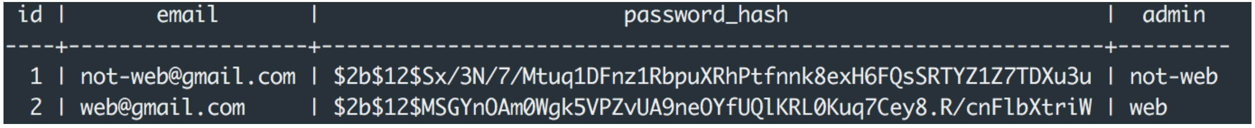
1.Prepare database:

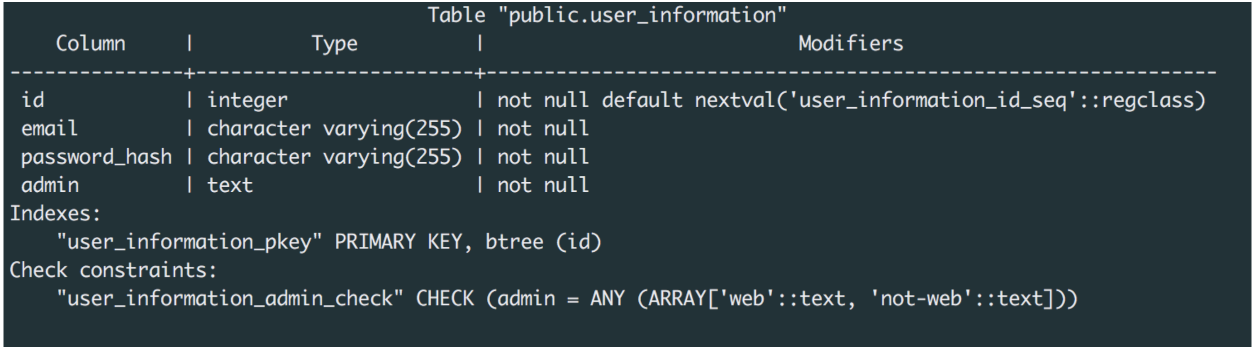
There are two ways to do the database preparation. The first way is that you could choose to directly dump the database ***walkthru***. There will be altogether two users in table ***user_information***:



```
 id |       email       |                         password_hash                          |  admin
----+-------------------+--------------------------------------------------------------+---------
  1 | not-web@gmail.com | $2b$12$Sx/3N/7/Mtuq1DFnz1RbpuXRhPtfnnk8exH6FQsSRTYZ1Z7TDXu3u | not-web
  2 | web@gmail.com     | $2b$12$MSGYnOAm0Wgk5VPZvUA9neOYfUQlKRL0Kuq7Cey8.R/cnFlbXtriW | web
```

The first of them is a user who has a web type and the second one has a not-web type. The id is primary key, email is string, password_bash is their hashed password and the admin is a enum type which only has 'not-web and web type'. Both of them have password: 123. You could directly use these two users in future API's.

If you want to do the migration, you need to first create database ***walkthru*** and then run the command: knex migrate:latest. Then you will get an empty ***user_information*** table as follow:



```
                                   Table "public.user_information"
    Column     |          Type          |                              Modifiers
---------------+------------------------+---------------------------------------------------------------
 id            | integer                | not null default nextval('user_information_id_seq'::regclass)
 email         | character varying(255) | not null
 password_hash | character varying(255) | not null
 admin         | text                   | not null
Indexes:
    "user_information_pkey" PRIMARY KEY, btree (id)
Check constraints:
    "user_information_admin_check" CHECK (admin = ANY (ARRAY['web'::text, 'not-web'::text]))
```

Notice: The database is empty.

2.Play with API's

Notice: The examples are for the first way of preparing the database which is directly dumping the database.

1.ALL-USER API:

a. GET /

Since it is very easy, I just create a very simple API for all users. If you run 'GET localhost:3000/' in postman, it will give back 'Hello World'.

b. POST /signup

It needs a json with email, password, and admin('web', or 'not-web'). If you POST a user with other admin type, it will send you back the error. An example is:

```
{
        "email": "web@gmail.com",
        "password": "123",
        "admin": "web"
}
```

2.ALL-AUTHENTICATED-USER API:

a. POST /getToken

It will give your authentication token if you are an authenticated user.

Given those two users in our table, you could run as follow:

```
{
	"email": "web@gmail.com",
	"password": "123",}
```

It will give you back a token like this:

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6MiwiaWF0IjoxNTM1OTIzMzg0fQ.hU-rU5hu9OIToxNUTW2SUpi3QzERQgGrpHozgf9d7Mg

3.ONLY-WEB-USER API:

a. POST /getWebToken

It will give your authentication token if you are an authenticated user who has a web type. If you are not-web type for admin, it will give you result as: I can not authorize you since your admin is web.

Example:

```
{
	"email": "not-web@gmail.com",
	"password": "123"
}
```

and

```
{
	"email": "web@gmail.com",
	"password": "123"
}
```