

Treballant amb Laravel i claus foranes amb un exemple des de zero

1- Hem de desenvolupar una aplicació per un botiga on-line. L'aplicació ha de permetre emmagatzemar dins d'una base de dades la següent informació sobre els **clients** de la botiga:

- dni del client
- nom complet del client
- email del client

També s'haurà d'emmagatzemar la següent informació sobre els **productes** que estan a la venda:

- identificador del producte
- nom del producte
- preu del producte

Finalment, cada cop que un **client compra** un **producte** s'haurà d'emmagatzemar:

- Quantitat d'unitats de producte comprades.
- Data en la qual el client va comprar el producte.

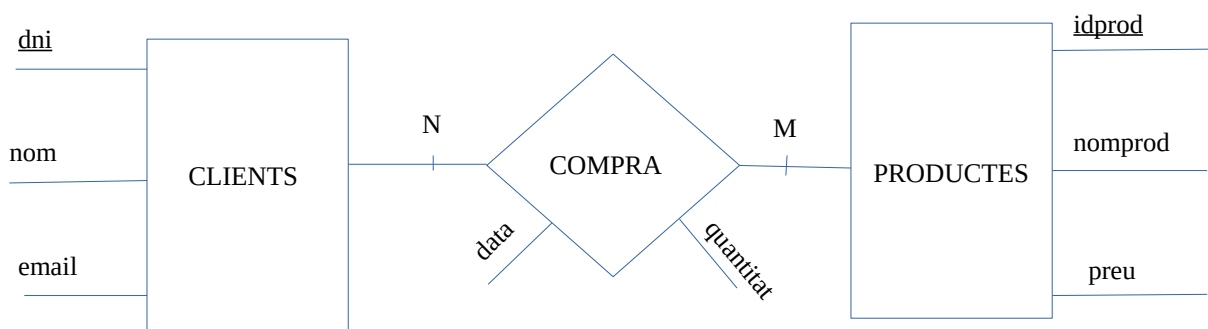
A partir d'aquesta informació hem de trobar:

- El model Entitat-Relació associat a la base de dades de l'aplicació
- El model Relacional associat al model Entitat-Relació

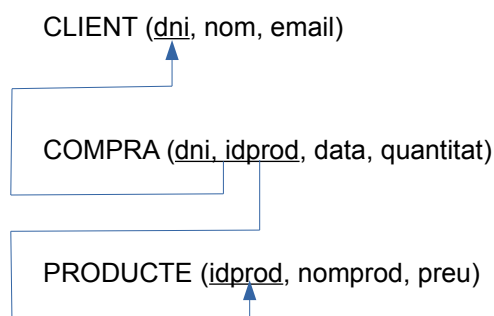
Un cop realitzada aquesta fase d'anàlisi, haurem de:

- Crear, utilitzant scripts SQL, una base de dades per la botiga i un usuari que tingui tots els permisos necessaris per accedir a la base de dades
- Utilitzant el framework de Laravel:
 - Ens connectarem a la base de dades
 - Crearem les taules que haurem deduït que són necessàries a partir de l'anàlisi realitzat previament.
 - Desenvoluparem l'aplicació que ens hauria de permetre la:
 - Creació, visualització, modificació i esborrament de clients.
 - Creació, visualització, modificació i esborrament de productes.
 - Creació, visualització, modificació i esborrament del registre de compres.

2- El model Entitat-Relació associat a la base de dades de l'aplicació serà aquest:



3- Del model Entitat-Relació anterior trobarem el següent model Relacional:



4- Ara que ja hem fet l'anàlisi ja podem començar el desenvolupament de l'aplicació. Per començar, crearem un script SQL de nom **botiga.sql** dins del directori **vagrant** de la màquina virtual **m07uf3** amb la qual hem estat treballant durant la UF3. El contingut del script serà aquest:

```
use mysql;
create user 'adm'@'localhost' identified by "FjeClot23@";
create database botiga;
use botiga;
grant all privileges on botiga.* to 'adm'@'localhost';
```

Aquest script, crearà un usuari de nom **adm** amb tots els permisos necessaris per treballar amb la base de dades **botiga**. A continuació accedirem des de dins del directori **vagrant** a MySQL executant:

```
sudo mysql -u root -p
```

i un cop dins de MySQL, executarem el script SQL que hem creat:

```
source ./botiga.sql
```

Finalment, comprovarem que s'ha creat l'usuari **adm** i la base de dades **botiga.sql**. Executa:

show databases; → Comprova que ha aparegut la base de dades **botiga**.

show grants for 'adm'@'localhost'; → Comprova que **adm** té tots els privilegis per treballar amb **botiga**.

