



Nombre de la práctica	Bases de Datos			No.	
Asignatura:	Taller de base de datos	Carrera:	Ingeniería en Sistemas Computacionales	Duración de la práctica (Hrs)	

I. Competencia(s) específica(s):

II. Lugar de realización de la práctica (laboratorio, taller, aula u otro):

Aula III. Material empleado: Laptop, Navicat

III. Desarrollo de la

práctica

- **Problemática:** Una escuela necesita gestionar los alumnos y los cursos en los que están inscritos. Cada alumno puede inscribirse en varios cursos y cada curso puede tener varios alumnos.

```

Objects | escuela @escuela (mysql80-localhost) - ...
Save | Query Builder | Beautify SQL | Code Snippet
mysql80-localhost | escuela | Run | Stop | Explain

1 CREATE TABLE Alumno(
2     id_alumno INT PRIMARY KEY AUTO_INCREMENT,
3     nombre VARCHAR(30) NOT NULL,
4     apellido1 VARCHAR(30) NOT NULL,
5     apellido2 VARCHAR(30) NOT NULL,
6     feha_nacimiento DATE
7 );
8
9 CREATE TABLE Curso(
10    id_curso INT PRIMARY KEY AUTO_INCREMENT,
11    nombre VARCHAR(30) NOT NULL,
12    credits INT CHECK(credits>0)
13 );
14
15 CREATE TABLE Inscripcion(
16    id_alumno INT,
17    id_curso INT,
18    PRIMARY KEY (id_alumno,id_curso),
19    FOREIGN KEY (id_alumno) REFERENCES alumno(id_alumno) ON UPDATE CASCADE ON DELETE RESTRICT,
20    FOREIGN KEY (id_curso) REFERENCES curso(id_curso) ON UPDATE CASCADE ON DELETE RESTRICT
21 );
22

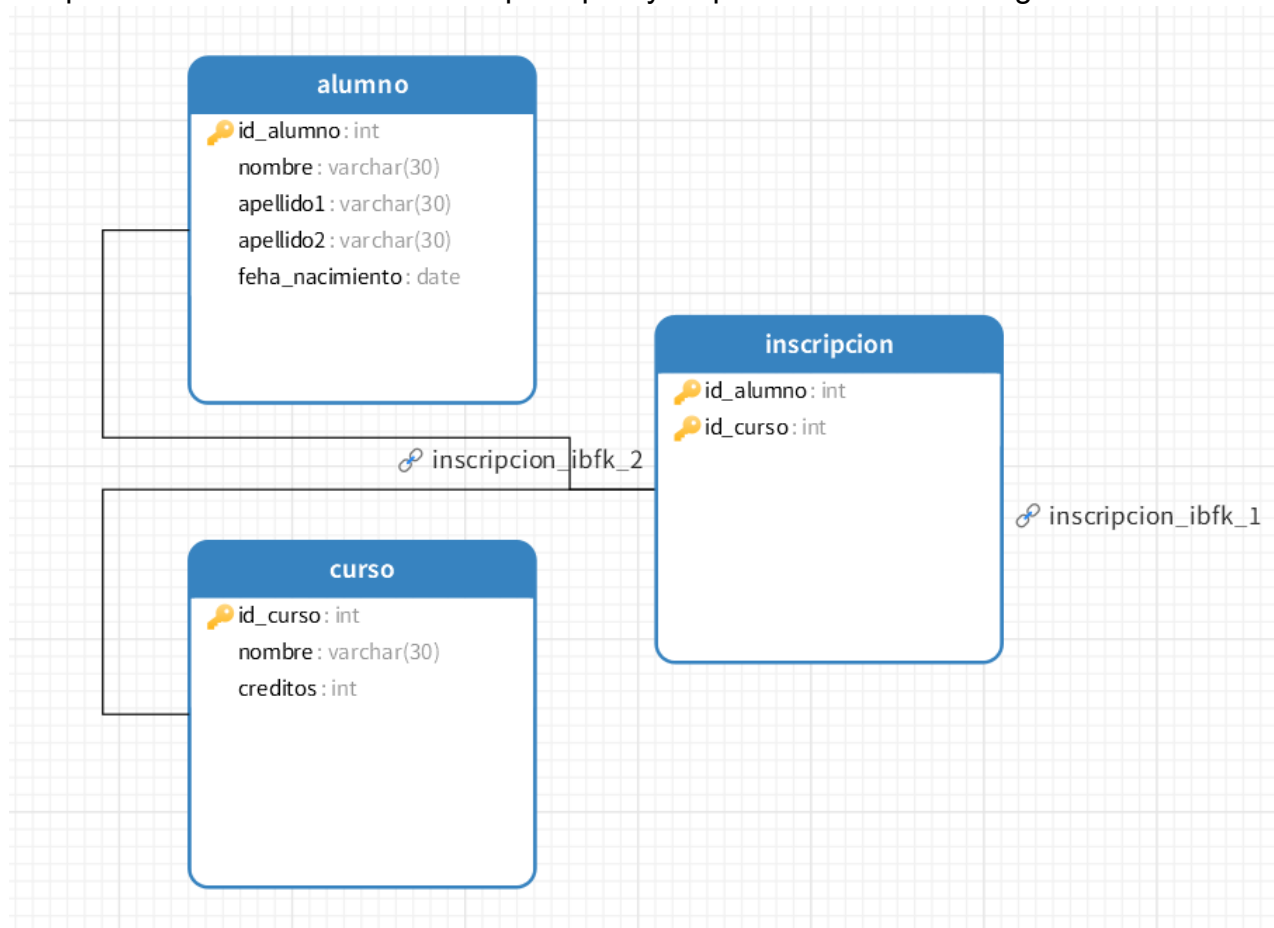
```

