USO DE LA CLÁUSULA “JOIN”

DAW 1 – Bases de Datos

Práctica 3.3

Consultas de bases de datos

Marco Valiente Rodríguez
Mvalienter2501@educantabria.es

Índice

Contenido

Planteamiento del Problema

En este ejercicio se nos proponen dos tipos de problemas al mismo tiempo. Uno es la de la creación de una base de datos, el segundo es la propia inserción de diferentes datos dentro de esta base de datos, donde se puede inventar aquella información la cual no sea proporcionada, y finalmente nos piden hacer consultas en base a los propios datos creados.

a. Creación de la Base de Datos en SQL

Empezamos con la tarea de crear la base de datos. Esta se va a hacer directamente en el código en MySQL usando el enunciado:

- Hay varios taxis. Cada taxi se identifica por su matrícula, marca, modelo, número de pasajeros y además queremos saber si está adaptado para clientes discapacitados.
- Cada taxi puede estar asignado a varios conductores, sin embargo, un conductor siempre conduce el mismo taxi (ya que trabajan a turnos). Un conductor se identifica por su DNI/NIE, nombre, apellidos y dirección.
- Los taxis hacen carreras. Cada carrera se identifica por su origen, destino y precio. También queremos saber si la carrera se realiza durante el turno de noche.

A la hora de crearlo, solo existe una complicación, que es la de un conductor siempre conduciendo el mismo taxi. Aunque en un primer momento parece que es una indicación que la Primary Key de taxi va insertada como una Foreign Key en conductores, esto no tiene por qué ser el caso ya que se trata de una relación que se podría conectar de por sí con una tabla intermedia que es carrera.

Esto sería una primera asunción, pero, posteriormente, se puede realizar la deducción de que no existe de forma directa del enunciado una petición para conectar carrera y conductor, sino que es algo que está únicamente relacionado a taxi. Así que taxi conecta tanto a conductor y como carrera, y estos dos últimos son los portadores de matrícula como su Foreign Key.

```
DROP DATABASE IF EXISTS taxisEj3;
CREATE DATABASE IF NOT EXISTS taxisEj3;
USE taxisEj3;

DROP TABLE IF EXISTS taxi,
                     conductor,
                     carrera;

CREATE TABLE taxi (
    matricula CHAR(7),
    marca VARCHAR(30) NOT NULL,
    modelo VARCHAR(40) NOT NULL,
    num_pasajeros TINYINT NOT NULL,
    adaptado_discapacitados BOOL,
```

```
    PRIMARY KEY (matricula)
);

CREATE TABLE conductor (
    dni_nie CHAR(9),
    nombre VARCHAR(30) NOT NULL,
    apellidos VARCHAR(50) NOT NULL,
    direccion VARCHAR(50) NOT NULL,
    matricula CHAR(7),
    FOREIGN KEY (matricula) REFERENCES taxi (matricula) ON DELETE
CASCADE,
    PRIMARY KEY (dni_nie)
);

CREATE TABLE carrera (
    id SMALLINT,
    matricula CHAR(7),
    origen VARCHAR(50) NOT NULL,
    destino VARCHAR(50) NOT NULL,
    precio DECIMAL(6,2) NOT NULL,
    turno_noche BOOL,
    FOREIGN KEY (matricula) REFERENCES taxi (matricula) ON DELETE
CASCADE,
    PRIMARY KEY (id)
);
```

Se escogen restricciones generosas para la base de datos, pero al menos, que sean razonables, y a partir de estas, completamos las diferentes columnas de cada tabla.