5- Entra al directori **/var/www/html** de la màquina virtual i crea un projecte de **laravel** executant:

```
laravel new botiga
```

6- Accedeix a la carpeta **botiga**. Modifica el fitxer **.env**. Fes que la connexió amb el servidor **MySQL** tingui els següents paràmetres:

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=botiga
DB_USERNAME=adm
DB_PASSWORD="FjeClot23@"
```

7- Ara crearem els fitxers de **Models**, **Migrations** i **Controllers** al mateix temps per les **3** taules que necessita la nostra aplicació: **clients**, **productes** i **compra**. Executa:

```
php artisan make:model Client --migration --controller --resource
php artisan make:model Producte --migration --controller --resource
php artisan make:model Compra --migration --controller --resource
```

i comprova ara que:

- Dins **app/Models** s'han creat els fitxers:
 - Client.php
 - Compra.php
 - Producte.php
- Dins **app/Http/Controllers** s'han creat els fitxers:
 - ClientController.php
 - CompraController.php
 - ProducteController.php
- Dins **database/migrations** s'han creat els fitxers:
 - xxxx_xx_xx_xxxxxx_create_clients_table.php
 - xxxx_xx_xx_xxxxxx_create_compras_table.php
 - xxxx_xx_xx_xxxxxx_create_productes_table.php

A on **xxxx_xx_xx_xxxxxx** és l'any, mes, dia i hora de creació del fitxer.

8- Editem `xxxx_xx_xx_xxxxxx_create_clients_table.php`. Aquí és molt important tenir en compte l'anàlisi realitzat als apartats 2 i 3. El nou codi de la funció `up()` serà:

```
public function up()
{
    Schema::create('clients', function (Blueprint $table) {
        $table->string('dni',9)->unique();
        $table->primary('dni');
        $table->string('nom',80);
        $table->string('email',50);
    });
}
```

En aquest cas és important adonar-se que hem definit el camp **dni** com un **string de 9 caràcters**, que és la **clau primària** i que **no pot repetir-se**. Això després s'haurà de tenir en compte en el model. D'altra banda, el camp **nom** serà un **string de 80 caràcters** i el camp **email** un string de **50 caràcters**. Hem **eliminitat** els **timestamps** perquè no els farem servir.

9- Editem `xxxx_xx_xx_xxxxxx_create_productes_table.php`. Novament, és molt important tenir en compte l'anàlisi realitzat als apartats 2 i 3. El nou codi de la funció `up()` serà:

```
public function up()
{
    Schema::create('productes', function (Blueprint $table) {
        $table->string('idprod',5)->unique();
        $table->primary('idprod');
        $table->string('nomprod',50);
        $table->float('preu');
    });
}
```

En aquest cas és important adonar-se que hem definit el camp **idprod** com un **string de 5 caràcters**, que és la **clau primària** i que **no pot repetir-se**. Això després s'haurà de tenir en compte en el model. D'altra banda, el camp **nom** serà un **string de 80 caràcters** i el camp **preu** un **float**. Hem **eliminitat** els **timestamps** perquè no els farem servir.

10- Editem `xxxx_xx_xx_xxxxxx_create_compras_table.php`. Novament, és molt important tenir en compte l'anàlisi realitzat als apartats 2 i 3. El nou codi de la funció `up()` serà:

```
public function up()
{
    Schema::create('compras', function (Blueprint $table) {
        $table->string('dni',9);
        $table->foreign('dni')->references('dni')->on('clients')->onDelete('cascade')->onUpdate('cascade');
        $table->string('idprod',5);
        $table->foreign('idprod')->references('idprod')->on('productes')->onDelete('cascade')->onUpdate('cascade');
        $table->primary(['dni','idprod']);
        $table->date('data');
        $table->integer('quantitat');
    });
}
```

En aquest cas és important adonar-se que:

- Hem definit el camp **dni** de **compras** com a camp **forani** relacionat (referenciat) amb el camp **dni** de **clients**.
- Hem definit el camp **idprod** de **productes** com a camp **forani** relacionat (referenciat) amb el camp **idprod** de **productes**.
- Si actualitzem o esborrem un registre dins de la taula **clients**, s'esborren o actualitzen en cascada tots els registres associats de la taula **compras**.
- Si actualitzem o esborrem un registre dins de la taula **productes**, s'esborren o actualitzen en cascada tots els registres associats de la taula **compras**.
- La **clau primària** de la taula **compras** està formada pels camps **dni** i **idprod**.
- Hem definit els camps **data** de tipus **date** i **quantitat** de tipus **integer**.

11- Ara farem uns canvis inicials dels models. Per començar modificarem la classe **Client** del model **app/Models/Client.php**. Bàsicament indicarem els camps que s'han d'omplir des del formulari i el nom de la clau primària. El codi de la classe **Client** serà aquest:

```
class Client extends Model
{
    use HasFactory;
    protected $primaryKey = 'dni';
    public $incrementing = false;
    protected $keyType = 'string';
    protected $fillable = [
        'dni',
        'nom',
        'email'
    ];
}
```

12- Ara la classe **Producte** del model **app/Models/Producte.php**. El codi inicial serà aquest:

```
class Producte extends Model
{
    use HasFactory;
    protected $primaryKey = 'idprod';
    public $incrementing = false;
    protected $keyType = 'string';
    protected $fillable = [
        'idprod',
        'nomprod',
        'preu'
    ];
}
```

13- Finalment la classe **Compra** del model **app/Models/Compra.php**. El codi inicial serà aquest:

```
class Compra extends Model
{
    use HasFactory;
    protected $primaryKey = ['dni', 'idprod'];
    public $incrementing = false;
    protected $fillable = [
        'data',
        'quantitat'
    ];
}
```

14- Ara ja podem crear les taules. Executa dins de la carpeta **botiga**:

php artisan migrate

i comprova que s'han creat les 3 taules clients, compres i productes dins de la base de dades botiga.

15- Ara cal desenvolupar els mètodes dels controladors i les vistes. El propòsit d'aquesta sessió era mostrar com treballar amb claus forànies i laravel i per tant ho deixem en aquest punt.