

BASES DE DATOS CONSULTAS  
@ING\_JIMMY ALEXANDER LOMBANA RIVERA

Una vez terminada la creación de la bases de datos comercial vamos a realizar unas series de consultas en todas las tablas que ingresaron los datos.

SELECT \* FROM Oficinas WHERE Departamento LIKE 'Va%';

Esta consulta utiliza la cláusula `LIKE` junto con el patrón `'Va%'` para buscar todos los departamentos que comienzan con las letras "Va". El símbolo `'%'` representa cualquier número de caracteres que pueda seguir después de las letras "Va".

```
Database changed
mysql> SELECT * FROM Oficinas WHERE Departamento LIKE 'Va%';
+-----+-----+-----+-----+-----+
| Id_oficina | Ciudad | Telefono | Direccion | Departamento |
| Pais      | Creado |          |           |              |
+-----+-----+-----+-----+-----+
|          3 | Cali   | 456789123 | Avenida 5 Norte # 23-45 | Valle del Cauca |
| Colombia | 2023-06-15 09:14:17 |           |           |              |
+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)

mysql>
```

**Actividad:** Van hacer esa consultas en diferentes letras que empiecen en todas las tablas que realizaron en la base de Datos anterior.

Mostrar los Id cuyo código está entre el 2 y el 9:

SELECT Id oficina FROM Oficinas WHERE Id oficina BETWEEN 2 AND 3;

```
mysql> SELECT Id_oficina FROM Oficinas WHERE Id_oficina BETWEEN 2 AND 3;
+-----+
| Id_oficina |
+-----+
|          2 |
|          3 |
+-----+
2 rows in set (0.00 sec)
```

Esta consulta utiliza la cláusula `WHERE` con el operador `BETWEEN` para filtrar los registros en los que el campo "Id oficina" está entre 2 y 3. El resultado será una lista de los Id que cumplen con esa condición.

Pero podemos traer más datos con esa consulta de `BETWEEN`

SELECT Id oficina, Departamento FROM Oficinas WHERE Id oficina BETWEEN 2 AND 3;

```
mysql>
mysql> SELECT Id_oficina, Departamento
-> FROM Oficinas
[ -> WHERE Id_oficina BETWEEN 2 AND 9;
+-----+-----+
| Id_oficina | Departamento |
+-----+-----+
|          2 | Antioquia    |
|          3 | Valle del Cauca |
|          4 | Atlántico    |
|          5 | Bolívar      |
|          6 | Risaralda    |
|          7 | Santander    |
|          8 | Norte de Santander |
|          9 | Caldas       |
+-----+-----+
8 rows in set (0.00 sec)
```

**Actividad:** Van realizar diferentes consultas con esa sintaxis en las diferentes tablas que realizaron en la base de datos.

## ACTUALIZAR Y ELIMINAR REGISTROS

Organicemos la base en la que vamos a trabajar.

### Base de datos de demostración:

Por favor crear una base con las siguientes características para que realicen cada uno de los pasos y casos de esta guía y además van a hacer con las demás bases de datos que hemos realizado durante las clases de formación, cargar el archivo .sql y el Word con los pantallazos al repo como actu.elim\_sql en un archivo zip.

1. **CREATE DATABASE** CADPH;
2. **USE** CADPH;
3. **CREATE TABLE** Titulada(

id INT (20) UNIQUE PRIMARY KEY,  
nombre\_apellido VARCHAR (50) UNIQUE NOT NULL,

## BASES DE DATOS CONSULTAS

@ING\_JIMMY ALEXANDER LOMBANA RIVERA

```
correo VARCHAR(50) UNIQUE NOT NULL,  
edad INT UNSIGNED NOT NULL,  
direccion VARCHAR(20) NOT NULL,  
ciudad VARCHAR(20) NOT NULL,  
estado ENUM('Activo', 'Inactivo') DEFAULT 'Inactivo',  
formación ENUM('Técnico', 'Tecnólogo') DEFAULT  
'Tecnólogo',  
creado TIMESTAMP DEFAULT CURRENT_TIMESTAMP  
);
```

