Upute za NoSQL projekt iz predmeta Napredni modeli i baze podataka

akademska godina 2014/2015

Upute su podijeljene u dva dijela:

1. **Upoznavanje s osnovnim Riak funkcionalnostima**
2. **Zadaci koje je potrebno samostalno riješiti**

1. Upoznavanje s osnovnim Riak funkcionalnostima

Pokrenite i spojite se na virtualno računalo.

Pokretanje Riak clustera:

Iz /usr/riak/riak-1.4.2/dev direktorija pokrenuti barem 3 Riak čvora:

```
> ulimit -n 4096
> dev1/bin/riak start
> dev2/bin/riak start
> dev3/bin/riak start
> dev4/bin/riak start
> dev5/bin/riak start
```

Provjeriti rade li:

```
> ps aux | grep beam
```

Doznajte iz odgovarajćih dev*/etc/app.config datoteka na kojem portu rade čvorovi.
Tražite konfiguracijsku naredbu tipa: {http, [{"192.168.56.12", PORTNUMBER}]}
Uvjerite se s:

```
> netstat -a -n | grep 12:100
```

da su Riak sluša na odgovarajućim portovima

Povežite čvorove u klaster:

```
> dev2/bin/riak-admin cluster join dev1@192.168.56.12
```

dalje je svejedno na koji se čvor spajate, svi su ravnopravni, npr:

```
> dev3/bin/riak-admin cluster join dev2@192.168.56.12
> dev4/bin/riak-admin cluster join dev3@192.168.56.12
> dev5/bin/riak-admin cluster join dev4@192.168.56.12
```

što čini pripremu, ali još ne i stvarno stvaranje klastera.
Sa sljedećom naredbom pogledati "plan" djelovanja:

```
> dev1/bin/riak-admin cluster plan
```

i konačno, obaviti povezivanje u klaster (još jednom, svejedno je koji čvor pitamo):

```
> dev2/bin/riak-admin cluster commit
```

Pogledati statistiku nekog čvora s:

```
> curl http://192.168.56.12:10018/stats
```

Potražite connected_nodes i ring_members.

Probajte i:

```
> curl http://192.168.56.12:10018/ping
```

Unos, izmjena i brisanje vrijednosti

Unesite, pročitajte, te obrišite neku vrijednost, npr.

```
> curl -v -X PUT http://192.168.56.12:10018/riak/testbucket/helloworld -H
"Content-Type: text/html" -d "<html><body>Hello world\!</body></html>"
```

Koji http kod vraća prethodna naredba?
Što taj kod znači?
Pročitajte sa svih čvorova (može i bez -v):

```
curl -v http://192.168.56.12:10018/riak/testbucket/helloworld
curl -v http://192.168.56.12:10028/riak/testbucket/helloworld
curl -v http://192.168.56.12:10038/riak/testbucket/helloworld
curl -v http://192.168.56.12:10048/riak/testbucket/helloworld
curl -v http://192.168.56.12:10058/riak/testbucket/helloworld
```

Što zaključujete?

Doznajte koliki je default replication factor za neki bucket (tražite "n_val"):

```
> curl http://192.168.56.12:10018/buckets/testbucket/props
```

Dodajmo još jedan zapis u bucket testbucket2:

```
> curl -v -X PUT http://192.168.56.12:10018/riak/testbucket2/helloworld -H
"Content-Type: text/html" -d "<html><body>Hello world\!</body></html>"
```

Dohvatimo sve postojeće buckete (možete dodati još kojeg)

```
> curl http://192.168.56.12:10018/riak?buckets=true
```

Dodajmo još jedan zapis u bucket testbucket2:

```
> curl -v -X PUT http://192.168.56.12:10018/riak/testbucket2/goodday -H
"Content-Type: text/html" -d "<html><body>Today is a good day.</body></html>"
```

i jedan bez ključa (Riak će generirati ključ). Primijetit da je potrebno koristiti POST a ne PUT:

```
> curl -v -X POST http://192.168.56.12:10018/riak/testbucket2 -H "Content-
Type: text/html" -d "<html><body>Generated key.</body></html>"
```

Koji je http response code?
Koji je ključ?
Dohvatite taj zapis (zamijenite ključ vlastitim).

```
> curl http://192.168.56.12:10018/riak/testbucket2/Jb3lPf8FJzv8IPfE6ZvwbWwcDnJ
```

Dohvatimo sve ključeve u bucketu testbucket2:

```
> curl http://192.168.56.12:10018/riak/testbucket2?keys=true
```

Konačno, obrišite jedan zapis iz testbucket2, npr. (u sljedećoj naredbi zamijenite ključ vlastitim):

```
> curl -v -X DELETE
http://192.168.56.12:10018/riak/testbucket2/Jb3lPf8FJzv8IPfE6ZvwbWwcDnJ
```

Koji je response code?
Probajte dohvatiti obrisani zapis.
Probajte još jednom obrisati (ponovite naredbu).

