Project 1

Payment Application using Heroku as a Web Hosting

FSDO002ONL013\_Deandri Firdaus

Hactiv8 Bootcamp OCBC NISP

1. **Link dan Keterangan Project**

Beberapa link terkait Project1 adalah sebagai berikut:

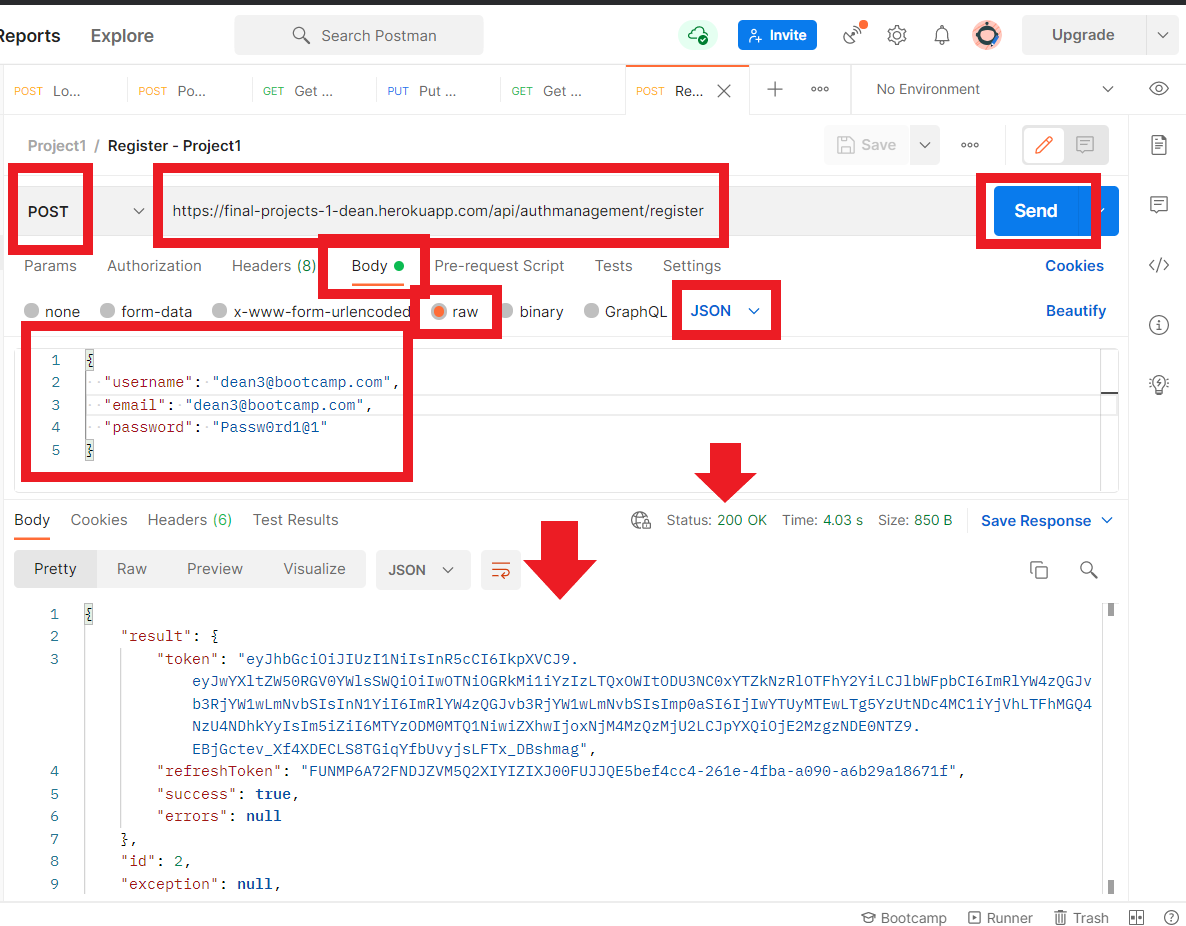
* + Project ini telah diupload seluruhnya ke Heroku dan GitHub.
  + Link GitHub =
  + Memasukkan link berikut ke postman => <https://final-projects-1-dean.herokuapp.com/api/project/masukkan_paymentDetailID>.

Akan dicoba untuk mengubah isi dari data

1. **Register**

Melakukan register dengan cara berikut:

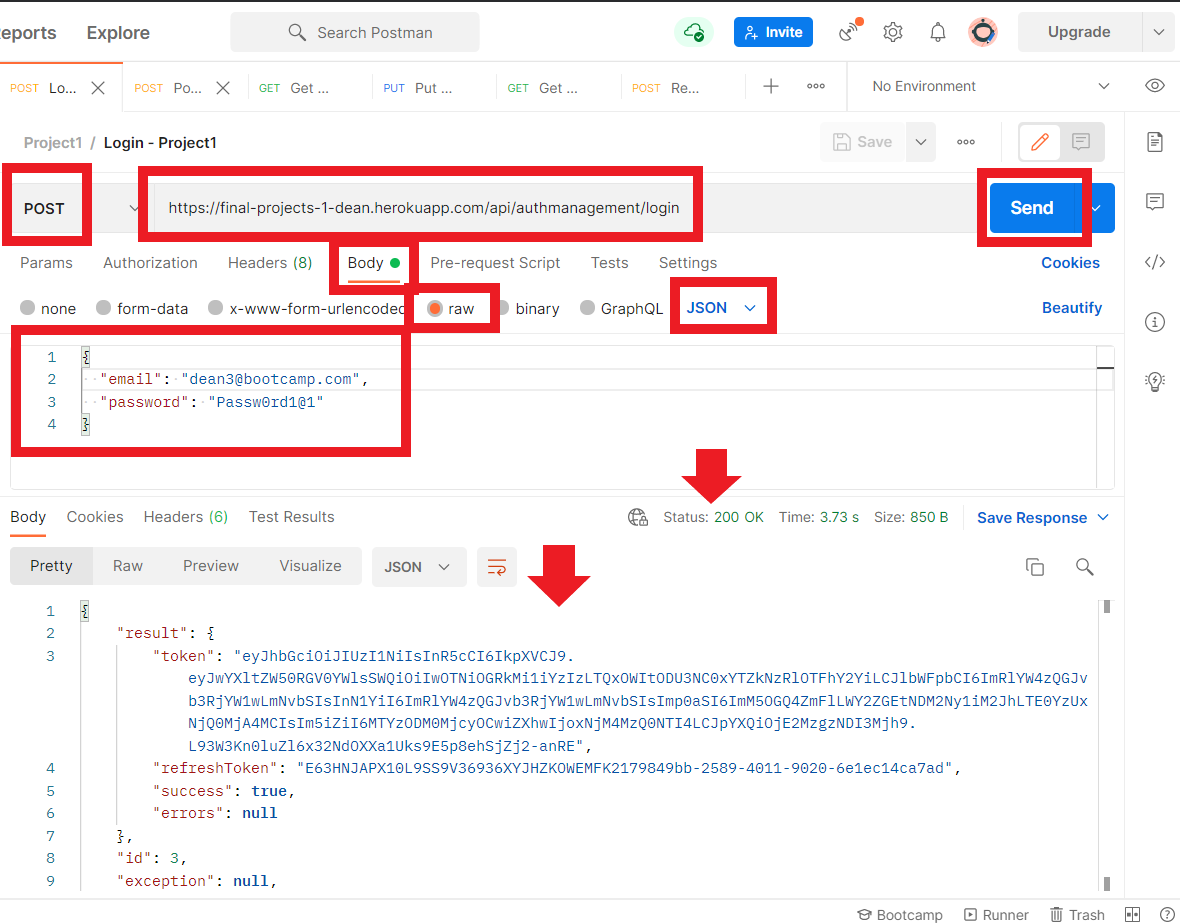
* + Memasukkan link berikut ke postman => <https://final-projects-1-dean.herokuapp.com/api/authmanagement/register>.
  + Mengganti method menjadi POST.
  + Mengubah dari Params menjadi Body.
  + Mengubah dari none menjadi raw.
  + Mengubah dari Text menjadi JSON.
  + Memasukkan username, email, dan password yang ingin didaftarkan. Hal ini perlu digunakan mengingat projek ini menggunakan Jwt dimana database tidak dapat diakses sebelum mendapatkan token.
  + Klik Send.
  + Tampil status bernilai Ok dan akan didapatkan token yang nantinya digunakan untuk mengakses database.
  + Untuk ke depannya, tidak perlu melakukan register lagi, hanya perlu melakukan login untuk mendapatkan token.



1. **Login**

Melakukan login dengan cara berikut:

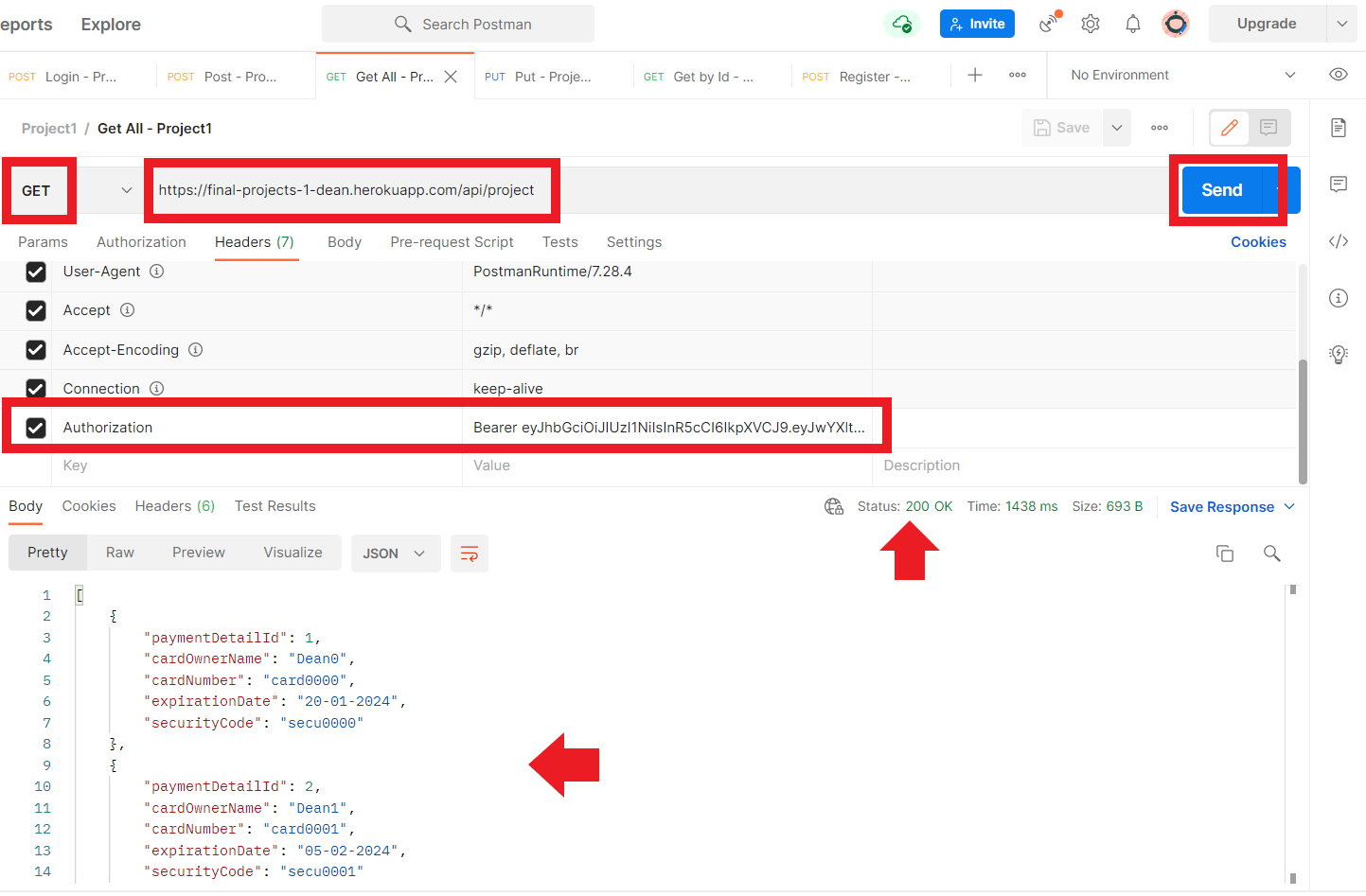
* + Memasukkan link berikut ke postman => <https://final-projects-1-dean.herokuapp.com/api/authmanagement/login>.
  + Mengganti method menjadi POST.
  + Mengubah dari Params menjadi Body.
  + Mengubah dari none menjadi raw.
  + Mengubah dari Text menjadi JSON.
  + Memasukkan email dan password yang telah didaftarkan sebelumnya. Hal ini perlu digunakan untuk mendapatkan token yang nantinya berfungsi untuk mengakses database.
  + Klik Send.
  + Tampil status bernilai Ok dan akan didapatkan token yang nantinya digunakan untuk mengakses database.



