

Ruby on Rails

CRUD işlemleri uygulama

1) Uygulamanın oluşturulması

Aşağıdaki komutlarla boş bir uygulama oluşturup, uygulama klasörünün içine giriyoruz. Diğer işlemleri bu klasör içinden gerçekleştireceğiz. «**crduygulama**» ismini kendinize göre değiştirebilirsiniz.

```
rails new crduygulama
```

```
cd crduygulama
```

2) Crud işlemleri için bir scaffold üretelim

- Aşağıdaki komutlarla /kitaps isminde restful tabanlı bir uygulama üretelim, bu uygulama aynı zamanda veri tabanında Kitap isimli bir model oluşturacak tablo isim,yazar, sayfa alanlarından oluşacak.

rails g scaffold kitap isim yazar sayfa:integer

rake db:migrate

rails s

3) CRUD işlemleri için konsolun açılması

- Bir önceki adımda «rails s» komutu ile uygulama çalışırken, Pazia ortamında soldaki menüyü kullanarak ikinci bir terminal açın (*web uygulaması ve CRUD konsol işlemlerini aynı anda görebilmek için*)

cd cruduygulama

rails c

4) Create işlemi

1.yol) işlemin başarılı olduğu ve çalışan sql kodlarına ilişkin konsol mesajlarına dikkat ediniz. Ayrıca modelde bizim eklediğimiz isim,yazar, sayfa alanlarına ilave olarak id, created_at, updated_at alanlarının rails tarafından eklendiğine dikkat ediniz.

Kitap.create(isim:"Python",sayfa:234,yazar:"Mustafa Başer")

```
irb(main):002:0> Kitap.create(isim:"Python",sayfa:234,yazar:"Mustafa Başer")
  (0.1ms) begin transaction
  Kitap Create (1.8ms) INSERT INTO "kitaps" ("isim", "yazar", "sayfa", "created_at", "updated_at") VALUES (?, ?, ?, ?, ?) [["isim", "Python"], ["yazar", "Mustafa Başer"], ["sayfa", 234], ["created_at", "2018-11-18 19:12:44.727169"], ["updated_at", "2018-11-18 19:12:44.727169"]]
  (6.2ms) commit transaction
=> #<Kitap id: 1, isim: "Python", yazar: "Mustafa Başer", sayfa: 234, created_at: "2018-11-18 19:12:44", updated_at: "2018-11-18 19:12:44">
irb(main):003:0> █
```

4) Create işlemi

2.yol) sql işlemleri **k.save** dedikten sonra çalışacak.

k.isim="Java"

k.yazar="Altuğ"

k.sayfa=232

k.save

4) Create işlemi

- a) 1 veya 2.yolu kullanarak Modele, kitap isimleri Python, Ruby, Java olan sayfa ve yazarları farklı birer kayıt ekleyin.
- b) 1 veya 2.yolu kullanarak Modele aynı yazar isminde örneğin «muharrem» isimli yazara ait farklı isimlerde kitaplar ekleyin.
- c) Modele sayfa sayısı 100 olan en az iki kitap ekleyin.

5) Read İşlemi

Read işlemi ile ilgili kullanılabilecek komutlar.

Kitap.all	#Tüm kayıtlar
Kitap.count	#Kayıt sayısı
Kitap.first	#İlk eklenen kayıt
Kitap.last	#Son eklenen kayıt
#Kitap.find(1)	#id=1 olan kayıt
Kitap.find_by(isim:"Java")	#Java ismiyle eklenen ilk kitap!

5) Read İşlemi

Read işlemi ile ilgili kullanılabilecek komutlar. *Devam*

Kitap.first.yazar

#İlk kitabın ismi

Kitap.find_by(isim:"Java")

#Java ismiyle eklenen ilk kitap!

Kitap.where(isim : "Java")

#İsmi Java olan tüm kitaplar

Kitap.where(isim : "Java").first

#İsmi Java olan ilk kitaplar

**Kitap.where(isim : "Java").first.sayfa
sayısı**

#İsmi Java olan ilk kitabın sayfa

5) Read işlemleri

- a) Kitap modeli üzerinde *find*, *find_by*, *where* ile ismi Python, Java olan kayıtları görüntüleyin. Find, find_by size tek kayıt yada nil döndürecektir.
- b) where sorgusu ile sayfa sayısı 100 olan, ismi Python olan, ismi Java olan kitapları görüntüleyin. Yazarı muharrem olan kitapları görüntüleyin (7.sayfada yapmış olmanız lazım)

6) Update işlemi

- 1) Tek bir kaydı update etmek:** find, find_by tek bir kayıt döndürür. Öncelikle kriterlere uyan update etmek istediğimiz bir kaydı seçiyoruz. Aşağıdaki örnekte ilk kaydın sayfa sayısını 150 olarak güncellemiş oluyoruz.

```
k=Kitap.first
```

```
k.sayfa=150
```

```
k.save
```

6) Update işlemi

- 1) **Tek bir kaydı update etmek:** find, find_by tek bir kayıt döndürür. Öncelikle kriterlere uyan update etmek istediğimiz bir kaydı seçiyoruz. Aşağıdaki örnekte kitap ismi java olan ilk kaydın yazarını «Altuğ Altıntaş» olarak güncelliyoruz.

```
k=Kitap.find_by(isim :"Java").first
```

```
k.yazar= "Altuğ Altıntaş"
```

```
k.save
```

6) Update işlemi

2) **Çoklu kayıt update etmek:** where sorguları birden fazla kayıt dönderir. Aşağıdaki örnekler ismi «Python» olan tüm kitapların isimlerini «pyton programlama» olarak değiştirir.

a)

```
k= Kitap.where(isim:"Python")  
k.update_all(isim: 'python programlama')  
k.save
```

Veya

b)

```
Kitap.where(isim:"Python").update_all(isim: 'python programlama')
```

6) Update işlemi

a) İsmi «Java» olan tüm kitapların ismini «Java programlama sanatı» olarak değiştirin.

b) İlk kaydı/son kaydı, find_by kriterine uyan ilk kaydı çekerek update işlemi uygulayınız.

c) Yazar ismi «muhammed» olan tüm kayıtların yazar ismini «muhammed taç» olarak güncelleyin

7) Delete işlemi

1) Tek bir kaydı silmek: update işleminde olduğu gibi *find*, *find_by* sorguları tek bir kaydı dönderir. Öncelikle kriterlere uyan silmek istediğimiz bir kaydı seçiyoruz. Aşağıdaki örnekte ilk kaydı siliyoruz.

a)

k=Kitap.first

k.delete

veya

b) Kitap.first.delete

7) Delete işlemi

2) Çoklu kayıt silmek: update işleminde olduğu gibi *where* sorguları çoklu kayıt dönderir. Öncelikle kriterlere uyan silmek istediğimiz bir kaydı seçiyoruz. Aşağıdaki örnekte ilk kaydı siliyoruz.

a)

```
k=Kitap.where(isim:"Python")
```

```
k.delete_all
```

veya

```
b) Kitap.where(isim:"Python").delete_all
```


7) Delete işlemi

3) **Tüm tabloyu silmek**: Aşağıdaki örnek tüm tabloyu siler

a)

Kitap.delete_all

Veya

b) **Kitap.destroy_all**

7) Delete işlemi

- a) Son kaydı çekerek siliniz.
- b) Yazar ismi «muharrem taç» olan tüm kayıtları silin
- c) Tüm tabloyu silin