Pristup s vanjskog računala

S vanjskog (host) računala (nakon što se instalirali curl) dodajte neki zapis s proizvoljnim ključem pomoću curl programa (isto kao prije).

Pogledajte ga putem internet preglednika.

Unesite neku sliku u Riak (naravno, stavite neku sliku u tekući direktorij i zamijenite „slika.png“):

```
    > curl -X PUT http://192.168.56.12:10028/riak/images/slika.jpg -H "Content-
type: image/png" --data-binary @slika.jpg
```

Dohvatite ju iz internet preglednika s te iste adrese.

Osim slika, zgodno je pohranjivati i druge datoteke na ovaj način, na primjer, kod izrade drugog zadatka (M/R), na ovaj način možete pohraniti js datoteku:

```
    > curl -v -X PUT http://192.168.56.12:10018/riak/nmbp/nmbp_map.js -H
"Content-Type: text/javascript" --data-binary @nmbp_map.js
```

Zaustavljanje i ponovo pokretanje Riaka

Zaustavite Riak cluster:

```
    > dev5/bin/riak stop
    > dev4/bin/riak stop
    > dev3/bin/riak stop
    > dev2/bin/riak stop
    > dev1/bin/riak stop
```

Ponovo ga pokrenite:

```
  > dev5/bin/riak start
    > dev4/bin/riak start
    > dev3/bin/riak start
    > dev2/bin/riak start
    > dev1/bin/riak start
```

Upitajte:

```
    > curl http://192.168.56.12:10018/stats
```

i potražite „connected_nodes“.
Treba li ponovo formirati cluster?
Probajte, npr.:

```
    > dev2/bin/riak-admin cluster join dev1@192.168.56.12
```

Probajte još jednom i:

```
    > curl http://192.168.56.12:10018/riak/testbucket2?keys=true
```

da se uvjerite da su zapisi sačuvani.

Vektorski satovi

Napravimo novi bucket s postavkom allow_mult = true (default je false) čime dopuštamo stvaranje siblinga (*sibling* = brat ili sestra) zapisa.

U Riak postoje dva formata kako se postavljaju svojstva bucketa, stari i novi (http://docs.basho.com/riak/latest/dev/references/http/set-bucket-props/):

```
    PUT /riak/bucket                # Old format
    PUT /buckets/bucket/props       # New format
```

```
    > curl -v -X PUT http://192.168.56.12:10018/buckets/dogovor/props -H
"Content-Type: application/json" -d '{"props":{"allow_mult":true}}'
```

Dohvatimo sva svojstva i provjerimo je li uspješno postavljeno:

```
  > curl -v  http://192.168.56.12:10018/buckets/dogovor/props
```

Pretpostavimo da se Ljilja, Krešo i Jasna hoće dogovoriti koju ocjenu će dati studentu Igoru iz domaće zadaće.

(a) Krešo predlaže jedan:

```
    > curl -v -X PUT http://192.168.56.12:10018/buckets/dogovor/keys/igor -H
"Content-Type: application/json" -H "X-Riak-ClientId: kreso" -d '{"ocjena":"1"}'
```

Dohvatite zapis i zapišite vektorski sat:

```
    > curl -v http://192.168.56.12:10018/buckets/dogovor/keys/igor
```

npr. X-Riak-Vclock: a85hYGBgzGDKBVIcKlYHQoIWzJiQwZTImMfKsCZA6gxfFgA=
(vaš će sigurno biti drugačiji)

(b) Ljilja je pročitala što je Krešo predložio i predložila dovoljan (dakle, znajući za predloženu negativnu ocjenu). Uključivši VClock u zahtjev, Ljilja daje do znanja Riaku koju (prethodnu) verziju zapisa je vidjela:

```
    > curl -i -X PUT http://192.168.56.12:10018/buckets/dogovor/keys/igor \
        -H "X-Riak-ClientId: ljilja" \
        -H "X-Riak-Vclock: a85hYGBgzGDKBVIcKlYHQoIWzJiQwZTImMfKsCZA6gxfFgA=" \
        -H "Content-Type: application/json" \
        -d '{"ocjena" : 2}'
```

Dohvatimo opet zapis, da se uvjerimo da je sada ocjena=2.

```
       > curl -v http://192.168.56.12:10018/buckets/dogovor/keys/igor
```

Kakav je vektorski sat?
Usporedite ga po duljini s prethodnim.

