MANIPULANDO DATOS

BD-
ACTIVIDAD 3.1

Trabajo de Alejandro Sainz Sainz

Manipulando Datos: Actividad 3.1

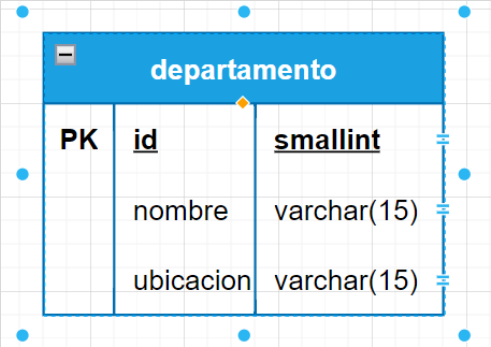
Alejandro Sainz Sainz

Miércoles 15 de enero de 2025

Alejandro Sainz Sainz

COMENZANDO POR EL RELACIONAL

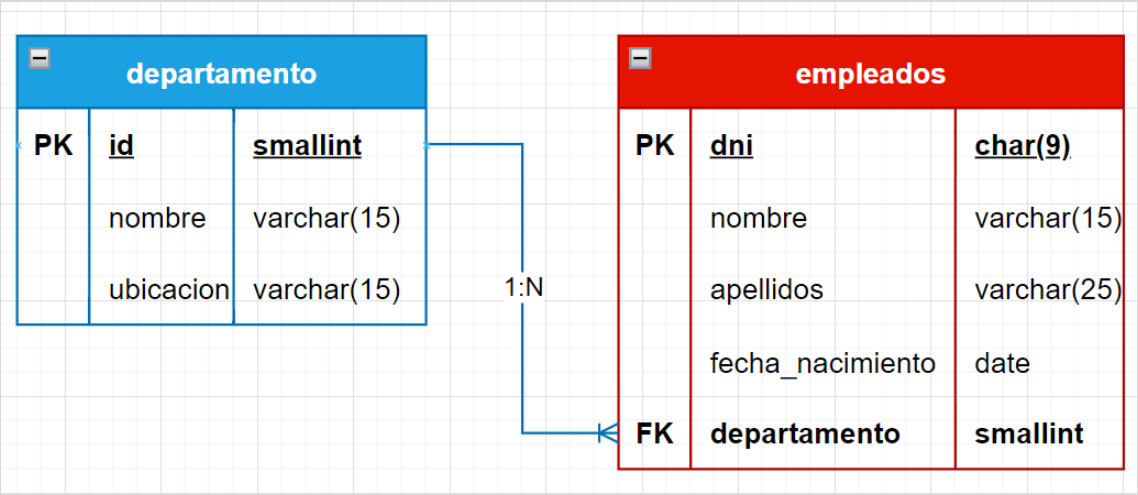
Pues como bien se solicita en el ejercicio vamos a comenzar creando el modelo relacional. Así que, sin más, creo el diagrama y razono el porque lo he creado así.



La primera tabla que creo es la tabla departamento, es la única que es independiente. Sus campos son:

Id: Es la PK, debe de ser un campo único, y para poder ser autoincrement la defino como un smallint. Un int también podría ser correcto.

Nombre y Ubicación: Yo los entiendo como campos texto, descriptivos. De ahí su tipo, varchar. La longitud de 15 es porque creo que es más que suficiente para ello.



Lo siguiente es crear la relación de departamento con empleados y la tabla empleados. Es una relación 1:N, ya que a cada empleado sólo le corresponde un departamento, pero cada departamento pude incluir varios empleados. El campo id de departamento pasa, por esa razón, como FK a la tabla empleados.