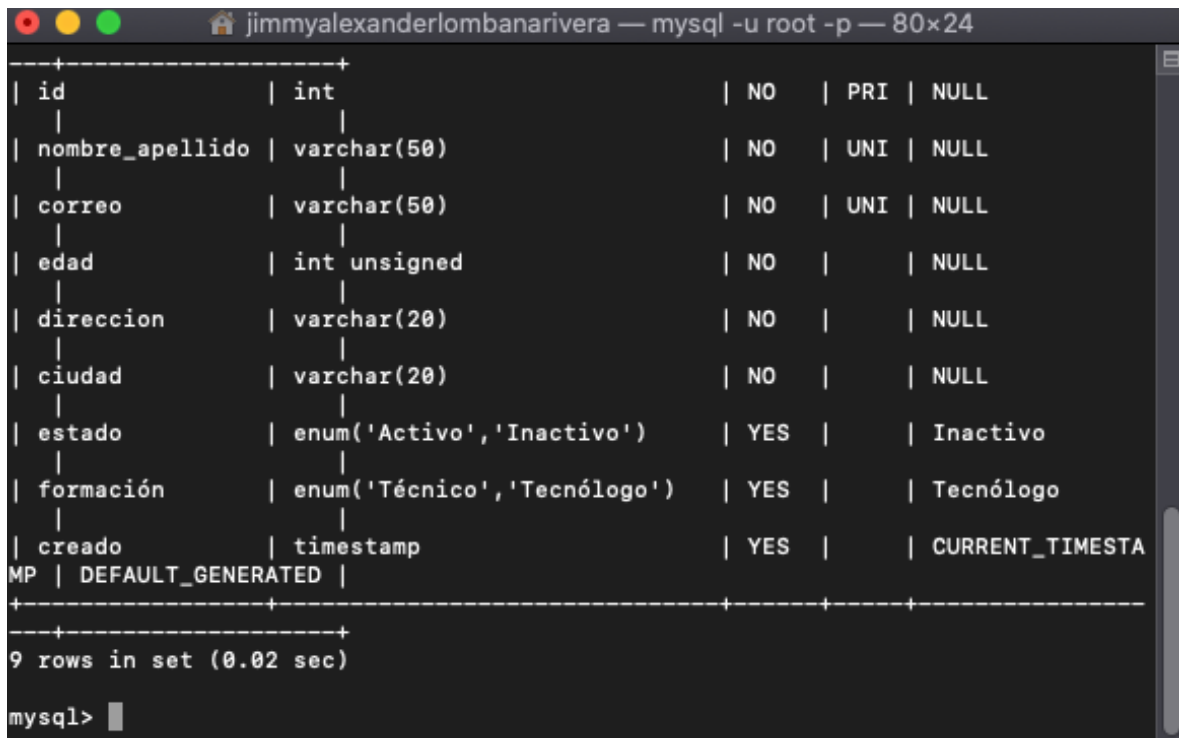
4. Realizar inserción de 20 registros.

```
INSERT INTO Titulada (id, nombre_apellido, correo, edad,  
direccion, ciudad, estado, formación)
```

```
VALUES ...
```

Debe quedar de la siguiente manera:

Tabla de Titulada:



The screenshot shows a MySQL terminal window with the title 'jimmyalexanderlombanarivera — mysql -u root -p — 80x24'. The terminal displays the table structure for 'Titulada' using the command 'DESCRIBE Titulada;'. The output is a table with 10 columns: id, nombre\_apellido, correo, edad, direccion, ciudad, estado, formación, creado, and DEFAULT\_GENERATED. The 'creado' column is of type 'timestamp' and has a default value of 'CURRENT\_TIMESTAMP'. The 'DEFAULT\_GENERATED' column is of type 'timestamp' and has a default value of 'CURRENT\_TIMESTAMP'. The terminal also shows '9 rows in set (0.02 sec)' and the prompt 'mysql>'.