(c) Neka je Jasna pročitala ocjenu prije Ljiljine izmjene (dok je ocjena bila nedovoljan). Jasna (nakon što je Ljilja već unijela dovoljan) potvrđuje nedovoljan i unosi:

```
    > curl -i -X PUT http://192.168.56.12:10018/buckets/dogovor/keys/igor \
        -H "X-Riak-ClientId: jasna" \
        -H "X-Riak-Vclock: a85hYGBgzGDKBVIcKlYHQoIWzJiQwZTImMfKsCZA6gxfFgA=" \
        -H "Content-Type: application/json" \
        -d '{"ocjena" : 1}'
```

Što bi sustav trebao napraviti?
Dohvatimo zapis:

```
    > curl -v http://192.168.56.12:10018/buckets/dogovor/keys/igor
```

Gdje su vrijednosti?
Koji je HTTP kod?

Kakav je vektorski sat?
Usporedite ga po duljini s prethodnim.

U ovom trenutku **treba razriješiti konflikt**.
Obje verzije (siblings, "brat i sestra") možete dohvatiti ako dodate -H "Accept: multipart/mixed":

```
   > curl -v http://192.168.56.12:10018/buckets/dogovor/keys/igor -H "Accept:
multipart/mixed"
```

ili ih možete dohvaćati jednog po jednog, koristeći šifre siblinga, npr. (zamijenite sa svojom šifrom):

```
 > curl -v
http://192.168.56.12:10018/buckets/dogovor/keys/igor?vtag=2vUo9hVift7TTf3q0LkG5h
  > curl -v
http://192.168.56.12:10018/buckets/dogovor/keys/igor?vtag=3dFQPS7ZQXoU9MUlXrYYdy
```

Konačno, **Ljilja** (srećom po Igora) **razrješava konflikt** s (zamijenite vektorski sat s vlastitim):

```
    > curl -i -X PUT http://192.168.56.12:10018/buckets/dogovor/keys/igor \
        -H "X-Riak-ClientId: ljilja" \
        -H "X-Riak-Vclock:
a85hYGBgymDKBVIccw9K7wlaMGNKBlMiYx4rQ2yQ1Bk+qJSK1YEQoNQEoBQzUEo0GCiVBQA=" \
        -H "Content-Type: application/json" \
        -d '{"ocjena" : 2}'
```

Provjerimo da je ostala jedna vrijednost (više nema siblinga):

```
    > curl -v http://192.168.56.12:10018/buckets/dogovor/keys/igor
```

Isprobajte (ili barem pročitajte) još i osnovni Riak MapReduce primjer:
http://docs.basho.com/riak/latest/dev/using/mapreduce/

Ovime smo isprobali osnove rada s Riakom.
Za dodatna pitanja studente upućujemo na dokumentaciju Riaka: http://docs.basho.com/riak/latest/

2. Zadaci koje je potrebno samostalno riješiti

Potrebno je napraviti dva zadatka opisana u nastavku.

1. NoSQL1: Minimalni portal temeljen na Riaku (10 bodova)

Napraviti web portal koji ispisuje najnovijih N (npr. N=10) vijesti.