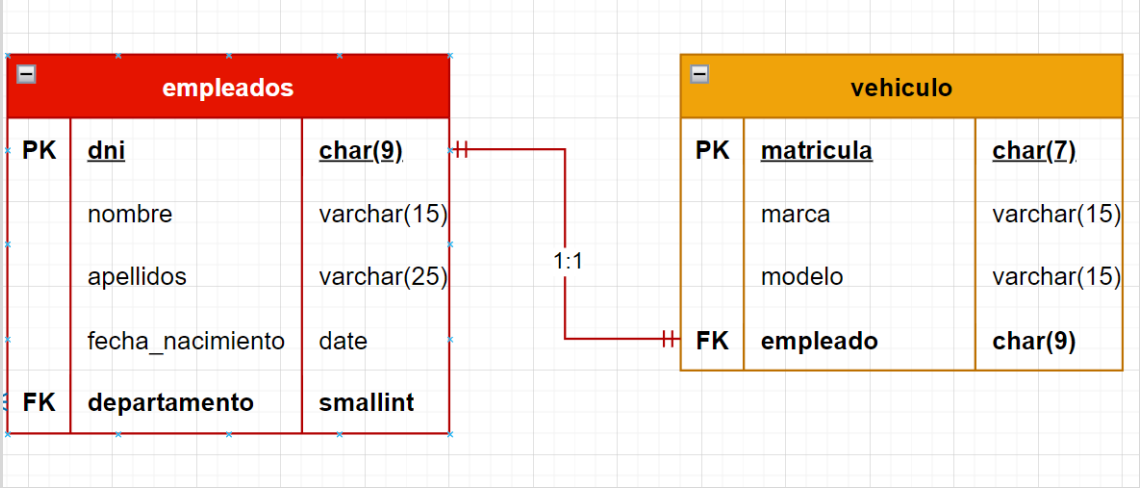
Ahora vamos a explicar los campos creados para la tabla:

Dni: La PK de la tabla, el identificador único. Como en otros ejercicios, el tipo es un char(9), ya que conocemos la longitud exacta de un dni y podemos definir así el campo.

Nombre y Apellidos: Dos campos del tipo texto, pero cuya longitud puede ser variable, así que varchar. A nombre una longitud de 15 y apellidos de 20 pues pueden necesitar más espacio.

Fecha de nacimiento: Al ser una fecha será tipo date. Nada más que añadir.

Departamento: Es el homónimo de id de la tabla departamento. Le he cambiado el nombre en la tabla empleado para que sea más descriptivo, poner simplemente id hace que no sepamos a que se puede referir. Es cierto que lo sabemos por la relación que se forma, pero no por lo descriptivo de su nombre. Pasa como FK. Su tipo, el mismo del campo al que referencia, smallint.



Por último, la tabla vehículo y su relación con empleados.

En este caso he considerado una relación con cardinalidad 1:1, explico el porqué.

El enunciado dice, que algunos empleados disponen de un vehículo de empresa. Eso quiere decir que habrá otros que no.

Podríamos considerar en este supuesto que la relación sería 0:1. Pero yo he tenido en cuenta, que a un coche, siempre le va a corresponder un empleado, a no ser que le empresa compre coches para tenerlos simplemente aparcados.

Por lo tanto, he resuelto que, en este caso, la cardinalidad será de 1:1, y que la clave principal de empleado (dni) pasará como FK a la tabla vehículo,

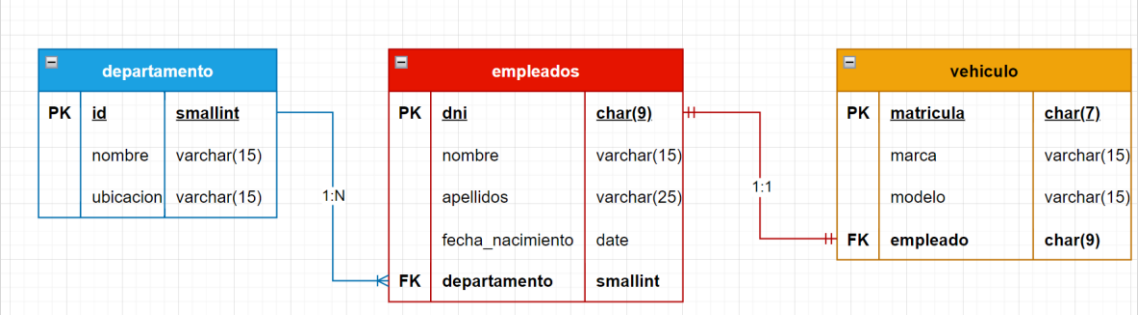
para que así cada vehículo tenga registrado el empleado al que está asignado, porque, aunque haya decidido que la cardinalidad sea 1:1, no puedo pasar la clave de vehículo a empleado como FK, pues habrá empleados que no tengan asociado ningún vehículo.

En cuanto a los atributos de la tabla y sus tipos:

Matrícula: PK de la tabla, de tipo char(7), que es la longitud normal de una matrícula si no tenemos en cuenta la letra del país.

Marca y Modelo: como son campos texto de longitud variable, varchar, de longitud 15 los dos que creo que es más que suficiente.

