Alquiler de Coches

Base de Datos
Zahira Zamora Camacho
Carlos Ignacio Domínguez Merchán
Enero 2013

Índice

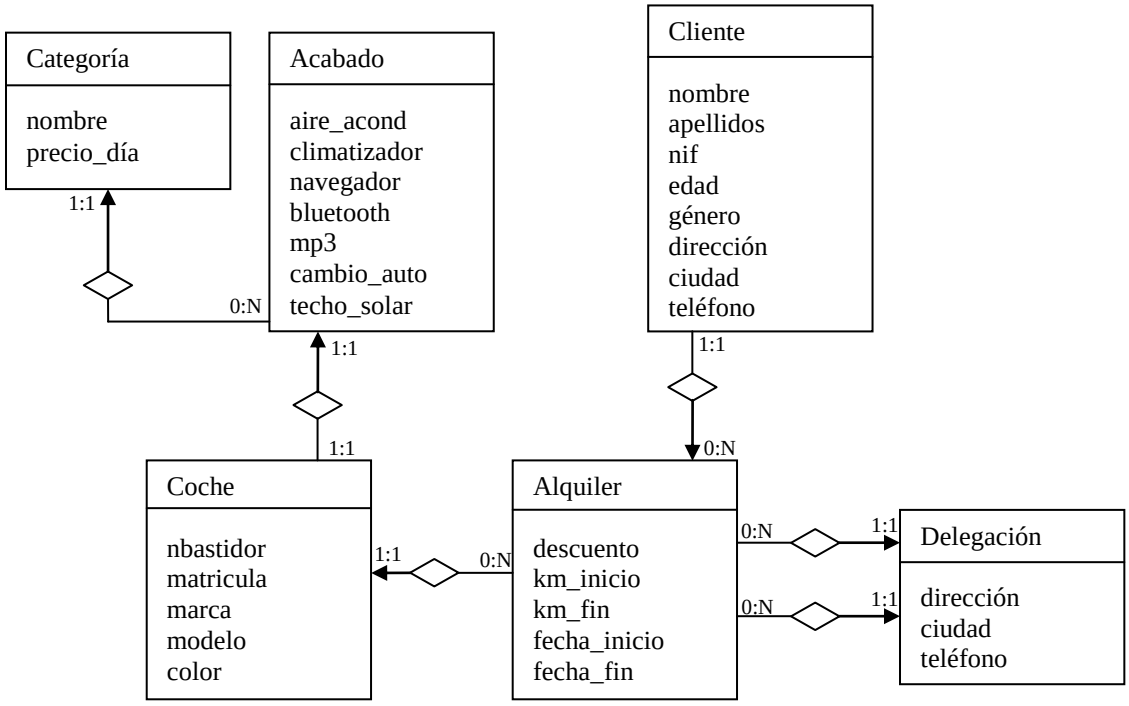
Objetivos de la Práctica

Realizar una base de datos para una compañía de alquiler de coches que desea informatizar su gestión. Para ello utilizaremos el diagrama Entidad-Relación, el paso a modelo Relacional y por último su implementación mediante lenguaje SQL del SGBD Oracle.

Desarrollo de la práctica

1.- PARTE 1

1.1.- Creación del modelo Entidad-Relación



1.2.- Paso a modelo relacional

COCHES (nbastidor, matricula, marca, modelo, color, cod_acabado)
PK (nbastidor)
UNIQUE (matricula)
FK (cod_acabado →ACABADOS.cod_acabado)
ACABADOS (cod_acabado, cod_categoria, aire_acond, climatizador, navegador, bluetooth, mp3, cambio_auto, techo_solar)
PK (cod_acabado)
FK (cod_categoria → CATEGORIAS.cod_categoria)
CATEGORIAS (cod_categoria, nombre, precio_dia)
PK (cod_categoria)
CLIENTES (cod_cliente, nombre, apellidos, nif, edad, género, dirección, ciudad, teléfono)
PK (cod_cliente)
UNIQUE (nif)