1. **Get All Payment Data**

Mendapatkan seluruh data payment dengan cara berikut:

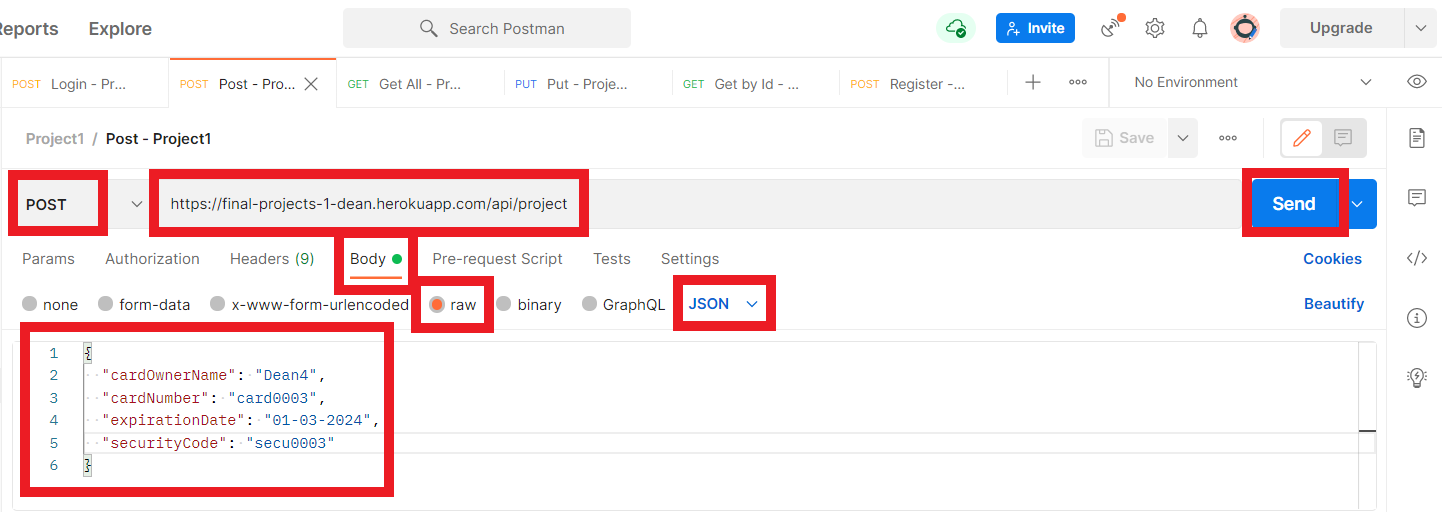
* + Memasukkan link berikut ke postman => <https://final-projects-1-dean.herokuapp.com/api/project>.
  + Menggunakan method GET.
  + Klik Header, scroll sampai bawah, lalu tambahkan key baru, yaitu Authorization.
  + Value-nya => Bearer {token}. Token didapatkan setelah register ataupun login.
  + Klik Send.
  + Tampil seluruh data payment yang telah dimasukkan.

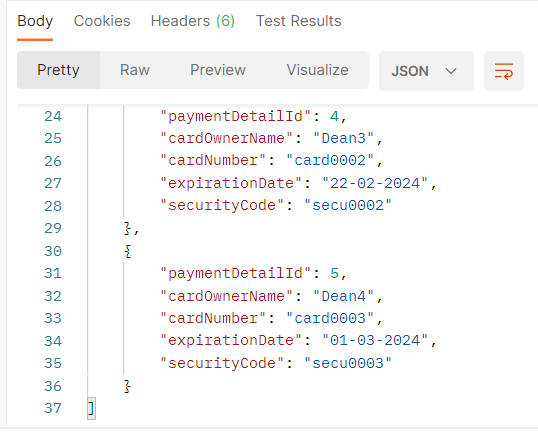


1. **Post New Payment Data**

Memasukkan data payment baru dengan cara berikut:

* + Memasukkan link berikut ke postman => <https://final-projects-1-dean.herokuapp.com/api/project>.
  + Mengganti method menjadi POST.
  + Klik Header, scroll sampai bawah, lalu tambahkan key baru, yaitu Authorization.
  + Value-nya => Bearer {token}. Token didapatkan setelah register ataupun login.
  + Mengubah dari Params menjadi Body.
  + Mengubah dari none menjadi raw.
  + Mengubah dari Text menjadi JSON.
  + Memasukkan data baru yang berisikan cardOwnerName, cardNumber, expirationDate, dan securityCode. Tidak perlu memasukkan paymentDetailId karena akan bernilai otomatis dan auto increment.
  + Klik Send.
  + Coba cek data tersebut menggunakan proses nomor 3.

