1. **One-To-One**

**hasOne**

1. If using foreign key in other table like user\_id use in this way in other table

Eg:table->phones

 $table->unsignedBigInteger('user\_id'); or $table->bigInteger('user\_id')->unsigned();

It will unset for negative integer it good approach and foreign key

$table->foreign('user\_id')->references('id')->on('users');

1. General error: 1364 Field 'phone' doesn't have a default value")

Sol : config->database.php->strict->false.

Sol: fake data so phoneNumber in fake data

'phone' => $faker->unique()->phoneNumber,

**UserController.php**

 public function index()

    {   $user = User::first();

       // $phone = Phone::first();

        //$phone->user\_id = $user->id;

        //$phone->save();

        dd($user->phoneX->number); //if only phoneX is passed it will return all Phone model data for User

    }

**User.php**

 //one to one relationship

 public function phoneX()

  {

   return $this->hasOne(Phone::class, 'user\_id','id');

//foreign key is user\_id of Phone and id is for User model

    }

**belongsTo**

UserController.php

 $phone = Phone::where('number', '1234567890')->first();

        if($phone){

            dd($phone->owner);

        }//

Phone.php

public function owner()

    {

        return $this->belongsTo(User::class,'user\_id');

    }

User.php

public function phoneX()

  {

   return $this->hasOne(Phone::class, 'user\_id','id');

//foreign key is user\_id of Phone and id is for User model

    }

1. Foreign key is added for owner object if user object is used then no need to add foreign key in both user and phone mode check for changes to return value
2. Protected $table= ‘tablename’ used in model when table name is different
3. **One-To-Many**

One user many own 0 or more posts. One post should strictly belong to one author.

**hasMany**

1. User.php

public function posts()

    {

        return $this->hasMany(Post::class);

    }

1. UserController.php

public function index()

    {

     //for data entry in table

        //$user = User::first();//first is for first input id

        //for post table fake data

       /\*$post = new Post;

     $post->title = Str::random(10);

     $post- >description = Str::random(10);//any random string for fake data

        $post->user\_id = 2;//for id 2 in User

     $post->save();\*/

        //one-to-many

      $user = user::find(2);//2 is id in User model find by id

         dd($user->posts);

    }

**belongsTo**

Post.php

public function posts()

    {

        return $this->hasMany(Post::class,'user\_id','id');

}

UserController.php

    public function index()

    {

        $post = Post::first();

        dd($post->author);

    }

1. **Many to Many**
2. Error: Cannot declare class CreateCoursesTable, because the name is already in use

Sol: refresh migration and reset table again check

1. Auti increment in User\_roles not required and even model also
2. This 3rd table is called pivot/intermediate/central table.
3. Contains only foreign\_id of both tables then many to many is used as belongsTomany and pivot tablename,foreign\_key,id should be mentioned

UserController.php

public function index()

    {

        //many-to-many

        $user = User::first();//for first id only

        $roles = Role::first();

        //$user->roles()->attach($roles->id);//for single id

        //or

        $user->roles()->attach([2,3,4]);//inserting all subject to one id

        //dd($user->roles);

        //or

        //in loop

        //echo $user->name;

        foreach(User::all() as $user){

           // echo '<p>'.$user->name.' '.$user->last\_name.'<p>';

            echo '<p>'.$user->name.'<p>';

            if ($user->roles) {

                echo '<ul>';

                foreach ($user->roles as $roles) {

                    echo '<li>'.$roles->name.'</li>';

                }

                echo '</ul>';

            }

        }

    }

Role.php

public function students()

    {   //many-to-many

        return $this->belongsToMany(User::class,'user\_roles');

    }

User.php

 public function roles()

    {

        return $this->belongsToMany(Role::class,'user\_roles');//pivot table name, foreign key and id

    }

**Additional column to pivot table**

1. Rollback migration for last table ie pivot table
2. Post model

**return $this->belongsToMany(Tag::class,'post\_tag','post\_id','tag\_id')->withTimestamps();**

1. Add column status to pivot table and in Post model column to be passed

 return $this->belongsToMany(Tag::class,'post\_tag','post\_id','tag\_id')