Field	Type	Null	Key	Extra
id	int	NO	PRI	
nombre_apellido	varchar(50)	NO	UNI	
correo	varchar(50)	NO	UNI	
edad	int unsigned	NO		
direccion	varchar(20)	NO		
ciudad	varchar(20)	NO		
estado	enum('Activo','Inactivo')	YES		Inactivo
formación	enum('Técnico','Tecnólogo')	YES		Tecnólogo
creado	timestamp	YES		CURRENT_TIMESTAMP
DEFAULT_GENERATED	timestamp	YES		CURRENT_TIMESTAMP

9 rows in set (0.02 sec)

mysql>

## BASE DE DATOS

@ING\_JIMMY ALEXANDER LOMBANA RIVERA

BASES DE DATOS CONSULTAS  
@ING\_JIMMY ALEXANDER LOMBANA RIVERA

Datos Ingresado en la Base de Datos:

```
mysql> SELECT * FROM titulada;
```

id	nombre_apellido	correo	edad	direccion	ciudad	estado	formación	creado
1	Juan Pérez	juan@hotmail.com	25	Calle 123	Ciudad A	Activo	Técnico	2023-06-20 19:33:39
2	María López	maria@gmail.com	30	Avenida 456	Ciudad B	Activo	Tecnólogo	2023-06-20 19:33:39
3	Carlos Ramírez	carlos@yahoo.es	28	Carrera 789	Ciudad A	Inactivo	Tecnólogo	2023-06-20 19:33:39
4	Ana Gómez	ana@sena.edu.co	23	Calle 789	Ciudad C	Activo	Tecnólogo	2023-06-20 19:33:39
5	Luisa Torres	luisa@unadvirtual.edu.co	29	Avenida 123	Ciudad B	Activo	Técnico	2023-06-20 19:33:39
6	Pedro Sánchez	pedro@gmail.com	26	Carrera 456	Ciudad A	Activo	Técnico	2023-06-20 19:33:39
7	Laura Martínez	laura@hotmail.com	27	Calle 456	Ciudad C	Inactivo	Tecnólogo	2023-06-20 19:33:39
8	Mario Castro	mario@gmail.com	31	Avenida 789	Ciudad B	Activo	Tecnólogo	2023-06-20 19:33:39
9	Isabel Herrera	isabel@sena.edu.co	24	Carrera 123	Ciudad A	Activo	Técnico	2023-06-20 19:33:39
10	Jorge Fernández	jorge@hotmail.com	32	Calle 789	Ciudad C	Activo	Técnico	2023-06-20 19:33:39

10 rows in set (0.00 sec)

## UPDATE

La palabra reservada Update nos permite realizar una actualización a las rows de nuestra tabla, aquí la sintaxis.

**UPDATE** <Nombre\_de\_la\_tabla> **SET** <Columna\_a\_modificar>= 'Lo que vamos a actualizar';

Sin embargo, el UPDATE nos permite ingresar una condición o regla, dado que como está la sintaxis arriba, nos actualizaría todos los registros contenidos para esa columna, siendo que, solo queremos actualizar un ID en específico para esa columna.

Entonces, esto quedaría así:

**UPDATE** <Nombre\_de\_la\_tabla> **SET** <Columna\_a\_modificar>= 'Lo que vamos a actualizar' **WHERE** <identificador de registro> = <valor>;

Si queremos actualizar el nombre del Id 1 realizamos la siguiente sintaxis

```
UPDATE Titulada
SET nombre_apellido = 'KAMELIN PANTEVIS'
WHERE id = 2;
```