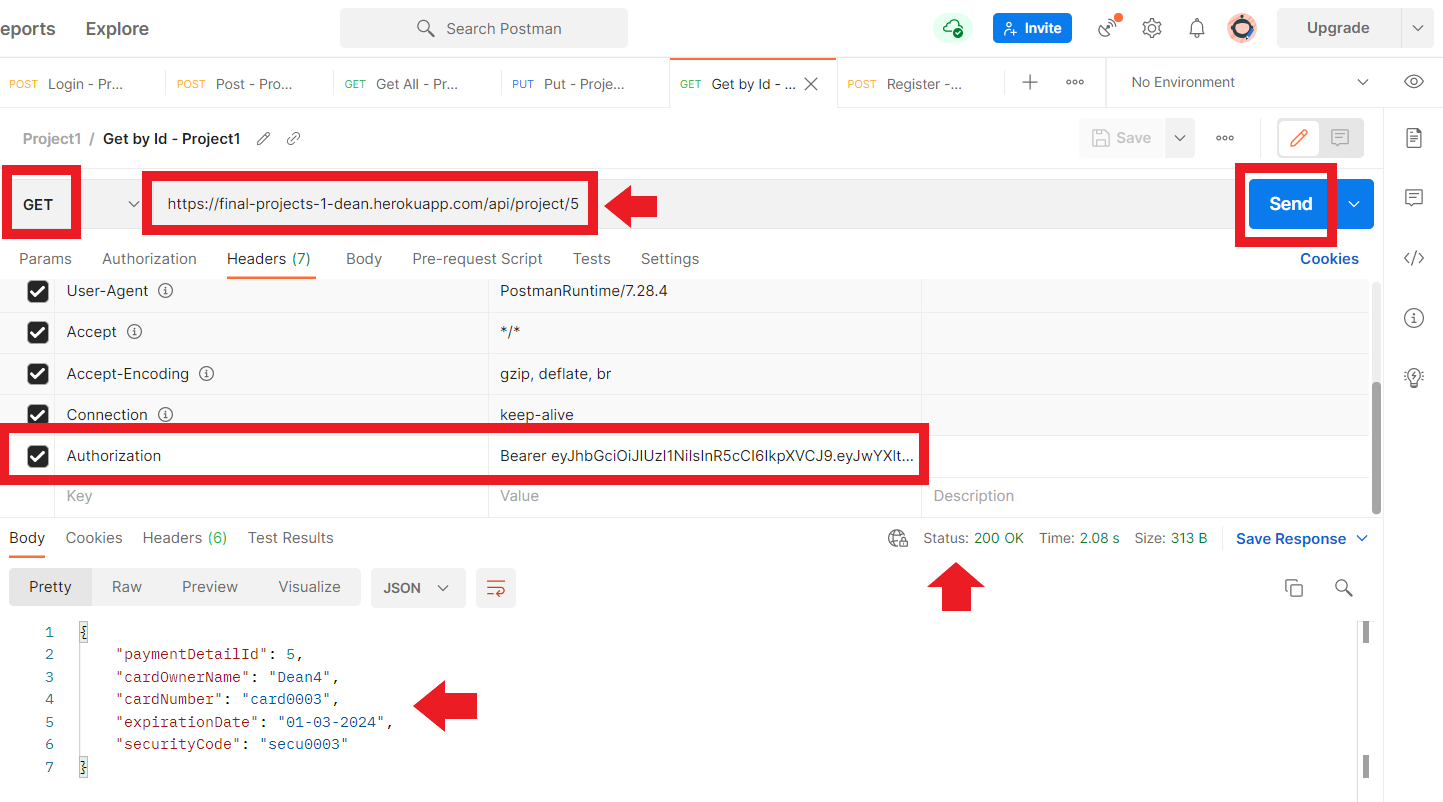




1. **Get Data by ID**

Mendapatkan data berdasarkan paymentDetailId dengan cara berikut:

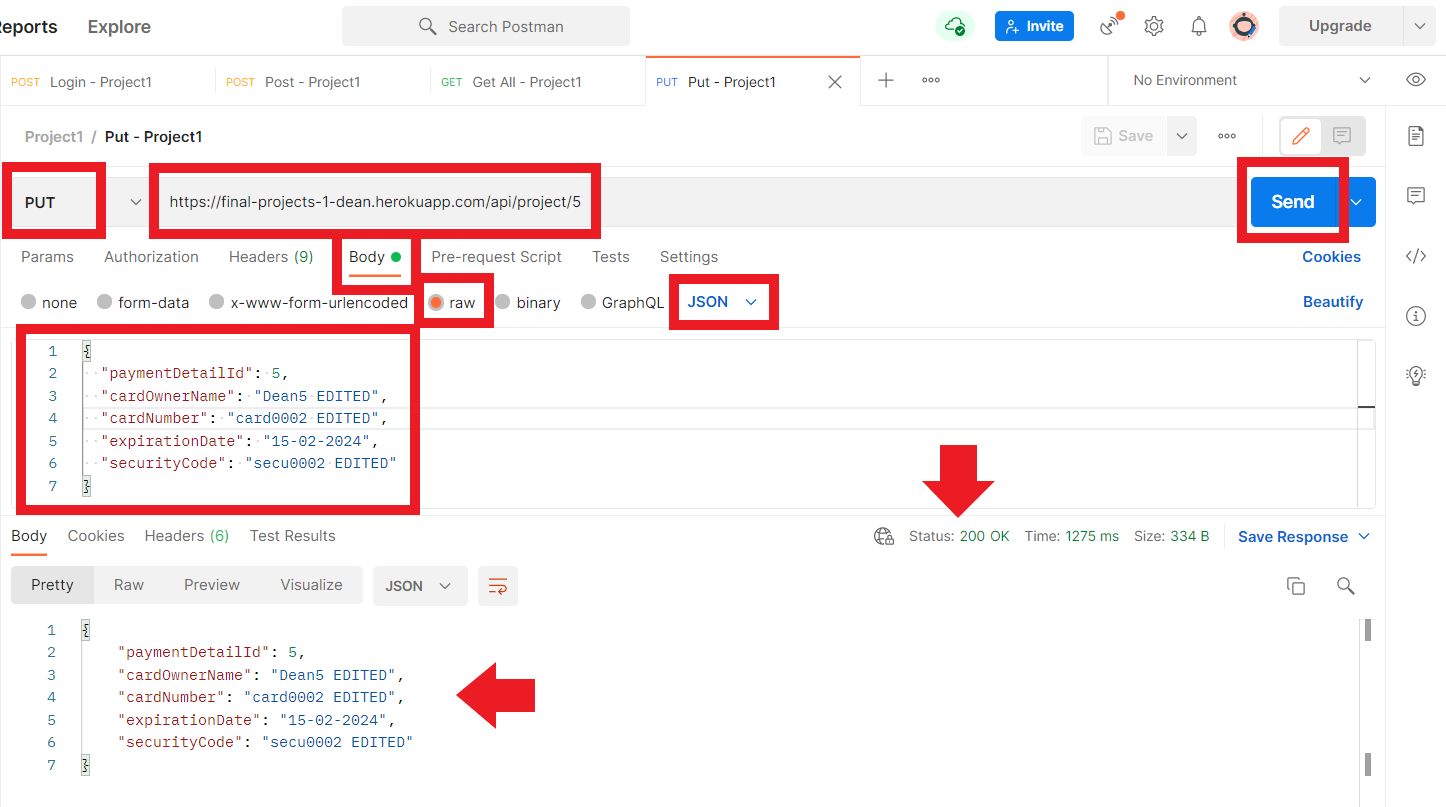
* + Memasukkan link berikut ke postman => <https://final-projects-1-dean.herokuapp.com/api/project/masukkan_paymentDetailID>.
  + Akan dicoba untuk memanggil data yang telah diinput sebelumnya dengan paymentDetailId bernilai 5.
  + Menggunakan method GET.
  + Klik Header, scroll sampai bawah, lalu tambahkan key baru, yaitu Authorization.
  + Value-nya => Bearer {token}. Token didapatkan setelah register ataupun login.
  + Klik Send.
  + Tampil detail data payment berdasarkan paymentDetailId yang telah dimasukkan.

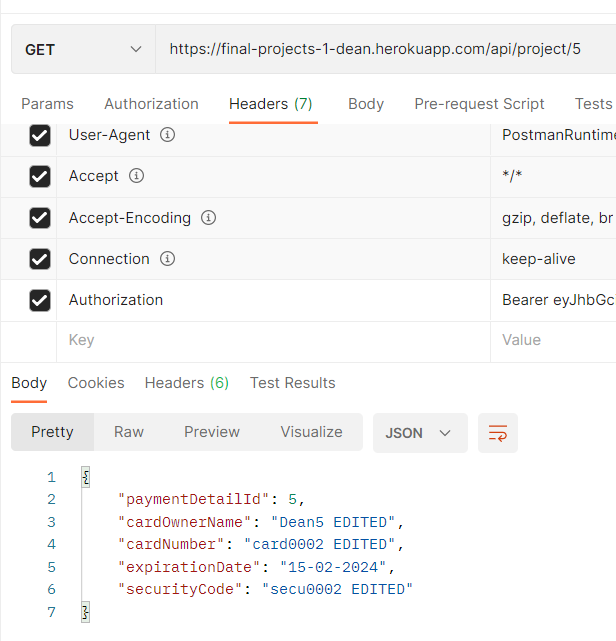


1. **Put Data by ID**

Mengedit salah satu data berdasarkan paymentDetailId dengan cara berikut:

* + Memasukkan link berikut ke postman => <https://final-projects-1-dean.herokuapp.com/api/project/masukkan_paymentDetailID>.
  + Akan dicoba untuk mengubah isi dari data yang telah diinput sebelumnya dengan paymentDetailId bernilai 5.
  + Menggunakan method PUT.
  + Klik Header, scroll sampai bawah, lalu tambahkan key baru, yaitu Authorization.
  + Value-nya => Bearer {token}. Token didapatkan setelah register ataupun login.
  + Klik Send.
  + Menampilkan data tersebut menggunakan proses 5.

