","Pediatria"),
17     ("1004","Wisin","Yandel","Otorrino"),
18     ("1005","Ash","Ketshup","Emergencias");
```

id_medico	nombre_medico	apellido_medico	especialidad
1001	Raul	Restrepo	Cirugia
1002	Fabio	Lopez	Dermatologo
1003	Julian	Ramirez	Pediatria
1004	Wisin	Yandel	Otorrino
1005	Ash	Ketshup	Emergencias
*	NULL	NULL	NULL

Tabla teléfono_medico

```
19
20 • create table telefono_medico(
21     id_medico varchar(20),
22     telefono varchar(50),
23     primary key(id_medico,telefono),
24     foreign key(id_medico) references tb_medico(id_medico)
25 );
26
27 • select * from telefono_medico;
28 • insert into telefono_medico(`id_medico`,`telefono`) values
29     ("1001","3135633231"),
30     ("1002","3135633232"),
31     ("1003","3135633233"),
32     ("1004","3135633234"),
33     ("1005","3135633235");
34
```

id_medico	telefono
1001	3135633231
1002	3135633232
1003	3135633233
1004	3135633234
1005	3135633235
*	NULL

Tabla tb_enfermero

```
42
43 • select * from tb_enfermero;
44 • insert into tb_enfermero (`id_enfermero`, `nombre_enfermero`, `apellido_enfermero`, `id_medico`) values
45   ("2001", "Laura", "Garcia", "1001"),
46   ("2002", "Maga", "Oscuro", "1002"),
47   ("2003", "Stuart", "Little", "1003"),
48   ("2004", "Trunks", "Delfuturo", "1004"),
49   ("2005", "Power", "Ranger", "1005");
```

id_enfermero	nombre_enfermero	apellido_enfermero	id_medico
2001	Laura	Garcia	1001
2002	Maga	Oscuro	1002
2003	Stuart	Little	1003
2004	Trunks	Delfuturo	1004
2005	Power	Ranger	1005
NULL	NULL	NULL	NULL

Tabla tlefono_enfermero

```
57
58 • select * from telefono_enfermero;
59 • insert into telefono_enfermero(`id_enfermero`, `telefono`) values
60   ("2001", "3135633236"),
61   ("2002", "3135633237"),
62   ("2003", "3135633238"),
63   ("2004", "3135633239"),
64   ("2005", "3135633210");
65
```

id_enfermero	telefono
2001	3135633236
2002	3135633237
2003	3135633238
2004	3135633239
2005	3135633210
NULL	NULL

Tabla procedimiento

```
65
66 • create table tb_procedimiento(
67   id_procedimiento varchar(20) primary key,
68   tipo_procedimiento varchar(50)
69 );
70
71 • select * from tb_procedimiento;
72 • insert into tb_procedimiento(`id_procedimiento`, `tipo_procedimiento`) values
73   ("3001", "Quirurjico"),
74   ("3002", "Terapia"),
75   ("3003", "Contra el cancer"),
76   ("3004", "Psicologico"),
77   ("3005", "Biopsia");
78
```

id_medico	id_procedimiento
1001	3001
1002	3002
1003	3003
1004	3004
1005	3005
NULL	NULL

Tabla dll_medico_procedimiento

```
79 • create table dll_medico_procedimiento(  
80     id_medico varchar(20),  
81     id_procedimiento varchar(20),  
82     primary key(id_medico,id_procedimiento),  
83     foreign key(id_medico)references tb_medico(id_medico),  
84     foreign key(id_procedimiento)references tb_procedimiento(id_procedimiento)  
85 );  
86  
87 • select * from dll_medico_procedimiento;  
88 • insert into dll_medico_procedimiento(`id_medico`,`id_procedimiento`) values  
89     ("1001","3001"),  
90     ("1002","3002"),  
91     ("1003","3003"),  
92     ("1004","3004"),  
93     ("1005","3005");  
94
```

id_medico	id_procedimiento
1001	3001
1002	3002
1003	3003
1004	3004
1005	3005

Tabla tb_paciente