Antes de la Actualización:

```
mysql> SELECT * FROM titulada;
```

id	nombre_apellido	correo	edad	direccion	ciudad	estado	formación	creado
1	Juan Pérez	juan@hotmail.com	25	Calle 123	Ciudad A	Activo	Técnico	2023-06-20 19:33:39
2	María López	maria@gmail.com	30	Avenida 456	Ciudad B	Activo	Tecnólogo	2023-06-20 19:33:39
3	Carlos Ramírez	carlos@yahoo.es	28	Carrera 789	Ciudad A	Inactivo	Tecnólogo	2023-06-20 19:33:39
4	Ana Gómez	ana@sena.edu.co	23	Calle 789	Ciudad C	Activo	Tecnólogo	2023-06-20 19:33:39
5	Luisa Torres	luisa@unadvirtual.edu.co	29	Avenida 123	Ciudad B	Activo	Técnico	2023-06-20 19:33:39
6	Pedro Sánchez	pedro@gmail.com	26	Carrera 456	Ciudad A	Activo	Técnico	2023-06-20 19:33:39
7	Laura Martínez	laura@hotmail.com	27	Calle 456	Ciudad C	Inactivo	Tecnólogo	2023-06-20 19:33:39
8	Mario Castro	mario@gmail.com	31	Avenida 789	Ciudad B	Activo	Tecnólogo	2023-06-20 19:33:39
9	Isabel Herrera	isabel@sena.edu.co	24	Carrera 123	Ciudad A	Activo	Técnico	2023-06-20 19:33:39
10	Jorge Fernández	jorge@hotmail.com	32	Calle 789	Ciudad C	Activo	Técnico	2023-06-20 19:33:39

10 rows in set (0.00 sec)

BASES DE DATOS CONSULTAS  
@ING\_JIMMY ALEXANDER LOMBANA RIVERA

Después de realizar la actualización:

```
mysql> SELECT * FROM titulada;
```

id	nombre_apellido	correo	edad	direccion	ciudad	estado	formación	creado
1	Jimmy Alexander	jalmpa77@gmail.com	25	Calle 123	Ciudad A	Activo	Técnico	2023-06-20 19:33:39
2	KAMELIN PANTEVIS	maria@gmail.com	30	Avenida 456	Ciudad B	Activo	Tecnólogo	2023-06-20 19:33:39
3	Carlos Ramírez	carlos@yahoo.es	28	Carrera 789	Ciudad A	Inactivo	Tecnólogo	2023-06-20 19:33:39
4	Ana Gómez	ana@sena.edu.co	23	Calle 789	Ciudad C	Activo	Tecnólogo	2023-06-20 19:33:39
5	Luisa Torres	luisa@unadvirtual.edu.co	29	Avenida 123	Ciudad B	Activo	Técnico	2023-06-20 19:33:39
6	Pedro Sánchez	pedro@gmail.com	26	Carrera 456	Ciudad A	Activo	Técnico	2023-06-20 19:33:39
7	Laura Martínez	laura@hotmail.com	27	Calle 456	Ciudad C	Inactivo	Tecnólogo	2023-06-20 19:33:39
8	Mario Castro	mario@gmail.com	31	Avenida 789	Ciudad B	Activo	Tecnólogo	2023-06-20 19:33:39
9	Isabel Herrera	isabel@sena.edu.co	24	Carrera 123	Ciudad A	Activo	Técnico	2023-06-20 19:33:39
10	Jorge Fernández	jorge@hotmail.com	32	Calle 789	Ciudad C	Activo	Técnico	2023-06-20 19:33:39

También podemos actualizar diferentes campos de la tabla a la vez:

UPDATE Titulada

SET nombre\_apellido = 'Juan Diego ', correo = 'jilombana@sena.edu.co', edad = '78'

WHERE id = 10;

```
mysql> SELECT * FROM titulada;
```

id	nombre_apellido	correo	edad	direccion	ciudad	estado	formación	creado
1	Jimmy Alexander	jalmpa77@gmail.com	25	Calle 123	Ciudad A	Activo	Técnico	2023-06-20 19:33:39
2	KAMELIN PANTEVIS	maria@gmail.com	30	Avenida 456	Ciudad B	Activo	Tecnólogo	2023-06-20 19:33:39
3	Carlos Ramírez	carlos@yahoo.es	28	Carrera 789	Ciudad A	Inactivo	Tecnólogo	2023-06-20 19:33:39
4	Ana Gómez	ana@sena.edu.co	23	Calle 789	Ciudad C	Activo	Tecnólogo	2023-06-20 19:33:39
5	Luisa Torres	luisa@unadvirtual.edu.co	29	Avenida 123	Ciudad B	Activo	Técnico	2023-06-20 19:33:39
6	Pedro Sánchez	pedro@gmail.com	26	Carrera 456	Ciudad A	Activo	Técnico	2023-06-20 19:33:39
7	Laura Martínez	laura@hotmail.com	27	Calle 456	Ciudad C	Inactivo	Tecnólogo	2023-06-20 19:33:39
8	Mario Castro	mario@gmail.com	31	Avenida 789	Ciudad B	Activo	Tecnólogo	2023-06-20 19:33:39
9	Isabel Herrera	isabel@sena.edu.co	24	Carrera 123	Ciudad A	Activo	Técnico	2023-06-20 19:33:39
10	Jorge Fernández	jorge@hotmail.com	32	Calle 789	Ciudad C	Activo	Técnico	2023-06-20 19:33:39