En la tabla Alumnos utilice **PRIMARY KEY** para marcar la llave principal y a esta misma le agregue un **AUTO\_INCREMENT** para que fuera incrementando automáticamente con cada registro y así evitar problemas de repetición del id.

A los atributos nombre, apellido1 y apellido 2 les coloque en **NOT NULL** porque quiero que sea obligatorio conocer esos datos del estudiante ya que el nombre es fundamental al momento de registrar datos, sin embargo a el atributo de fecha no lo puse obligatoriamente para llenar, de modo que si algún alumno decide no colocar su fecha de nacimiento no habrá problema ya que no lo considero un dato que influiría al cursar sus materias.

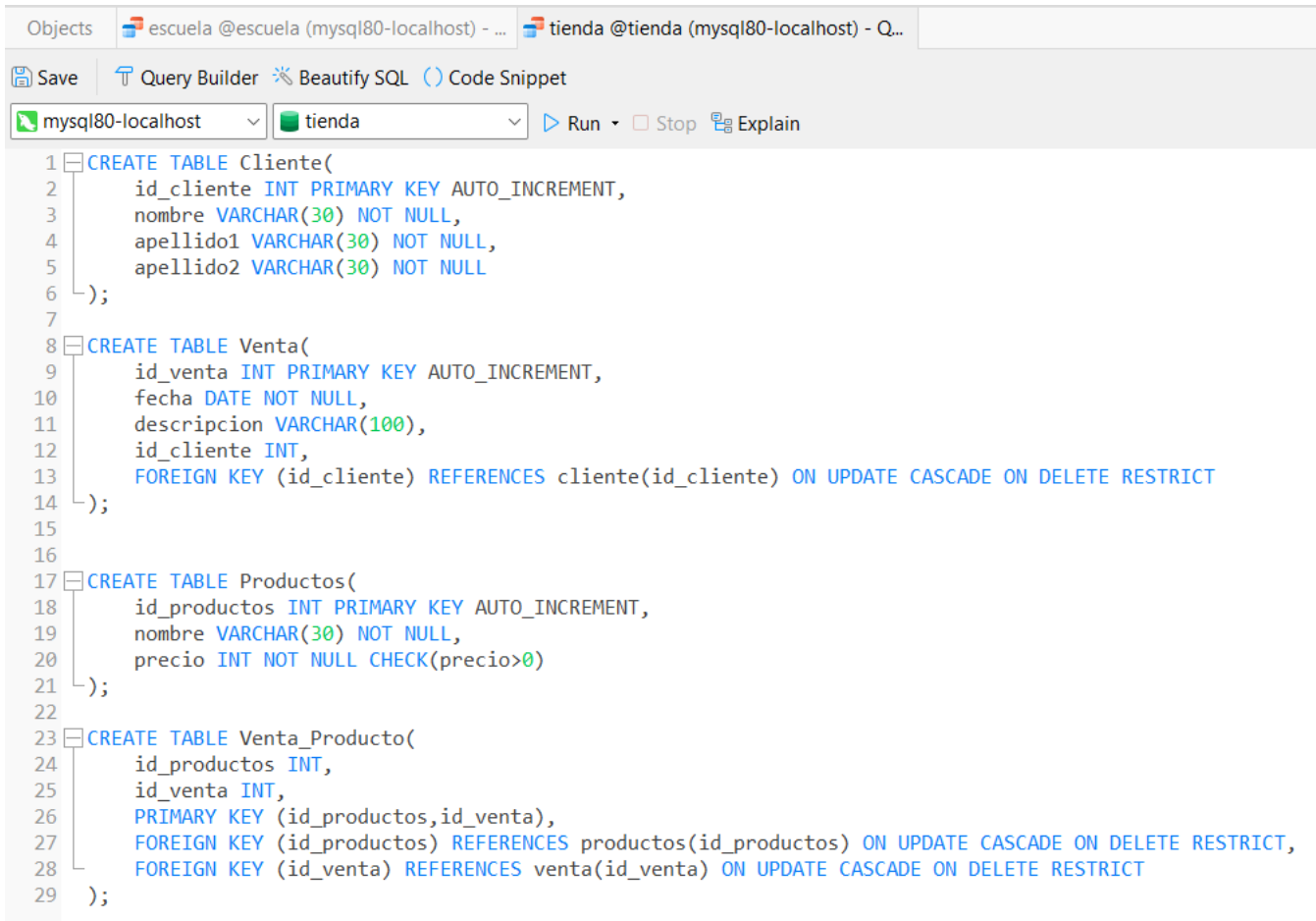
En la tabla Curso igualmente tengo mi llave primaria auto incrementada, el nombre tiene un **NOT NULL** para que sea obligatorio colocar el nombre del curso al cual se inscribirán y el atributo créditos tiene una restricción ya que necesitamos que el número de créditos que tenga ese curso al menos sea mayor a 0.

