**CLASE FS 23-04-2019**

**ELOQUENT**

Para trabajar con bases de datos en Laravel no necesitamos escribir SQL de forma manual. Laravel nos permite interactuar con la base de datos a un nivel mucho más alto a través del ORM Eloquent. ORM tenemos la posibilidad de trabajar con los datos que hay en las bases de datos por medio de objetos

Utilizando Eloquent podemos trabajar con modelos, que son clases que representan nuestras tablas en la base de datos y nos proveen de métodos con los que podemos interactuar en una interfaz más “elocuente” y orientada a objetos.

**Realizar consultas**

Podemos utilizar los modelos para hacer consultas a la base de datos. Utilizando el método all() obtenemos todo el contenido de la tabla:

***$productos = Producto::all();***

Esta consulta nos devuelve un objeto JSON que necesita iterarse con un foreach:

*[{“atributo1a”:valor1a,”atributo2a”:valor2a},{“atributo1b”:”valor1b”,”atributo2b:valor2b”}]*

Podemos retornar un resultado dependiendo de su id mediante el método find():

***$productos = Producto::find(1);***

Esta consulta nos devuelve un objeto que no necesita iterarse:

*{“atributo1”:valor1,”atributo2”:valor2}*

También podemos encadenar métodos del constructor de consultas para obtener resultados más específicos:

**$productos = Producto::where('nombre','Manzana')->value('id');**

Unas de las instrucciones basicas de este son **get()** que nos regresa todos los registros de la BD y **first()** que nos regresa el primer registro de una selección.

**$productos = Producto::where('precio','>',400)->get();** //Requiere foreach

**$productos = Producto::where('precio','>',400)->first();** //No requiere foreach

**Sumatoria:**

$valor\_total = Producto::where('precio','>',1000)->where('estado\_id',1)->sum('precio');

**Promedio:**

$valor\_total = Producto::where('precio','>',1000)->where('estado\_id',1)->avg('precio');

**Select:**

$productos = Producto::where('precio','>',1000)->select('nombre','cantidad')->get();

**Facade DB**

use Illuminate\Support\Facades\DB;

$datos = DB::table('movimientos')->where('empresa\_id',$empresa)

->where('cuenta','LIKE','4%')->where('periodo\_id',$periodo)->get();

**orWhere:** O

**orderBy:** Ordenar por

whereNotNull

whereBetween('fecha', [$from, $to])

groupBy

**DB:raw():**

Si queremos usar alguna función de mysql, podemos usar DB::raw()

$usuarios = User::orderBy('nombres','asc')->select(DB::raw('CONCAT(nombres," ",apellidos," - ",email) as datos'),'id')->get();

**JOINS**

*$usuarios = User::join("perfiles","users.perfil\_id","=","perfiles.id")*

*->get();*

*$usuarios = User::join("perfiles","users.perfil\_id","=","perfiles.id")*

*->where('users.ciudad\_id','=',1)*

*->get();*

*$usuarios = User::join("perfiles","users.perfil\_id","=","perfiles.id")*

*->select("users.nombres","perfiles.nombre")*

*->get();*

$ciudades = Ciudad::join('departamentos','ciudades.departamento\_id','=','departamentos.id')

->select('ciudades.nombre',DB::raw("(SELECT nombre FROM departamentos WHERE id=ciudades.departamento\_id) as nombre\_departamento"))->get();

**Ejemplo: Buscar un término**

$search = 'Palabra a buscar';

$posts = Post::from('posts as a')

->where(function ($query) use ($search) {

$query = $query->orWhere('a.titulo','like',"%$search%");

$query = $query->orWhere('a.descripcion','like',"%$search%");

$query = $query->orWhere('a.tags','like',"%$search%");

});

$posts = $posts->where('a.estado','=',1)

->get();