ALQUILERES (cod_alquiler, nbastidor, cod_cliente, cod_deleg_ori, cod_deleg_des, descuento, km_inicio, km_fin, fecha_inicio, fecha_fin)
PK (cod_alquiler)
FK (nbastidor → COCHES.nbastidor)
FK (cod_cliente → CLIENTES.cod_cliente)
FK (cod_deleg_ori → DELEGACIONES.cod_delegacion)
FK (cod_deleg_des → DELEGACIONES.cod_delegacion)

DELEGACIONES (cod_delegacion, dirección, ciudad, telefono)
PK (cod_delegacion)

Nota: No hemos incluido los NOT NULL en el modelo Relacional de este documento pero si lo hemos tenido en cuenta a la hora de crear las sentencias SQL.

1.3.- Creación de las tablas en la base de datos

Para la creación de la base de datos con el modelo relacional que tenemos hay que realizar los siguientes pasos:

1.- Creación de las siete tablas con sus campos.

```
CREATE TABLE nombreTabla (nombreCampo1  tipo1, … , nombreCampoN  tipoN);
```

2.- Añadimos la información de quienes son Primary Keys.

```
ALTER TABLE nombreTabla add constraint nombreTabla_pk
  primary key (nombreCampo);
```

siendo “nombreTabla_pk” el nombre que le ponemos a la restricción (constraint).

3.- Añadimos la información de quienes son Foreign Keys.

```
ALTER TABLE nombreTabla add constraint nombreTabla_nombreTablaOriginal
  foreign key (nombreCampo) references nombreTablaOriginal(nombreCampo);
```

siendo “nombreTabla_nombreTablaOriginal” el nombre que le ponemos a la restricción.

4.- Añadimos la información de qué campos son Uniques.

```
ALTER TABLE nombreTabla add constraint nombreCampo_nombreTabla_unique
  unique (nombreCampo);
```

siendo “nombreCampo_nombreTabla_unique” el nombre que le ponemos a la restricción.

5.- Añadimos la información de los campos que no pueden ser nulos.

```
ALTER TABLE nombreTabla add constraint nombreCampo_nombreTabla_not_null
  check (nombreCampo is not null);
```

siendo “nombreCampo_nombreTabla_not_null” el nombre que le ponemos a la restricción.

6.- Como no existe el tipo boolean, hemos tenido que crearlos como NUMBER(1,0), para asignarles 0 si es falso y 1 si es verdadero. Por tanto tenemos que restringir que sólo se puedan añadir ceros y unos.

```
ALTER TABLE nombreTabla add constraint nombreCampo_nombreTabla_check_bool
  check (nombreCampo=0 or nombreCampo=1);
```

7.- Para el caso concreto del género, hemos hecho que sea un carácter. Cuando el cliente sea de género femenino escribiremos “F” y cuando sea masculino “M”. Para ello hay que incluir otra restricción.

```
ALTER TABLE clientes add constraint gen_clientes_check_boolean
  check (genero='F' or genero='M');
```

Nota: Las sentencias descritas están en forma genérica para no repetir código innecesario que está incluido en los archivos adjuntos .sql.

2.- PARTE 2: Insertamos datos en la base de datos

Los datos se insertan por filas en cada tabla, es decir, en un solo comando insertamos dentro de una tabla un valor para cada campo. Los campos que puedan ser nulos, pueden no aparecer. La estructura es la siguiente:

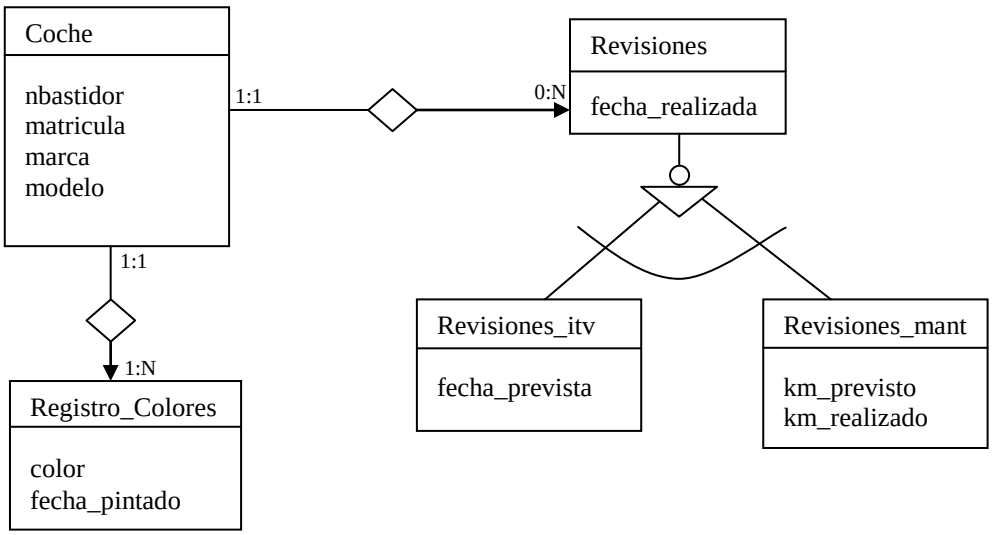
```
INSERT INTO nombreTabla (nombreCampo1, nombreCampo2,…, nombreCampoN)
  values(valor1, valor2, … , valorN);
```

Cuando el valor del campo sea un String (VARCHAR2) se introducirá entre comillas simples (‘valorCaracteres’).

3.- PARTE 3: Modificación de la base de datos

Para solucionar los nuevos requerimientos de la empresa necesitamos crear una nueva entidad “Revisiones”, que es entidad padre de “Revisiones_itv” y “Revisiones_mant”. En estas entidades podremos guardar todas las revisiones de itv y de mantenimiento que se han hecho y que están previstas hacer para cada coche.
Por otro lado se nos pide un registro de los colores de los coches actuales y antiguos ya que los coches pueden ser repintados. Para ello creamos otra entidad “Registro_Colores” en la que incluimos la fecha de cuando se ha pintado de ese color. Al crear esta entidad, sobra el campo de “color” en la tabla de “COCHES”, por tanto hay que borrarla.
Además de lo anterior, se nos pide añadir otro campo a la tabla “ACABADOS”. El nuevo campo es “calefactados” para cuando los asientos del coche son calefactados.

Además del modelo relacional anteriormente descrito, habría que añadir lo siguiente:



1.- Creamos las tres tablas: “revisiones_itv”, “revisiones_mant” y “registro_colores”.
2.- Añadimos las constraints de primary keys, foreign keys y not null.
3.- Insertamos los colores de los coches (guardados en el campo color de la tabla COCHES) en la nueva tabla registro_colores para no perder los datos antes de borrar el campo color.
4.- Borramos el campo “color” de COCHES.

```
ALTER TABLE coches DROP COLUMN color;
```

5.- Añadimos el campo “calefactados” en ACABADOS. Le restringimos a que sólo pueda ser 0 o 1 como los demás campos que son de tipo booleano y le decimos que sean campos no nulos. Para poder restringir a que sean los valores no nulos, previamente debemos darles un valor ya que hay registros dentro de la tabla y al añadir el nuevo campo se han creado como nulos.

```
ALTER TABLE acabados ADD (calefactados NUMBER(1,0));
UPDATE acabados SET calefactados=0;
ALTER TABLE acabados add constraint calef_acabados_not_null
  check (calefactados is not null);
```

4.- PARTE 4: Nuevos datos a ingresar

En este apartado únicamente tenemos que introducir los nuevos datos en la base de datos. Para ello utilizamos el mismo comando que utilizamos en el apartado 2.

```
INSERT INTO nombreTabla (nombreCampo1, nombreCampo2,…, nombreCampoN)
  values(valor1, valor2, … , valorN);
```

Vamos introduciendo por cada coche las revisiones de itv, las revisiones de mantenimiento y el registro de colores de dicho coche.
Por último le añadimos los asientos calefactados al coche con matrícula 1111ABC. Este coche tiene el cod_acabado 4, por tanto la sentencia quedaría del siguiente modo:

```
UPDATE acabados SET calefactados=1 WHERE cod_acabado=4;
```