Como lo que tengo es una relación muchos a muchos de esta se creó una nueva tabla llamada Inscripción, dicha tabla tendrá como llave primaria las dos llaves primarias de las tablas anteriores, pero antes de describir eso tenemos que declararlas dentro de esta nueva tabla, posteriormente indicamos que son mi llave primaria, pero así mismo como son mi llave primaria también son llaves foráneas por lo cual con la palabra **REFERENCES** indico de que tabla es originaria esa llave y con **ON UPDATE CASCADE ON DELETE RESTRICT** que este dato no se pueda eliminar en la tabla principal ya que afectaría los registros de esta tabla.



- **PRIMARY KEY:** Identifica de manera única cada fila en una tabla. Los valores en esta columna no se repiten ni pueden ser nulos.
- **AUTO\_INCREMENT:** Se usa en columnas numéricas para que el sistema asigne automáticamente un valor único y creciente cuando se añaden nuevas filas.
- **NOT NULL:** Asegura que una columna no pueda tener valores vacíos, es decir, siempre debe tener un dato.
- **CHECK (créditos > 0):** Restringe los valores permitidos en una columna. En este caso, asegura que los valores sean mayores a 0.
- **FOREIGN KEY:** Crea una relación entre dos tablas, asegurando que el valor en una tabla esté presente en otra.
- **REFERENCES:** Especifica qué columna de otra tabla está relacionada con la clave externa.
- **ON UPDATE CASCADE:** Si el valor de una clave externa en la tabla principal cambia, se actualiza automáticamente en las tablas relacionadas.
- **ON DELETE RESTRICT:** Evita que se elimine una fila de la tabla principal si está relacionada con otras tablas.

- **Problemática:** Una tienda necesita gestionar las ventas, los clientes y los productos. Cada venta tiene un cliente asociado y puede incluir varios productos.



```
1 CREATE TABLE Cliente(  
2     id_cliente INT PRIMARY KEY AUTO_INCREMENT,  
3     nombre VARCHAR(30) NOT NULL,  
4     apellido1 VARCHAR(30) NOT NULL,  
5     apellido2 VARCHAR(30) NOT NULL  
6 );  
7  
8 CREATE TABLE Venta(  
9     id_venta INT PRIMARY KEY AUTO_INCREMENT,  
10    fecha DATE NOT NULL,  
11    descripcion VARCHAR(100),  
12    id_cliente INT,  
13    FOREIGN KEY (id_cliente) REFERENCES cliente(id_cliente) ON UPDATE CASCADE ON DELETE RESTRICT  
14 );  
15  
16  
17 CREATE TABLE Productos(  
18     id_productos INT PRIMARY KEY AUTO_INCREMENT,  
19     nombre VARCHAR(30) NOT NULL,  
20     precio INT NOT NULL CHECK(precio>0)  
21 );  
22  
23 CREATE TABLE Venta_Producto(  
24     id_productos INT,  
25     id_venta INT,  
26     PRIMARY KEY (id_productos,id_venta),  
27     FOREIGN KEY (id_productos) REFERENCES productos(id_productos) ON UPDATE CASCADE ON DELETE RESTRICT,  
28     FOREIGN KEY (id_venta) REFERENCES venta(id_venta) ON UPDATE CASCADE ON DELETE RESTRICT  
29 );
```

