1. **El trabajo debe ser INDIVIDUAL y ORIGINAL (creado por el propio estudiante)**

La base de datos **empresa\_camiones** está diseñada para gestionar y organizar la información clave de una empresa de transporte de camiones, he tomado esta referencia gracias a mi padre el cual la fundó hace más de 20 años y nunca ha ido mejor. Esta empresa ha crecido y evolucionado a lo largo de las décadas, y la base de datos refleja esta evolución

* **Empleados:** Almacena la información de los empleados de la empresa, incluyendo datos personales, puesto, salario y fecha de contratación.
* **Clientes:** Contiene los datos de los clientes, como nombre, apellidos, correo electrónico y teléfono.
* **Camiones:** Registra detalles sobre los camiones disponibles en la empresa, incluyendo marca, modelo, año de fabricación, precio y disponibilidad.
* **Ventas:** Guarda información sobre las ventas de camiones, enlazando clientes y camiones con detalles de fecha y precio de venta.
* **Tipos de Mercancía:** Define los diferentes tipos de mercancía que la empresa transporta.
* **Envíos:** Captura los detalles de los envíos realizados, incluyendo origen, destino, tipo de mercancía y fecha de envío.
* **Mantenimiento:** Almacena registros de mantenimiento realizados en los camiones, con descripciones, costos y fechas.
* **Historial de Pagos (pagos):** Registra los pagos asociados a las ventas de camiones, detallando el total pagado y la fecha de pago.
* **Detalles de Envíos:** Incluye información detallada de cada envío, como cantidad y peso de la mercancía.
* **Comprobación:** Vincula los registros de mantenimiento con los empleados responsables de realizar el mantenimiento

1. **Piensa en un escenario de base de datos. Escribe las especificaciones de diseño, tal como hicimos en el Tema 1. El modelo debe tener al menos 8 tablas relacionadas entre sí.**

Aquí hablaremos las tablas y sus especificaciones, empezando por la tabla de camión:

**create table camiones** *(*

*id\_camion int auto\_increment primary key,*

*marca varchar(30) not null,*

*modelo varchar(30) not null,*

*ano\_fabricacion date,*

*precio decimal(10, 2),*

*disponible char(1) not null default 'n'*

*);*

Esta tabla consiste por un **id\_camion** y su especificación de **auto\_increment**, este hará que su valor se incremente automáticamente cada vez que se inserta una nueva fila.

Además podemos ver los componentes de esta tabla camión empezando por marca que tiene **varchar(30)** esto significa que sus datos **no podrán sobrepasar 30 caracteres**, a continuación nos fijamos en la fila de disponible, esta es **Char(1),** la cual significa que solo podrá tener un valor y **“not default ‘n’”** que significa que por defecto **aparecerá con la letra ‘N’**.

También deberemos de tener en cuenta la fila “ano\_fabricacion” ya que su clave es **date**, esto significa que sus valores serán **solo fechas** como por ejemplo: ‘2024-06-30’.

Por último deberemos fijarnos en la clave **Decimal (10,2),** este hace normalmente referencia a la fila Precio, Salario, Pago etc…

*Continuamos con la tabla empleados:*

**create table empleados** *(*

*id\_empleado int auto\_increment primary key,*

*nombre varchar(30) not null,*

*apellido varchar(30),*

*puesto varchar(30),*

*salario decimal(10, 2),*

*fecha\_contratacion date*

*);*

Esta tabla consiste por un **id\_empleado** con la misma especificación que la anterior “auto\_increment”. Esta también cuenta con la clave **date** en “fecha\_contratacion”. Aquí podemos ver la fila “Puesto”, esta está constituida por los cargos de la empresa, los cuales veremos más adelante

*Continuamos con la tabla de cliente:*

**create table clientes** *(*

*id\_cliente int auto\_increment primary key,*

*nombre varchar(30) not null,*

*apellidos varchar(30),*

*email varchar(50),*

*telefono char(10)*

*);*

Esta tabla consiste por un **id\_cliente** pero no tiene nada nuevo, en ella se hospedaran toda la información de los clientes los cuales hagan alguna compra en nuestra empresa, estas compras se registraran en la tabla de ventas

