

Consultas y vistas

Primera actividad:

1. Utilizando la sentencia INSERT INTO de SQL, se crean registros en las tablas Autor, Libro, Cliente, Editorial, Libro_cliente, Libro_Autor y teléfono_cliente.
 - Se insertan 4 registros en la tabla Editorial con las siguientes sentencias. Estas sentencias pueden ser consultadas en el archivo scprit-insert-editorial:

```
INSERT INTO libreriabuscalibre.editorial (nombre, ciudad, complemento, Telefono) VALUES ('Acantilado', 'Medellin', 'calle10sur58', '169263422');
```

```
INSERT INTO libreriabuscalibre.editorial (nombre, ciudad, complemento, Telefono) VALUES ('Altea', 'Medellin', 'carrera44-11', '60443338699');
```

```
INSERT INTO libreriabuscalibre.editorial (nombre, ciudad, complemento, Telefono) VALUES ('Babel', 'Medellin', 'circular1-220', '72678146');
```

```
INSERT INTO libreriabuscalibre.editorial (nombre, ciudad, complemento, Telefono) VALUES ('Planeta', 'Medellin', 'transversal2-12', '93583301');
```

	nombre	ciudad	complemento	Telefono
	Acantilado	Medellin	calle10sur58	169263422
	Altea	Medellin	carrera44-11	60443338699
	Babel	Medellin	circular1-220	72678146
▶	Planeta	Medellin	transversal2-12	93583301
*	NULL	NULL	NULL	NULL

Figura 1: Tabla Editorial

```
INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('91679210', '20 leguas de viaje submarino', '200', 'Acantilado');
```

```
INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('3362398', 'De la tierra a la luna', '300', 'Acantilado');
```

```
INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('153498214', 'El castillo de los Carpatos', '158', 'Acantilado');
```

```
INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('120778212', 'Drácula', '250', 'Acantilado');
```

INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('10606345', 'El padrino', '254', 'Altea');

INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('183861911', 'La casa de las dos palmas', '300', 'Altea');

INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('193291785', 'Frutos de mi tierra', '154', 'Altea');

INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('150927636', 'El testamento del paisa', '400', 'Babel');

INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('98862798', 'La marquesa de Yolombo', '200', 'Planeta');

INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('79620954', 'El conde de Montecristo', '158', 'Planeta');

INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('96371970', 'El tulipan negro', '140', 'Planeta');

INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('89427792', '1984', '220', 'Babel');

INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('71625169', 'Rebelión en la granja', '100', 'Babel');

INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('195647355', 'Crimen y castigo', '300', 'Planeta');

INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('197457990', 'La Eneida', '150', 'Planeta');

INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('193347930', 'La Iliada', '150', 'Acantilado');

INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('85124527', 'Los venenos de la corona', '180', 'Babel');

INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('175499761', 'Tormenta de espadas', '450', 'Acantilado');

INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('189799297', 'Danza de dragones', '450', 'Planeta');

INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('153951522', 'El Silmarillion', '300', 'Planeta');

- Se insertan 20 registros en la tabla libros. El script puede consultarse en el archivo script-insert-libro.

```
INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('91679210', '20 leguas de viaje submarino', '200', 'Acantilado');
```

```
INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('3362398', 'De la tierra a la luna', '300', 'Acantilado');
```

```
INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('153498214', 'El castillo de los Carpatos', '158', 'Acantilado');
```

```
INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('120778212', 'Drácula', '250', 'Acantilado');
```

```
INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('10606345', 'El padrino', '254', 'Altea');
```

```
INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('183861911', 'La casa de las dos palmas', '300', 'Altea');
```

```
INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('193291785', 'Frutos de mi tierra', '154', 'Altea');
```

```
INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('150927636', 'El testamento del paisa', '400', 'Babel');
```

```
INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('98862798', 'La marquesa de Yolombo', '200', 'Planeta');
```

```
INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('79620954', 'El conde de Montecristo', '158', 'Planeta');
```

```
INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('96371970', 'El tulipan negro', '140', 'Planeta');
```

```
INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('89427792', '1984', '220', 'Babel');
```

```
INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('71625169', 'Rebelión en la granja', '100', 'Babel');
```

```
INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('195647355', 'Crimen y castigo', '300', 'Planeta');
```

```
INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('197457990', 'La Eneida', '150', 'Planeta');
```

```
INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('193347930', 'La Iliada', '150', 'Acantilado');
```

INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('85124527', 'Los venenos de la corona', '180', 'Babel');

INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('175499761', 'Tormenta de espadas', '450', 'Acantilado');

INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('189799297', 'Danza de dragones', '450', 'Planeta');

INSERT INTO libreriabuscalibre.libro (ISBN, titulo, numero_paginas, nombre_editorial) VALUES ('153951522', 'El Silmarillion', '300', 'Planeta');

	ISBN	titulo	numero_paginas	nombre_editorial
▶	10606345	El padrino	254	Altea
	120778212	Drácula	250	Acantilado
	150927636	El testamento del paisa	400	Babel
	153498214	El castillo de los Carpatos	158	Acantilado
	153951522	El Silmarillion	300	Planeta
	175499761	Tormenta de espadas	450	Acantilado
	183861911	La casa de las dos palmas	300	Altea
	189799297	Danza de dragones	450	Planeta
	193291785	Frutos de mi tierra	154	Altea
	193347930	La Iliada	150	Acantilado
	195647355	Crimen y castigo	300	Planeta
	197457990	La Eneida	150	Planeta
	3362398	De la tierra a la luna	300	Acantilado
	71625169	Rebelión en la granja	100	Babel
	79620954	El conde de Montecristo	158	Planeta
	85124527	Los venenos de la corona	180	Babel
	89427792	1984	220	Babel
	91679210	20 leguas de viaje submarino	200	Acantilado
	96371970	El tulipan negro	140	Planeta
	98862798	La marquesa de Yolombo	200	Planeta

Figura 2: Tabla Libro

- Se insertan 7 registros en la tabla Cliente. El archivo con el script se puede encontrar en el archivo script-insert-cliente.

	cedula	nombre
	28570803	Paola Restrepo
	180053541	Natalia Velez
	146947827	Leonel Alvarez
	87971349	Hernan Torres
	24422640	Doris Salas
	162803699	David Gonzales
▶	80172781	Alejandro Restrepo
✱	NULL	NULL

Figura 3: Tabla Cliente

- Se insertan 11 registros en la tabla Autor. El script con las sentencias SQL se pueden consultar en el archivo script-insert-autor.

```
INSERT INTO libreriabuscalibre.autor (id, `fecha de nacimiento`, nacionalidad, nombre) VALUES ('28247212', '08/02/1828', 'Francesa', 'Julio Verne');
```

```
INSERT INTO libreriabuscalibre.autor (id, `fecha de nacimiento`, nacionalidad, nombre) VALUES ('169283941', '17/01/1858', 'Colombiana', 'Tomas Carrasquilla');
```

```
INSERT INTO libreriabuscalibre.autor (id, `fecha de nacimiento`, nacionalidad, nombre) VALUES ('122123299', '23/04/1923', 'Colombiana', 'Manuel Mejia Vallejo');
```

```
INSERT INTO libreriabuscalibre.autor (id, `fecha de nacimiento`, nacionalidad, nombre) VALUES ('70024041', '15/10/1920', 'Estadounidense', 'Mario Puzzo');
```

```
INSERT INTO libreriabuscalibre.autor (id, `fecha de nacimiento`, nacionalidad, nombre) VALUES ('33374070', '24/07/1870', 'Francesa', 'Alejandro Dumas');
```

```
INSERT INTO libreriabuscalibre.autor (id, `fecha de nacimiento`, nacionalidad, nombre) VALUES ('116792844', '02/02/1923', 'Colombiana', 'Agustín Jaramillo');
```

```
INSERT INTO libreriabuscalibre.autor (id, `fecha de nacimiento`, nacionalidad, nombre) VALUES ('41133823', '03/012/1892', 'Inglesa', 'J.R.R. Tolkien');
```

```
INSERT INTO libreriabuscalibre.autor (id, `fecha de nacimiento`, nacionalidad, nombre) VALUES ('171591866', '70 AC', 'Romano', 'Virgilio');
```

```
INSERT INTO `libreriabuscalibre`.`autor` (id, `fecha de nacimiento`, nacionalidad, nombre) VALUES ('107362972', 'siglo VIII AC', 'Griego', 'Homero');
```

```
INSERT INTO `libreriabuscalibre`.`autor` (id, `fecha de nacimiento`, nacionalidad, nombre) VALUES ('74643161', '20/09/1948', 'Estadounidense', 'George R.R. Martin');
```

```
INSERT INTO `libreriabuscalibre`.`autor` (id, `fecha de nacimiento`, nacionalidad, nombre) VALUES ('121166070', '14/04/1918', 'Francesa', 'Maurice Druon');
```