Ahora:

```
mysql> SELECT * FROM titulada;
```

id	nombre_apellido	correo	edad	direccion	ciudad	estado	formación	creado
1	Jimmy Alexander	jalmpa77@gmail.com	25	Calle 123	Ciudad A	Activo	Técnico	2023-06-20 19:33:39
2	KAMELIN PANTEVIS	maria@gmail.com	30	Avenida 456	Ciudad B	Activo	Tecnólogo	2023-06-20 19:33:39
3	Carlos Ramírez	carlos@yahoo.es	28	Carrera 789	Ciudad A	Inactivo	Tecnólogo	2023-06-20 19:33:39
4	Ana Gómez	ana@sena.edu.co	23	Calle 789	Ciudad C	Activo	Tecnólogo	2023-06-20 19:33:39
5	Luisa Torres	luisa@unadvirtual.edu.co	29	Avenida 123	Ciudad B	Activo	Técnico	2023-06-20 19:33:39
6	Pedro Sánchez	pedro@gmail.com	26	Carrera 456	Ciudad A	Activo	Técnico	2023-06-20 19:33:39
7	Laura Martínez	laura@hotmail.com	27	Calle 456	Ciudad C	Inactivo	Tecnólogo	2023-06-20 19:33:39
8	Mario Castro	mario@gmail.com	31	Avenida 789	Ciudad B	Activo	Tecnólogo	2023-06-20 19:33:39
9	Isabel Herrera	isabel@sena.edu.co	24	Carrera 123	Ciudad A	Activo	Técnico	2023-06-20 19:33:39
10	Juan Diego	jilombana@sena.edu.co	78	Calle 789	Ciudad C	Activo	Técnico	2023-06-20 19:33:39

10 rows in set (0.00 sec)

## Ejercicio 1

Actualizar las 8 columnas, en por lo menos 3 valor, para los 20 registros

## DELETE

La palabra reservada Delete, nos permite eliminar las rows o registros de nuestra tabla, aquí la sintaxis.

**DELETE FROM** <Nombre de la tabla>;

Sin embargo, el DELETE nos permite ingresar una condición o regla, dado que como está la sintaxis arriba, nos eliminaría todos los registros contenidos para esa tabla, siendo que, solo queremos un registro en específico. Entonces, esto quedaría así:

**DELETE FROM** <Nombre de la tabla> **WHERE** <identificador de registro> = <valor>;

Si quisiera eliminar el registro con **ID 10** Quedaría así:

**DELETE FROM** titulada **WHERE** id = 10;

Como Podemos Observar no elimino todo el registro del Id 10

id	nombre_apellido	correo	edad	direccion	ciudad	estado	formación	creado
1	Jimmy Alexander	jalmpa77@gmail.com	25	Calle 123	Ciudad A	Activo	Técnico	2023-06-20 19:33:39
2	KAMELIN PANTEVIS	maria@gmail.com	30	Avenida 456	Ciudad B	Activo	Tecnólogo	2023-06-20 19:33:39
3	Carlos Ramírez	carlos@yahoo.es	28	Carrera 789	Ciudad A	Inactivo	Tecnólogo	2023-06-20 19:33:39
4	Ana Gómez	ana@sena.edu.co	23	Calle 789	Ciudad C	Activo	Tecnólogo	2023-06-20 19:33:39
5	Luisa Torres	luisa@unadvirtual.edu.co	29	Avenida 123	Ciudad B	Activo	Técnico	2023-06-20 19:33:39
6	Pedro Sánchez	pedro@gmail.com	26	Carrera 456	Ciudad A	Activo	Técnico	2023-06-20 19:33:39
7	Laura Martínez	laura@hotmail.com	27	Calle 456	Ciudad C	Inactivo	Tecnólogo	2023-06-20 19:33:39
8	Mario Castro	mario@gmail.com	31	Avenida 789	Ciudad B	Activo	Tecnólogo	2023-06-20 19:33:39
9	Isabel Herrera	isabel@sena.edu.co	24	Carrera 123	Ciudad A	Activo	Técnico	2023-06-20 19:33:39

