Department of Information Technology
IT 350 - Web Technologies
Fall - 2021

# Databases and Eloquent

## Lecture Six

Rebin M. Ahmed
rebin.mohammed@tiu.edu.iq

## Previous Lecture

- Collecting and Handling User Data
- Validation
- Blade Templating
- Template Inheritance

## Contents

## Introduction

Laravel makes interacting with databases extremely simple across a variety of database backends using either **raw SQL**, the fluent **query builder**, and the **Eloquent ORM**. Currently, Laravel supports four databases:

- ➔ MySQL 5.6+
- ➔ PostgreSQL 9.4+
- ➔ SQLite 3.8.8+
- ➔ SQL Server 2017+

## Introduction

- Raw SQL

```
DB::select('select * from users where id = :id', ['id' => 1]);
```

- Query Builder

```
DB::table('users')->where('name', 'John')->first();
```

- Eloquent ORM.

```
App\Flight::where('active', 1)
```

## configuration

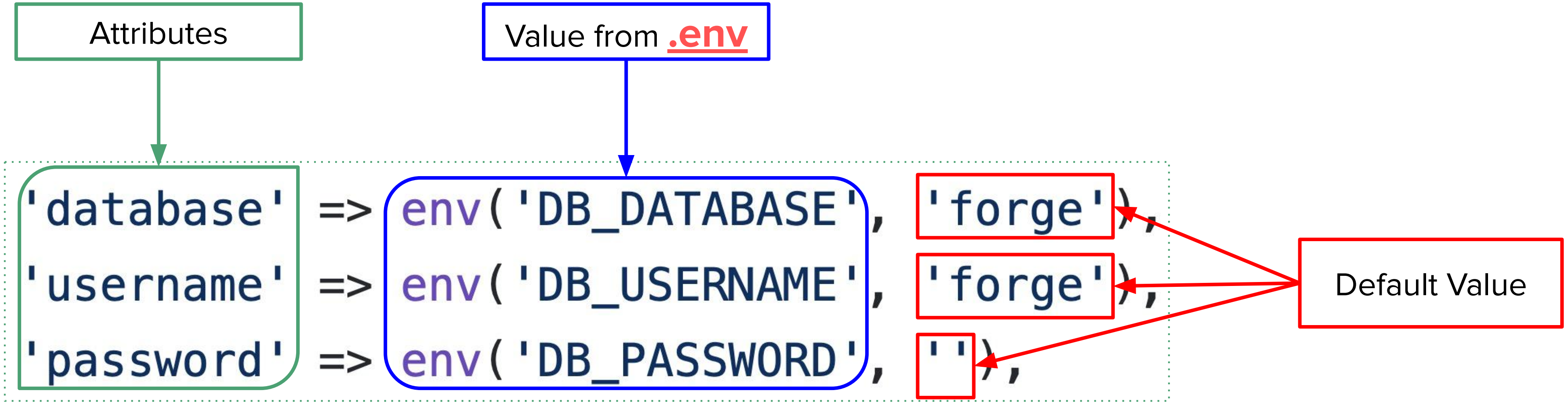
The database configuration for your application is located at **config/database.php**. In this file you may define all of your database connections, as well as specify which connection should be used by default. Examples for most of the supported database systems are provided in this file.

## configuration

```
'mysql' => [
    'driver' => 'mysql',
    'url' => env('DATABASE_URL'),
    'host' => env('DB_HOST', '127.0.0.1'),
    'port' => env('DB_PORT', '3306'),
    'database' => env('DB_DATABASE', 'forge'),
    'username' => env('DB_USERNAME', 'forge'),
    'password' => env('DB_PASSWORD', ''),
    'unix_socket' => env('DB_SOCKET', ''),
    'charset' => 'utf8mb4',
    'collation' => 'utf8mb4_unicode_ci',
    'prefix' => '',
    'prefix_indexes' => true,
    'strict' => true,
    'engine' => null,
    'options' => extension_loaded('pdo_mysql') ? array_filter([
        PDO::MYSQL_ATTR_SSL_CA => env('MYSQL_ATTR_SSL_CA'),
    ]) : [],
],
```