	id	fecha de nacimiento	nacionalidad	nombre
	28247212	08/02/1828	Francesa	Julio Verne
	169283941	17/01/1858	Colombiana	Tomas Carrasquilla
	122123299	23/04/1923	Colombiana	Manuel Mejia Vallejo
	70024041	15/10/1920	Estadounidense	Mario Puzzo
	33374070	24/07/1870	Francesa	Alejandro Dumas
	116792844	02/02/1923	Colombiana	Agustín Jaramillo
	41133823	03/012/1892	Inglesa	J.R.R. Tolkien
	171591866	70 AC	Romano	Virgilio
	107362972	siglo VIII AC	Griego	Homero
	74643161	20/09/1948	Estadounidense	George R.R. Martin
▶	121166070	14/04/1918	Francesa	Maurice Druon
*	NULL	NULL	NULL	NULL

Figura 4: Tabla Autor

- Se insertan 10 registros en la tabla Libro_autor. El script con las secuencias SQL puede encontrarse en el archivo script-insert-libro_autor.

```
INSERT INTO libreriabuscalibre.libro_autor (ISBN_libro, id_autor) VALUES ('10606345',
'70024041');
```

```
INSERT INTO libreriabuscalibre.libro_autor (ISBN_libro, id_autor) VALUES ('150927636',
'116792844');
```

```
INSERT INTO libreriabuscalibre.libro_autor (ISBN_libro, id_autor) VALUES ('153498214',
'28247212');
```

```
INSERT INTO libreriabuscalibre.libro_autor (ISBN_libro, id_autor) VALUES ('153951522',
'41133823');
```

```
INSERT INTO libreriabuscalibre.libro_autor (ISBN_libro, id_autor) VALUES ('175499761',
'74643161');
```

```
INSERT INTO libreriabuscalibre.libro_autor (ISBN_libro, id_autor) VALUES ('183861911',
'122123299');
```

```
INSERT INTO libreriabuscalibre.libro_autor (ISBN_libro, id_autor) VALUES ('193291785',
'169283941');
```

```
INSERT INTO libreriabuscalibre.libro_autor (ISBN_libro, id_autor) VALUES ('79620954',
'33374070');
```

```
INSERT INTO libreriabuscalibre.libro_autor (ISBN_libro, id_autor) VALUES ('91679210',
'28247212');
```

```
INSERT INTO libreriabuscalibre.libro_autor (ISBN_libro, id_autor) VALUES ('96371970',
'33374070');
```

ISBN_libro	id_autor
10606345	70024041
150927636	116792844
153498214	28247212
153951522	41133823
175499761	74643161
183861911	122123299
193291785	169283941
79620954	33374070
91679210	28247212
96371970	33374070

Figura 5: Tabla Libro_autor

- Se insertan 16 registros en la tabla teléfono_cliente. El script con las secuencias para insertar la información en la tabla se encuentra en el archivo script-insert-telefono_cliente_

```
INSERT INTO libreriabuscalibre.telefono_cliente (cedula_cliente, numero) VALUES ('146947827',
'60456817498');
```

```
INSERT INTO libreriabuscalibre.telefono_cliente (cedula_cliente, numero) VALUES ('146947827',
'40589360');
```

```
INSERT INTO libreriabuscalibre.telefono_cliente (cedula_cliente, numero) VALUES ('146947827',
'176331154');
```

```
INSERT INTO libreriabuscalibre.telefono_cliente (cedula_cliente, numero) VALUES ('80172781',
'165554977');
```

```
INSERT INTO libreriabuscalibre.telefono_cliente (cedula_cliente, numero) VALUES ('80172781',
'108880275');
```

```
INSERT INTO libreriabuscalibre.telefono_cliente (cedula_cliente, numero) VALUES ('162803699',
'60488515453');
```

```
INSERT INTO libreriabuscalibre.telefono_cliente (cedula_cliente, numero) VALUES ('87971349',
'126225290');
```

```
INSERT INTO libreriabuscalibre.telefono_cliente (cedula_cliente, numero) VALUES ('87971349',
'60426123261');
```

```
INSERT INTO libreriabuscalibre.telefono_cliente (cedula_cliente, numero) VALUES ('24422640', '91320800');
```

```
INSERT INTO libreriabuscalibre.telefono_cliente (cedula_cliente, numero) VALUES ('24422640', '192670886');
```

```
INSERT INTO libreriabuscalibre.telefono_cliente (cedula_cliente, numero) VALUES ('24422640', '6043634490');
```

```
INSERT INTO libreriabuscalibre.telefono_cliente (cedula_cliente, numero) VALUES ('28570803', '60433803341');
```

```
INSERT INTO libreriabuscalibre.telefono_cliente (cedula_cliente, numero) VALUES ('28570803', '177782150');
```

```
INSERT INTO libreriabuscalibre.telefono_cliente (cedula_cliente, numero) VALUES ('180053541', '60493926757');
```

```
INSERT INTO libreriabuscalibre.telefono_cliente (cedula_cliente, numero) VALUES ('180053541', '163132007');
```

```
INSERT INTO libreriabuscalibre.telefono_cliente (cedula_cliente, numero) VALUES ('180053541', '135492588');
```

	cedula_cliente	numero
	146947827	60456817498
	146947827	40589360
	146947827	176331154
	80172781	165554977
	80172781	108880275
	162803699	60488515453
	87971349	126225290
	87971349	60426123261
	24422640	91320800
	24422640	192670886
	24422640	6043634490
	28570803	60433803341
	28570803	177782150
	180053541	60493926757
	180053541	163132007
	180053541	135492588
▶	NULL	NULL

Figura 6: Tabla telefono_cliente

- Se insertan 10 registros en la tabla libro_cliente. El script son las sentencias SQL para insertar la información en la tabla se encuentra en el archivo script-insert-libro_cliente.


```

INSERT INTO libreriabuscalibre.libro_cliente (ISBN_libro_cliente, id_cliente) VALUES ('10606345',
'146947827');

INSERT INTO libreriabuscalibre.libro_cliente (ISBN_libro_cliente, id_cliente) VALUES ('150927636',
'146947827');

INSERT INTO libreriabuscalibre.libro_cliente (ISBN_libro_cliente, id_cliente) VALUES ('153498214',
'80172781');

INSERT INTO libreriabuscalibre.libro_cliente (ISBN_libro_cliente, id_cliente) VALUES ('153951522',
'162803699');

INSERT INTO libreriabuscalibre.libro_cliente (ISBN_libro_cliente, id_cliente) VALUES ('175499761',
'162803699');

INSERT INTO libreriabuscalibre.libro_cliente (ISBN_libro_cliente, id_cliente) VALUES ('183861911',
'162803699');

INSERT INTO libreriabuscalibre.libro_cliente (ISBN_libro_cliente, id_cliente) VALUES ('193291785',
'87971349');

INSERT INTO libreriabuscalibre.libro_cliente (ISBN_libro_cliente, id_cliente) VALUES ('79620954',
'24422640');

INSERT INTO libreriabuscalibre.libro_cliente (ISBN_libro_cliente, id_cliente) VALUES ('91679210',
'28570803');

INSERT INTO libreriabuscalibre.libro_cliente (ISBN_libro_cliente, id_cliente) VALUES ('96371970',
'180053541');