*Continuamos con la tabla ventas:*

**create table ventas** *(*

*id\_venta int auto\_increment primary key,*

*id\_cliente int,*

*id\_camion int,*

*fecha\_venta date,*

*precio\_venta decimal(10, 2),*

*foreign key (id\_cliente) references clientes(id\_cliente),*

*foreign key (id\_camion) references camiones(id\_camion)*

*);*

Esta tabla consiste por un **id\_venta** con el valor auto\_increment, id\_cliente haciendo referencia a la tabla de cliente a través de **foreign key** y **references**. Esto lo podemos ver también con id camión haciendo referencia a la tabla camion. En esta tabla se registraran todas las ventas

*Continuamos con la tabla Tipos de mercancía:*

create table **tipos\_mercancia***(*

*id\_mercancia int auto\_increment primary key,*

*nombre varchar(30) not null*

*);*

Esta tabla consiste por un **id\_mercancia** con el valor “auto\_increment” y con el nombre, aquí usaremos los nombre de los tipos de mercancía que podemos distribuir, en nuestro caso usaremos: Electrónicos, Material de construcción, Muebles, Productos químicos, Ropa)

Continuamos con la tabla envíos:

**create table envios** *(*

*id\_envio int auto\_increment primary key,*

*id\_camion int,*

*tipo\_mercancia int,*

*origen varchar(40),*

*destino varchar(40),*

*fecha\_envio date,*

*foreign key (id\_camion) references camiones(id\_camion),*

*foreign key (tipo\_mercancia) references tipos\_mercancia(id\_mercancia)*

*);*

Esta tabla consiste por un **id\_envio** con el valor auto\_increment, una foreign key haciendo referencia a la tabla **camión** y otra haciendo referencia a la tabla **tipo\_mercancia** visto anteriormente. Esta tabla contará con todos los envíos realizados por camiones, añadiendo así su transportista y su destino

*Continuamos con la tabla mantenimiento:*

**create table mantenimiento** *(*

*id\_mantenimiento int auto\_increment primary key,*

*id\_camion int,*

*descripcion varchar(200),*

*precio decimal(10, 2),*

*fecha\_mantenimiento date,*

*foreign key (id\_camion) references camiones(id\_camion)*

*);*

Esta tabla consiste por un **id\_mantenimiento** con el valor auto\_increment, una foreign key haciendo referencia a la tabla **camión,** podemos ver que en esta tabla existe una fila llamada Descripción con **varchar(200),** este varchar funciona igual que varchar(30) solo que al ser una descripción necesitamos más espacio y he puesto 200 caracteres

*Continuamos con la tabla Historia de pagos (pagos):*

**create table pagos** *(*

*id\_pago int auto\_increment primary key,*

*id\_venta int,*

*total\_pagado decimal(10, 2),*

*fecha\_pago date,*

*foreign key (id\_venta) references ventas(id\_venta)*

*);*

En esta tabla he acortado el nombre, pero igualmente consiste en un **id\_pago** y con una serie de filas para los mismo, por único, recalcar que hace referencia a la tabla ventas a través de la **foreign key (id\_venta).**

*Continuamos con la tablas detalles de envios:*

**create table detalles\_envios** *(*

*id\_detalles int auto\_increment primary key,*

*id\_envio int,*

*cantidad int,*

*peso decimal(10, 2),*

*foreign key (id\_envio) references envios(id\_envio)*

*);*

Esta tabla consiste de un **id\_detalle**.

*Continuamos con la tabla comprobación:*

**create table comprobacion** *(*

*id\_comprobacion int auto\_increment primary key,*

*id\_mantenimiento int,*

*id\_empleado int,*

*foreign key (id\_mantenimiento) references mantenimiento(id\_mantenimiento),*

*foreign key (id\_empleado) references empleados(id\_empleado)*

*);*

En la tabla de comprobación que está formada por un **id\_comprobacion** y dos referencias a las tablas de mantenimiento y empleado, en ella daremos por confirmada la comprobación del mantenimiento realizado, ya que muchas veces puede salir algún vehículo en mal estado.