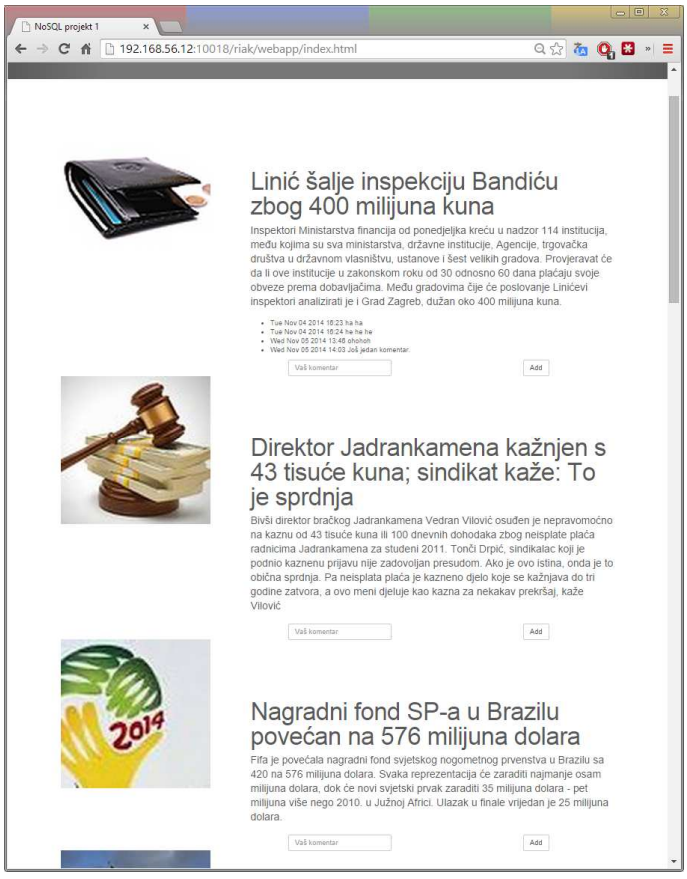
> Osmisliti kako, odnosno upotrijebiti prikladnu strukturu podataka za to.
> Na primjer, ako pretpostavimo da u bazi imamo 10.000 članaka, **nije dobra** strategija dohvatiti svih 10.000 zapisa na klijenta, sortirati i uzeti prvih deset.

Svaka vijest treba biti (otprilike) sljedećeg oblika:

Naslov	Slika
Tekst	
Autor	

Omogućiti komentiranje vijesti: ispod svake vijesti dodati polje za unos komentara.
Nije potrebno napraviti sučelje za unos vijesti, odnosno možete ih sve unijeti ručno koristeći npr. curl.

Npr.:



2. NoSQL2: MapReduce upit (3 + 7 = 10 bodova)

Napisati:

(a) (3 boda) MapReduce upit koji vraća listu članaka poredanu silazno po broju komentara.

(b) (7 bodova) MapReduce upit koji za svakog autora vraća prvih 10 najkorištenijih riječi.
Pojam „riječ“ shvatiti u najjednostavnijem mogućem obliku (niz slova odvojen od drugih s razmakom, zarezom ili točkom).
Nije potrebno raditi nikakve leksičke transformacije na riječima (svođene na korijen i sl.).
Nije potrebno ispisati i 11. riječ ako se ima isti broj korištenja kao 10. riječ.

Priznavati će se (sa smanjenim bodovima) i polovična rješenja, npr. sve riječi koje je autor koristio (a ne prvih 10) i sl.

Rezultat upita možete prikazati u istoj web aplikaciji, ali i ne morate, priznati će se i ako ga znate pokrenuti iz konzole.

Savjeti:

- Isprobajte prvo js programski kod u npr. Chrome js konzoli, te ga potom pohranite u neki bucket u Riaku
- Riak dokumentaciju s M/R javascript primjerima pogledajte na starijoj verziji dokumentacije (1.4): http://docs.basho.com/riak/1.4.1/dev/advanced/mapreduce/
- Debagiranje Riak js (preseno s Riak stranica):

Debugging Javascript MapReduce Phases

There are currently two facilities for debugging MapReduce phases. If there was an exception in the Javascript VM you can view the error in the `log/sasl-error.log` file. In addition to viewing exceptions you can write to a specific log file from your map or reduce phases using the ejsLog function.
Javascript:

```
ejsLog('/tmp/map_reduce.log', JSON.stringify(value))
```

Note that when used from a map phase the ejsLog function will create a file on each node on which the map phase runs. The output of a reduce phase will be located on the node you queried with your MapReduce function.

Za oba zadatka:

Rješenja projekta trebaju biti strukturirana na sljedeći način:

```
|-NoSql1
   |- main1.txt
   |- (...)         // ostale datoteke, npr. Slike, tekst, json i sl.
   |- (projekt.zip) // zip webapp projekta ako je riješeno na taj način.
|-NoSql2
   |- main2.txt
   |- (...) // ostale datoteke, npr. js datoteke.
```

U oba slučaja main?.txt datoteka treba korak po korak sadržavati opis i naredbe koje je potrebno obaviti kako bi se ostvario rezultat (nalik Uputama iz točke dva ovog dokumenta)

Studente se kod predavanja može pitati da:

- objasne nešto iz rješenja (main.txt)
- objasne neki koncept iz Uputa (npr. vektorski sat)
- pokrenu tj. demonstriraju rješenje (i npr. dodaju vijest na portal)
- naprave neke manje modifikacije (npr. blago modificirati M/R upit)
- itd.