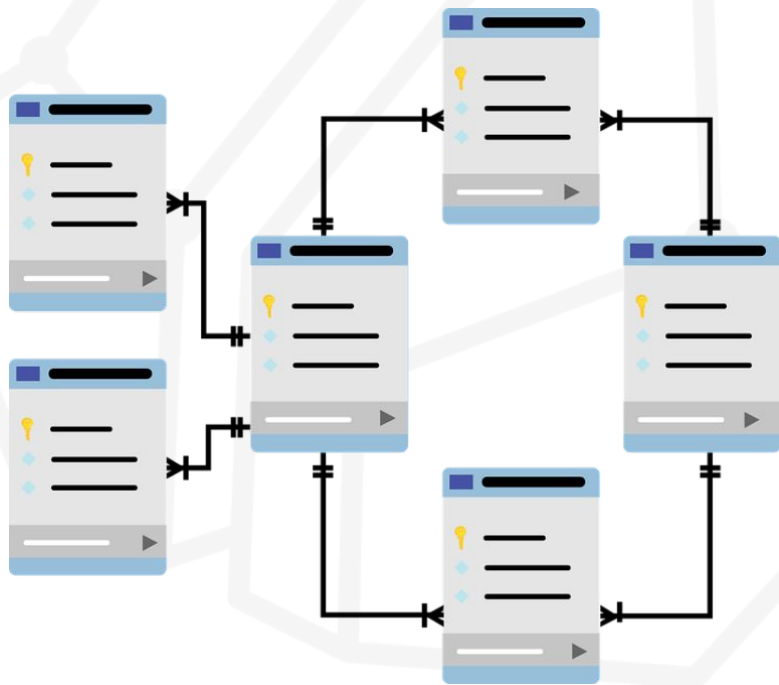


Laravel

Eloquent relationship - one to many





Modelli e relazioni

Le tabelle di un database sono spesso collegate tra di loro e per questo è necessario “insegnare” a Laravel, attraverso Eloquent, a gestire le relazioni tra queste tabelle.



Le relazioni in Eloquent

- Eloquent non permette solo di mappare tabelle in oggetti (Modelli), ma riesce ad esprimere anche relazioni tra di essi.
- Le colonne di una tabella vengono definite come proprietà dell'oggetto
- Le relazioni tra le entità sono definite, invece, come funzioni
- Si possono progettare le seguenti relazioni
 - one to one
 - **one to many**
 - many to many
 - *has one through*
 - *has many through*
 - *polymorphic*

ONE-TO-ONE	ONE-TO-MANY	MANY-TO-ONE	MANY-TO-MANY
			

La relazione one to many

Questa relazione è molto frequente ed è possibile trovarla, ad esempio, nella relazione tra post e commenti: un post ha più commenti, mentre un commento appartiene solamente ad un post.

Permette di associare un particolare modello con una lista di altri modelli che sono legati ad un solo modello padre.

Esistono due modi per vedere questa relazione: in maniera diretta oppure inversa, a seconda da quale modello sia il soggetto della relazione.

One to many diretta



```
class Post extends Model
{
    /**
     * Get the comments for the blog post.
     */
    public function comments()
    {
        return $this->hasMany( 'App\Comment' );
    }
}
```

One to many - inversa



```
class Comment extends Model
{
    /**
     * Get the post that owns the comment.
     */
    public function post()
    {
        return $this->belongsTo('App\Post');
    }
}
```

One to many - nelle risposte delle REST API

Utilizzando la funzione `with()` è possibile specificare le relazioni da includere nella query e quindi nella risposta emessa dalle REST API.



```
//Esempio lettura autore
$author = Author::with('books')->findOrFail($id);
return response()->json($author, 200)

//Esempio lettura libro
$book = Book::with('author')->findOrFail($id);
return response()->json($book, 200);
```



Q & A