1. **Dibuja el modelo de BD acorde a las especificaciones del punto anterior.**

En las bases de datos nos podemos encontrar con 3 tipos de relaciones que son:

1. **Relación de uno a uno:** Cada fila de una tabla está vinculada a una sola fila de otra tabla y viceversa
2. **Relación de uno a muchos:** Una fila de una tabla puede estar relacionada con múltiples filas de otra tabla, pero una fila de la segunda tabla solo puede estar relacionada con una fila de la primera tabla.
3. **Relación de muchos a muchos**: Las filas de una tabla pueden estar relacionadas con múltiples filas de otra tabla y viceversa

El modelo de esta base de datos consiste en un eje central que es: Camiones, dentro de ella nos encontramos con las siguientes relaciones

* **Relaciones de camiones:**

1. Tiene una relación uno a muchos con la tabla de ventas.
2. Tiene una relación uno a muchos con la tabla mantenimiento.
3. Tiene una relación uno a muchos con la tabla envios.

* **Relaciones de ventas:**

1. Tiene una relación muchos a uno con la tabla clientes.
2. Tiene una relación muchos a uno con la tabla camiones..
3. Tiene una relación uno a muchos con la tabla pagos

* **Relaciones de pagos:**

1. Tiene una relación muchos a uno con la tabla de ventas.

* **Relaciones de clientes:**

1. Tiene una relación uno a muchos con la tabla ventas.

* **Relaciones de envios:**

1. Tiene una relación muchos a uno con la tabla camiones.
2. Tiene una relación muchos a uno con la tabla tipos\_mercancia.
3. Tiene una relación uno a muchos con la tabla detalles\_envios.

* **Relaciones de detalles\_envios:**

1. Tiene una relación muchos a uno con la tabla envios.

* **Relaciones de tipos\_mercancia:**

1. Tiene una relación uno a muchos con la tabla envios

* **Relaciones de mantenimiento:**

1. Tiene una relación muchos a uno con la tabla camiones..
2. Tiene una relación uno a muchos con la tabla comprobacion

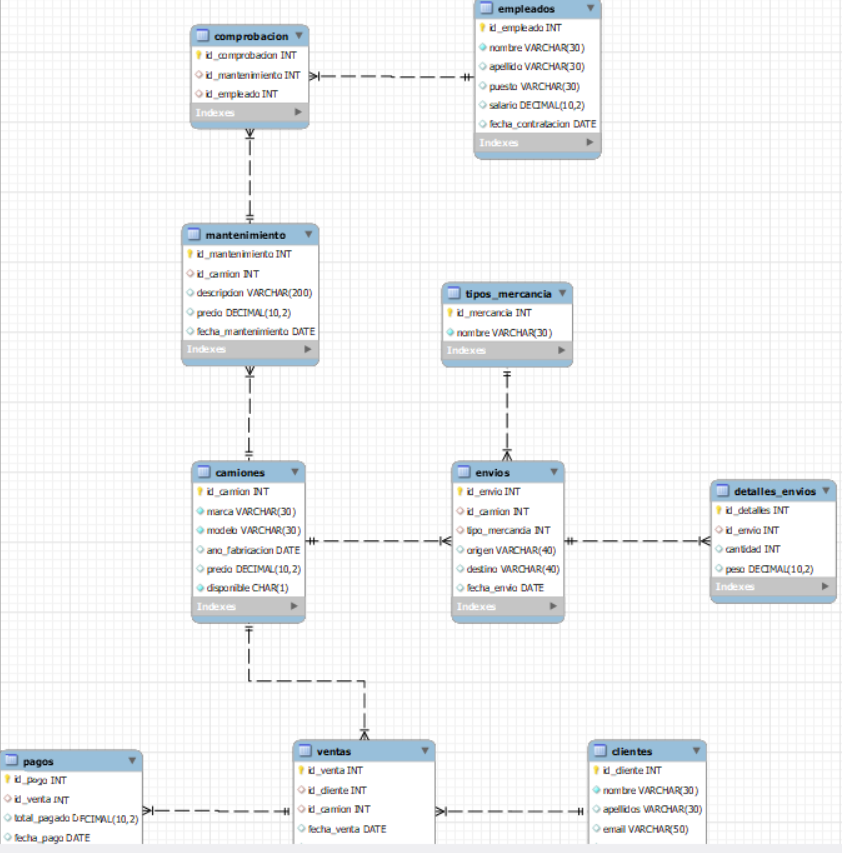
* **Relaciones de comprobacion:**

1. Tiene una relación muchos a uno con la tabla mantenimiento.
2. Tiene una relación muchos a uno con la tabla empleados

* **Relaciones de empleados:**

1. Tiene una relación uno a muchos con la tabla de comprobacion

Al terminar de unir nuestras tablas deberíamos de tener una imagen de la base de datos parecida a la siguiente:



**Puede que no sea exactamente igual, pero no te preocupes ya que lo puedes ordenar de la forma más eficiente. Lo único que tenemos que tener igual son las tablas y sus relaciones**

1. **Crea la Base de Datos**

PASOS PARA CREAR UNA BASE DE DATOS MYSQL:

1. **PRIMER PASO:**

Usando Workbench crearemos nuestro propio modelo de base de datos, el cual usaremos para comprobar las relaciones entre sí, como hemos visto en la imagen anterior

1. **SEGUNDO PASO:**

Una vez creado el diseño entraremos en cmd y usaremos el comando:

* *sudo mysql -u root -p*

1. **TERCER PASO:**

Crearemos un usuario con contraseña, o mejor dicho, el nombre de la actividad para realizar la actividad dentro de él, con el comando:

* *mysql> create user ejemplo1 identified by "ejemplo1" ;*
* *mysql> create user prueba1 identified by "camiones" ;*

1. **CUARTO PASO:**

Aplicaremos privilegios al propio usuario

* mysql> grant all privileges on ejemplo1.\* to ejemplo1;
* mysql> grant all privileges on empresa\_camiones.\* to prueba1;

1. **QUINTO PASO:**

Una vez aplicado, entraremos dentro del usuario, para ello cerraremos la terminal y usaremos el comando

* mysql -u empresa\_camiones-p

Pedirá la contraseña y usaremos la anteriormente aplicada (vehículos)

1. **SEXTO PASO:**

Una vez dentro necesitaremos crear la base de datos con el comando:

* create database empresa\_camiones ;

1. **SÉPTIMO PASO**

Si no tenemos ningun error ya podremos usar la base de datos con el comando

* use empresa\_camiones

**Una vez creada la base de datos, desde la misma terminal añadiremos los siguientes datos:**

create table empleados (

id\_empleado int auto\_increment primary key,

nombre varchar(30) not null,

apellido varchar(30),

puesto varchar(30),

salario decimal(10, 2),

fecha\_contratacion date

);

create table clientes (

id\_cliente int auto\_increment primary key,

nombre varchar(30) not null,

apellidos varchar(30),

email varchar(50),

telefono char(10)

);

create table camiones (

id\_camion int auto\_increment primary key,

marca varchar(30) not null,

modelo varchar(30) not null,

ano\_fabricacion date,

precio decimal(10, 2),

disponible char(1) not null default 'n'

);

create table ventas (

id\_venta int auto\_increment primary key,

id\_cliente int,

id\_camion int,

fecha\_venta date,

precio\_venta decimal(10, 2),

foreign key (id\_cliente) references clientes(id\_cliente),

foreign key (id\_camion) references camiones(id\_camion)

);

create table tipos\_mercancia (

id\_mercancia int auto\_increment primary key,

nombre varchar(30) not null

);

create table envios (

id\_envio int auto\_increment primary key,

id\_camion int,

tipo\_mercancia int,

origen varchar(40),

destino varchar(40),

fecha\_envio date,

foreign key (id\_camion) references camiones(id\_camion),

foreign key (tipo\_mercancia) references tipos\_mercancia(id\_mercancia)

);

create table mantenimiento (

id\_mantenimiento int auto\_increment primary key,

id\_camion int,

descripcion varchar(200),

precio decimal(10, 2),

fecha\_mantenimiento date,

foreign key (id\_camion) references camiones(id\_camion)

);

create table pagos (

id\_pago int auto\_increment primary key,

id\_venta int,

total\_pagado decimal(10, 2),

fecha\_pago date,

foreign key (id\_venta) references ventas(id\_venta)

);

create table detalles\_envios (

id\_detalles int auto\_increment primary key,

id\_envio int,

cantidad int,

peso decimal(10, 2),

foreign key (id\_envio) references envios(id\_envio)

);

create table comprobacion (

id\_comprobacion int auto\_increment primary key,

id\_mantenimiento int,

id\_empleado int,

foreign key (id\_mantenimiento) references mantenimiento(id\_mantenimiento),

foreign key (id\_empleado) references empleados(id\_empleado)