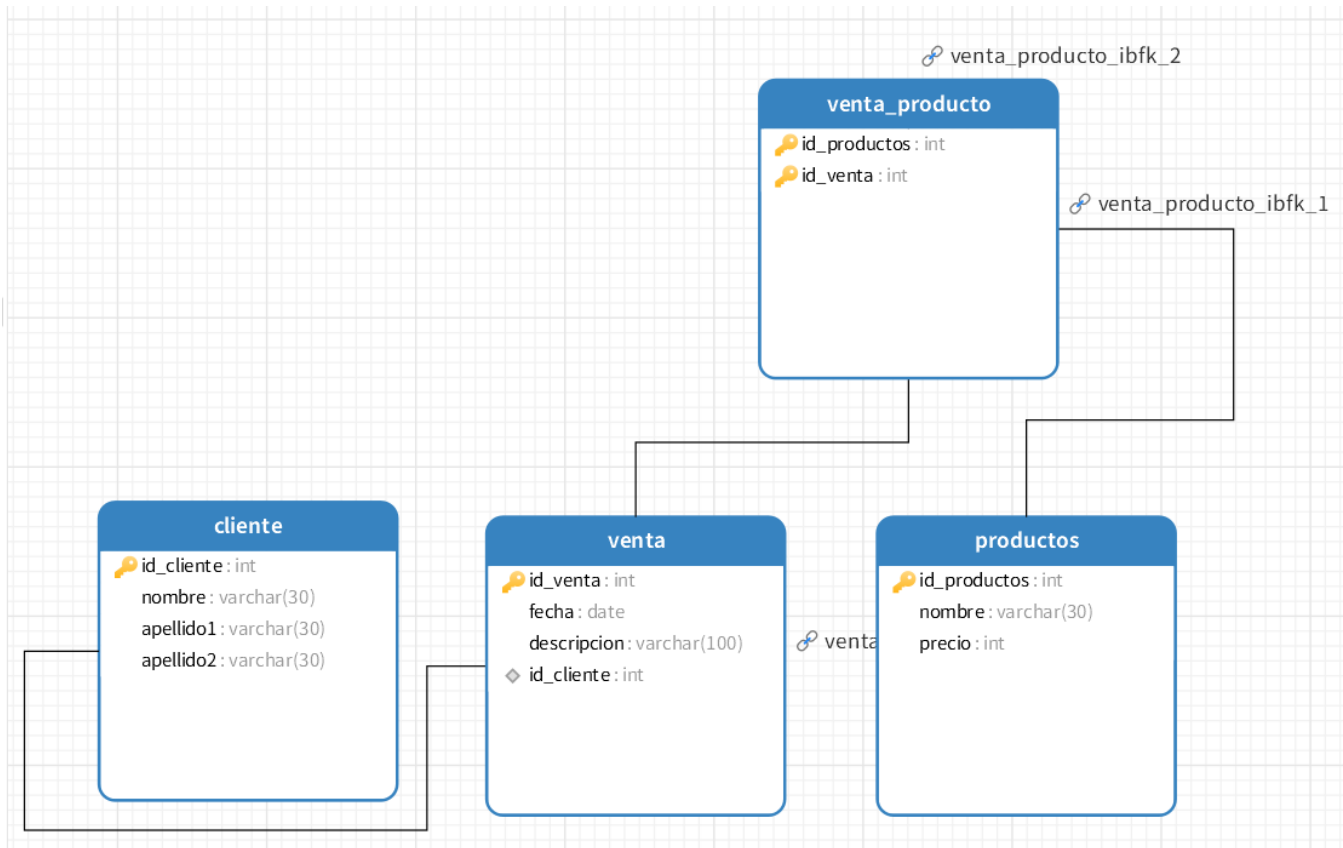
Primero cree mi tabla Cliente la cual lleva un id auto incrementado, y cuyos atributos de nombre y apellido son obligatorios con un **NOT NULL**.

