Its very simple we have to go to **BlogPost.php** and add

public function scopeLatest(Builder $query){

return $query->orderBy(static::CREATED\_AT,'desc');

}

In case of showing relations

One way is go to **PostsController.php**

return view('posts.show',['posts'=>BlogPost::with(['comment'=>function($query){

return $query->latest();

}])->findorFail($id)]);

Much easir way go to **BlogPost.php**

**public function comment()**

**{**

**return $this->hasMany('App\Comment')->latest();**

**}**

**Local Query Scope-Most Commented Posts**

BlogPost::mostCommented()->get()->pluck('comment\_count');

**Problem**

Illuminate/Database/QueryException with message 'SQLSTATE[42S22]: Column not found: 1054 Unknown column 'comments\_count' in 'order clause'

(SQL: select `blog\_posts`.\*, (select count(\*) from `comments` where `blog\_posts`.`id` = `comments`.`blog\_post\_id` and `comments`.`deleted\_at` is null) as `comment\_count` from `blog\_posts` where `blog\_posts`.`deleted\_at` is null order by `comments\_count` desc)'

**Problem solved**

BlogPost::mostCommented()->get()->pluck('comment\_count');

BlogPost.php

public function scopemostCommented(Builder $query){

//we will produce a field name comments count

// return $query->withCount('comment')->orderBy('comments\_count','desc');

return $query->withCount('comment')->orderBy('comment\_count', 'desc');

}

'comment\_count' the first letter you need to check the relation if relation name is comment and you write comments you will face the above error Illuminate/Database/QueryException with message 'SQLSTATE[42S22]: Column not found: 1054 Unknown column 'comments\_count' in 'order clause'

**In PostsController you will write**

**return view(**

**'posts.index',**

**[**

**'posts' => BlogPost::latest()->withCount('comment')->get(),**

**'most\_commented' => BlogPost::mostCommented()->take(5)->get(),**

**'mostActive' => User::WithMostBlogPosts()->take(5)->get(),**

**// 'mostActiveLastMonth' => User::withMostBlogPostsLastMonth()->take(5)->get(),**

**]**

**);**

**Users with most Blog posts in Last month**

**public function scopeWithMostBlogPostslastMonth(Builder $query)**

**{**

**{**

**return $query->withCount(['blogPost' => function (Builder $query) {**

**$query->whereBetween(static::CREATED\_AT, [now()->subMonths(1), now()]);**

**}])->having('blog\_post\_count', '>=', 2)**

**->orderBy('blog\_post\_count', 'desc');**

**}**

**Practical Global Query Scope**

**-Admin can see deleted posts**

**At first we will create a DeletedAdminScope.php inside scope folder**

**public static function boot()**

**{**

**// static::addGlobalScope(new LatestScope);**

**static::addGlobalScope(new DeletedAdminScope);**

**parent::boot();**

**static::deleting(function (BlogPost $blogPost) {**

**$blogPost->comment()->delete();**

**});**

**static::restoring(function (BlogPost $blogPost) {**

**$blogPost->comment()->restore();**

**});**

**}**

**If post is trashed than if we want that admin cant delete it again**

**We have to create a scope in Scope/DeletedAdminScope**

in DeletedAdminScope.php

**public function apply(Builder $builder, Model $model)**

**{**

**if (Auth::check() && Auth::user()->is\_admin) {**

**// $builder->withTrashed();**

**$builder->withoutGlobalScope('Illuminate\Database\Eloquent\SoftDeletingScope');**

**}**

}

}

in BlogPost.php we need to write

public static function boot()

{

// static::addGlobalScope(new LatestScope);

static::addGlobalScope(new DeletedAdminScope);

parent::boot();

static::deleting(function (BlogPost $blogPost) {

$blogPost->comment()->delete();

});

static::restoring(function (BlogPost $blogPost) {

$blogPost->comment()->restore();

});

}

in delete.blade.php if posts are deleted we wont show deleted button anymore

**@can('delete',$post)**

**@if($post->trashed())**

**<form class="fm-inline d-none" method="POST"action="{{ route('posts.destroy', ['post' => $post->id]) }}">**

**@csrf**

**@method('DELETE')**

**<input class="btn btn-danger" type="submit" value="Delete!" />**

**</form>**

**@endcan**

**@endif**

**In Index.blade.php**

**@forelse($posts as $post)**

**@if($post->trashed())**

**<del>**

**@endif**

**<h1>{{ $loop->index+1 }}.<a class="{{$post->trashed() ? 'text-danger':''}}" href="{{ route('posts.show', ['post' => $post->id]) }}" target="\_blank">{{ $post->title }}</a></h1>**

**</del>**