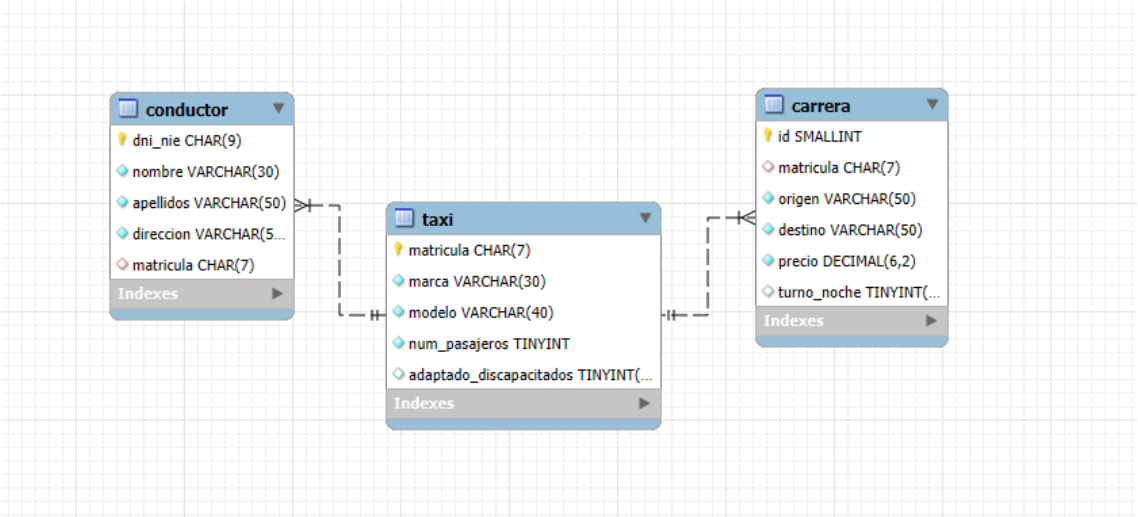
#	Time	Action	Message
2	11:19:01	CREATE DATABASE IF NOT EXISTS taxisEj3	1 row(s) affected
3	11:19:01	USE taxisEj3	0 row(s) affected
4	11:19:01	DROP TABLE IF EXISTS taxi, conductor, carrera	0 row(s) affected,
5	11:19:01	CREATE TABLE taxi (matricula CHAR(7), marca VARCHAR(30) NOT NULL, modelo VARCHAR(40) NOT NULL, num_pasajeros TINYINT NO...	0 row(s) affected
6	11:19:01	CREATE TABLE conductor (dni_nie CHAR(9), nombre VARCHAR(30) NOT NULL, apellidos VARCHAR(50) NOT NULL, direccion VARCHAR...	0 row(s) affected
7	11:19:01	CREATE TABLE carrera (id SMALLINT, matricula CHAR(7), origen VARCHAR(50) NOT NULL, destino VARCHAR(50) NOT NULL, precio ...	0 row(s) affected

Aunque en calidad de imagen no se puede apreciar, lo que se puede comprobar es que no hay errores ni advertencias, con la excepción de que estaba intentando hacer un drop de algo que no existe.

b. Revisión Ingeniería Reversa

Al estar haciendo una tabla desde cero, necesitamos comprobar si efectivamente lo que hemos diseñado corresponde a lo que nosotros deseamos, que es, la tabla general de la base de datos.

Realizamos la ingeniería reversa:



El resultado entra dentro de lo deseado. Una base de datos donde taxi esté relacionado con conductor y con carrera, y de la cual no se necesita añadir ninguna relación más entre estas.

c. Inserción de Datos

Se nos pide, a continuación, la inserción de la siguiente tabla de datos, reflejada dentro de nuestra propia base de datos.

Carrera (ID)	Taxi (Matrícula)	Conductor (DNI/NIE)
	1234 CHB	
1	3243 HGF	11111111H
2	2345 CKD	22222222J
3	2345 CKD	22222222J
4	2378 FHG	33333333P
5	2378 FHG	33333333P
6	7069 DLV	44444444A
7	7069 DLV	44444444A
8	7069 DLV	44444444A
		55555555B

De esta tabla, vamos a hacer las siguientes asunciones, debido a su ambigüedad:

1. Si una tabla no tiene datos relacionados a su alrededor, dicha entidad existe dentro de nuestra base de datos, pero no tiene ninguna relación en este momento.
2. Solo existen los datos introducidos en esta tabla, y, por tanto, no añadiremos entidades que no se encuentren reflejadas en estos datos.
3. El resto de datos vacíos son de elección propia.