```
95 • create table tb_paciente(  
96     id_paciente varchar(20) primary key,  
97     nombre_paciente varchar(50),  
98     apellido_paciente varchar(50),  
99     direccion varchar(50),  
100     id_procedimiento varchar(20),  
101     foreign key(id_procedimiento) references tb_procedimiento(id_procedimiento)  
102 );  
103  
104 • select * from tb_paciente;  
105 • insert into tb_paciente(`id_paciente`,`nombre_paciente`,`apellido_paciente`,`direccion`,`id_procedimiento`) values  
106     ("4001","Jaime","Garzon","calle falsa 121","3001"),  
107     ("4002","Calamardo","Tentaculos","calle falsa 122","3002"),  
108     ("4003","Bob","Esponja","calle falsa 123","3003"),  
109     ("4004","Patricio","Estrella","calle falsa 124","3004"),  
110     ("4005","Don","Cangrejo","calle falsa 125","3005");  
111
```

id_paciente	nombre_paciente	apellido_paciente	direccion	id_procedimiento
4001	Jaime	Garzon	calle falsa 121	3001
4002	Calamardo	Tentaculos	calle falsa 122	3002
4003	Bob	Esponja	calle falsa 123	3003
4004	Patricio	Estrella	calle falsa 124	3004
4005	Don	Cangrejo	calle falsa 125	3005

Tabla teléfono_paciente

```
112 • create table telefono_paciente(  
113     id_paciente varchar(20),  
114     telefono varchar(50),  
115     primary key(id_paciente,telefono),  
116     foreign key(id_paciente) references tb_paciente(id_paciente)  
117 );  
118  
119 • select * from telefono_paciente;  
120 • insert into telefono_paciente(`id_paciente`,`telefono`) values  
121     ("4001","3135633216"),  
122     ("4002","3135633227"),  
123     ("4003","3135633438"),  
124     ("4004","3135633249"),  
125     ("4005","3135633250");
```

id_paciente	telefono
4001	3135633216
4002	3135633227
4003	3135633438
4004	3135633249
4005	3135633250

Tabla tb_facture

```
127 • create table tb_facture(  
128     id_factura varchar(20) primary key,  
129     fecha varchar(20),  
130     valor_total double,  
131     id_paciente varchar(20),  
132     foreign key(id_paciente) references tb_paciente(id_paciente)  
133 );  
134  
135 • select * from tb_facture;  
136 • insert into tb_facture('id_factura','fecha','valor_total','id_paciente') values  
137     ("5001","01/01/2023",60000,"4001"),  
138     ("5002","08/02/2023",60000,"4002"),  
139     ("5003","15/03/2023",60000,"4003"),  
140     ("5004","06/12/2022",60000,"4004"),  
141     ("5005","25/11/2022",60000,"4005");
```

Result Grid

	id_factura	fecha	valor_total	id_paciente
▶	5001	01/01/2023	60000	4001
	5002	08/02/2023	60000	4002
	5003	15/03/2023	60000	4003
	5004	06/12/2022	60000	4004
	5005	25/11/2022	60000	4005
•	total	total	total	total

Tabla tb_medicamento