```

	ISBN_libro_cliente	id_cliente
	10606345	146947827
	150927636	146947827
	153498214	80172781
	153951522	162803699
	175499761	162803699
	183861911	162803699
	193291785	87971349
	79620954	24422640
	91679210	28570803
▶	96371970	180053541
•	NULL	NULL

Figura 7: Tabla libro_cliente

2. Ahora se pretende obtener información valiosa de las tablas de la base de datos.

- Primero se implementa una consulta para conocer el nombre y la fecha de nacimiento de cada autor registrado en la tabla Autor. Esta sentencia se encuentra en **scripts/script-select-nombre-fecha-autor**:

```
1 • SELECT autor.nombre, autor.`fecha de nacimiento` FROM libreriaescalibre.autor;
```

En este caso se utiliza una sentencia SELECT, donde se seleccionan las columnas nombre y fecha de nacimiento de la tabla Autor. El resultado se presenta en la figura 8.

	nombre	fecha de nacimiento
▶	Homero	siglo VIII AC
	Agustín Jaramillo	02/02/1923
	Maurice Druon	14/04/1918
	Manuel Mejía Vallejo	23/04/1923
	Tomas Carrasquilla	17/01/1858
	Virgilio	70 AC
	Julio Verne	08/02/1828
	Alejandro Dumas	24/07/1870
	J.R.R. Tolkien	03/01/1892
	Mario Puzzo	15/10/1920
	George R.R. Martin	20/09/1948

Figura 8: Resultado consulta Nombre y fecha nacimiento Autor

- Sentencia para conocer la cantidad de libros vendidos:

```
1 • SELECT COUNT(ISBN_libro_cliente) AS cantidad_libros_vendidos
2 FROM libreriaescalibre.libro_cliente;
```

La anterior sentencia se encuentra en **scripts/script-count-libros**.

En este caso se utiliza COUNT DISTINCT para contar el número de registros diferentes en la tabla libro_clientes; a la cual se le aplica el alias de “cantidad_libros_vendidos”. El resultado se presenta en la figura 9.

	cantidad_libros_vendidos
▶	10

Figura 9: Resultado consulta cantidad libros vendidos

- Sentencia para listar clientes con sus respectivos números telefónicos:

```

1 • SELECT cliente.nombre, telefono_cliente.numero
2   FROM cliente INNER JOIN telefono_cliente
3   ON cliente.cedula = telefono_cliente.cedula_cliente;

```

La anterior sentencia se encuentra en **scripts/script-join-telefono-cliente**.

En este caso se utiliza un INNER JOIN entre las tablas cliente y teléfono_cliente, y se seleccionan las columnas nombre de la tabla Cliente y numero de la tabla teléfono_cliente. El resultado se presenta en la figura 10

	nombre	numero
▶	Leonel Alvarez	176331154
	Leonel Alvarez	40589360
	Leonel Alvarez	60456817498
	David Gonzales	60488515453
	Natalia Velez	135492588
	Natalia Velez	163132007
	Natalia Velez	60493926757
	Doris Salas	192670886
	Doris Salas	6043634490
	Doris Salas	91320800
	Paola Restrepo	177782150
	Paola Restrepo	60433803341
	Alejandro Restrepo	108880275
	Alejandro Restrepo	165554977
	Hernan Torres	126225290
	Hernan Torres	60426123261

Figura 10: Resultado consulta para listar clientes y su número telefónico

- Sentencia para listar los libros, acompañados de su autor

```
1 • SELECT libro.titulo, autor.nombre
2 FROM ((libreriabuscalibre.libro
3 INNER JOIN libreriabuscalibre.libro_autor ON libro.ISBN = libro_autor.ISBN_libro)
4 INNER JOIN libreriabuscalibre.autor ON id_autor = autor.id);
```

La anterior sentencia se encuentra en **scripts/script-join-libro-autor**.

En este caso se utiliza un doble INNER JOIN. El primero se realiza entre la tabla libro y libro_autor, para relacionar cada libro con su autor. Posteriormente se realiza el segundo INNER JOIN entre la tabla resultante del primer join y la tabla autor. El resultado se presenta en la figura 11.

	titulo	nombre
►	El testamento del paisa	Agustín Jaramillo
	La casa de las dos palmas	Manuel Mejía Vallejo
	Frutos de mi tierra	Tomas Carrasquilla
	El castillo de los Carpatos	Julio Verne
	20 leguas de viaje submarino	Julio Verne
	El conde de Montecristo	Alejandro Dumas
	El tulipán negro	Alejandro Dumas
	El Silmarillion	J.R.R. Tolkien
	El padrino	Mario Puzzo
	Tormenta de espadas	George R.R. Martin

Figura 11: Resultado consulta para listar libro y su autor

- Sentencia para listar las editoriales que han logrado vender libros

```
1 • SELECT DISTINCT nombre_editorial
2 FROM libreriabuscalibre.libro
3 LEFT JOIN libreriabuscalibre.libro_cliente
4 ON libro.ISBN = libro_cliente.ISBN_libro_cliente
```

La anterior sentencia se encuentra en **scripts/scrip-leftjoin-editorial**

En este caso, se desea obtener información de la tabla Libro y de la tabla Libro_cliente; la cual registra las ventas realizadas por la librería. Sin embargo, no nos interesa en esta consulta los datos de los clientes que adquirieron los libros, únicamente el nombre de la editorial. Por este motivo se decide realizar la consulta utilizando un left join. El resultado se presenta en la figura 12.

	nombre_editorial
▶	Acantilado
	Altea
	Babel
	Planeta

Figura 12: Resultado consulta editoriales que han vendido libros

3. Se decide crear la vista cantidad_libros_vendidos.

```
CREATE VIEW cantidad_libros_vendidos AS  
SELECT COUNT(ISBN_libro_cliente) AS cantidad_libros_vendidos  
FROM libreriabuscalibre.libro_cliente;
```

La razón para elegir esta vista en particular es porque, a pesar de ser una consulta simple, esta se estaría realizando constantemente para llevar un registro de las ventas de la librería; por lo que sería conveniente tenerla definida.

La segunda vista que se crea es la de editoriales que han vendido libros:

```
CREATE VIEW ventas_editorial AS  
SELECT DISTINCT nombre_editorial  
FROM libreriabuscalibre.libro  
LEFT JOIN libreriabuscalibre.libro_cliente  
ON libro.ISBN = libro_cliente.ISBN_libro_cliente;
```

Se decide crear esta vista porque permite consultar las editoriales que han logrado vender libros; esto ayuda a tomar decisiones a la hora de adquirir libros de estas editoriales en el futuro.