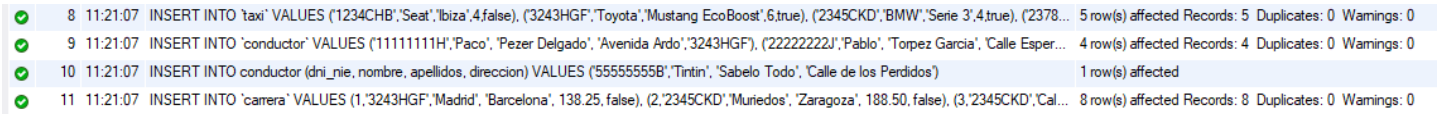
A partir de esto creamos nuestro código en MySQL.

```
INSERT INTO `taxi` VALUES ('1234CHB','Seat','Ibiza',4,false),
('3243HGF','Toyota','Mustang EcoBoost',6,true),
('2345CKD','BMW','Serie 3',4,true),
('2378FHG','Seat','Arona',6,true),
('7069DLV','Seat','Leon',4,false);

INSERT INTO `conductor` VALUES ('11111111H','Paco', 'Pezer Delgado',
'Avenida Ardo','3243HGF'),
('22222222J','Pablo', 'Torpez Garcia', 'Calle Esperanza','2345CKD'),
('33333333P','Pepe', 'Gutierrez Lopez', 'Calle Casablanca','2378FHG'),
('44444444A','Chavo', 'Blanco', 'Avenida Rocosa','7069DLV');
INSERT INTO conductor (dni_nie, nombre, apellidos, direccion) VALUES
('55555555B','Tintin', 'Sabelo Todo', 'Calle de los Perdidos');
```

```
INSERT INTO `carrera` VALUES (1,'3243HGF','Madrid', 'Barcelona', 138.25,
false),
(2,'2345CKD','Muriedos', 'Zaragoza', 188.50, false),
(3,'2345CKD','Calle Del Valle', 'Avenida Calle del Valle', 29.33,false),
(4,'2378FHG','Loscuales', 'Esperadito', 38.25,true),
(5,'2378FHG','Ursumiminto', 'Lapleluquin', 452.34,true),
(6,'7069DLV','Madrid', 'Paris', 3255.99,false),
(7,'7069DLV','Paris', 'Madrid', 3922.45,true),
(8,'7069DLV','Barcelona', 'Madrid', 127.65,false);
```

Tenemos su ejecución que resulta en:



8	11:21:07	INSERT INTO `taxi` VALUES ('1234CHB','Seat','Ibiza',4,false), ('3243HGF','Toyota','Mustang EcoBoost',6,true), ('2345CKD','BMW','Serie 3',4,true), ('2378...	5 row(s) affected	Records: 5	Duplicates: 0	Warnings: 0
9	11:21:07	INSERT INTO `conductor` VALUES ('11111111H','Paco', 'Pezer Delgado', 'Avenida Ardo','3243HGF'), ('22222222J','Pablo', 'Torpez Garcia', 'Calle Esper...	4 row(s) affected	Records: 4	Duplicates: 0	Warnings: 0
10	11:21:07	INSERT INTO conductor (dni_nie, nombre, apellidos, direccion) VALUES ('55555555B','Tintin', 'Sabelo Todo', 'Calle de los Perdidos')	1 row(s) affected			
11	11:21:07	INSERT INTO `carrera` VALUES (1,'3243HGF','Madrid', 'Barcelona', 138.25, false), (2,'2345CKD','Muriedos', 'Zaragoza', 188.50, false), (3,'2345CKD','Cal...	8 row(s) affected	Records: 8	Duplicates: 0	Warnings: 0

Nos habla de diferentes números de filas modificadas, y nos lo marca en verde, por lo que se puede deducir que se han realizado los cambios de forma adecuada. Como único dato a comentar, para elaborar un INSERT de conductor, al no tener una tabla llena, la de matrícula, optamos por crear una nueva línea de inserción para tener bien controlado la información que cada contacto tiene.

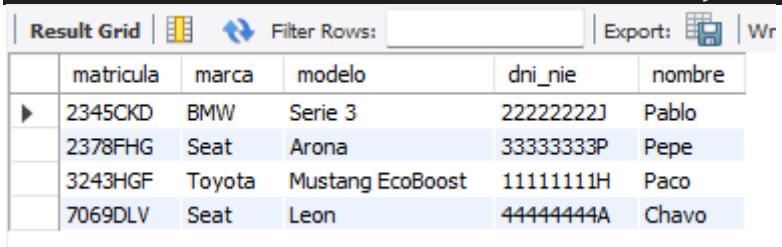
d. Consultas de datos

Como parte del ejercicio, ahora se nos pide hacer consultas a los datos usando el comando SELECT para seleccionar los datos, y el comando JOIN para unir las diferentes tablas y poder realizar de forma apropiada las consultas.

