

Eloquent cheat sheet for Laravel 13

Queries, relationships, eager loading, scopes, casts, mass assignment, chunking and the N+1 fixes, in the Laravel 13 syntax (model attributes like #[Fillable] and #[Scope]).



Laravel 13

Model setup

Attributes instead of properties

```
use Illuminate\Database\Eloquent\Attributes\{Fillable, Hidden, Table};

#[Table('flights', key: 'flight_id')]
#[Fillable(['name', 'destination'])]
#[Hidden(['secret'])]
class Flight extends Model
{
    use HasFactory, SoftDeletes;
}
```

Strict mode in AppServiceProvider::boot()

```
Model::preventLazyLoading(! $this->app->isProduction());
Model::preventSilentlyDiscardingAttributes($this->app->isLocal());
Model::shouldBeStrict(! $this->app->isProduction()); // all of the above + missing attributes
```

Queries

Retrieve

```
Flight::find(1); // Flight::findOrFail(1);
Flight::where('active', 1)->orderBy('name')->limit(10)->get();
Flight::firstWhere('number', 'FR 900');
Flight::where('legs', '>', 3)->firstOr(fn () => null);
Flight::where('active', 1)->count(); // sum, max, avg...
Flight::whereIn('id', $ids)->pluck('name', 'id');
Flight::latest()->paginate(15); // simplePaginate, cursorPaginate
```

Create, update, upsert

```
$flight = Flight::create(['name' => 'London to Paris']);
$flight->update(['name' => 'Paris to London']);
Flight::firstOrCreate(['name' => 'X'], ['delayed' => 1]);
Flight::updateOrCreate(['departure' => 'OAK'], ['price' => 99]);
Flight::upsert($rows, uniqueBy: ['departure', 'destination'], update: ['price']);
$flight->isDirty('name'); $flight->wasChanged(); $flight->getOriginal('name');
```

Subquery select and order

```
Destination::addSelect(['last_flight' => Flight::select('name')->whereColumn('destination_id', 'destinations.id')->latest('arrived_at')->limit(1)])->get();
```

Relationships

Define

```
public function posts(): HasMany { return $this->hasMany(Post::class); }
public function user(): BelongsTo { return $this->belongsTo(User::class); }
public function roles(): BelongsToMany
{
    return $this->belongsToMany(Role::class)->withPivot('active')->withTimestamps();
}
public function latestOrder(): HasOne { return $this->hasOne(Order::class)->latestOfMany(); }
public function comments(): MorphMany { return $this->morphMany(Comment::class, 'commentable'); }
```

Query and write through relations

```
$user->posts()->where('active', 1)->get();
Post::has('comments', '>=', 3)->get();
Post::whereHas('comments', fn ($q) => $q->where('approved', true))->get();
Post::whereRelation('comments', 'is_approved', false)->get();
Post::whereBelongsTo($user)->get();
$user->posts()->create(['title' => 'Hi']);
$user->roles()->attach($id); ->detach($id); ->sync([1, 2]); ->toggle([3]);
```

Eager loading

Load up front

```
Book::with('author')->get();
Book::with(['author.contacts', 'publisher'])->get();
Book::with('author.id,name')->get(); // include the foreign/primary keys
User::with(['posts' => fn ($q) => $q->latest()])->get();
User::withWhereHas('posts', fn ($q) => $q->where('featured', true))->get();
```

Counts and aggregates without loading

```
Post::withCount('comments')->get(); // $post->comments_count
Post::withExists('comments')->get(); // $post->comments_exists
Post::withSum('orders', 'total')->get(); // withAvg, withMin, withMax
$books->load('author'); $books->loadMissing('author'); $post->loadCount('likes');
```

Scopes

Local scope with #[Scope]

```
use Illuminate\Database\Eloquent\Attributes\Scope;

#[Scope]
protected function ofType(Builder $query, string $type): void
{
    $query->where('type', $type);
}

User::ofType('admin')->popular()->get();
User::popular()->orWhere('active')->get();
```

Global scope

```
#[ScopedBy([AncientScope::class])]
class User extends Model {}

User::withoutGlobalScope(AncientScope::class)->get();
User::withoutGlobalScopes()->get();
```

Casts and accessors

casts() method

```
protected function casts(): array
{
    return [
        'is_admin' => 'boolean',
        'options' => AsArrayObject::class,
        'tags' => AsCollection::class,
        'status' => OrderStatus::class, // backed enum
        'paid_at' => 'immutable_datetime',
        'secret' => 'encrypted',
        'password' => 'hashed',
    ];
}
```

Accessor / mutator

```
protected function firstName(): Attribute
{
    return Attribute::make(
        get: fn (string $value) => ucfirst($value),
        set: fn (string $value) => strtolower($value),
    );
}
```

Big tables: chunk, lazy, cursor

Pick the right one

```
Flight::chunk(200, fn (Collection $flights) => /* ... */);
Flight::where('departed', true) // updating the
    filtered column?
    ->chunkById(200, fn ($flights) => $flights->each-
>update(['departed' => false]));
foreach (Flight::lazy() as $flight) { } // chunked
queries, one stream
Flight::where('departed', true)->lazyById(200)->each-
>update(['departed' => false]);
foreach (Flight::cursor() as $flight) { } // one query, no
eager loading
```

N+1 fixes

Spot it, then fix it

```
// N+1: one query per book
foreach (Book::all() as $book) { echo $book->author->name; }

// Fix: eager load (2 queries)
foreach (Book::with('author')->get() as $book) { echo $book-
>author->name; }

// Counting? Don't load the rows
Post::withCount('comments')->get();

// Child needs its parent back: chaperone() sets the inverse
without a query
public function comments(): HasMany { return $this-
>hasMany(Comment::class)->chaperone(); }

// Opt in to automatic lazy eager loading (whole app or one
collection)
Model::automaticallyEagerLoadRelationships();
$users->withRelationshipAutoloading();
```

Turn on Model::preventLazyLoading() outside production so every N+1 throws in development and tests.

Checked 2026-10-15 against: <https://laravel.com/docs/13.x/eloquent> · <https://laravel.com/docs/13.x/eloquent-relationships> · <https://laravel.com/docs/13.x/eloquent-mutators>

Online version with copy buttons: laravel-vuejs.space/cheatsheets/eloquent · More guides and the monthly newsletter at laravel-vuejs.space