Posteriormente esta la tabla Ventas, esta también tiene un id auto incrementado, una fecha de venta la cual quiero que lleve **NOT NULL** para que sea obligatorio conocer la fecha ya que así podre obtener los cortes de caja o ventas por día, lleva una descripción de la venta a la cual le asigne mayores caracteres en Varchar pero la cual no es obligatoria porque no lleva **NOT NULL**. La tabla cliente tiene una relación 1:N con la tabla ventas por lo cual la tabla ventas va a tener como llave foránea el id\_cliente, con **REFERENCES** indico de donde viene ese id y con **ON UPDATE CASCADE ON DELETE RESTRICT** indico que esta llave no se elimine en la tabla original porque afectaría los registros de esta tabla.

Después tengo la tabla productos con un id, con el nombre del producto obligatorio y el precio además de ser obligatorio por el **NOT NULL** tiene un **CHECK** el cual hace referencia a que el precio debe ser mayor a 0.

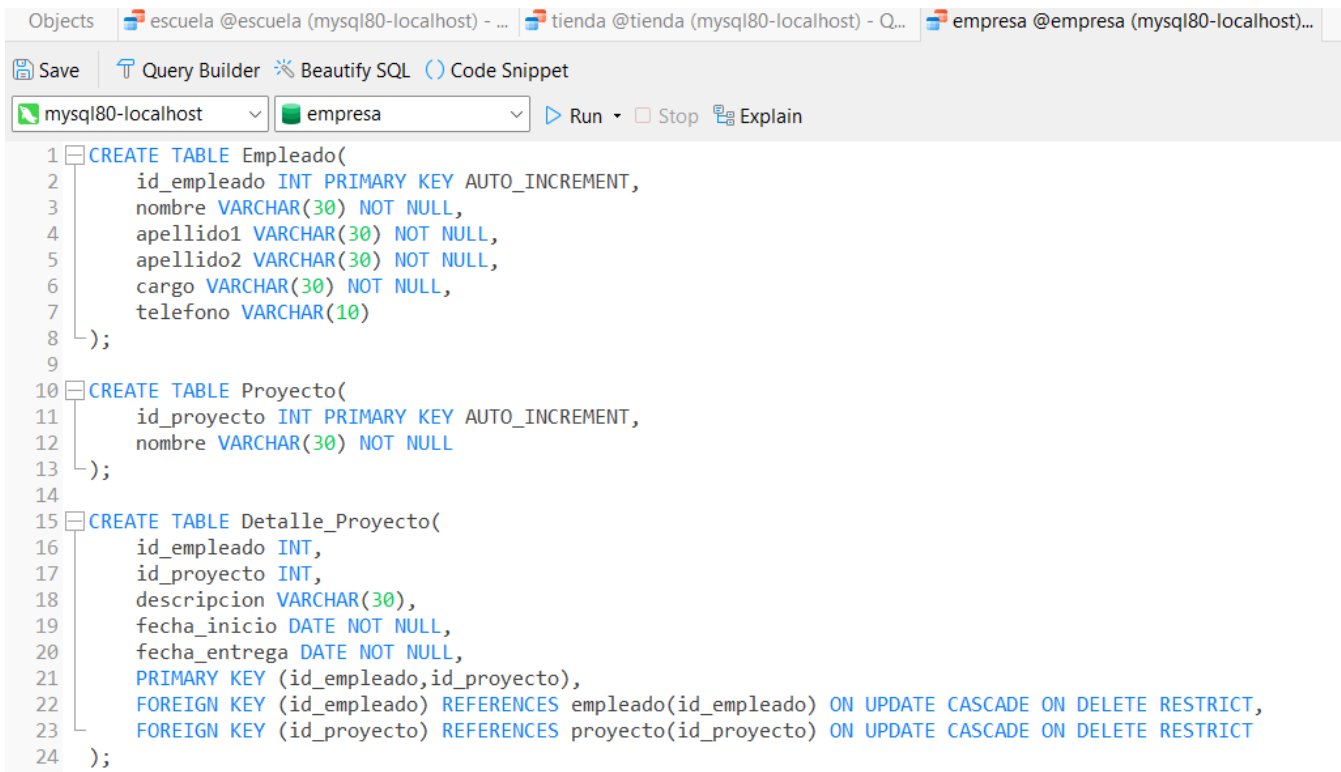
La tabla ventas y la tabla productos tienen una relación muchos a muchos por lo cual se creó una nueva tabla llamada Ventas\_Productos, declare la llave primaria a la llave de la tabla productos así como a la de la tabla ventas y también las coloque como llaves foráneas, referencie de que tablas provenían y coloque las restricción **ON UPDATE CASCADE ON**

**DELETE RESTRICT** para indicar que esta llave no se elimine en la tabla original porque afectaría los registros de esta tabla.