Segunda actividad:

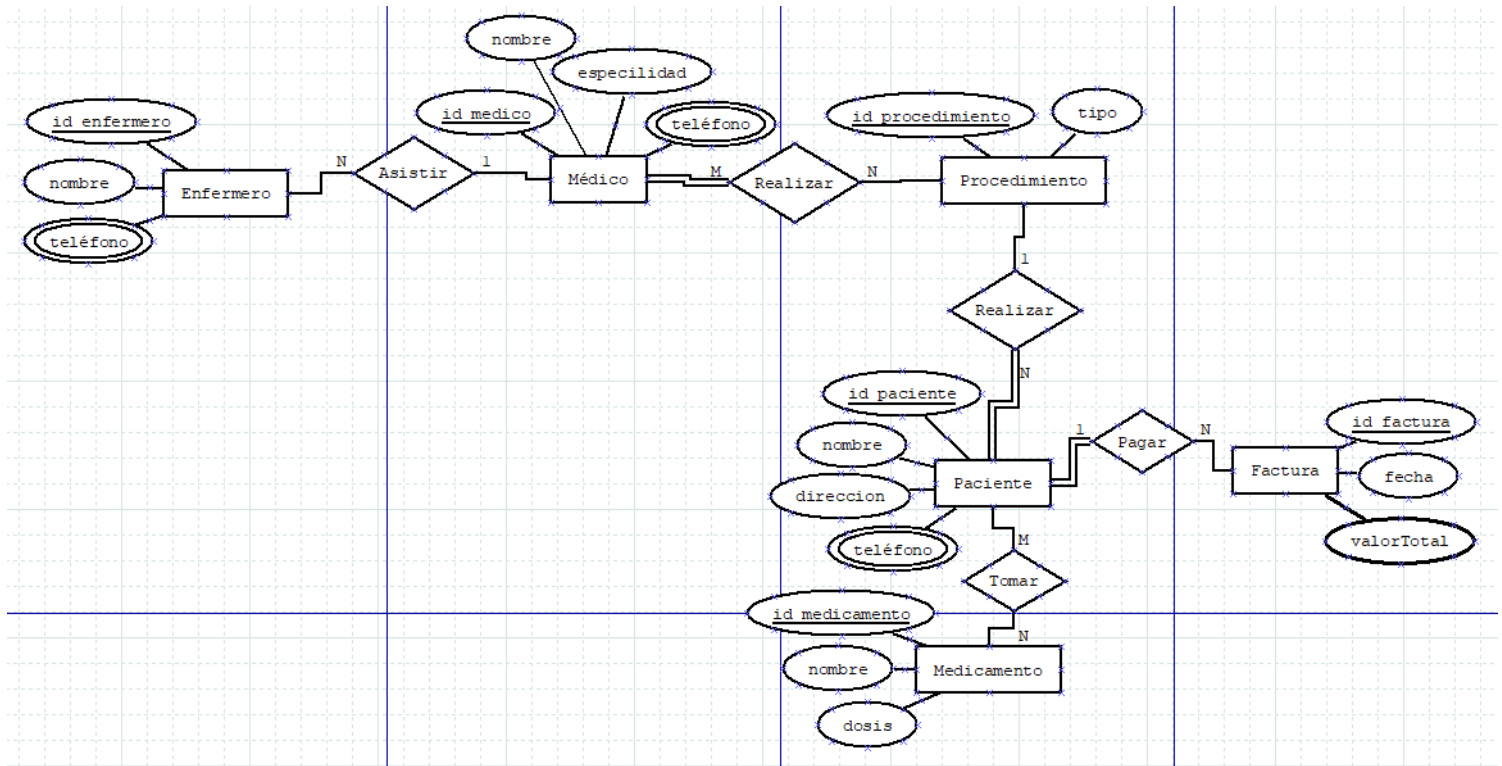


Figura 13: MER procedimiento médico

1. Se parte del diagrama que se presenta en la figura 13, para generar el Modelo relacional en Worbench. El MR resultante se observa en la figura 14.

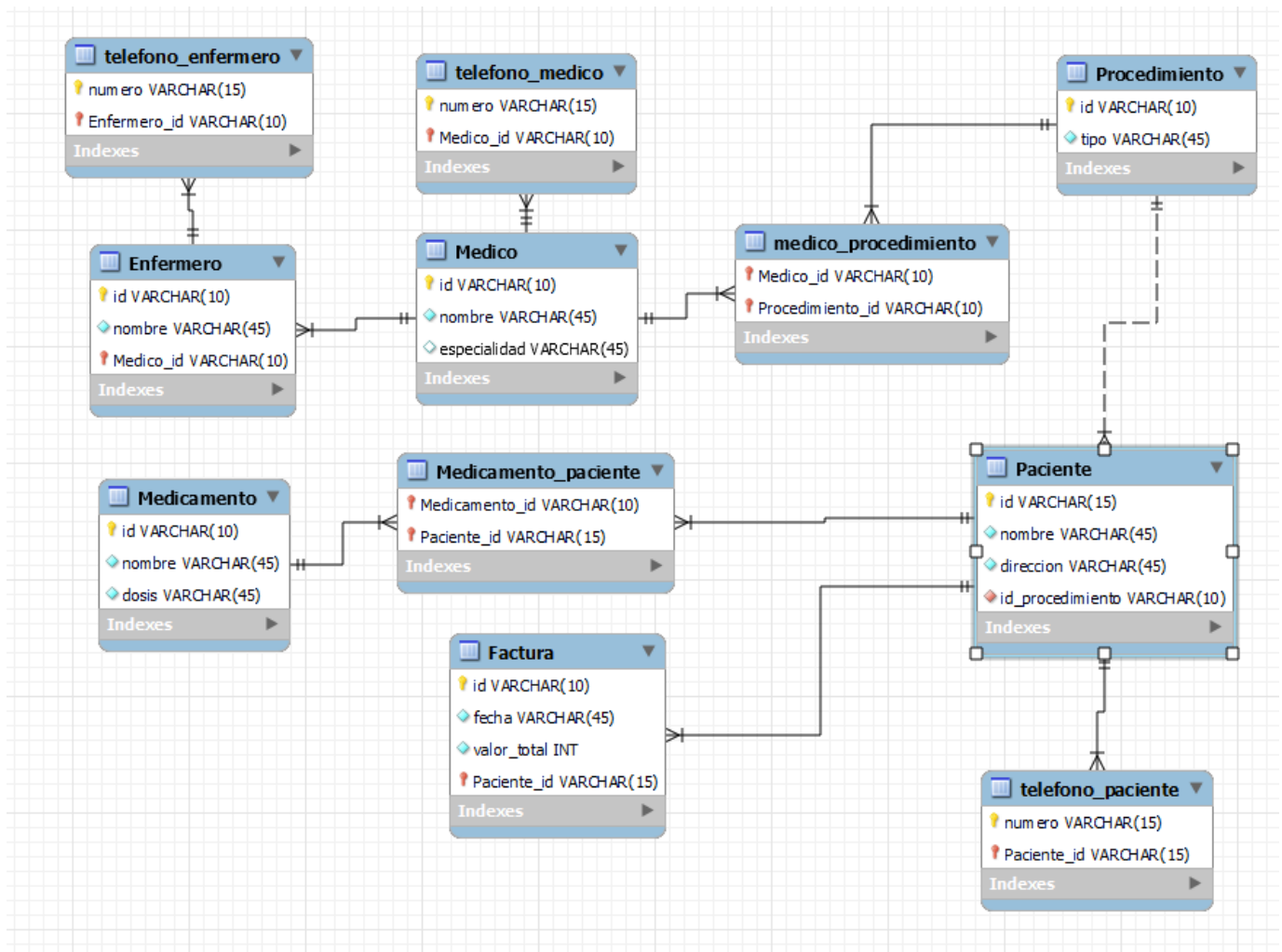


Figura 14: MR procedimiento_medico

- Una vez generado el MR, se procede a crear las tablas de la base de datos, utilizando Forward Engineer. Se generan 5 registros por tabla, los cuales se presentan a continuación.

	id	nombre	especialidad
	1165563701	Mariela Delagado	general
	1192410757	Andrés Salgado	medicina deportiva
	1889548204	Jorge Lopez	neurólogo
	35591894	Natalia Ramirez	cardiología
▶	820220422	Ester Gomez	pediatria
*	NULL	NULL	NULL

Figura 15: Tabla Médico

	id	nombre	Medico_id
	1270427242	Hugo Morales	1165563701
	2360106380	Martín Cáceres	1192410757
	2543160056	Lucia Gutierrez	35591894
	279893944	Sofía Lopez	820220422
▶	2979613909	Lucas Hernandez	1889548204
✱	NULL	NULL	NULL