);

**Una vez copiados todos los datos, podremos entrar en Workbench para comprobar que la base de datos empresa\_camiones ya tiene todas las tablas creadas correctamente**

1. **Inserta datos que aporten sentido a la Base de Datos**

**Para ello copiaremos los siguientes datos y desde la misma terminal podemos pegarlos. Una vez pegados y comprobados que no haya ningún error, entraremos en Workbench para comprobar que los datos están correctos**

insert into empleados (nombre, apellido, puesto, salario, fecha\_contratacion) values

('Juan', 'González', 'Chofer', 2500.00, '2023-01-15'),

('María', 'López', 'Mecánico', 2800.00, '2022-08-20'),

('Pedro', 'Martínez', 'Gerente', 3500.00, '2021-05-10'),

('Laura', 'Pérez', 'Recepcionista', 2000.00, '2023-03-01'),

('Carlos', 'García', 'Chofer', 2400.00, '2023-02-10');

insert into clientes (nombre, apellidos, email, telefono) values

('Ana', 'Ruiz', 'ana@gmail.com', 612345678),

('Luis', 'Sánchez', 'luis@gmail.com', 655432189),

('Elena', 'Gómez', 'elena@gmail.com', 699887766),

('Pablo', 'Díaz', 'pablo@gmail.com', 671234567),

('Sofía', 'Fernández', 'sofia@gmail.com', 688998877);

insert into camiones (marca, modelo, ano\_fabricacion, precio, disponible) values

('Volvo', 'vnl 860', '2020-01-01', 120000.00, 'N'),

('Kenworth', 't680', '2019-05-15', 110000.00, 'N'),

('Freightliner', 'cascadia', '2021-03-10', 130000.00, 'N'),

('Peterbilt', '579', '2018-12-20', 105000.00, 'N'),

('Mack', 'anthem', '2022-02-28', 125000.00, 'N');

insert into ventas (id\_cliente, id\_camion, fecha\_venta, precio\_venta) values

(1, 1, '2024-04-10', 115000.00),

(3, 3, '2024-03-25', 125000.00),

(2, 4, '2024-02-15', 107000.00),

(4, 2, '2024-01-05', 112000.00),

(5, 5, '2024-04-01', 120000.00);

insert into tipos\_mercancia (nombre) values

('Electrónicos'),

(‘Materiales de construcción’),

('Muebles'),

('Productos químicos'),

('Ropa');

insert into envios (id\_camion, tipo\_mercancia, origen, destino, fecha\_envio) values

(1, 1, 'Madrid', 'Barcelona', '2024-04-20'),

(3, 3, 'Sevilla', 'Valencia', '2024-03-30'),

(2, 2, 'Bilbao', 'Alicante', '2024-02-25'),

(4, 4, 'Málaga', 'Zaragoza', '2024-01-15'),

(5, 5, 'Valencia', 'Valladolid', '2024-04-05');

insert into mantenimiento (id\_camion, descripcion, precio, fecha\_mantenimiento) values

(1, 'cambio de aceite y filtros', 200.00, '2024-03-01'),

(2, 'revisión del sistema de frenos', 300.00, '2024-02-10'),

(3, 'reparación de motor', 800.00, '2024-01-20'),

(4, 'alineación y balanceo', 150.00, '2024-04-05'),

(5, 'cambio de llantas', 600.00, '2024-02-28');

insert into pagos (id\_venta, total\_pagado, fecha\_pago) values

(1, 115000.00, '2024-04-12'),

(3, 125000.00, '2024-03-28'),

(2, 107000.00, '2024-02-18'),

(4, 112000.00, '2024-01-08'),

(5, 120000.00, '2024-04-03');

insert into detalles\_envios (id\_envio, cantidad, peso) values

(1, 100, 500.00),

(3, 50, 300.00),

(2, 80, 700.00),

(4, 120, 1000.00),

(5, 200, 1500.00);

insert into comprobacion (id\_mantenimiento, id\_empleado) values

(1, 2),

(2, 3),

(3, 1),

(4, 4),

(5, 5);

select \* from empleados;

select id\_cliente, nombre, apellidos, email, telefono

from clientes

order by nombre;