## configuration

Note that all the attributes in config/database.php have default values, in case you didn't define it in the .env file, the default value will be used.



```
'database' => env('DB_DATABASE', 'forge'),
'username' => env('DB_USERNAME', 'forge'),
'password' => env('DB_PASSWORD', ''),
```

## configuration

Database configurations from .env file

| | |
|---|---|
| DB_CONNECTION=mysql | (Type of the connection as shown in slide 5) |
| DB_HOST=127.0.0.1 | (Database Host IP Address, usually it is 127.0.0.1) |
| DB_PORT=3306 | (Mysql Port, usually it is 3306) |
| DB_DATABASE=laravel | (Name of your database) |
| DB_USERNAME=root | (Username of  your database) |
| DB_PASSWORD= | (Password of  your database, if there is password) |

# Raw SQL

Once you have configured your database connection, you may run queries using the DB facade. The DB facade provides methods for each type of query: **select**, **update**, **insert**, **delete**, and statement.

```
DB::select('select * from users where id = :id', ['id' => 1]);
```

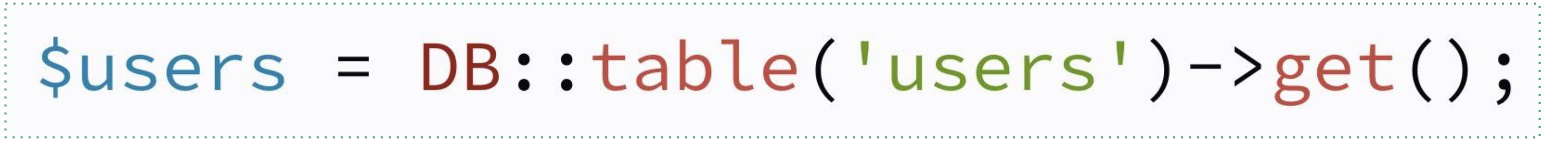
```
DB::update('update users set votes = 100 where name = ?', ['John']);
```

[illegible]

```
DB::delete('delete from users');
```

## Query Builder

Laravel's database query builder provides a convenient, fluent interface to creating and running database queries. It can be used to perform most database operations in your application and works on all supported database systems.



```
$users = DB::table('users')->get();
```

## Query Builder

You may use the table method on the DB facade to begin a query. The table method returns a fluent query builder instance for the given table, allowing you to chain more constraints onto the query and then finally get the results using the get method:

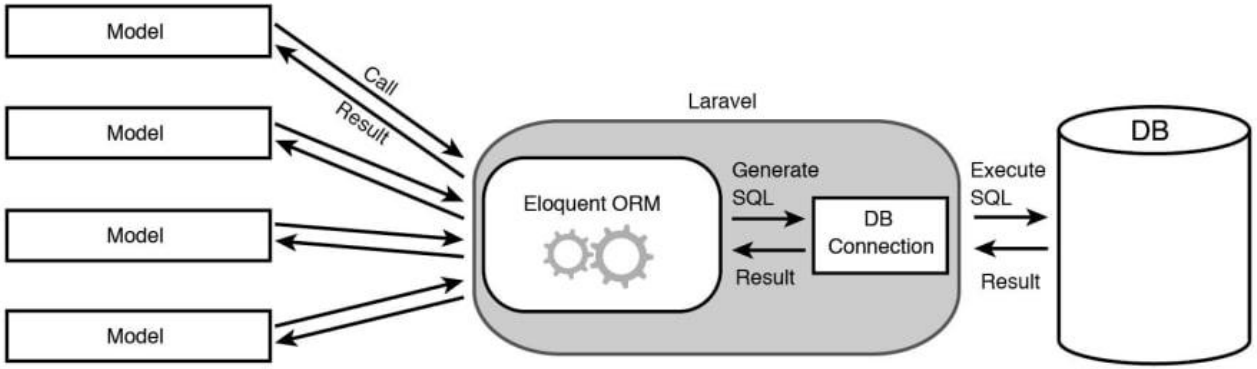
```
$users = DB::table('users')->get();
```