```
149 • select * from tb_medicamento;  
150 • insert into tb_medicamento('id_medicamento','nombre_medicamento','dosis') values  
151     ("6001","Semilla del leonitiano","2 mg al dia"),  
152     ("6002","Paracetamol","6mg cada 8 horas"),  
153     ("6003","Loratadina","mucho"),  
154     ("6004","Dolex gripa","2 veces al dia 5mg"),  
155     ("6005","Descansar","Siempre");  
156  
157 • create table dll_paciente_medicamento(  
158     id_paciente varchar(20),  
159     id_medicamento varchar(20),  
160     primary key(id_paciente,id_medicamento),  
161     foreign key(id_paciente) references tb_paciente(id_paciente),  
162     foreign key(id_medicamento) references tb_medicamento(id_medicamento)  
163 );  
164
```

Result Grid

	id_medicamento	nombre_medicamento	dosis
▶	6001	Semilla del leonitiano	2 mg al dia
	6002	Paracetamol	6mg cada 8 horas
	6003	Loratadina	mucho
	6004	Dolex gripa	2 veces al dia 5mg
	6005	Descansar	Siempre
•	total	total	total

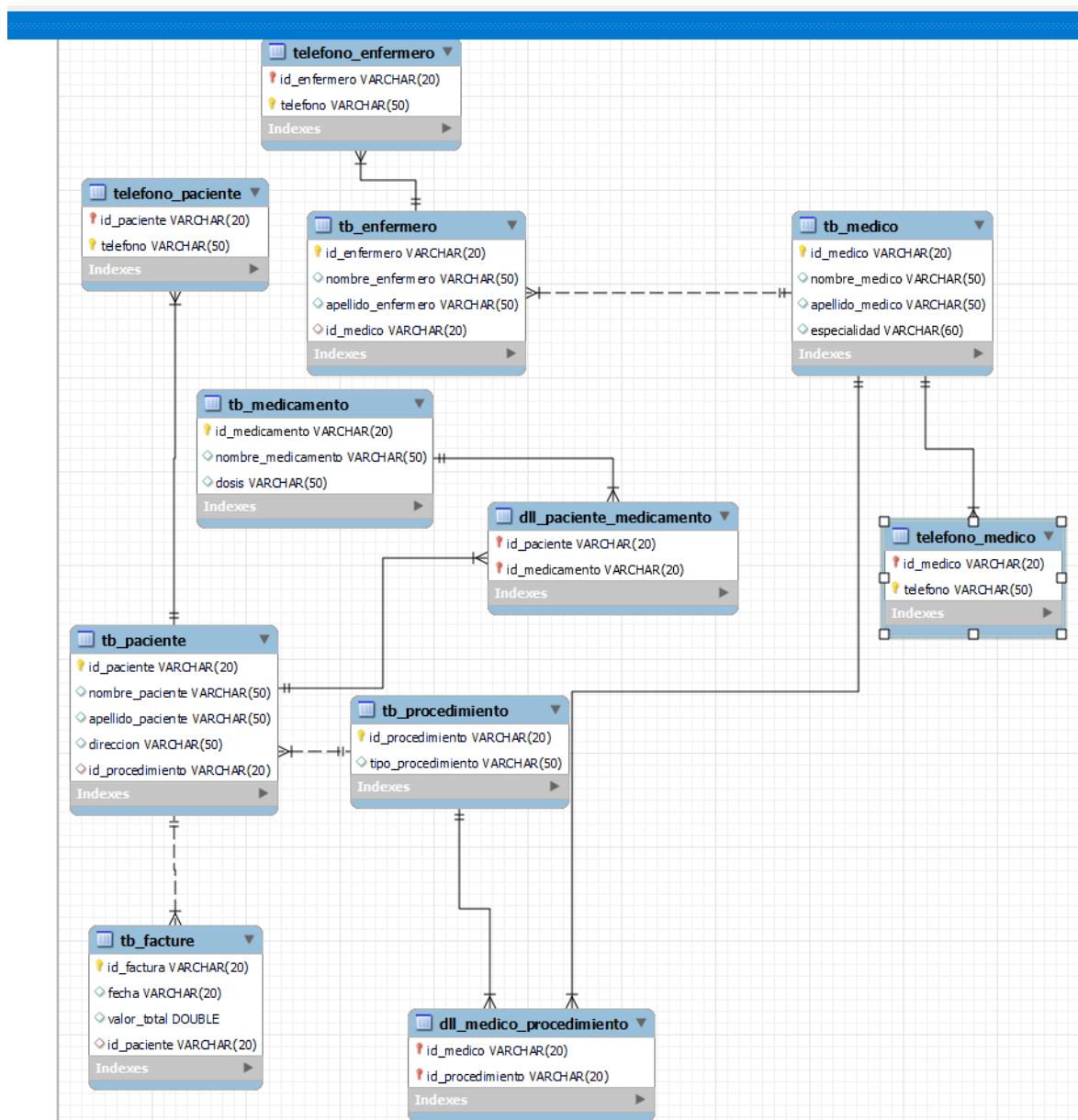
Tabla dll_paciente_medicamento

```
157 • create table dll_paciente_medicamento(  
158     id_paciente varchar(20),  
159     id_medicamento varchar(20),  
160     primary key(id_paciente,id_medicamento),  
161     foreign key(id_paciente) references tb_paciente(id_paciente),  
162     foreign key(id_medicamento) references tb_medicamento(id_medicamento)  
163 );  
164  
165 • select * from dll_paciente_medicamento;  
166 • insert into dll_paciente_medicamento('id_paciente','id_medicamento') values  
167     ("4001","6001"),  
168     ("4002","6002"),  
169     ("4003","6003"),  
170     ("4004","6004"),  
171     ("4005","6005");
```

Result Grid

	id_paciente	id_medicamento
▶	4001	6001
	4002	6002
	4003	6003
	4004	6004
	4005	6005
•	total	total

Aplicando ingeniería inversa podemos ver el diagrama en workbench



Consultas:

Realice una consulta que me permita conocer que medicamentos a tomado cada paciente y la dosis suministrada.