Figura 16: Tabla Enfermero

	numero	Enfermero_id
	1300862640	1270427242
	6041479001675	1270427242
	894690590	2360106380
	2041782121	2543160056
	604926816110	279893944
	201362350	2979613909
▶	2752548590	2979613909
✱	NULL	NULL

Figura 17: Tabla telefono_enfermero

	numero	Medico_id
	1182133689	1165563701
	2173487391	1165563701
	132465672	1192410757
	578688925	1889548204
	1117936992	35591894
	476353625	820220422
▶	604232649228	820220422
✱	NULL	NULL

Figura 18: Tabla telefono_médico

	id	tipo
	1028107570	cita general
	1525532732	cita cardiología
▶	2052524500	terapia
	2953104284	lectura exámenes
	827989848	toma presión
*	NULL	NULL

Figura 19: Tabla Procedimiento

	id	nombre	direccion	id_procedimiento
▶	1278080243	Daniel Rodriguez	carrera50-20	2052524500
	2323492316	Camilo Tapias	circular9-212	2953104284
	2846306193	Paola Restrepo	calle 10a5	1525532732
	2978732323	Luisa Monedero	calle 12sur 56	1028107570
	483205679	Giovanny Moreno	transversal32a23	827989848
*	NULL	NULL	NULL	NULL

Figura 20: Tabla Paciente

	numero	Paciente_id
	30025469	1278080243
	6043435810	2323492316
	6042558904	2846306193
	258963147	2978732323
▶	22225698745	483205679
*	NULL	NULL

Figura 21: Tabla telefono_paciente

	id	nombre	dosis
	1672934475	Amoxicilina	20
	1705333834	Omeprazol	10
	2190413320	Silimarina	30
	2200265884	Acetaminofen	20
▶	2659799262	Paracetamol	30
*	NULL	NULL	NULL

Figura 22: Tabla Medicamento

id	fecha	valor_total	Paciente_id
1494166189	12/02/2023	12000	1278080243
2020858438	12/02/2023	10000	483205679
2479978458	12/02/2023	50000	2846306193
2523280659	12/02/2023	30000	2323492316
369751182	12/02/2023	20000	2978732323
NULL	NULL	NULL	NULL

Figura 23: Tabla Factura

Medico_id	Procedimiento_id
1165563701	1028107570
820220422	1028107570
35591894	1525532732
1192410757	2052524500
1889548204	2953104284
1165563701	827989848
NULL	NULL

Figura 24: Tabla Medico_procedimiento

Medicamento_id	Paciente_id
1672934475	1278080243
1705333834	2323492316
2190413320	2846306193
2200265884	2978732323
2659799262	483205679
NULL	NULL

Figura 25: Tabla Medicamento_paciente

- Ahora se implementa una consulta para conocer los medicamentos que consume cada paciente, con la dosis suministrada. En el archivo **scripts/script-join-medicamento-paciente**.

```

1 • SELECT medicamento.nombre AS medicamento, medicamento.dosis, paciente.nombre AS paciente
2 FROM ((procedimiento_medico.medicamento
3 INNER JOIN procedimiento_medico.medicamento_paciente ON medicamento.id = medicamento_paciente.medicamento_id)
4 INNER JOIN procedimiento_medico.paciente ON paciente.id = paciente_id);
5

```

	medicamento	dosis	paciente
▶	Amoxicilina	20	Daniel Rodriguez
	Omeprazol	10	Camilo Tapias
	Silimarina	30	Paola Restrepo
	Acetaminofen	20	Luisa Monedero
	Paracetamol	30	Giovanny Moreno

Figura 26: Resultado de la consulta para listar medicamentos recetados a cada paciente

Para obtener el resultado de la tabla 27 fue necesario llevar a cabo 2 INNER JOIN.

El primero entre las tablas Medicamento y medicamento_paciente, para relacionar cada medicamento con el id del paciente. Luego se realiza un join entre la tabla resultante y la tabla Paciente, para agregar el atributo nombre del paciente a la tabla resultante final.

4. Para realizar la consulta que me permita conocer los enfermeros que asistieron cada procedimiento de los pacientes se utiliza el script de la figura 28. Este puede ser encontrado en **scripts/script-join-enfermero-paciente**

```

1 • SELECT enfermero.nombre AS nombre_enfermero, procedimiento.tipo AS tipo_procedimiento, paciente.nombre AS nombre_paciente
2 FROM procedimiento_medico.enfermero
3 INNER JOIN procedimiento_medico.medico_procedimiento ON enfermero.medico_id = medico_procedimiento.medico_id
4 INNER JOIN procedimiento_medico.procedimiento ON procedimiento_id = procedimiento.id
5 INNER JOIN procedimiento_medico.paciente ON procedimiento_id = paciente.id_procedimiento;

```

	nombre_enfermero	tipo_procedimiento	nombre_paciente
▶	Hugo Morales	cita general	Luisa Monedero
	Sofía Lopez	cita general	Luisa Monedero
	Lucía Gutierrez	cita cardiología	Paola Restrepo
	Martín Cáceres	terapia	Daniel Rodríguez
	Lucas Hernandez	lectura exámenes	Camilo Tapias
	Hugo Morales	toma presión	Giovanny Moreno

Figura 27: Resultado consulta para listar enfermero-tipo procedimiento-paciente

Para obtener la tabla de la figura 28 fue necesario llevar a cabo 3 INNER JOIN.

El primero entre las tablas Enfermero y medico_procedimiento, para relacionar al enfermero con el id del procedimiento mediante la llave foránea medico_id.

El segundo INNER JOIN se realiza entre la tabla resultante del primer join con la tabla procedimiento, para obtener de esta ultima el tipo de procedimiento.

Por ultimo se relacionan el resultado del segundo join con la tabla Paciente, para agregar el nombre del paciente a la tabla resultante.

5. Ahora se procede a crear 3 vistas de consultas que se realicen de manera frecuente y que aporten información valiosa para el negocio.
 - La primera vista sería para visualizar cada paciente con el procedimiento recibido, y el medico que lo realizó. Para crear la vista se utiliza el script del archivo **scripts/script-view-medico-procedimiento-paciente**.

```

1 • Use procedimiento_medico;
2 • CREATE VIEW medico_procedimiento_paciente AS
3 SELECT medico.nombre AS nombre_medico, procedimiento.tipo AS tipo_procedimiento, paciente.nombre AS nombre_paciente
4 FROM   procedimiento_medico.medico
5        INNER JOIN procedimiento_medico.medico_procedimiento ON medico.id = medico_procedimiento.medico_id
6        INNER JOIN procedimiento_medico.procedimiento ON procedimiento_id = procedimiento.id
7        INNER JOIN procedimiento_medico.paciente ON procedimiento_id = paciente.id_procedimiento;

```

	nombre_medico	tipo_procedimiento	nombre_paciente
►	Mariela Delagado	cita general	Luisa Monedero
	Ester Gomez	cita general	Luisa Monedero
	Natalia Ramirez	cita cardiología	Paola Restrepo
	Andrés Salgado	terapia	Daniel Rodriguez
	Jorge Lopez	lectura exámenes	Camilo Tapias
	Mariela Delagado	toma presión	Giovanny Moreno

Figura 28: Vista médico-procedimiento-paciente

Se elije crear esta vista, porque permite llevar un historial de los procedimientos que recibe cada paciente y el médico que los lleva a cabo; información valiosa que debe ser registrada en la historia clínica de cada paciente.

- La segunda vista se implementa para visualizar la fecha del procedimiento, el valor total a pagar y el paciente que recibe el procedimiento. El script para crear la vista se encuentra en **scripts/script-view-fecha-valor-paciente**

```

1 • USE procedimiento_medico;
2 • CREATE VIEW factura_paciente AS
3   SELECT factura.valor_total, factura.fecha, paciente.nombre
4   FROM factura INNER JOIN paciente
5   ON factura.Paciente_id = paciente.id;

```

	valor_total	fecha	nombre
►	12000	12/02/2023	Daniel Rodriguez
	10000	12/02/2023	Giovanny Moreno
	50000	12/02/2023	Paola Restrepo
	30000	12/02/2023	Camilo Tapias
	20000	12/02/2023	Luisa Monedero

Figura 29: Vista factura-cliente

Esta vista se implementa, pues es una consulta que se realiza de manera frecuente para generar la factura de cada cliente, por lo que es conveniente tenerla predefinida.