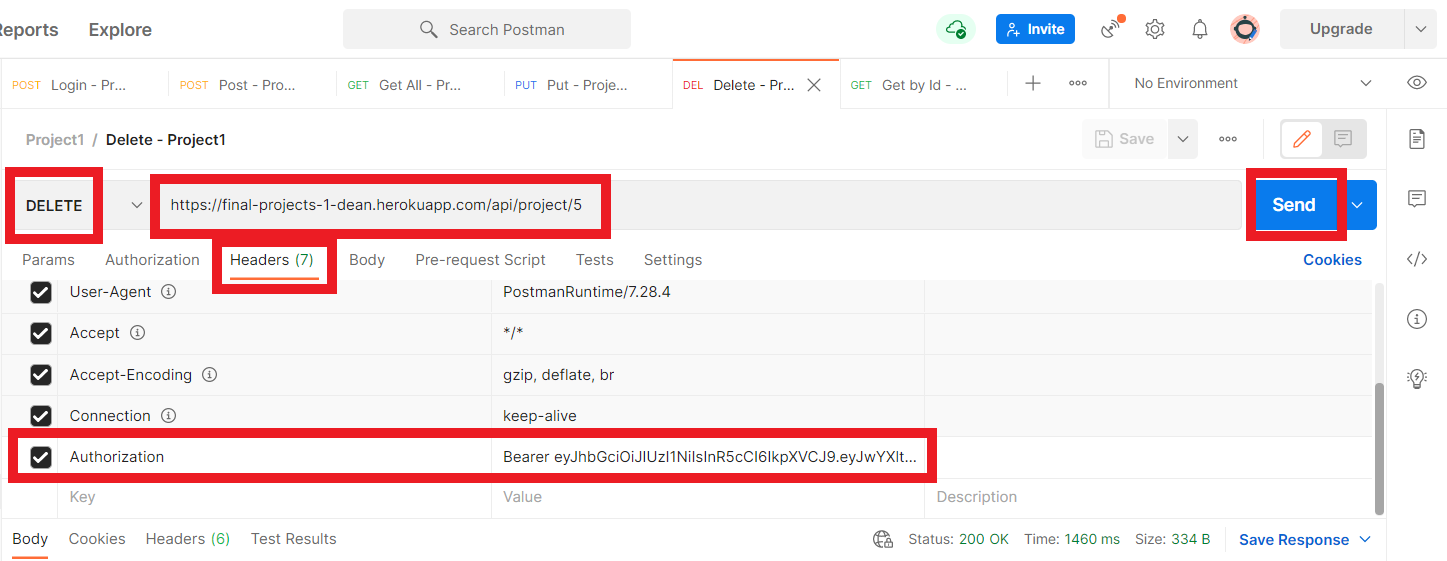


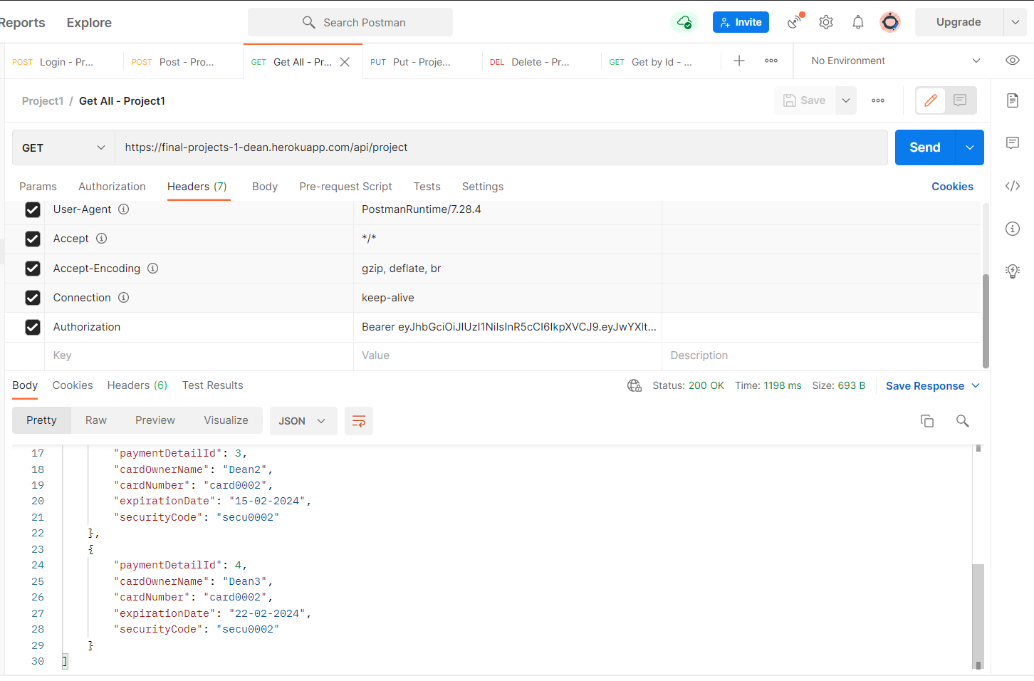


1. **Delete Data by ID**

Menghapus salah satu data berdasarkan paymentDetailId dengan cara berikut:

* + Memasukkan link berikut ke postman => <https://final-projects-1-dean.herokuapp.com/api/project/masukkan_paymentDetailID>.
  + Akan dicoba untuk menghapus data yang telah diinput sebelumnya dengan paymentDetailId bernilai 5.
  + Menggunakan method DELETE.
  + Klik Header, scroll sampai bawah, lalu tambahkan key baru, yaitu Authorization.
  + Value-nya => Bearer {token}. Token didapatkan setelah register ataupun login.
  + Klik Send.
  + Menampilkan semua data menggunakan proses 3.





1. Menggunakan database **assignment2** dengan kueri **use** dan membuat tabel baru dengan nama **products** dengan beberapa atribut sebagai berikut:
   * productCode dengan tipe data varchar(8) dan sebagai *primary key* yang berisikan kode produk (unique).
   * productName dengan tipe data varchar(50) yang berisikan nama produk.
   * productLine dengan tipe data varchar(8) yang berisikan nama *brand* atau merk. Atribut ini akan menjadi *foreign key* setelah tabel produclines dibuat. Relasi ini menunjukkan jalur produk (NKE, MSFT, PEP) atas produk yang telah dibuat atau akan dibuat.
   * productScale dengan tipe data int yang berisikan kualitas dan kesiapan produk dari skala 1 sampai 100.
   * productVendor dengan tipe data varchar(50) yang berisikan nama vendor.
   * productDescription dengan tipe data varchar(100) yang berisikan deskripsi dari produk yang telah atau akan dibuat.
   * quantityInStock dengan tipe data int yang berisikan total stok untuk produk terkait saat ini.
   * buyPrice dengan tipe data int yang berisikan harga beli product untuk customer.
   * MSRP dengan tipe data int yang berisikan harga beli product untuk retail-retail competitor yang ada di market dan harga yang disarankan oleh vendor atau pabrik pembuat produk.

Graphical user interface, text, application

Description automatically generated

1. Menggunakan database **assignment2** dengan kueri **use** dan membuat tabel baru dengan nama **productlines** dengan beberapa atribut sebagai berikut:
   * productLine dengan tipe data varchar(8) dan sebagai *primary key* yang berisikan kode jalur produk (unique). Misalkan beberapa produk yang telah dibuat akan dimasukkan ke jalur produk makanan atau pakaian.
   * textDescription dengan tipe data varchar(100) yang berisikan nama deskripsi atau penjelasan atas jalur produk.
   * htmlDescription dengan tipe data varchar(50) yang berisikan deskripsi dari produk yang dapat ditelusuri di internet, seperti sepatu, baju, keripik, dan seterusnya. Bisa disebut sebagai metadata.
   * image dengan tipe data varbinary(max) yang berisikan gambar dari jalur produk agar customer tertarik untuk membeli.

