Updating our Factory Relationships (Laravel 8+)

1. Introduction

If you open up DatabaseSeeder.php you'll see this line of code in run method:

```
Company::factory()->hasContacts(5)->count(50)->create();
```

If you remember, this code will create 50 companies, and for each generated company it will generate 5 contacts.

Now since we have added user_id column in both companies and contacts tables to link them to users table, we need to make a bit change to our factory relationships.

In this lesson I'm going to show you two options on how to update factory relationships.

2. 1st option: Simple option

First, before we generate companies and contacts table with Factory, we generate some users, let's say 5 users.

```
User::factory()->count(5)->create();
Company::factory()->hasContacts(5)->count(50)->create();
```

Second, let's open CompanyFactory. In definition method we add in the returned array a new item. The array key is user_id, while the value is contain id of the user that taken from user model randomly. Don't forget to import the user model namespace.

database/factories/CompanyFactory.php

Last, let's open ContactFactory. And we do almost the same way with CompanyFactory. Except, the value of user_id is not taken directly from User model. Why? Because if you remember contacts table also has relationship to companies table. We need to make both tables to have the same user id.

To do that we can randomly choose a company id, using that id to find the model instance, and then get its user id.

```
Company::pluck('id')->random()
Company::find(Company::pluck('id')->random())
Company::find(Company::pluck('id')->random())->user_id
```

Here how we implement that in ContactFactory.

database/factories/ContactFactory.php

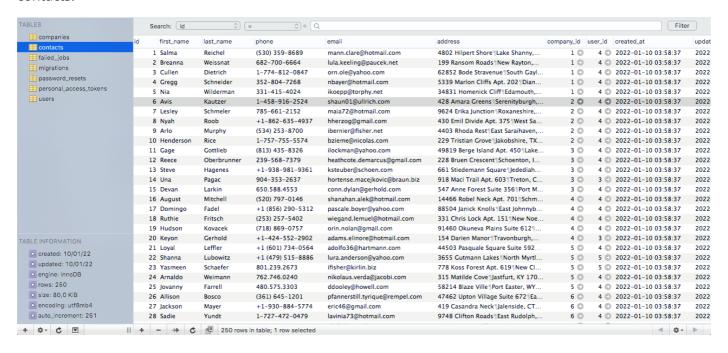
An alternative way is to get all companies, take one item randomly, and get its user id. It's much simpler that the code before.

```
Company::all()->random()->user_id
```

Go to terminal and run this command.

```
php artisan migrate:fresh --seed
```

If you now see your database, you'll have 5 users, 50 companies with random user id, and then 250 contacts.



3. 2nd option: Advance

First, like the first option, before we generate companies and contacts table with Factory, we generate some users, we'll create 5 users in this case. Then we loop through each user intance using each method, and pass a function.

```
User::factory()->count(5)->create(function ($user) {
});
```

Second, we move the compan::factory code into the function. We also change the number of companies that will be generated from 50 to 10. However it will produce the same result: 5*10 = 50.

```
User::factory()->count(5)->create(function ($user) {
   Company::factory()->hasContacts(5)->count(10)->create();
});
```

Third, change hascontacts magic method to has method, and pass the contact::factory in.

```
User::factory()->count(5)->create()->each(function ($user) {
   Company::factory()->has(
        Contact::factory()->count(5)
   )
   ->count(10)->create();
});
```

Fourth, specify user_id in company model creation by specifying an array contains user_id. We can get the value from <code>\$user_id</code> in stance.

```
User::factory()->count(5)->create()->each(function ($user) {
   Company::factory()->has(
        Contact::factory()->count(5)
   )
   ->count(10)->create([
        'user_id' => $user->id
   ]);
});
```

Last, we need to pass <code>user_id</code> to contact model creation. Since we're not call <code>create</code> method we cannot do the same way with the previous step. As alternative we can use <u>factory state</u> which return an array contains <code>user_id</code>. It's value is taken from the company instance.

```
User::factory()->count(5)->create()->each(function ($user) {
   Company::factory()->has(
        Contact::factory()->count(5)->state(function ($attributes, Company $company) {
          return ['user_id' => $company->user_id];
      })
   )
   ->count(10)->create([
        'user_id' => $user->id
   ]);
});
```

That's it, we don't need to touch CompanyFactory, ContactFactory or other files. Go to terminal and run this migrate & seed command.

```
php artisan migrate:fresh --seed
```

If you now see your database, you'll have 5 users, 50 companies with random user id, and then 250 contacts. You'll see in first 50 rows have user_id 1, next 50 rows have user_id 2, and so on.