- Por último, se crea una vista a partir de la consulta que relaciona la dosis asignada de medicamentos a cada paciente. El script se encuentra en **scripts/script-view-dosis-paciente**.

```
1 • USE procedimiento_medico;
2 • CREATE VIEW dosis_paciente AS
3   SELECT medicamento.nombre AS medicamento, medicamento.dosis, paciente.nombre AS paciente
4   FROM ((procedimiento_medico.medicamento
5   INNER JOIN procedimiento_medico.medicamento_paciente ON medicamento.id = medicamento_paciente.medicamento_id)
6   INNER JOIN procedimiento_medico.paciente ON paciente.id = paciente_id);
7
```

	medicamento	dosis	paciente
►	Amoxicilina	20	Daniel Rodriguez
	Omeprazol	10	Camilo Tapias
	Silimarina	30	Paola Restrepo
	Acetaminofen	20	Luisa Monedero
	Paracetamol	30	Giovanny Moreno

Figura 30: Vista dosis-paciente

Tercera Actividad:

1. Se procede a generar 4 procedimientos almacenados para agregar, actualizar, consultar y borrar, en una de las tablas de la librería Busca Libre (primera actividad).
- Se inicia con un procedimiento que permite buscar un libro por su ISBN.

```

1  USE libreriaescalibre;
2
3  DELIMITER //
4  • CREATE PROCEDURE buscar_por_ISBN(in ISBN_libro VARCHAR(12))
5  ○ BEGIN
6    SELECT * FROM libro where libro.ISBN = ISBN_libro;
7  END //
8  DELIMITER ;
9
10 • CALL buscar_por_ISBN("10606345");

```

Figura 31: Sentencia para crear el procedimiento buscar_por_ISBN

El anterior script se encuentra en **scripts/script-procedure-libroisbn**

	ISBN	titulo	numero_paginas	nombre_editorial
►	10606345	El padrino	254	Altea

Figura 32: Resultado al ejecutar el procedimiento buscar_por_ISBN

- Ahora se implementa un procedimiento para agregar libros a la tabla Libro. El script se encuentra en **scripts/script-procedure-agregar-libro**

```

1 • USE libreriaescalibre;
2
3 DELIMITER //
4 • create procedure agregar_libro(in ISBN_libro VARCHAR(12),
5     titulo_libro VARCHAR(45),
6     numero_paginas_libro VARCHAR(45),
7     editorial VARCHAR(45))
8 • BEGIN
9     INSERT INTO libro VALUES (ISBN_libro, titulo_libro, numero_paginas_libro, editorial);
10 END //
11 DELIMITER ;
12
13 • CALL agregar_libro("14403325", "Los tres Mosqueteros", "300", "Planeta");

```

Figura 33: Sentencia para crear procedimiento agregar_libro

Al utilizar el procedimiento almacenado buscar_por_ISBN, se obtiene la información del libro agregado al utilizar el procedimiento agregar_libro. El resultado se presenta en la figura 34.

	ISBN	titulo	numero_paginas	nombre_editorial
►	14403325	Los tres Mosqueteros	300	Planeta

Figura 34: Libro agregado usando procedimiento agregar_libro

- Se implementa un procedimiento almacenado para actualizar información de un libro en la tabla Libro. El script se encuentra en **scripts/script-procedure-actualizar-libro**.


```

1 • USE libreriabuscalibre;
2
3 DELIMITER //
4 • CREATE PROCEDURE actualizar_libro(in ISBN_libro VARCHAR(12),
5     titulo_libro VARCHAR(45),
6     numero_paginas_libro VARCHAR(45),
7     editorial VARCHAR(45))
8 BEGIN
9     UPDATE libro
10    SET titulo = titulo_libro, numero_paginas = numero_paginas_libro, nombre_editorial = editorial
11    WHERE ISBN = ISBN_libro;
12 END //
13 DELIMITER ;

```

Figura 35: Script para crear procedimiento actualizar_libro

El procedimiento actualizar_libro se ejecuta para modificar el número de páginas del libro recientemente agregado, Los tres Mosqueteros, pasando de tener 300 a 350 páginas (comparar figura 35 y figura 36)

Call stored procedure libreriabuscalibre.actualizar_libro

Enter values for parameters of your procedure and click <Execute> to create an SQL editor and run the call:

ISBN_libro: 14403325 [IN] VARCHAR(12)

titulo_libro: s Tres Mosqueteros [IN] VARCHAR(45)

numero_paginas_libro: 350 [IN] VARCHAR(45)

editorial: Planeta [IN] VARCHAR(45)

Execute Cancel

Figura 36: Actualización libro Los tres Mosqueteros

	ISBN	titulo	numero_paginas	nombre_editorial
►	14403325	Los tres Mosqueteros	350	Planeta

Figura 37: Resultado al ejecutar el procedimiento actualizar_libro

- Por último, se crea un procedimiento almacenado para eliminar un libro de la tabla Libro. El script se encuentra en **scripts/script-procedure-borrar-libro**.

```
1 • USE libreriabuscalibre;  
2  
3 DELIMITER //  
4  
5 • CREATE PROCEDURE eliminar_libro(in ISBN_libro VARCHAR(12))  
6 BEGIN  
7 DELETE FROM libro WHERE ISBN = ISBN_libro;  
8 END //  
9 DELIMITER ;
```

Figura 37: Script para crear procedimiento eliminar_libro

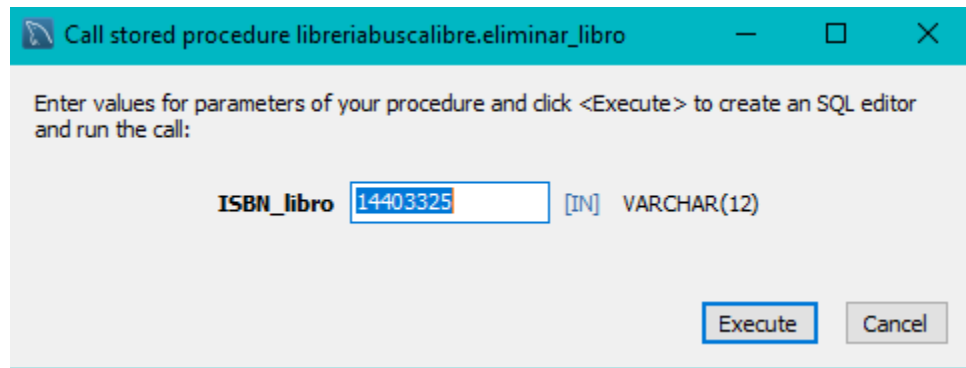


Figura 38: Eliminación libro Los tres Mosqueteros

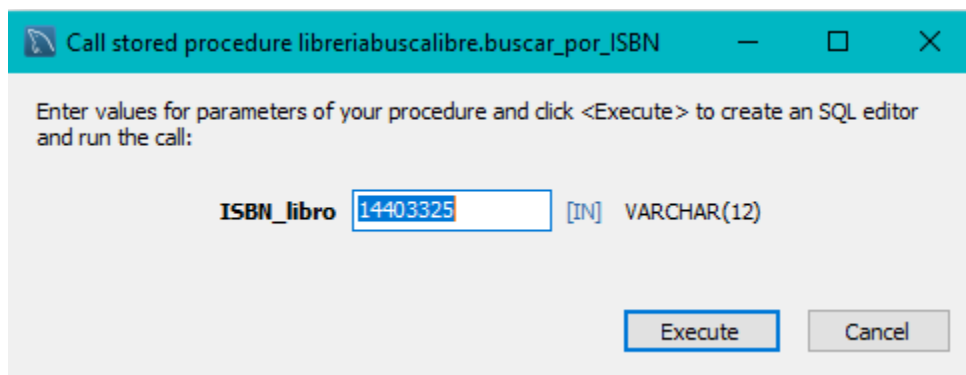


Figura 39: Búsqueda libro Los tres Mosqueros con su ISBN

Al tratar de buscar el libro Los tres Mosqueteros, ingresando su ISBN, el resultado es una tabla vacía, ya que el libro fue eliminado exitosamente al ejecutar el procedimiento eliminar_libro.