select nombre, apellido, puesto, salario

from empleados

where salario > 2500;

select count(\*) as total\_ventas, sum(precio\_venta) as valor\_total\_ventas

from ventas;

select marca, count(\*) as cantidad\_de\_camiones

from camiones

group by marca;

select id\_cliente, sum(precio\_venta) as total\_ventas

from ventas

group by id\_cliente;

select puesto, avg(salario) as salario\_promedio

from empleados

group by puesto;

select id\_cliente, sum(precio\_venta) as total\_gastado

from ventas

group by id\_cliente

order by total\_gastado desc

limit 1;

select year(ano\_fabricacion) as año, avg(precio) as precio\_promedio

from camiones

group by year(ano\_fabricacion);

select tm.nombre as tipo\_mercancia, e.id\_envio, e.origen, e.destino, e.fecha\_envio

from tipos\_mercancia tm

left join envios e on tm.id\_mercancia = e.tipo\_mercancia;

select c.nombre, sum(v.precio\_venta) as total\_ventas

from ventas v

inner join clientes c on v.id\_cliente = c.id\_cliente

group by c.nombre;

select c.modelo, sum(m.precio) as precio\_total\_mantenimiento

from camiones c

left join mantenimiento m on c.id\_camion = m.id\_camion

group by c.modelo;

select \*

from empleados

where salario = (select max(salario) from empleados);

select id\_detalles, id\_envio, cantidad, peso

from detalles\_envios

where cantidad >

( select avg(cantidad)

from detalles\_envios

);

select \*

from camiones c

where precio = (

select max(precio)

from camiones

where marca = c.marca

);

**function realizar\_venta(a\_id\_cliente int, a\_id\_camion int, a\_fecha\_venta date, a\_precio\_venta decimal(10, 2)) returns int**

drop function if exists realizar\_venta;

delimiter //

create function realizar\_venta(a\_id\_cliente int, a\_id\_camion int, a\_fecha\_venta date, a\_precio\_venta decimal(10, 2)) returns int

reads sql data

begin

declare v\_venta int;

insert into ventas (id\_cliente, id\_camion, fecha\_venta, precio\_venta)

values (a\_id\_cliente, a\_id\_camion, a\_fecha\_venta, a\_precio\_venta);

set v\_venta = last\_insert\_id();

return v\_venta;

end //

delimiter ;

**funcion f\_realizar\_envio(a\_id\_camion int, a\_tipo\_mercancia int, a\_origen varchar(40), a\_destino varchar(40), a\_fecha\_envio date ) returns int**

drop function if exists f\_realizar\_envio;

delimiter //

create function f\_realizar\_envio(a\_id\_camion int, a\_tipo\_mercancia int, a\_origen varchar(40), a\_destino varchar(40), a\_fecha\_envio date ) returns int

reads sql data

begin

declare v\_envio int;

insert into envios (id\_camion, tipo\_mercancia, origen, destino, fecha\_envio)

values (a\_id\_camion, a\_tipo\_mercancia, a\_origen, a\_destino, a\_fecha\_envio);

set v\_envio = last\_insert\_id();

return v\_envio;

end //

delimiter ;

**funcion f\_mantenimiento(a\_id\_camion int) returns decimal(10, 2)**

drop function if exists f\_mantenimiento;

delimiter //

create function f\_mantenimiento(a\_id\_camion int) returns decimal(10, 2)

reads sql data

begin

declare total\_mantenimiento decimal(10, 2);

select sum(precio) into total\_mantenimiento

from mantenimiento

where id\_camion = a\_id\_camion;

if total\_mantenimiento is null then

set total\_mantenimiento = 0;

end if;

return total\_mantenimiento;

end //

delimiter ;

**funcion f\_salario\_promedio(a\_id\_empleado int) returns decimal(10, 2)**

drop function if exists f\_salario\_promedio;

delimiter //

create function f\_salario\_promedio(a\_id\_empleado int) returns decimal(10, 2)

reads sql data

begin

declare salario\_promedio decimal(10, 2);

select avg(salario) into salario\_promedio

from empleados

where id\_empleado = a\_id\_empleado;

if salario\_promedio is null then

set salario\_promedio = 0;

end if;

return salario\_promedio;

end //

delimiter ;