- **PRIMARY KEY**: Identifica de manera única cada fila en una tabla. No puede haber valores repetidos ni nulos.
- **AUTO\_INCREMENT**: Asigna automáticamente un valor numérico único y creciente a cada nueva fila en una columna, generalmente en la clave primaria.
- **NOT NULL**: Asegura que una columna siempre tenga un valor y no permita valores vacíos (nulos).
- **FOREIGN KEY**: Crea una relación entre tablas, garantizando que los valores en una columna de la tabla hija existan en la columna referenciada de la tabla padre.
- **CHECK (precio > 0)**: Restringe los valores de la columna precio para que sean mayores a 0, validando que siempre haya un precio positivo.
- **REFERENCES**: Define a qué tabla y columna se refiere una clave externa (foreign key), creando una relación entre tablas.
- **ON UPDATE CASCADE**: Si el valor en la tabla referenciada cambia, el cambio se refleja automáticamente en la tabla relacionada.
- **ON DELETE RESTRICT**: Impide eliminar una fila de la tabla principal si está relacionada con otra tabla, protegiendo los datos referenciados.

- **Problemática:** Una empresa necesita gestionar los proyectos y los empleados que trabajan en ellos. Cada empleado puede trabajar en varios proyectos y cada proyecto puede tener varios empleados.



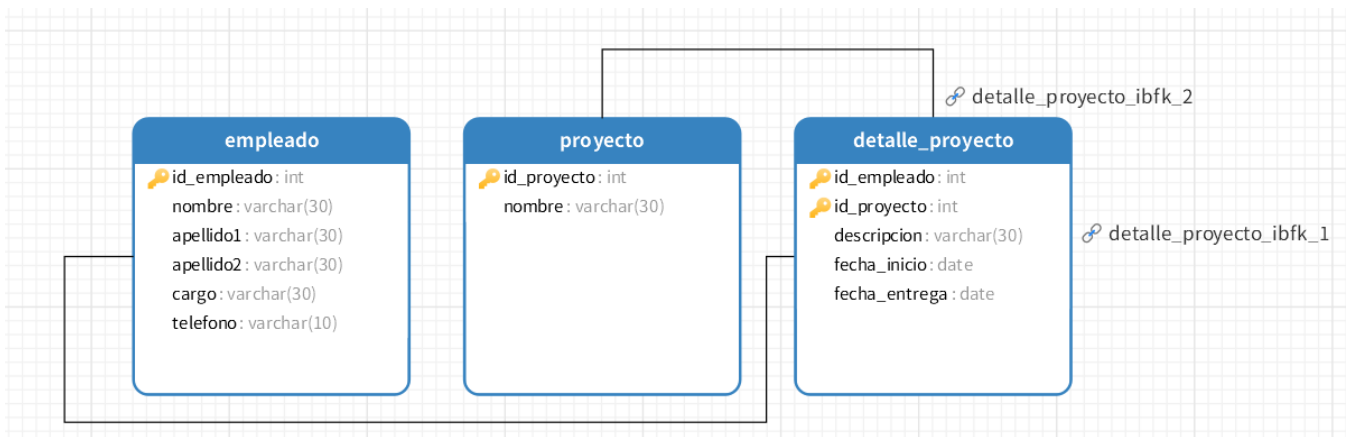
```
1 CREATE TABLE Empleado(  
2     id_empleado INT PRIMARY KEY AUTO_INCREMENT,  
3     nombre VARCHAR(30) NOT NULL,  
4     apellido1 VARCHAR(30) NOT NULL,  
5     apellido2 VARCHAR(30) NOT NULL,  
6     cargo VARCHAR(30) NOT NULL,  
7     telefono VARCHAR(10)  
8 );  
9  
10 CREATE TABLE Proyecto(  
11     id_proyecto INT PRIMARY KEY AUTO_INCREMENT,  
12     nombre VARCHAR(30) NOT NULL  
13 );  
14  
15 CREATE TABLE Detalle_Proyecto(  
16     id_empleado INT,  
17     id_proyecto INT,  
18     descripcion VARCHAR(30),  
19     fecha_inicio DATE NOT NULL,  
20     fecha_entrega DATE NOT NULL,  
21     PRIMARY KEY (id_empleado, id_proyecto),  
22     FOREIGN KEY (id_empleado) REFERENCES empleado(id_empleado) ON UPDATE CASCADE ON DELETE RESTRICT,  
23     FOREIGN KEY (id_proyecto) REFERENCES proyecto(id_proyecto) ON UPDATE CASCADE ON DELETE RESTRICT  
24 );
```

Primero cree una tabla llamada empleados, la cual tiene un id auto incrementado, el nombre, apellido1, apellido2 así como el cargo son datos Varchar que serán obligatorios de llenar por lo cual les coloque el **NOT NULL**.