Van a realizar la eliminación de 5 registros de la tabla de Bases de Datos en la actual y demás bases que se han realizado durante las actividades anteriores.

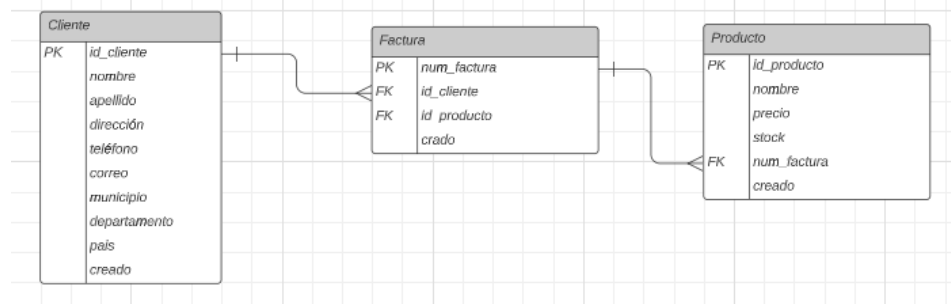
## RELACIONES

Cuando se está realizando el maquetado de nuestras tablas, utilizamos principalmente 3 tipos de relaciones:

- Uno a uno.
- Uno a muchos.
- Muchos a muchos.

Crearemos la siguiente base con sus correspondientes tablas, la relación más normal de encontrar o utilizar es de **uno a muchos**, para esto aremos uso de llaves foráneas, que no es más que una columna que almacena una llave primaria de otra tabla o tabla externa.

### MODELO ENTIDAD RELACIÓN – EJERCICIO EN CLASE FACTURACION



Un cliente → Puede tener muchas facturas.

Una factura → Puede tener muchos productos.

Es decir que para este ejercicio aremos uso de la relación uno a muchos.

### **Ejemplo de uno a uno**

Un ejemplo de relación uno a uno sería una tabla de "Usuarios" y una tabla de "Perfiles", donde un usuario tiene un perfil asociado y un perfil pertenece a un usuario específico.

### **Ejemplo de Muchos a Muchos**

Un ejemplo de relación muchos a muchos podrían ser una base de datos para una tienda en línea, donde tienes una tabla de "Productos" y una tabla de "Categorías". Cada producto puede pertenecer a múltiples categorías, y cada categoría puede tener múltiples productos asociados. Para representar esta relación, se necesita una tabla intermedia llamada "Producto Categoría" que vincule los productos y las categorías.

BASES DE DATOS CONSULTAS  
@ING\_JIMMY ALEXANDER LOMBANA RIVERA

```
CREATE DATABASE FACTURACIÓN;
USE FACTURACIÓN;

CREATE TABLE cliente(
    id_cliente VARCHAR (30) UNIQUE PRIMARY KEY,
    nombre VARCHAR (25) UNIQUE NOT NULL,
    apellido VARCHAR (25) UNIQUE NOT NULL,
    direccion VARCHAR(20) NOT NULL,
    telefono VARCHAR(20) NOT NULL,
    correo VARCHAR(50) UNIQUE NOT NULL,
    municipio VARCHAR(20),
    departamento VARCHAR(20),
    pais VARCHAR (20) CHECK (pais='Colombia'), -- CHECK permite limitar a una respuesta
    creado TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

CREATE TABLE factura(
    num_factura VARCHAR (20) UNIQUE PRIMARY KEY,
    id_cliente VARCHAR (30),
    id_producto VARCHAR (30),
    creado TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (id_cliente) REFERENCES cliente(id_cliente),
    FOREIGN KEY (id_producto) REFERENCES productos(id_producto)
);

CREATE TABLE productos(
    id_producto VARCHAR (30) UNIQUE PRIMARY KEY,
    nombre VARCHAR (25) NOT NULL,
    precio INT (25) NOT NULL,
    stock INT (25) NOT NULL,
    num_factura VARCHAR (20),
    creado TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (num_factura) REFERENCES factura(num_factura)
);
```