        ->withTimestamps()

        ->withPivot('status');

**Attach, detach and sync (Defining intermediate custom model)**

1. Post model

 return $this->belongsToMany(Tag::class)

        ->using(PostTag::class)//create relationship for post\_tag table

        ->withTimestamps()

        ->withPivot('status');

1. No event handling for attach detach in laravel so handle such event like delete update inserted then use lifecycle in pivot model created, deleted in PostTag model

**Hasmanythrough**

1. Returns all product belong to that project id from

Error: column not found: 1054 Unknown column 'users.projects\_id' in 'where clause' (SQL: select \* from `users` where `users`.`projects\_id` = 1 and `users`.`projects\_id` is not null)

Sol: add foreign key and id

For: $project = Projects::find(1);

    return $project->tasks;

Error: Column not found: 1054 Unknown column 'tasks.projects\_id' in 'where clause' (SQL: select \* from `tasks` where `tasks`.`projects\_id` = 1 and `tasks`.`projects\_id` is not null)//

Sol:4.1(11:00):has many through for tasks model to project model

return $this->hasManyThrough(Task::class, User::class);//or

return $this->hasManyThrough(Task::class, User::class, 'project\_id','user\_id','id','id');

**HasOnethrough**

1. Returns only one project reference to from tasks

return $this->hasOneThrough(Task::class, User::class, 'project\_id','user\_id','id','id');

**HasManyThrough**  using pivot table

**PolyMorphic**

1. **One-one and one-many**

Create post comments in comment model using comments() method for one-one

Comment model have

 $table->morphs('commentable');//creates 2 col commentable\_id and commentable\_type

$post->comments()->create([

        'user\_id' => $user->id,

        'body' => 'comment for post'

    ]);

Video model

    //retrieve collection of comments

    public function comments()

    {

        return $this->morphMany(Comment::class,'commentable');//commentable from table morph

    }

    //retrieve single comments

    public function comment()

    {

        return $this->morphOne(Comment::class,'commentable')->latest();//commentable from table morph

    }

Post model

    //retrieve collection of comments

    public function comments()

    {

        return $this->morphMany(Comment::class,'commentable');//commentable from table morph

    }

    //retrieve single comments

public function comment()

    {

        return $this->morphOne(Comment::class,'commentable')->latest();//commentable from table morph

    }

Comment model

 public function commentable()

    {

        return $this->morphTo();//this will belongs to one post/video

    }

If method commentable() with other name then

 public function subject()

    {

        return $this->morphTo('commentable');//commentable identifier

    }

Retrive data from comment model

 //retrive post

   /\*$post = Post::find(1);

    dd($post->comments);\*/

     //retrive video

   /\*$video = Video::find(1);

    dd($video->comments);\*/

     //retrive comment directly using commentable method

    $comment = Comment::find(3);

    return $comment->commentable;

return $comment->subject; //retrive subject method from comment model

1. **Many-many**

App\Providers\Appserviceprovider.php for model in pivot column taggable\_type save App\Post but using code below only Model name Post

 public function boot()

    {

         Relation::morphMap([

            'Post' => \App\Post::class,

            'Video' => \App\Video::class,

            'Tag' => \App\Tag::class

        ]);

    }

**Tag.php**

 //reverse

    public function posts()

    {

        return $this->morphedByMany(Post::class, 'taggable','taggables');//taggables is pivot table

                                                                         //but here not required

                                                                         //as it will take taggable in plural as table

    }

    public function videos()

    {

        return $this->morphedByMany(Video::class, 'taggable','taggables');

    }

Tag create video link below code

  $tags = Tag::find(2);

    $tags->videos()->create([

        'title' => 'video  check'

    ]);

    dd($tags->posts);

**Post.php**

public function tags()

    {

        return $this->morphToMany(Tag::class,'taggable')->withTimestamps();

    }

**Video.php**

public function tags()

    {

        return $this->morphToMany(Tag::class,'taggable','taggables')->withTimestamps();

//taggable is for column taggable\_id and taggable\_type

    }