2. Ahora se crea la tabla control_cambios_libreria, la cual contiene 3 atributos; usuario, acción y fecha.

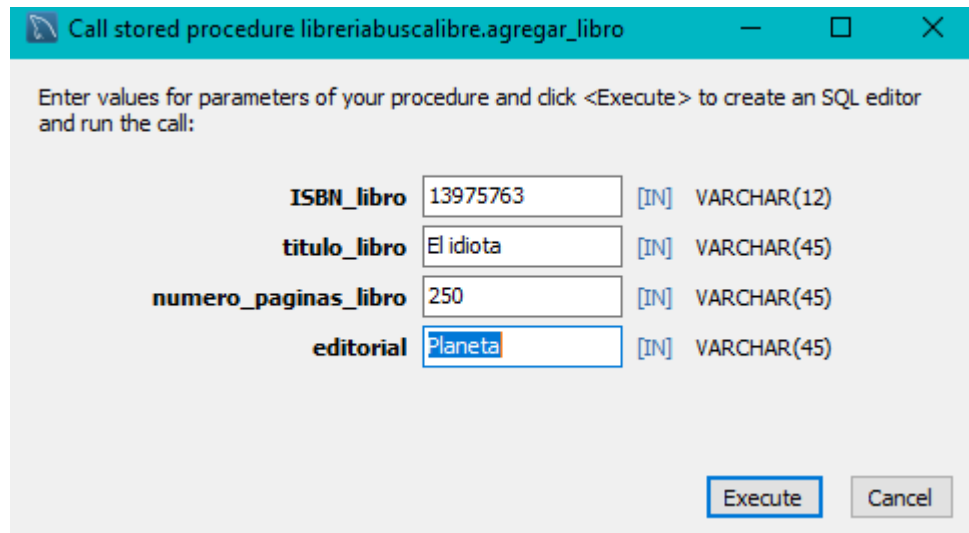
Con la siguiente sentencia se crea la tabla con los atributos solicitados:

```
1 • USE libreriaescalibre;
2
3 • CREATE TABLE IF NOT EXISTS control_cambios_libreria (
4     usuario VARCHAR(45) NOT NULL,
5     accion VARCHAR(45) NOT NULL,
6     fecha DATETIME DEFAULT CURRENT_TIMESTAMP
7 );
```

Se crea un trigger llamado insert_libro, el cual se activa cuando un usuario agrega un registro en la tabla Libro, y agrega un registro en la tabla control_cambios_hospital.

```
1 • USE libreriaescalibre;
2
3     DELIMITER //
4 • CREATE TRIGGER insert_libro AFTER INSERT ON libro
5     FOR EACH ROW
6     BEGIN
7         INSERT INTO control_cambios_libreria
8         VALUES (user(), "Agregó libro", now());
9     END //
10    DELIMITER ;
```

Se agrega un registro a la tabla Libro, usando el procedimiento almacenado agregar_libro (figura 40), luego se verifica el funcionamiento del trigger con una sentencia SELECT. El resultado se presenta en la figura 40.



Call stored procedure libreria.uscalibre.agregar_libro

Enter values for parameters of your procedure and click <Execute> to create an SQL editor and run the call:

ISBN_libro	13975763	[IN]	VARCHAR(12)
titulo_libro	El idiota	[IN]	VARCHAR(45)
numero_paginas_libro	250	[IN]	VARCHAR(45)
editorial	Planeta	[IN]	VARCHAR(45)

Execute Cancel

Figura 40: Ejecución Procedimiento almacenado agregar_libro

	usuario	accion	fecha
▶	root@localhost	Agregó libro	2023-02-14 21:32:42

Figura 41: Registro Tabla control_cambios_libreria

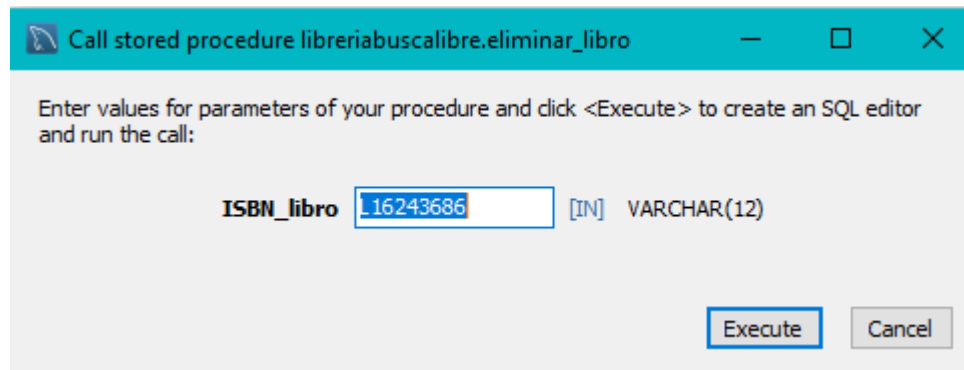
Se crea un segundo trigger llamado delete_libro, el cual se activa cuando un usuario elimina un registro de la tabla Libro, y crea un resgistro en la tabla control_cambios_hospital.

```

1 • USE libreriabuscalibre;
2
3 DELIMITER //
4 • CREATE TRIGGER delete_libro AFTER DELETE ON libro
5   FOR EACH ROW
6   BEGIN
7       INSERT INTO control_cambios_libreria
8       VALUES (user(), "Borró libro", now());
9   END //
10  DELIMITER ;

```

Para probar el funcionamiento del trigger, se utiliza el procedimiento almacenado eliminar_libro (figura 42). Posteriormente se realiza una consulta SELECT de la tabla control_cambio_libreria. El resultado se presenta en la figura 43.



Call stored procedure libreriabuscalibre.eliminar_libro

Enter values for parameters of your procedure and click <Execute> to create an SQL editor and run the call:

ISBN_libro [IN] VARCHAR(12)

Execute Cancel

Figura 42: Ejecución Procedimiento eliminar_libro

	usuario	accion	fecha
▶	root@localhost	Agregó libro	2023-02-14 21:32:42
	root@localhost	Borró libro	2023-02-14 21:37:50

Figura 43: Registro tabla control_cambios_libreria

3. Ahora se crean procedimientos para agregar, consultar, actualizar y eliminar registros en la tabla Medicamento.
 - Primero se crea el procedimiento agregar_medicamento. Se encuentra en el archivo **scripts/script-buscar-medicamento**.

```

1 • USE procedimiento_medico;
2
3 DELIMITER //
4 • CREATE PROCEDURE buscar_medicamento(in id_medicamento VARCHAR(10))
5 BEGIN
6     SELECT * FROM medicamento WHERE medicamento.id = id_medicamento;
7 END //
8 DELIMITER ;

```

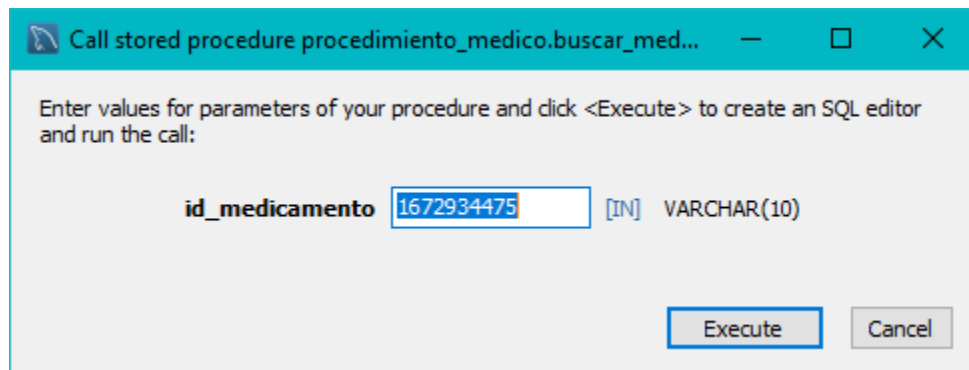


Figura 44: Ejecución procedimiento buscar_medicamento

	id	nombre	dosis
▶	1672934475	Amoxicilina	20

Figura 45: Resultado al ejecutar procedimiento buscar_medicamento

Se implementa un procedimiento llamado agregar_medicamento, para agregar registros en la tabla Medicamento. El script se encuentra en **scripts/script-agregar-medicamento**. Se llama al procedimiento buscar_medicamento y el resultado se presenta en la figura 46.