Empleado: como en la relación anterior, le cambio el nombre al atributo que se recibe como FK, para que sea más descriptivo. Del mismo tipo que el atributo del que es origen.



Así quedaría finalmente el diagrama de mi ejercicio.

GENERANDO NUESTRO SCRIPT

Vamos a ir paso a paso creando el script de sql para workbench.

```
#comienzo con los comandos básicos de la base de datos
drop database if exists empresa;
create database empresa;

use empresa;
```

Lo primero es crear la base de datos, la dropamos en caso de que exista, ya que como se va a ejecutar varias veces el script para hacer las pruebas, obligatoriamente la vamos a tener que eliminar al inicio de cada ejecución.

Después con créate database se crea nuestra BD.

Acto seguido indicamos que queremos usar la BD que hemos creado.

```
create table departamento(
    id smallint auto_increment not null,
    nombre varchar(15) not null,
    ubicacion varchar(15) not null,
    constraint id primary key (id)
);
```

Como en el caso del modelo relacional, y obligatoriamente, puesto que de no ser así nuestro script fallaría, debemos de crear la única tabla que no tiene dependencias, en este caso departamento.

La he creado ciñéndome al diagrama que he creado.

```
create table empleados(
    dni char(9) unique not null,
    nombre varchar(15) not null,
    apellidos varchar(25) not null,
    fecha_nacimiento date not null,
    departamento smallint,
    constraint dni primary key (dni),
    constraint departamento foreign key(departamento) references departamento(id)
    );
```

De nuevo, si seguimos nuestro modelo, creamos la tabla empleados y, mediante el constraint, indicamos cual es la PK y la FK (indicando en esta a que atributo de que tabla hace referencia).

```
create table vehiculo(
    matricula char(7) unique not null,
    marca varchar(15),
    modelo varchar(15),
    empleado char(9),
    constraint matricula primary key (matricula),
    constraint empleado foreign key (empleado) references empleados(dni)
);
```

Por último, creamos la tabla vehículo y su relación mediante la FK. En las tablas anteriores la mayoría de los atributos son not null, pero en este caso, y debido a un ejercicio posterior, modelo, marca y empleado pueden recibir valores nulos.

Crea al empleado #1, al departamento #1 y al coche de empresa #1

Una vez creadas todas las tablas, damos paso a los diferentes subejercicios dentro del mismo script, que es la manipulación de datos.

En este caso, como indica el título, debo de introducir un registro en cada tabla.

```
insert into departamento (nombre,ubicacion) values('CONTABILIDAD','TERCERA PLANTA');
insert into empleados values('13000001A','ANTONIO','GARCIA FERNANDEZ','2008-11-12',1);
insert into vehiculo values('0000AAA','RENAULT','RAPIDO','13000001A');
```

Estos son los comandos. Mediante insert into indico que quiero insertar un registro, a continuación, indico el nombre de la tabla y luego los valores con una pequeña aclaración.

En el primer registro de la tabla departamento, como se vio en clase, al ser la PK un campo autoincrement debo indicar las columnas en las que voy a insertar datos, en este caso nombre y ubicación, que son la segunda y la tercera respectivamente.

Luego en cada comando entre paréntesis voy indicando los datos de cada atributo del registro, entre comillas simples cada uno de ellos aquellos que sean char, varchar y date.

Crea 3 empleados más, utilizando una única sentencia

Este comando es un poco largo, así que le voy a copiar aquí en el texto para que se vea mejor.

insert into empleados (dni, nombre, apellidos, fecha_nacimiento)

values ('13000002B','PEPE','RODRIGUEZ','2000-12-12'),

('13000003C','JUAN','DIAZ PEREZ','1995-11-06'),

('13000004D','MARIA','GONZALEZ PI','1996-09-07');

Bien, no tiene más que explicar de lo que hice en el campo anterior. Al indicar al crear las tablas que departamento puede recibir valores nulos no tengo que incluir el campo entre los requeridos, así que tengo que indicar en el comando que voy a introducir los otros cuatro atributos de cada empleado.

Luego tras la palabra values inserto cada empleado entre paréntesis separado del siguiente por una coma.

Crea el coche de empresa #2, el cual sólo se identifica por su matrícula

```
insert into vehiculo (matricula) values ('1234ABC');
```

En este caso, como sólo vamos a introducir uno de los atributos de la tabla, debemos indicarlo después del nombre de la tabla (matricula).