```
173 • select tb_paciente.nombre_paciente as "paciente", tb_medimento.nombre_medimento as "medimento", tb_medimento.dosis as "dosis"
174 from tb_paciente inner join dll_paciente_medimento on id_paciente=dll_paciente_medimento.id_paciente_m
175 inner join tb_medimento on id_medimento=dll_paciente_medimento.id_medimento_m;
```

paciente	medimento	dosis
Jaime	Semilla del leonitiano	2 mg al dia
Calamardo	Paracetamol	6mg cada 8 horas
Bob	Loratadina	mucho
Patricio	Dolex gripa	2 veces al dia 5mg
Don	Descansar	Siempre

Uso inner join para conectar las 3 tablas necesarias y muestro los datos requeridos. Conecto la tabla tb_paciente con dll_paciente_medimento mediante el atributo id_paciente, y después conecto la consulta generada con la tabla tb_medimento mediante el atributo id_medimento.

Se hacen esas conexiones ya que la tabla tb_paciente contiene el nombre del paciente, la tabla dll_medimento_paciente es necesaria para conectar las dos tablas (tb_paciente y tb_medimento), y la tabla tb_medimento contiene el nombre del medicamento y la dosis de este mismo.

Realice una consulta que me permita conocer que enfermeros estuvieron en los procedimientos de los pacientes.

```
176
177 • select tb_enfermeros.nombre_enfermero as "enfermero", tb_pacientes.nombre_paciente as "paciente", tb_procedimiento.tipo_procedimiento as "procedimiento"
178 from tb_pacientes inner join tb_procedimiento on id_procedimiento_f=tb_procedimiento.id_procedimiento
179 inner join dll_medico_procedimiento on id_procedimiento=dll_medico_procedimiento.id_procedimiento_p
180 inner join tb_medico on id_medico = dll_medico_procedimiento.id_medico_p
181 inner join tb_enfermeros on id_medico_e=tb_medico.id_medico;
```

enfermero	paciente	procedimiento
Laura	Jaime	Quirujico
Maga	Calamardo	Terapia
Stuart	Bob	Contra el cancer
Trunks	Patricio	Psicologico
Power	Don	Biopsia

Para mostrar esta consulta se deben usar 4 inner join, uno que conecte la tabla tb_pacientes con tb_procedimientos mediante el atributo id_procedimiento, después se conecta la tabla generada con dll_medico_procedimiento mediante el atributo id_procedimiento, después se debe conectar con la tabla tb_medico mediante el id_medico y por ultimo se conecta con la tabla enfermero mediante el atributo id_medico.

Vistas:

Vista1 DatosProcedimiento

Esta vista se crea con el fin de generar todos los datos importantes sobre un procedimiento, el doctor que lo realiza con su respectivo enfermero y el tipo de procedimiento que será realizado, y obviamente el paciente que lo va a recibir.

```
177
178 • select * from DatosProcedimiento;
179 • create view DatosProcedimiento as
180 select tb_enfermeros.nombre_enfermero as "enfermero", tb_pacientes.nombre_paciente as "paciente", tb_procedimiento.tipo_procedimiento as "procedimiento",
181        tb_medico.nombre_medico as "doctor"
182 from tb_pacientes inner join tb_procedimiento on id_procedimiento_f=tb_procedimiento.id_procedimiento
183        inner join dll_medico_procedimiento on id_procedimiento=dll_medico_procedimiento.id_procedimiento_p
184        inner join tb_medico on id_medico = dll_medico_procedimiento.id_medico_p
185        inner join tb_enfermeros on id_medico_e=tb_medico.id_medico;
186
```

enfermero	paciente	procedimiento	doctor
Laura	Jaime	Quirujico	Raul
Maga	Calamardo	Terapia	Fabio
Stuart	Bob	Contra el cancer	Julian
Trunks	Patricio	Psicologico	Wisin
Power	Don	Biopsia	Ash

Vista2 DatosPaciente

Esta vista se crea con la finalidad de tener en una tabla con toda la información sobre un paciente, desde sus doctores, su procedimiento, y sus medicamentos, puede ser útil para llevar una historia clínica del paciente.

```
173 • select * from DatosPaciente;
174 • create view DatosPaciente as
175 select tb_paciente.nombre_paciente as "paciente", tb_paciente.direccion as "direccion", tb_medimento.nombre_medimento as "medicamento", tb_medimento.dosis as "dosis"
176 from tb_paciente inner join dll_paciente_medimento on id_paciente=dll_paciente_medimento.id_paciente_m
177        inner join tb_medimento on id_medimento=dll_paciente_medimento.id_medimento_m;
178
179
```

paciente	direccion	medicamento	dosis
Jaime	calle falsa 121	Semilla del lemitario	2 mg al dia
Calamardo	calle falsa 122	Paracetamol	6mg cada 8 horas
Bob	calle falsa 123	Loratadina	mucho
Patricio	calle falsa 124	Dolex gripa	2 veces al dia 5mg
Don	calle falsa 125	Descansar	Siempre

Vista3 DatosCompleto

Se conectan casi todas las relaciones para guardar toda la información sobre un paciente, enfermeros, médicos, medicamentos y procedimiento.