Call stored procedure procedimiento_medico.agregar_me... — □ ×

Enter values for parameters of your procedure and click <Execute> to create an SQL editor and run the call:

id_medicamento [IN] varchar(10)

nombre_medicamento [IN] varchar(45)

dosis_medicamento [IN] varchar(45)

	id	nombre	dosis
▶	24240798	Naproxeno	20

Figura 46: Resultado al ejecutar procedimiento agregar_medicamento

- Se implementa el procedimiento almacenado actualizar_medicamento. Se puede encontrar en **scripts/script-actualizar-medicamento**.

```

1 • USE procedimiento_medico;
2
3 DELIMITER //
4
5 • CREATE PROCEDURE actualizar_medicamento(in id_medicamento varchar(10),
6     nombre_medicamento varchar(45),
7     dosis_medicamento varchar(45))
8 BEGIN
9     UPDATE medicamento
10    SET nombre = nombre_medicamento, dosis = dosis_medicamento
11    WHERE id = id_medicamento;
12 END //
13 DELIMITER ;

```

Call stored procedure procedimiento_medico.agregar_me...

Enter values for parameters of your procedure and click <Execute> to create an SQL editor and run the call:

id_medicamento 24240798 [IN] varchar(10)

nombre_medicamento Naproxeno [IN] varchar(45)

dosis_medicamento 30 [IN] varchar(45)

Execute Cancel

Luego de ejecutar el procedimiento actualizar_medicamento, se observa en la figura 47 y comparándola con la figura 46, que la dosis de Naproxeno fue actualizada de 20 a 30.

	id	nombre	dosis
▶	24240798	Naproxeno	30

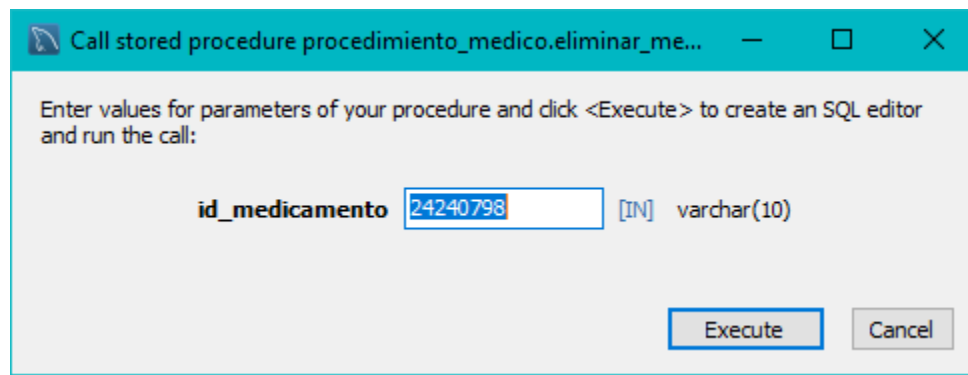
Figura 47: Resultado luego de ejecutar procedimiento actualizar_medicamento

- Por último, se agrega el procedimiento eliminar_medicamento, para borrar un registro de la tabla medicamento. El script se encuentra en **scripts/script-eliminar-medicamento**.

```

1 • USE procedimiento_medico;
2
3 DELIMITER //
4 • CREATE PROCEDURE eliminar_medicamento(in id_medicamento varchar(10))
5   BEGIN
6     DELETE FROM medicamento WHERE id = id_medicamento;
7   END //
8 DELIMITER ;

```

Luego de ejecutar el procedimiento eliminar_medicamento, se ejecuta el procedimiento buscar_medicamento, y el resultado es una tabla vacía (figura 48).

	id	nombre	dosis

Figura 48: Resultado luego de ejecutar procedimiento eliminar_medicamento

- Ahora se crea la tabla control_cambios_hospital, con los atributos usuario, acción y fecha.

```

1 • ○ CREATE TABLE IF NOT EXISTS control_cambios_hospital (
2     usuario varchar(45) not null,
3     accion varchar(45) not null,
4     fecha datetime default current_timestamp
5 );

```

Se crea el trigger inserta_medicamento, el cual agrega un registro en la tabla control_cambios_hospital, cada que se inserta un registro en la tabla Medicamento

```

1 • USE procedimiento_medico;
2
3 DELIMITER //
4 • CREATE TRIGGER inserta_medicamento AFTER INSERT ON medicamento
5   FOR EACH ROW
6   BEGIN
7       INSERT INTO control_cambios_hospital
8       VALUES (user(), "Agregó medicamento", now());
9   END //
10  DELIMITER ;

```

Luego se agrega un registro a la tabla Medicamento, usando el procedimiento almacenado agregar_medicamento. Posteriormente, se realiza una consulta SELECT en la tabla control_cambios_hospital para verificar el funcionamiento del trigger implementado. El resultado se presenta en la figura 49.

Call stored procedure procedimiento_medico.agregar_me...

Enter values for parameters of your procedure and click <Execute> to create an SQL editor and run the call:

id_medicamento	<input type="text" value="8809086"/>	[IN]	varchar(10)
nombre_medicamento	<input type="text" value="Ibuprofeno"/>	[IN]	varchar(45)
dosis_medicamento	<input type="text" value="10"/>	[IN]	varchar(45)

	usuario	accion	fecha
▶	root@localhost	Agregó medicamento	2023-02-14 22:33:44

Figura 49: Resultado luego de ejecutar procedimiento agregar_medicamento

- Por último, se crea el trigger elimina_medimento, el cual agrega un registro a la tabla control_cambios_hospital cada que se elimina un registro de la tabla Medicamentos.

```
1 • USE procedimiento_medico;
2
3 DELIMITER //
4 • CREATE TRIGGER elimina_medimento AFTER DELETE ON medicamento
5   FOR EACH ROW
6   BEGIN
7       INSERT INTO control_cambios_hospital
8       VALUES (user(), "Eliminó medicamento", now());
9   END //
10  DELIMITER ;
```

Para verificar el funcionamiento del trigger elimina_medimento, se elimina un registro de la tabla Medicamento, usando el procedimiento eliminar_medimento. Luego se ejecuta una sentencia SELECT a la tabla control_cambios_hospital para evaluar si se registró correctamente la acción. El resultado se presenta en la figura 50.

Call stored procedure procedimiento_medico.eliminar_me... — □ ×

Enter values for parameters of your procedure and click <Execute> to create an SQL editor and run the call:

id_medimento [IN] varchar(10)

Execute Cancel

	usuario	accion	fecha
▶	root@localhost	Agregó medicamento	2023-02-14 22:33:44
	root@localhost	Eliminó medicamento	2023-02-14 22:45:12

Figura 50: Resultado luego de ejecutar el procedimiento eliminar_medimento

5. Para obtener mayor información del modelo planteado del hospital; agregaría en las entidades medicamento y procedimiento el costo unitario de cada procedimiento y medicamento recetado, para posteriormente obtener el costo total en la entidad factura.