1. Muestra la matrícula, marca y modelo de todos los coches que tienen asignado un conductor (y los datos de los conductores -DNI/NIE y nombre-).

Comenzamos con la realización de la primera consulta. Para realizar esta consulta, tenemos que mezclar dos tablas que están directamente conectadas.

```
SELECT t.matricula, t.marca, t.modelo, c.dni_nie, c.nombre FROM taxi t
JOIN conductor c ON t.matricula = c.matricula;
```



Result Grid | Filter Rows: | Export: | Wr

matricula	marca	modelo	dni_nie	nombre
2345CKD	BMW	Serie 3	22222222J	Pablo
2378FHG	Seat	Arona	33333333P	Pepe
3243HGF	Toyota	Mustang EcoBoost	11111111H	Paco
7069DLV	Seat	Leon	44444444A	Chavo

La tabla nos muestra de forma correcta los datos unidos con JOIN.

2. Muestra el origen y destino de todos los servicios, así como la matrícula del taxi que se utilizó y si estaba adaptado para clientes discapacitados.

Ahora nos toca realizar un JOIN entre las tablas de carrera y coche.

```
SELECT c.origen, c.destino, t.matricula, t.adaptado_discapacitados FROM
taxi t JOIN carrera c ON t.matricula = c.matricula;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

origen	destino	matricula	adaptado_discapacitados
Muriedos	Zaragoza	2345CKD	1
Calle Del Valle	Avenida Calle del Valle	2345CKD	1
Loscuales	Esperadito	2378FHG	1
Ursumiminto	Lapleluquin	2378FHG	1
Madrid	Barcelona	3243HGF	1
Madrid	Paris	7069DLV	0
Paris	Madrid	7069DLV	0
Barcelona	Madrid	7069DLV	0

Conseguimos mostrar las 8 carreras con sus respectivos destinos y matrículas.

3. Lista todos los nombres de conductores y DNI/NIE y sus respectivos coches (marca, modelo, matrícula y número de pasajeros).

Repetimos la consulta, una muy similar a la original.

```
SELECT c.nombre, c.dni_nie, t.marca, t.modelo, t.matricula,
t.num_pasajeros FROM taxi t JOIN conductor c ON t.matricula =
c.matricula;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

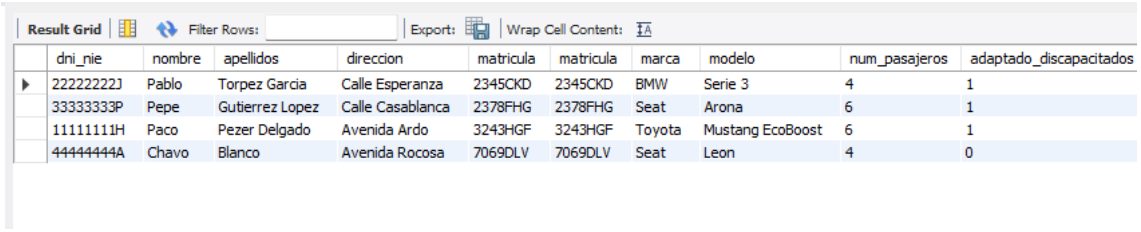
nombre	dni_nie	marca	modelo	matricula	num_pasajeros
Pablo	22222222J	BMW	Serie 3	2345CKD	4
Pepe	33333333P	Seat	Arona	2378FHG	6
Paco	11111111H	Toyota	Mustang EcoBoost	3243HGF	6
Chavo	44444444A	Seat	Leon	7069DLV	4

Los datos son recolectados de forma exitosa.

