LARAVEL RELATIONSHIPS

In Laravel, Eloquent relationships allow you to define how different database tables are related to each other. Relationships enable you to retrieve and work with related data easily. Laravel supports several types of relationships, including:

1. One-to-One Relationship:

Define Relationship:

```php
class User extends Model
{
    public function phone()
    {
        return $this->hasOne(Phone::class);
    }
}
```

Access Relationship:

```php
```

```
$user = User::find(1);
$phone = $user->phone; // Access the related phone
```

2.One-to-Many Relationship:

Define Relationship:

```php
class Post extends Model
{
    public function comments()
    {
        return $this->hasMany(Comment::class);
    }
}
```

Access Relationship:

```php
$post = Post::find(1);
$comments = $post->comments; // Access the related comments
```

3. Many-to-One Relationship (Inverse of One-to-Many):

Define Relationship:

```php
class Comment extends Model
{
    public function post()
    {
        return $this->belongsTo(Post::class);
    }
}
```

Access Relationship:

```php
$comment = Comment::find(1);
$post = $comment->post; // Access the related post
```

4. Many-to-Many Relationship:

Define Relationship:

```php
class User extends Model
{
    public function roles()
    {
        return $this->belongsToMany(Role::class);
    }
}
```

Access Relationship:

```php
$user = User::find(1);
$roles = $user->roles; // Access the related roles
```

5. Has-Many-Through Relationship:

Define Relationship:

```php
class Country extends Model
{
    public function posts()
    {
        return $this->hasManyThrough(Post::class, User::class);
    }
}
```

Access Relationship:

```php
$country = Country::find(1);
$posts = $country->posts; // Access the related posts
```

6. Polymorphic Relationships:

Define Relationship:

```php
class Image extends Model
{
    public function imageable()
    {
        return $this->morphTo();
    }
}

class Post extends Model
{
    public function images()
    {
        return $this->morphMany(Image::class, 'imageable');
    }
}

class User extends Model
{
```

```
    public function images()
    {
        return $this->morphMany(Image::class, 'imageable');
    }
}
```

Access Relationship:

```php
$post = Post::find(1);
$images = $post->images; // Access the related images
```

These are the fundamental Eloquent relationships in Laravel. Utilizing these relationships helps in simplifying database queries and creating expressive, maintainable code. Each type of relationship serves a specific purpose, allowing you to model different scenarios in your application.