El teléfono lo defino como VARCHAR ya que no lo necesito para realizar operaciones aritméticas, solo como información y le coloque una longitud de 10 porque son los 10 dígitos de un número telefónico.

Posteriormente cree una tabla de proyecto la cual lleva un id auto incrementable y el nombre del proyecto con **NOT NULL** para obligar a colocarle un nombre a los proyectos para después relacionarlos con los detalles del mismo o con los empleados.

La relación que tiene la tabla empleados con la tabla proyectos es de muchos a muchos por lo cual se origina una nueva tabla llamada Detalles\_Proyecto, en ella coloque algunos atributos como una descripción, la fecha de inicio y fin la cual al tener NOT NULL es obligatoria de colocar. Después declare las variables de los id que serían mi llave primaria. Y seguido de esto también coloque que mis llaves primarias serían al mismo tiempo llaves foráneas, esto con **FOREIGN KEY**, para después referenciarlas con **REFERENCES** y después colocar la restricción **ON UPDATE CASCADE ON DELETE RESTRICT** para que no eliminarán los id originales ya que afectarían la existencia de esta tabla.



- **PRIMARY KEY:** Identifica de forma única cada fila de una tabla. No se permiten valores duplicados o nulos en esta columna.
- **AUTO\_INCREMENT:** Asigna automáticamente un número único y creciente a cada nueva fila insertada en una columna, generalmente usada con la clave primaria.
- **NOT NULL:** Asegura que una columna siempre tenga un valor y no permita valores vacíos.
- **FOREIGN KEY:** Crea una relación entre dos tablas, asegurando que los valores en una tabla existan en la otra.
- **REFERENCES:** Especifica a qué tabla y columna se refiere la clave externa (foreign key).
- **ON UPDATE CASCADE:** Si el valor en la tabla principal cambia, se actualiza automáticamente en las tablas relacionadas.
- **ON DELETE RESTRICT:** Evita eliminar una fila de la tabla principal si está relacionada con otra tabla, impidiendo la eliminación de datos clave.

- **Problemática:** Un hospital necesita gestionar a sus pacientes, médicos y las consultas que se realizan. Cada paciente puede tener múltiples consultas, y cada consulta es atendida por un médico.

```

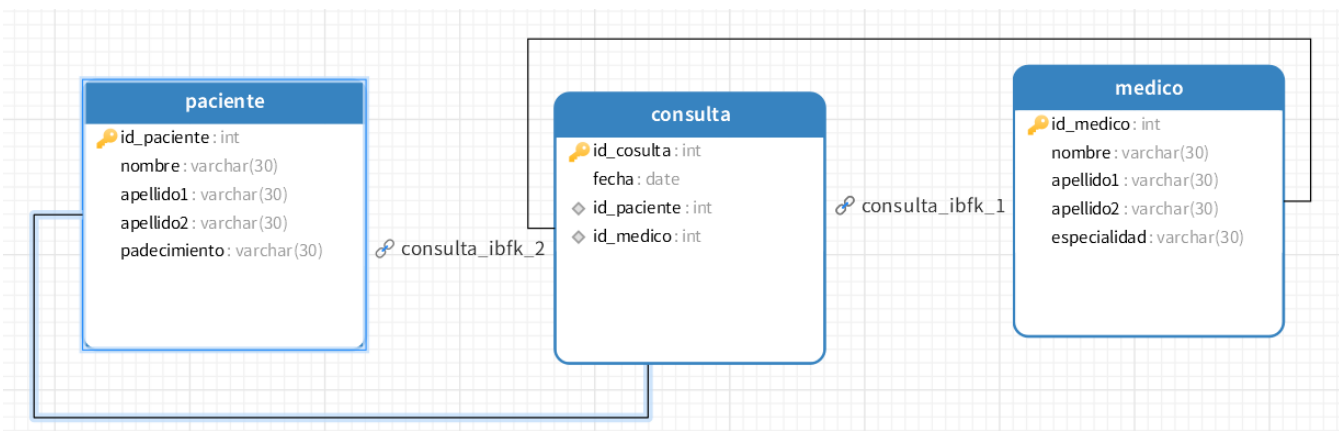
1 CREATE TABLE Paciente(
2   id_paciente INT PRIMARY KEY AUTO_INCREMENT,
3   nombre VARCHAR(30) NOT NULL,
4   apellido1 VARCHAR(30) NOT NULL,
5   apellido2 VARCHAR(30) NOT NULL,
6   padecimiento VARCHAR(30) NOT NULL
7 );
8
9 CREATE TABLE Consulta(
10  id_consulta INT PRIMARY KEY AUTO_INCREMENT,
11  fecha DATE NOT NULL,
12  id_paciente INT,
13  id_medico INT,
14  FOREIGN KEY (id_paciente) REFERENCES paciente(id_paciente) ON UPDATE CASCADE ON DELETE RESTRICT,
15  FOREIGN KEY (id_medico) REFERENCES medico(id_medico) ON UPDATE CASCADE ON DELETE RESTRICT
16 );
17
18
19
20 CREATE TABLE Medico(
21  id_medico INT PRIMARY KEY AUTO_INCREMENT,
22  nombre VARCHAR(30) NOT NULL,
23  apellido1 VARCHAR(30) NOT NULL,
24  apellido2 VARCHAR(30) NOT NULL,
25  especialidad VARCHAR(30) NOT NULL
26 );