Si al crear las tablas hubiese indicado modelo y matrícula como not null, este comando me habría dado error, pues sólo queremos introducir uno de los atributos.

Cambia el nombre del empleado #2 a "Carlos"

```
#En el enunciado del ejercicio dice cambiar el nombre al empleado numero 2, en nuestro caso, como el numero
#2 no existe, si no que los identificamos por el dni es aquel cuyo dni sea el 13000002B

update empleados set nombre = 'CARLOS' where dni='13000002B';
```

Como explico en el comentario del script, la clave de nuestros empleados es el dni, así que la condición que debe de cumplir el registro para que se cambie su nombre es un dni concreto, en este caso where dni='13000002B', que es el que coincide con nuestro empleado número 2.

Alejandro Sainz Sainz

Cambia el nombre del departamento #1 a "IT"

```
#cambiar el nombre del departamento 1, el único que hay, por IT

update departamento set nombre ='IT' where id=1;
```

Este es el mismo caso que el ejercicio anterior, pero aquí si tenemos departamento 1, ya que nuestra id es un campo autoincrement.

Así que la condición tras el where es id = '1'.

Alejandro Sainz Sainz

Cambia la marca a "Seat" y el modelo a "Ibiza" en el coche de empresa #2

```
#En el mismo caso que cambiar el nombre del empleado, para el vehiculo el where será la matricula

update vehiculo set marca='SEAT', modelo='IBIZA' where matricula='1234ABC';
```

Sigue siendo igual a los anteriores. Aquí lo que nos identifica al vehículo es la matrícula, y la del número 2 (recordemos que no habíamos introducido marca y modelo para este registro) es la que se muestra en la captura.

Esa matrícula será la condición detrás del where. En este comando tendremos que indicar los otros dos atributos y sus valores después del comando SET.

Elimina al empleado #4

```
#Eliminar al empleado numero 4, en este caso por dni 13000004D

delete from empleados where dni='13000004D';
```

Para eliminar al trabajador 4 tendremos que ver cual es su dni, después con el comando delete from indicaremos de que tabla queremos borrar un registro y tras el where indicaremos que el registro que se quiere eliminar es aquel cuyo dni sea igual al que se muestra en la imagen.

Elimina coche de empresa #2

```
#elimina el coche de empresa numero 2, where, matricula = 1234ABC

delete from vehiculo where matricula='1234ABC';
```

Mismo caso que el apartado anterior, sólo que en este caso indicaremos que el vehículo que se quiere eliminar es aquel cuya matrícula es igual a la que se muestra en la imagen.

Elimina todos los registros

```
delete from vehiculo;
delete from empleados;
delete from departamento;
```

Para poder borrar todos los registros de forma correcta, tendremos que eliminarlos en este orden.

Si lo hacemos de otra forma, probablemente no dará error, porque quizá intentemos eliminar registros que reciben una FK de otra tabla, y hasta que no eliminemos los registros de esa tabla no podemos eliminar los que queremos.

El orden es:

Eliminar primero los registros de vehículo: reciben un atributo de empleados como FK y si intentamos eliminar primero empleados el script da un error.

Segundo, eliminamos empleados: Una vez eliminamos registros de vehículo, que reciben atributos de empleados, podemos proceder a borrar los registros de empleados, ya no supone un conflicto borrarlos.

Por último, borramos registros de departamento, no podíamos hacerlo hasta no borrar los registros de empleado, ya que algunos de ellos reciben el atributo id de departamento como FK.

16	18:59:52	update departamento set nombre ='IT' where id=1	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.000 sec
17	18:59:52	update vehiculo set marca='SEAT', modelo='IBIZA' where matricula='1234ABC'	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.000 sec
18	18:59:52	delete from empleados where dni='13000004D'	1 row(s) affected	0.000 sec
19	18:59:52	delete from vehiculo where matricula='1234ABC'	1 row(s) affected	0.000 sec
20	18:59:52	delete from vehiculo	1 row(s) affected	0.000 sec
21	18:59:52	delete from empleados	3 row(s) affected	0.000 sec
22	18:59:52	delete from departamento	1 row(s) affected	0.000 sec

Ejemplo de la ejecución del script. Hasta la última línea, sin fallos ni warnings.