You may access each column's value by accessing the column as a property of the object.

```
foreach ($users as $user) {
    echo $user->name;
}
```

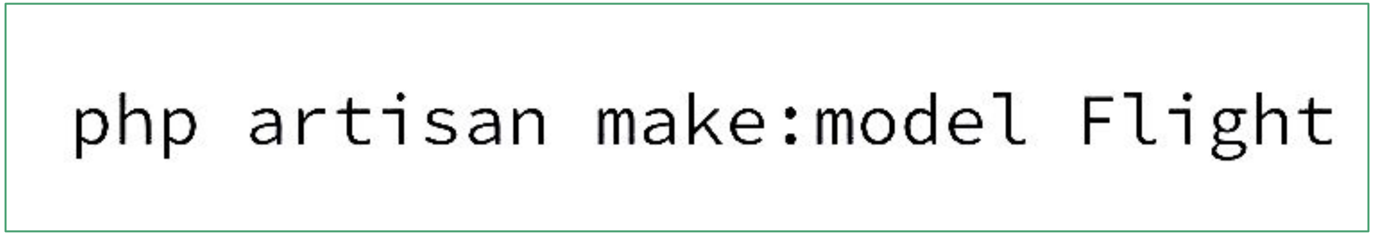
## Eloquent ORM

The Eloquent ORM included with Laravel provides a beautiful, simple ActiveRecord implementation for working with your database. Each database table has a corresponding "**Model**" which is used to interact with that table. Models allow you to query for data in your tables, as well as insert new records into the table.



## Eloquent ORM

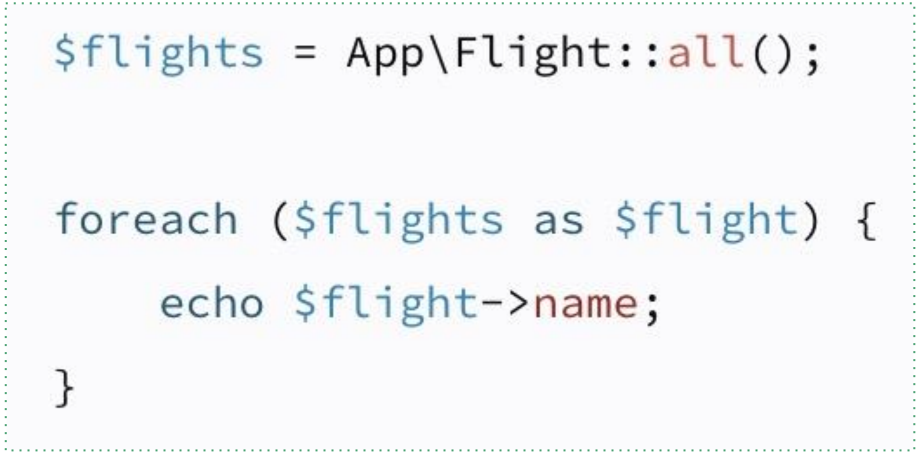
To get started, let's create an Eloquent model. Models typically live in the app directory.The easiest way to create a model instance is using the *make:model* Artisan command:



```
php artisan make:model Flight
```

## Eloquent ORM

Once you have created a model and its associated database table, you are ready to start retrieving data from your database. Think of each Eloquent model as a powerful query builder allowing you to fluently query the database table associated with the model. For example:



```
$flights = App\Flight::all();


foreach ($flights as $flight) {
    echo $flight->name;
}
```

For more details on topics of this lecture:
Read Chapter 05

## Activities and Next Week Topics

**This Week:**

- Read Chapter 05 of Laravel: Up & Running, for more information on Databases and Eloquent.
- Practice different types of databases and connections.
- Create simple database for your project.

**Next Week:**

- Database Migrations.

## References / Further Readings

- Laravel.com : Laravel’s official Documentation.
- Matt Stauffer, 2019. Laravel: Up & Running: A Framework for Building Modern PHP Apps. O’Reilly Media.
- Dayle Rees, 2016. Laravel: Code Smart.