```

Cree la tabla paciente, la cual contiene un id incrementable, así como un nombre, apellidos y padecimiento obligatoria mente debido al **NOT NULL**.

La tabla de médico también contiene el id del médico auto incrementable, su nombre y apellidos obligatoriamente, así como su especialidad o área, la cual va a acompañada de un **NOT NULL** para considerar que es un campo que debe llenarse. Cree la tabla paciente, la cual contiene un id incrementable, así como un nombre, apellidos y padecimiento obligatoria mente debido al **NOT NULL**.

La tercera tabla fue la de consultas, esta tabla tiene su propio id auto incrementable, así como la fecha de la consulta obligatoriamente para tener un control en las citas. Esta tabla tiene una relación 1 a 1 con la tabla médico y paciente por lo cual estas tablas le seden su id como clave foránea, así que con **REFERENCES** identifico de dónde vienen y con **ON UPDATE CASCADE ON DELETE RESTRICT** restringió qué puedan eliminar estos id ya que afectarían a esta tabla.





- **PRIMARY KEY:** Define una columna o conjunto de columnas que identifican de manera única cada fila en una tabla. No puede haber valores duplicados o nulos en esta columna.
- **AUTO\_INCREMENT:** Se usa para generar automáticamente un número único y creciente en una columna, generalmente utilizada junto con la clave primaria para asignar identificadores únicos a nuevas filas.
- **NOT NULL:** Asegura que una columna no acepte valores nulos, obligando a que siempre se proporcione un valor cuando se inserte o actualice una fila.
- **FOREIGN KEY:** Establece una relación entre dos tablas, garantizando que los valores en una columna de una tabla existan en la columna de otra tabla. Esto ayuda a mantener la integridad referencial entre las tablas.
- **REFERENCES:** Especifica la tabla y la columna a la que se refiere una clave externa (foreign key), estableciendo así la relación entre las tablas.
- **ON UPDATE CASCADE:** Si el valor de la columna referenciada en la tabla principal cambia, ese cambio se reflejará automáticamente en las tablas relacionadas que tienen una clave externa con esta columna.
- **ON DELETE RESTRICT:** Impide la eliminación de una fila en la tabla principal si está siendo referenciada por una fila en otra tabla. Esto protege la integridad de los datos al evitar la eliminación de datos que todavía están en uso en otras tablas.

## Conclusiones:

En MySQL, la gestión de una base de datos se basa en varios conceptos clave que aseguran la organización y consistencia de los datos. La **PRIMARY KEY** garantiza que cada fila de una tabla tenga un identificador único, evitando duplicados y valores vacíos. Cuando se usa **AUTO\_INCREMENT** en una columna, MySQL asigna automáticamente valores únicos y crecientes.

Las restricciones como **NOT NULL** aseguran que todas las columnas tengan datos válidos, previniendo la entrada de valores vacíos en los campos. Para mantener la integridad entre diferentes tablas, se utilizan **FOREIGN KEY** y **REFERENCES**, que establecen y verifican relaciones entre tablas, asegurando que los datos en una tabla correspondan con los de otra.

Además, **ON UPDATE CASCADE** permite que los cambios en una tabla principal se reflejen automáticamente en las tablas relacionadas, mientras que **ON DELETE RESTRICT** previene la eliminación de datos que aún están en uso en otras tablas. Estos mecanismos ayudan a mantener una base de datos bien estructurada y libre de inconsistencias, garantizando que los datos sean precisos y estén bien relacionados.