CODIGO SQL

Para que se puede observar con más claridad, incluyo aquí todo el código sql de mi script.

```
#comienzo con los comandos básicos de la base de datos
drop database if exists empresa;
create database empresa;

use empresa;

create table departamento(
  id smallint auto_increment not null,
    nombre varchar(15) not null,
    ubicacion varchar(15) not null,
    constraint id primary key (id)
);

create table empleados(
  dni char(9) unique not null,
    nombre varchar(15) not null,
    apellidos varchar(25) not null,
    fecha_nacimiento date not null,
    departamento smallint,
    constraint dni primary key (dni),
    constraint departamento foreign key(departamento) references departamento(id)
    );
```

Alejandro Sainz Sainz

```
create table vehiculo(
  matricula char(7) unique not null,
   marca varchar(15),
   modelo varchar(15),
   empleado char(9),
   constraint matricula primary key (matricula),
   constraint empleado foreign key (empleado) references empleados(dni)
);


Insert into departamento (nombre,ubicacion) values('CONTABILIDAD','TERCERA PLANTA');
insert into empleados values('13000001A','ANTONIO','GARCIA FERNANDEZ','2008-11-12',1);
insert into vehiculo values('0000AAA','RENAULT','RAPIDO','13000001A');


insert into empleados (dni, nombre, apellidos, fecha_nacimiento) values ('13000002B','PEPE','RODRIGUEZ','2000-12-12'),('13000003C','JUAN','DIAZ PEREZ','1995-11-06'),('13000004D','MARIA','GONZALEZ PI','1996-09-07');
insert into vehiculo (matricula) values ('1234ABC');


#En el enunciado del ejercicio dice cambiar el nombre al empleado numero 2, en nuestro caso, como el numero
#2 no existe, si no que los identificamos por el dni es aquel cuyo dni sea el 13000002B


update empleados set nombre = 'CARLOS' where dni='13000002B';


#cambiar el nombre del departamento 1, el único que hay, por IT


update departamento set nombre ='IT' where id=1;


#En el mismo caso que cambiar el nombre del empleado, para el vehiculo el where será la matricula
```

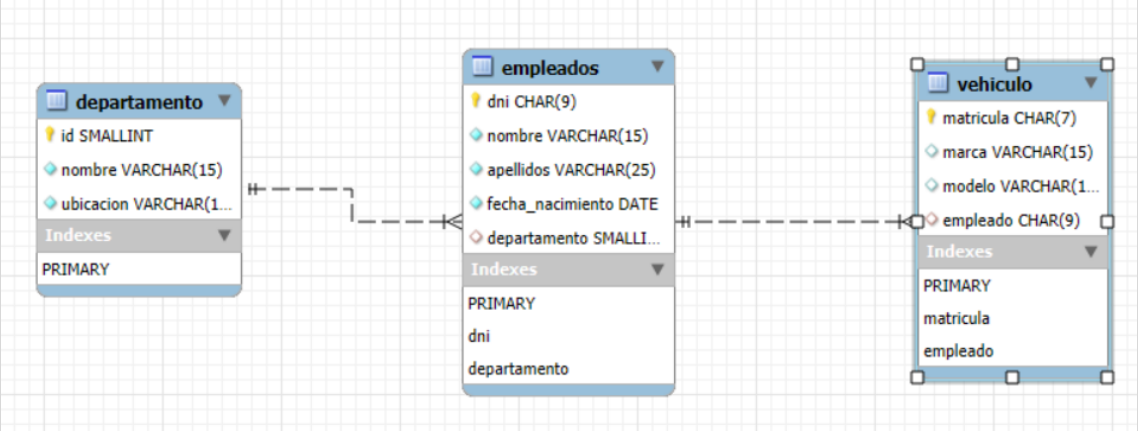
Alejandro Sainz Sainz

```
update vehiculo set marca='SEAT', modelo='IBIZA' where matricula='1234ABC';


#Eliminar al empleado numero 4, en este caso por dni 13000004D


delete from empleados where dni='13000004D';


#elimina el coche de empresa numero 2, where, matricula = 1234ABC


delete from vehiculo where matricula='1234ABC';


delete from vehiculo;
delete from empleados;
delete from departamento;


Este es todo mi código.
```

Alejandro Sainz Sainz

INGENIERIA INVERSA

Como se pide en el ejercicio, vamos a realizar un proceso de ingeniería inversa, para ver como queda el diagrama de nuestra BD.



Al ejecutar la ingeniería inversa este es el resultado.