1.ManytoMany

2.ManytoManyMigration

At first we have to create a tags table

**php artisan make:migration create\_tags\_table**

**the**

**php artisan make:migration createBlogPostTagTable**

**Relation between tags and BlogPost Model**

**public function up()**

**{**

**Schema::create('blog\_post\_tag', function (Blueprint $table) {**

**$table->bigIncrements('id');**

**$table->unsignedBigInteger('blog\_post\_id')->index();**

**$table->foreign('blog\_post\_id')**

**->references('id')**

**->on('blog\_posts')**

**->onDelete('cascade');**

**$table->unsignedBigInteger('tag\_id')->index();**

**$table->foreign('tag\_id')**

**->references('id')**

**->on('tags')**

**->onDelete('cascade');**

**$table->timestamps();**

**});**

**}**

**3.Defining ManytoMany on models**

**Go to BlogPost.php**

**public function tags(){**

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

**}**

**Go to Tag.php**

**public function blogPost(){**

**$this->belongsToMany(BlogPost::class);**

**}**

**4.Assosiating Model in ManytoMany**

**tinker code**

**$tag1=new App\Tag;**

**$tag1->name='Science';**

**$tag1->save();**

**$blogPost=BlogPost::find(8);**

**$blogPost->tags()->attach($tag1);**

**In case of attach it wont care it will attach same tags to different blogposts**

**in case of**

**syncwithoutDetaching it will checek if there is any duplication**

**If we want to detach all tags just simply write**

**$blogPost->tags()->detach();**