Cuando comenzamos a crear la Tabla factura no está generando un error en la última llave foránea.

```
mysql> USE FACTURACIÓN;
Database changed
mysql> CREATE TABLE cliente(
-> id_cliente VARCHAR (30) UNIQUE PRIMARY KEY,
-> nombre VARCHAR (25) UNIQUE NOT NULL,
-> apellido VARCHAR (25) UNIQUE NOT NULL,
-> direccion VARCHAR(20) NOT NULL,
-> telefono VARCHAR(20) NOT NULL,
-> correo VARCHAR(50) UNIQUE NOT NULL,
-> municipio VARCHAR(20),
-> departamento VARCHAR(20),
-> pais VARCHAR (20) CHECK (pais='Colombia'), -- CHECK permite limitar a una respuesta
-> creado TIMESTAMP DEFAULT CURRENT_TIMESTAMP
-> );
Query OK, 0 rows affected (0.04 sec)

mysql> CREATE TABLE factura(
-> num_factura VARCHAR (20) UNIQUE PRIMARY KEY,
-> id_cliente VARCHAR (30),
-> id_producto VARCHAR (30),
-> creado TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
-> FOREIGN KEY (id_cliente) REFERENCES cliente(id_cliente),
-> );
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near ')' at line 7
```



Básicamente es porque no podemos declarar una llave foránea que toma como referencia una tabla que no hemos creado, como es el caso de productos, y si crean primero productos pues tampoco lo podrán hacer dado que la tabla factura no estaría creada.

Ejercicio: Entonces investigue como asignar esta clave foránea faltante, haciendo uso de ALTER TABLE (Mediante la instrucción ALTER TABLE se puede modificar una tabla existente de varias formas) o de otro método para esta alteración o modificación.

Una vez solucionado el error de nuestras tablas procedemos a realizar este Ejercicio

**Posterior inserte 10 productos, 5 clientes y 10 facturas.**

## **Funciones de agregación**

Las funciones de agregación en SQL nos permiten efectuar operaciones sobre un conjunto de resultados, pero devolviendo un único valor agregado para todos ellos. Es decir, nos permiten obtener medias, máximos, etc.... sobre un conjunto de valores.

Las funciones de agregación básicas que soportan todos los gestores de datos son las siguientes:

- COUNT: devuelve el número total de filas seleccionadas por la consulta.
- MIN: devuelve el valor mínimo del campo que especifiquemos.
- MAX: devuelve el valor máximo del campo que especifiquemos.
- SUM: suma los valores del campo que especifiquemos. Sólo se puede utilizar en columnas numéricas.
- AVG: devuelve el valor promedio del campo que especifiquemos. Sólo se puede utilizar en columnas numéricas.

Cabe aclarar que éstas se realizan con valores numéricos

Función SUM() → Devuelve la suma de todos los valores de una columna.  
SELECT SUM (columna) FROM nombre\_tabla;

Función AVG() → Devuelve la media o el promedio de una columna.  
SELECT AVG (columna) FROM nombre\_tabla;

Función MIN() → Devuelve el valor mínimo de una columna.  
SELECT MIN (columna) FROM nombre\_tabla;

Función MAX() → Devuelve el valor máximo de una columna.  
SELECT MAX (columna) FROM nombre\_tabla;

Función COUNT() → Devuelve el número de filas.  
SELECT COUNT(columna) FROM nombre\_tabla;

### **EJERCICIO**

Realice en base a cada una de estas funciones de agregación por lo menos una consulta en la tabla FACTURA.