Graphical user interface, text, application, email

Description automatically generated

1. Menjadikan atribut **productLine** di tabel **products** menjadi *foreign key* yang mereferensikan *primary key* **productLine** di tabel **productlines**

Graphical user interface, text, application, email

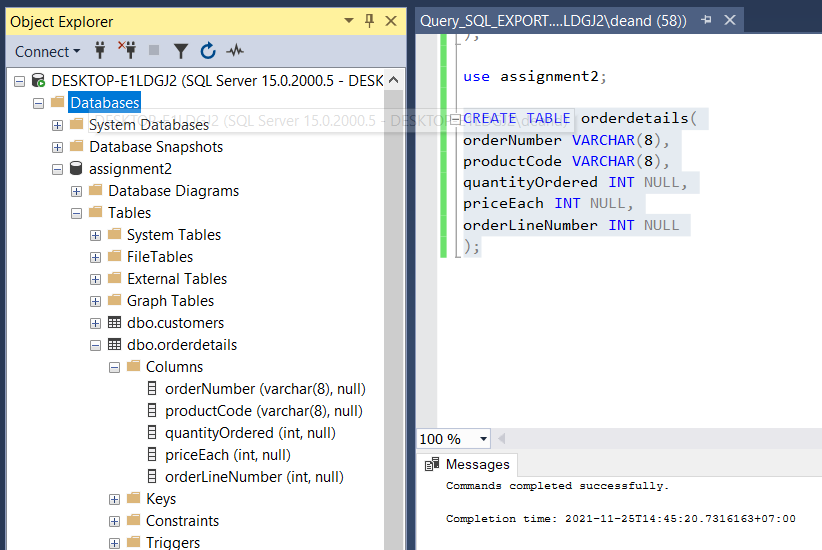
Description automatically generated

1. Menggunakan database **assignment2** dengan kueri **use** dan membuat tabel baru dengan nama **orders** dan dengan beberapa atribut sebagai berikut:
   * orderNumber dengan tipe data varchar(8) dan sebagai *primary key* yang berisikan kode transaksi (unique).
   * orderDate dengan tipe data date yang berisikan tanggal terjadinya transaksi.
   * requiredDate dengan tipe data date yang berisikan tanggal yang diinginkan customer.
   * shippedDate dengan tipe data date yang berisikan tanggal barang atau product dikirimkan.
   * status dengan tipe data varchar(15) yang berisikan keterangan mengenai proses barang saat ini, misal Packing.
   * comments dengan tipe data varchar(100) yang berisikan komentar mengenai proses barang saat ini.
   * customerNumber dengan tipe data varchar(8) dan sebagai *foreign key* yang berisikan nomor induk customer (unique) yang diferensikan oleh tabel **customers**.

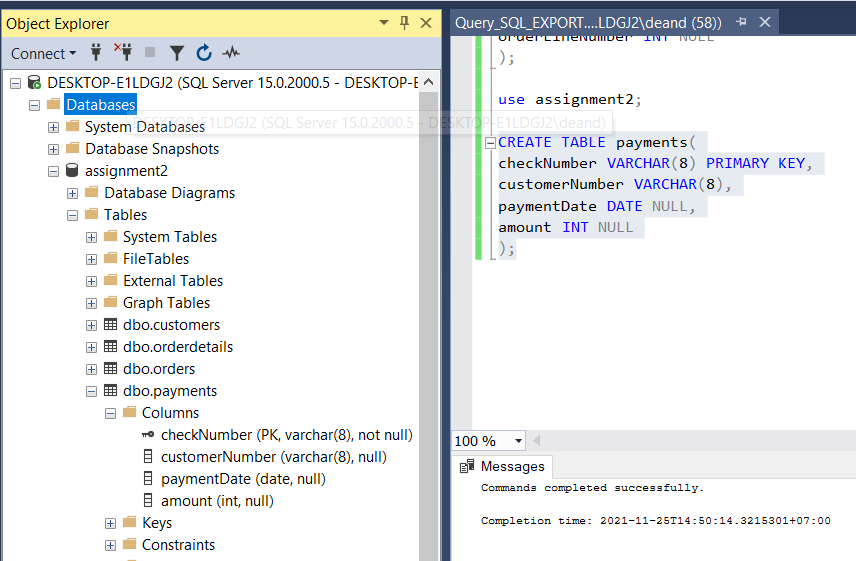
Graphical user interface, text, application, email

Description automatically generated

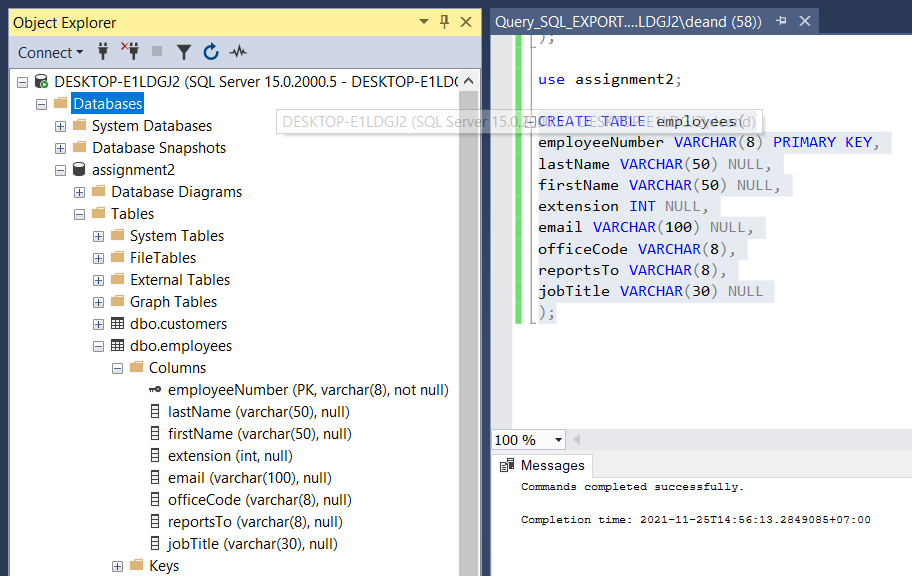
1. Menggunakan database **assignment2** dengan kueri **use** dan membuat tabel baru dengan nama **orderdetails** dan dengan beberapa atribut sebagai berikut:
   * orderNumber dengan tipe data varchar(8) dan sebagai *foreign key* yang berisikan kode transaksi (unique). Atribut ini direferensikan oleh tabel **orders**.
   * productCode dengan tipe data varchar(8) dan sebagai *foreign key* yang berisikan kode produk (unique). Atribut ini direferensikan oleh tabel **products**.
   * quantityOrdered dengan tipe data int yang berisikan banyaknya barang yang dipesan untuk setiap produk.
   * priceEach dengan tipe data int yang berisikan harga satuan tiap produk.
   * orderLineNumber dengan tipe data int yang berisikan data antrian pembelian.