```
180 • select * from DatosCompleto;
181 • create view DatosCompleto as
182 select tb_enfermeros.nombre_enfermero as "enfermero", tb_pacientes.nombre_paciente as "paciente", tb_procedimiento.tipo_procedimiento as "procedimiento",
183        tb_medico.nombre_medico as "doctor", tb_medimento.nombre_medimento as "medicamento", tb_medimento.dosis as "dosis"
184 from tb_pacientes inner join tb_procedimiento on id_procedimiento_f=tb_procedimiento.id_procedimiento
185        inner join dll_medico_procedimiento on id_procedimiento=dll_medico_procedimiento.id_procedimiento_p
186        inner join tb_medico on id_medico = dll_medico_procedimiento.id_medico_p
187        inner join tb_enfermeros on id_medico_e=tb_medico.id_medico
188        inner join dll_paciente_medimento on id_paciente_m = tb_pacientes.id_paciente
189        inner join tb_medimento on id_medimento=dll_paciente_medimento.id_medimento_m;
190
191
```

enfermero	paciente	procedimiento	doctor	medicamento	dosis
Laura	Jaime	Quirujico	Raul	Semilla del lemitario	2 mg al dia
Maga	Calamardo	Terapia	Fabio	Paracetamol	6mg cada 8 horas
Stuart	Bob	Contra el cancer	Julian	Loratadina	mucho
Trunks	Patricio	Psicologico	Wisin	Dolex gripa	2 veces al dia 5mg
Power	Don	Biopsia	Ash	Descansar	Siempre

Pregunta final

¿Qué le agregaría al modelo para dar más información y esa información cual sería?

R) Creo que lo mas importante al tener una clase factura es agregarle los costos a lo medicamentos y tener una tabla intermedia que podría llamarse pedido, que guarde el total a pagar, y un lista de medicamentos con su respectiva cantidad.

También se podría agregar el salario a los trabajadores, costo a los procedimientos y en realidad se podrían ser muchísimas más cosas que complicarían muchísimo la actividad.

TERCERA ACTIVIDAD

Procedimientos actividad 1

Agregar:

```
// Procedimiento que crea un registro en la tabla autor o prescrite según el parámetro nombre_tabla //
DELIMITER //
create procedure crear_registro(in dato1 varchar(20),in dato2 varchar(50),in dato3 varchar(50),in dato4 varchar(100), in nombre_tabla varchar(50))
BEGIN
if nombre_tabla = "libro" then
insert into libro values
(dato1,dato2,dato3,dato4);
elseif nombre_tabla="autor" then
insert into autor values
(dato1,dato2,dato3,dato4);
END IF;
END
//
```

El procedimiento crear_registro recibe 5 parametros, 4 de estos indican los datos que serán ingresados y el ultimo parámetro llamado nombre_tabla indica en que tabla será agregado el nuevo registro. En este caso solo funciona con las tablas autor y libro.

```
119 • call crear_registro("ABC321","El inicio del fin","255","Editorial1","libro");
120 • call crear_registro("1006","05/05/95","Argentino","Pedro Sanchez","autor");
```

Al ejecutar estos comandos debería crear el registro con llave primaria ABC321 en la tabla libro, y el registro con llave primaria 1006 en la tabla autor.

ISBN	titulo	numero_paginas	nombre_editorial
ABC130	Aprenda ingles	1000	Editorial3
ABC131	Aprenda frances	1200	Editorial1
ABC132	Aprenda programar	80	Editorial4
ABC133	Matematica finanzas...	54	Editorial2
ABC134	Calculo diferencial	453	Editorial1
ABC135	Calculo vectorial	415	Editorial3
ABC136	Fisica	57	Editorial1
ABC137	Leyes de newton	756	Editorial4
ABC138	La biblia	745	Editorial4
ABC139	El senior de los anillos	57	Editorial1
ABC140	The last of us	213	Editorial1
ABC141	Batman	214	Editorial3
ABC142	La caída de Edgar	456	Editorial2
ABC321	El inicio del fin	255	Editorial1

id	fecha de nacimiento	nacionalidad	nombre
1001	01/01/90	Colombiano	Juan Garcia
1002	12/06/95	Mexicano	Pedro Mancha
1003	13/10/86	Venezolano	Jhon Cena
1004	06/05/70	Colombiano	Mario Bross
1005	25/01/78	Peruano	Pablo Escobar
1006	05/05/95	Argentino	Pedro Sanchez

Buscar:

```
DELIMITER //
• create procedure buscar_libro(in titulo_libro varchar(50))
BEGIN
select * from libro where titulo=titulo_libro;
END
//
DELIMITER ;
```


El procedimiento buscar_libro recibe como parámetro el título de un libro y hacer la consulta en la tabla libro.

```

152
153 • call buscar_libro("El amanecer");
154

```

ISBN	título	numero_paginas	nombre_editorial
ABC123	El amanecer	100	Editorial4

Eliminar:

```

123
124 DELIMITER //
125 • create procedure eliminar_registro(in identificador varchar(20),in nombre_tabla varchar(40))
126 BEGIN
127 if nombre_tabla="libro" then
128 delete from libro where ISBN=identificador;
129 elseif nombre_tabla="autor" then
130 delete from autor where id=identificador;
131 elseif nombre_tabla="editorial" then
132 delete from editorial where nombre=identificador;
133 elseif nombre_tabla="cliente" then
134 delete from cliente where cedula=identificador;
135 END IF;
136 END
137 //
138 DELIMITER ;

```

El proceso eliminar_registro recibe un identificador (la llave primaria de una tabla) y el nombre la tabla de la cual desea eliminar el registro.

```

138 DELIMITER ;
139 • call eliminar_registro("ABC321","libro");
140 • select * from libro;
141 • call eliminar_registro("1006","autor");
142 • select * from autor;

```

Al ejecutar el procedimiento con esos datos, eliminaremos los registros creados en el ejemplo anterior.

```

141 • call eliminar_registro("1006","autor");
142 • select * from autor;

```

ISBN	título	numero_paginas	nombre_editorial
ABC129	Zelda	3600	Editorial1
ABC130	Aprenda ingles	1000	Editorial3
ABC131	Aprenda frances	1200	Editorial1
ABC132	Aprenda programar	80	Editorial4
ABC133	Matematica financ...	54	Editorial2
ABC134	Calculo diferencial	453	Editorial1
ABC135	Calculo vectorial	415	Editorial3
ABC136	Fisica	57	Editorial1
ABC137	Leyes de newton	756	Editorial4
ABC138	La biblia	745	Editorial4
ABC139	El señor de los anillos	57	Editorial1
ABC140	The last of us	213	Editorial1
ABC141	Batman	214	Editorial3
ABC142	La caída de Edgar	456	Editorial2

```

141 • call eliminar_registro("1006","autor");
142 • select * from autor;

```

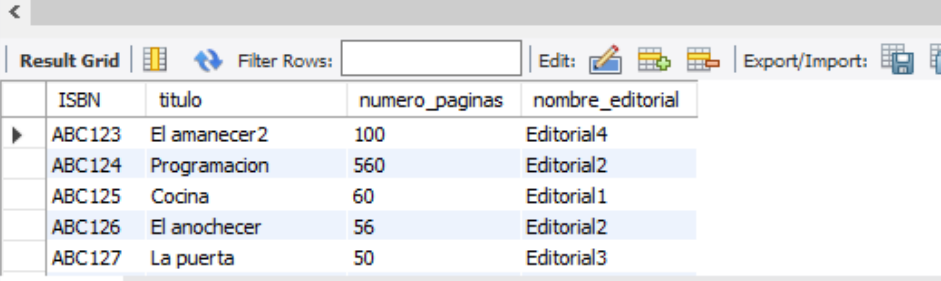
id	fecha de nacimiento	nacionalidad	nombre
1001	01/01/90	Colombiano	Juan Garcia
1002	12/06/95	Mexicano	Pedro Mancha
1003	13/10/86	Venezolano	Jhon Cena
1004	06/05/70	Colombiano	Mario Bross
1005	25/01/78	Peruano	Pablo Escobar

Editar:

```
DELIMITER //
create procedure editar_registro(in dato varchar(50), in atributo varchar(50),in identificador varchar(20),in tabla varchar(50))
BEGIN
if tabla="libro" then
    if atributo="titulo" then
        update libro
        set titulo = dato
        where ISBN=identificador;
    elseif atributo="numero_paginas" then
        update libro
        set numero_paginas = dato
        where ISBN=identificador;
    end if;
elseif tabla="autor" then
    if atributo="nombre" then
        update autor
        set nombre=dato
        where id=identificador;
    elseif atributo="nacionalidad" then
        update autor
        set nacionalidad=dato
        where id=identificador;
    end if;
elseif tabla="cliente" then
    update cliente
    set nombre=dato
    where cedula=identificador;
END IF;
END
//
DELIMITER ;
```

Este procedimiento recibe 4 parámetros, la tabla que se va a modificar, la columna que se va a modificar, el nuevo dato que será ingresado y el identificador del registro que se desea modificar.