4. Enumera todos los detalles de todos los taxis y todos los conductores.

Necesitamos usar el asterisco para poder llamar a todos los elementos de la tabla.

```
SELECT c.*, t.* FROM taxi t JOIN conductor c ON t.matricula =
c.matricula;
```



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

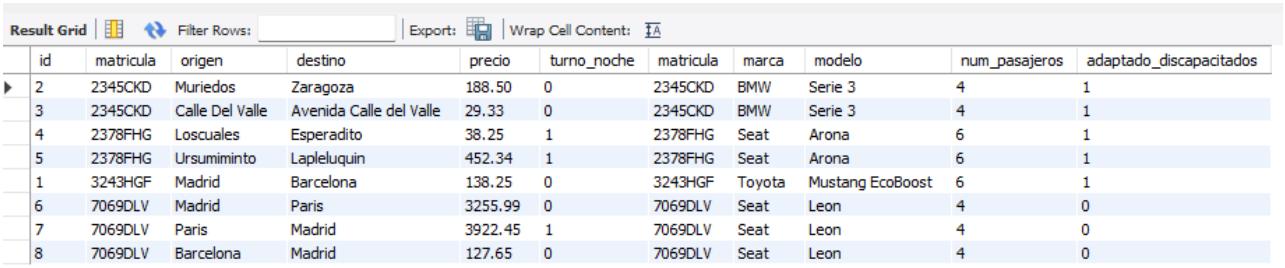
dni_nie	nombre	apellidos	direccion	matricula	matricula	marca	modelo	num_pasajeros	adaptado_discapacitados
22222222J	Pablo	Torpez Garcia	Calle Esperanza	2345CKD	2345CKD	BMW	Serie 3	4	1
33333333P	Pepe	Gutierrez Lopez	Calle Casablanca	2378FHG	2378FHG	Seat	Arona	6	1
11111111H	Paco	Pezer Delgado	Avenida Ardo	3243HGF	3243HGF	Toyota	Mustang EcoBoost	6	1
44444444A	Chavo	Blanco	Avenida Rocosa	7069DLV	7069DLV	Seat	Leon	4	0

La tabla refleja, gracias al asterisco, a todos los datos.

5. Muestra todos los detalles de todos los servicios y todos los taxis.

Se nos pide lo mismo, pero para carrera.

```
SELECT c.*, t.* FROM taxi t JOIN carrera c ON t.matricula = c.matricula;
```



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

id	matricula	origen	destino	precio	turno_noche	matricula	marca	modelo	num_pasajeros	adaptado_discapacitados
2	2345CKD	Muriedos	Zaragoza	188.50	0	2345CKD	BMW	Serie 3	4	1
3	2345CKD	Calle Del Valle	Avenida Calle del Valle	29.33	0	2345CKD	BMW	Serie 3	4	1
4	2378FHG	Loscuales	Esperadito	38.25	1	2378FHG	Seat	Arona	6	1
5	2378FHG	Ursumiminto	Lapleluquin	452.34	1	2378FHG	Seat	Arona	6	1
1	3243HGF	Madrid	Barcelona	138.25	0	3243HGF	Toyota	Mustang EcoBoost	6	1
6	7069DLV	Madrid	Paris	3255.99	0	7069DLV	Seat	Leon	4	0
7	7069DLV	Paris	Madrid	3922.45	1	7069DLV	Seat	Leon	4	0
8	7069DLV	Barcelona	Madrid	127.65	0	7069DLV	Seat	Leon	4	0

Con el uso del asterisco, podemos ver, de nuevo, todas las tablas.

e. Revisión y Validación

Quiero comentar, que, en respecto a esta actividad, aunque todo funciona correctamente y está hecho bien, esto no fue así desde un primer momento, sino que hubo dificultades y tuve que rehacer partes de este documento de forma retroactiva.

En un primer momento, tomamos la hipótesis de que la mejor forma de solucionarlo es la de tomar la tabla de carrera como una intermedia entre conductor y carrera, aunque esta no tiene nada que la una a conductor. El motivo de este fallo tiene que ver por el uso de la tabla de datos como asunción que, en algún momento, sería posible obtener esos datos, y que, por tanto, toda esa información se pudiese encontrar dentro de la propia tabla de carrera.

A la hora de realizar consulta de datos, fui capaz de lograr lo que cada ejercicio resolvía, pero los comandos eran bastante más complejos que los que en esta reinterpretación son, por lo que igual no es prudente realizar dichos cambios en la primera práctica de consultas enlazadas de MySQL.

Dicho esto, los cambios aquí se pueden verificar como correctos.

Bibliografía

- Recurso del curso de Desarrollo de Aplicaciones Web, Documento de Consultas Básicas Avanzadas.