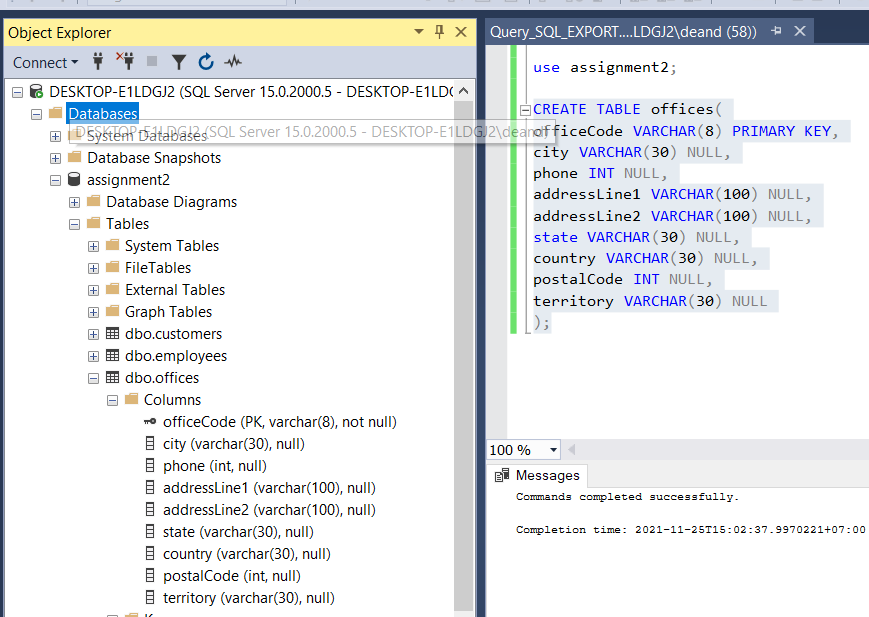
1. Menggunakan database **assignment2** dengan kueri **use** dan membuat tabel baru dengan nama **payments** dan dengan beberapa atribut sebagai berikut:
   * checkNumber dengan tipe data varchar(8) dan sebagai *primary key* yang berisikan kode pembayaran (unique).
   * customerNumber dengan tipe data varchar(8) dan sebagai *foreign key* yang berisikan nomor induk customer (unique). Atribut ini direferensikan oleh tabel **customers**.
   * paymentDate dengan tipe data int yang berisikan tanggal pembayaran.
   * amount dengan tipe data int yang berisikan total pembayaran.



1. Menggunakan database **assignment2** dengan kueri **use** dan membuat tabel baru dengan nama **employees** dan dengan beberapa atribut sebagai berikut:
   * employeeNumber dengan tipe data varchar(8) dan sebagai *primary key* yang berisikan nomor induk pegawai (unique).
   * lastName dengan tipe data varchar(50) yang berisikan nama belakang pegawai.
   * firstName dengan tipe data varchar(50) yang berisikan depan belakang pegawai.
   * extension dengan tipe data int yang berisikan kode ekstensi dan biasanya digunakan untuk jalur telepon internal maupun eksternal.
   * email dengan tipe data varchar(100) yang berisikan alamat email pegawai.
   * officeCode dengan tipe data varchar(8) dan sebagai *foreign key* yang berisikan nomor kode kantor (unique). Atribut ini direferensikan oleh tabel **offices**.
   * reportsTo dengan tipe data varchar(8) dan sebagai *foreign key* yang berisikan nomor induk pegawai yang bertindak sebagai manager (unique). Atribut ini direferensikan oleh tabel **employees**.
   * jobTitle dengan tipe data varchar(30) yang berisikan jabatan pegawai.



1. Menggunakan database **assignment2** dengan kueri **use** dan membuat tabel baru dengan nama **offices** dan dengan beberapa atribut sebagai berikut:
   * officeCode dengan tipe data varchar(8) dan sebagai *primary key* yang berisikan nomor kode kantor (unique).
   * city dengan tipe data varchar(30) yang berisikan kota lokasi kantor.
   * phone dengan tipe data int yang berisikan nomor telepon kantor.
   * addressLine1 dengan tipe data varchar(100) yang berisikan alamat kantor.
   * addressLine2 dengan tipe data varchar(100) yang berisikan alamat kantor.
   * state dengan tipe data varchar(30) yang berisikan provinsi lokasi kantor.
   * country dengan tipe data varchar(30) yang berisikan negara lokasi kantor.
   * postalCode dengan tipe data int yang berisikan kode pos kantor.
   * territory dengan tipe data varchar(30) yang berisikan wilayah atau penjelasan mengenai lokasi kantor, misal “Banyak competitor”, “Wilayah banjir”.



1. Menambahkan 10 baris baru untuk setiap tabel.