```
185
186 • call editar_registro("El amanecer2","titulo","ABC123","libro");
187 • select * from libro;
188
```



	ISBN	titulo	numero_paginas	nombre_editorial
▶	ABC123	El amanecer2	100	Editorial4
	ABC124	Programacion	560	Editorial2
	ABC125	Cocina	60	Editorial1
	ABC126	El anocheceer	56	Editorial2
	ABC127	La puerta	50	Editorial3

libro 47 x

Se puede observar como cambio el titulo del libro (ABC123), antes era el amanecer y ahora es el amanecer2.

Triggers

Inicio creando la tabla control_de_cambio_libreria

```
188
189 • create table control_de_cambios_libreria(
190     usuario varchar(50),
191     accion varchar(50),
192     fecha datetime default current_timestamp
193 );
194
```

Creación de triggers

```
194
195 delimiter //
196 • create trigger ins_libro after insert on libro
197     for each row
198     begin
199         insert into control_de_cambios_libreria values
200             (user(),"agrego un libro",now());
201     end;
202 //
203 delimiter ;
204 delimiter //
205 • create trigger del_libro after delete on libro
206     for each row
207     begin
208         insert into control_de_cambios_libreria values
209             (user(),"elimino un libro",now());
210     end;
211 //
212 delimiter ;
```

Se crean los triggers ins_libro y del_libro, uno se encarga de los registros de creación y el otro de los registros de eliminación respectivamente, y estos registros son guardados en la tabla control_de_cambio_libreria.

```
214 • select * from libro;
215 • insert into libro values
216     ("ABC321","El inicio del fin","255","Editorial1");
217
218 • delete from libro
219     where ISBN="ABC321";
220 • select * from control_de_cambios_libreria;
221
222
```

Result Grid

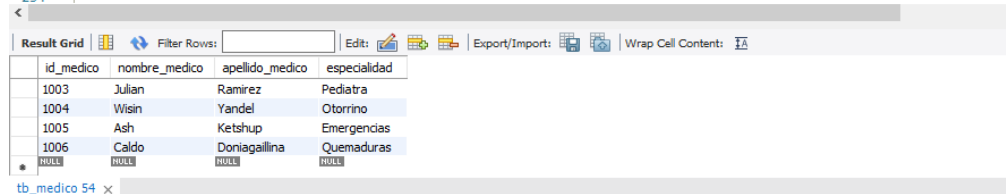
usuario	accion	fecha
root@localhost	agrego un libro	2023-02-14 14:01:43
root@localhost	agrego un libro	2023-02-14 14:04:09
root@localhost	elimino un libro	2023-02-14 14:04:17

Como se puede observar, funcionan perfectamente.

Procedimientos actividad 2:

Agregar:

```
223
224 DELIMITER //
225 • create procedure crear_medico(in dato1 varchar(20),in dato2 varchar(50),in dato3 varchar(50),in dato4 varchar(100))
226 BEGIN
227     insert into tb_medico values
228     (dato1,dato2,dato3,dato4);
229 END
230 //
231 DELIMITER ;
232 • select * from tb_medico;
233 • call crear_medico("1006","Caldo","Doniagallina","Quemaduras");
234
```



The screenshot shows a database client interface. The top part displays the SQL code entered in the previous block. Below the code, there is a 'Result Grid' section. It contains a table with the following data:

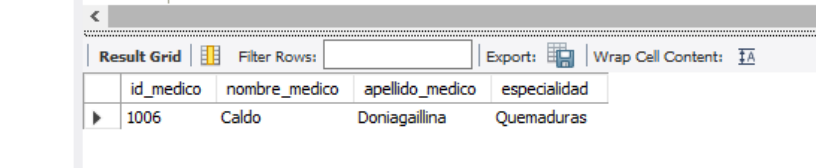
id_medico	nombre_medico	apellido_medico	especialidad
1003	Julian	Ramirez	Pediatra
1004	Wisin	Yandel	Otorrino
1005	Ash	Ketchup	Emergencias
1006	Caldo	Doniagallina	Quemaduras

At the bottom of the screenshot, a tab labeled 'tb_medico 54 x' is visible.

Se crea el procedimiento agregar_medico y se genera un registro utilizándolo.

Buscar:

```
256
257 DELIMITER //
258 • create procedure buscar_medico(in idmedico varchar(50))
259 BEGIN
260     select * from tb_medico where id_medico=idmedico;
261 END
262 //
263 DELIMITER ;
264 • call buscar_medico("1006");
```



The screenshot shows a database client interface. The top part displays the SQL code entered in the previous block. Below the code, there is a 'Result Grid' section. It contains a table with the following data:

id_medico	nombre_medico	apellido_medico	especialidad
1006	Caldo	Doniagallina	Quemaduras

Se crea el procedimiento buscar_medico y se consulta al medico creado en el ejemplo anterior, utilizando el procedimiento.

Editar:

```
265
266 DELIMITER //
267 • create procedure editar_medico(in dato varchar(50), in atributo varchar(50),in identificador varchar(20))
268 BEGIN
269 if atributo="nombre_medico" then
270     update tb_medico
271     set nombre_medico = dato
272     where id_medico=identificador;
273 elseif atributo="apellido_medico" then
274     update tb_medico
275     set apellido_medico = dato
276     where id_medico=identificador;
277 elseif atributo="especialidad" then
278     update tb_medico
279     set especialidad = dato
280     where id_medico=identificador;
281 end if;
282 end
283 //
284 DELIMITER ;
285 • call editar_medico("Jaimico","nombre_medico","1006");
286 • select * from tb_medico;
```

id_medico	nombre_medico	apellido_medico	especialidad
1001	Raul	Restrepo	Cirugia
1002	Fabio	Lopez	Dermatologo
1003	Julian	Ramirez	Pediatra
1004	Wisin	Yandel	Otorrino
1005	Ash	Ketshup	Emergencias
1006	Jaimico	Doniagallina	Quemaduras

Este procedimiento puede editar una de las columnas que no sea el identificador de la tabla tb_medico, recibe como parámetros el identificador del médico, el atributo a modificar y el dato con el que reemplazara la información que estaba antes.

Eliminar:

```
236
237 DELIMITER //
238 • create procedure eliminar_medico(in identificador varchar(20))
239 BEGIN
240 delete from tb_medico where id_medico=identificador;
241 END
242 //
243 DELIMITER ;
244 • call eliminar_medico("1006");
245 • select * from tb_medico;
```

id_medico	nombre_medico	apellido_medico	especialidad
1001	Raul	Restrepo	Cirugia
1002	Fabio	Lopez	Dermatologo
1003	Julian	Ramirez	Pediatra
1004	Wisin	Yandel	Otorrino
1005	Ash	Ketshup	Emergencias
NULL	NULL	NULL	NULL

Se crea el procedimiento eliminar_medico y se ejecuta para eliminar al medico con id=1006 que fue creado en los ejemplos anteriores.

Triggers

Se crea la tabla control_de_cambios_hospital

```
281
282 • create table control_de_cambios_hospital(
283     usuario varchar(50),
284     accion varchar(50),
285     fecha datetime default current_timestamp
286 );
287
```

Se crean ambos triggers

```
287
288 delimiter //
289 • create trigger ins_medico after insert on tb_medico
290     for each row
291     begin
292         insert into control_de_cambios_hospital values
293         (user(),"agrego un medico",now());
294     end;
295 //
296 delimiter ;
297 delimiter //
298 • create trigger del_medico after delete on tb_medico
299     for each row
300     begin
301         insert into control_de_cambios_hospital values
302         (user(),"elimino un medico",now());
303     end;
304 //
305 delimiter ;
```

Estos triggers guardan un registro en la tabla de control_de_cambios_hospital, cuando un usuario crea un nuevo médico o lo elimina.

```
308 • insert into tb_medico values
309     ("1006","Dominic","Toreto","Carros");
310
311 • delete from tb_medico
312     where id_medico="1006";
313 • select * from control_de_cambios_hospital;
314
```

Result Grid		
usuario	accion	fecha
root@localhost	agrego un medico	2023-02-14 14:35:33
root@localhost	elimino un medico	2023-02-14 14:35:37

Se crea y se elimina al medico con id=1006 para ejemplificar el funcionamiento correcto de los triggers.