

Visoko skalabilen NewSQL sistem za upravljanje s podatkovnimi bazami CockroachDB

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko

Avtor: Matjaž Mav

Mentor: izr. prof. dr. Matjaž Kukar

Ljubljana, 2018

Pozdravljeni,
sem **Matjaž Mav** in sedaj vam bom predstavil diplomsko delo z naslovom „**Visoko skalabilen NewSQL sistem za upravljanje s podatkovnimi bazami CockroachDB**“.

00:15

Vsebina

- Kaj je NewSQL
- Podatkovna baza CockroachDB
- Izvedba primerjalne analize
- Rezultati in ugotovitve

V diplomskem delu smo želeli **podrobneje pregledati kaj** nam jih ponuja podatkovna baza **CockroachDB**. V predstavitvi pa bom na kratko predstavil samo **idejo NewSQL** podatkovnih baz. Opisal bom podatkovno bazo **CockroachDB**, izvedbo **primerjalne analize**, **rezultate** in naše **ugotovitve**.

00:35

Kaj je NewSQL?

- SQL
- ACID transakcije
- skalabilnost
- visoka razpoložljivost
- porazdeljena okolja

Kratika **NewSQL** stoji za podatkovne baze, ki združujejo lastnosti tako **tradicionalnih relacijskih SQL**, karko tudi **novejših NoSQL** podatkovnih baz. S SQL sveta ohranijo „standardni“ **poizvedovalni jezik SQL** in **visoko konsistenco podatkov**, ki jo zagotavljajo preko **ACID transakcij**. Z NoSQL sveta pa **skalabilnost in visoko razpoložljivost**.

01:00

Podatkovna baza CockroachDB

- NewSQL
- kompatibilna s PostgreSQL
- odprtokodna
- dostopna komurkoli
- enostavna za uporabo
- aktivna skupnost



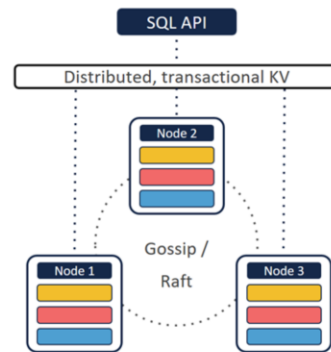
Slika 1: Celostna grafična podoba podatkovne baze CockroachDB
(vir: <https://github.com/cockroachdb/cockroach>)

Podatkovna baza **CockroachDB** je ena od **novih NewSQL** podatkovnih baz. Dostopna je komurkoli saj je odprtokodna, deluje pa na vseh glavnih operajskih sistemih, primerna pa je tudi za oblak. Je zelo enostavna za postavitve in upravljanje. Poleg tega pa ima aktivno skupnost.

01:30

Arhitektura podatkovne baze CockroachDB

- plast SQL
- transakcijska plast
- porazdelitvena plast
- replikacijska plast
- shranjevalna plast



Slika 2: Arhitekturni pregled
(vir: http://cs.ulb.ac.be/public/media/teaching/cockroachdb_2017.pdf)

Sama arhitektura podatkovne baze CockroachDB je razdeljena na **5 funkcionalnih plasti**. Če pogledamo **poenostavljen primer**:

Ker so vozlišča v gruči **simetrična**, se lahko odjemalec preko **PostgreSQL žičnega protokola** poveže do katerega koli vozlišča in izvrši **SQL poizvedbo**. Plast SQL to poizvedbo prevede v **množico KV operacij**, s katerimi noto operira naprej.

Vozlišče ugotovi, katero od vozlišč zna obdelati ta zahtevek in to posreduje naprej. To ugotovi preko **dvo nivojskega meta indeksa**, kateri prizvezo omejuje velikost podatkovne baze na **4EiB=2⁽²⁶⁺³⁶⁾B** (exbibyte 2⁶⁰B).

Podatki so razdeljeni v **obsege** (angl. ranges) kateri nosijo do 64MiB (2²⁶B). Vsak obseg je **repliciran 3 krat** in shranjen na 3 različnih vozliščih. Le eno od vozlišč lahko za določen obseg obdelava zahtevek, temu obsegu pravijo **najemnik** (angl. leaseholder).

Vsak pisalni zahtevek ustvari novo **verzijo podatka** in ga shrani v **KV shrambo**. Za KV shrambo CockroachDB uporablja odprtokodno shrambo **RocksDB**.

Postavitev testnega okolja

- štirje računalniki
- gigabitno omrežje
- Ubuntu Server 16.04 LTS
- Docker 18.03.0-ce
- CockroachDB 2.0.1
- PostgreSQL 10.3 + Citus 7.3.0

Za izvedbo primerjalne analize smo najprej **postavili testno okolje**. Uporabili smo **4 starejše računalnike**. En v vlogi odjemalca in tirje v vlogi strežnika. Vsi računalniki so bili med seboj povezani v **gigabitno ethernet omrežje**.

Na njih je tekel operacijski sistem **Ubuntu Server**. Zaradi lažjega upravljanja, ponovljivosti in primerljivosti rezultatov pa smo uporabili **Docker**. Tehnologije Docker v začetku še nismo dobro poznali, za to smo imeli v začetni fazi manjše **težave z zmogljivostjo**.

Pri primerjalni analizi zmogljivosti smo primerjali podatkovno bazo **CockroachDB 2.0.1** s že dobro uveljavljeno podatkovno bazo **PostgreSQL 10.3** z nameščeno razširitvijo **Citus 7.3.0**. Citus je **razširitev**, ki omogoča enostavno horizontalno skaliranje **večnajmeniških** (oziroma angleško multi tenant) **aplikacij**.

4:00

Izvedba primerjalne analize zmogljivosti

- orodje YCSB
- priprava podatkov
- parametri
- avtomatizacija testiranja
- github.com/matjazmav/diploma-ycsb

```
CREATE TABLE usertable (  
  YCSB_KEY VARCHAR(255) PRIMARY KEY,  
  FIELD0 TEXT, FIELD1 TEXT,  
  FIELD2 TEXT, FIELD3 TEXT,  
  FIELD4 TEXT, FIELD5 TEXT,  
  FIELD6 TEXT, FIELD7 TEXT,  
  FIELD8 TEXT, FIELD9 TEXT  
);
```

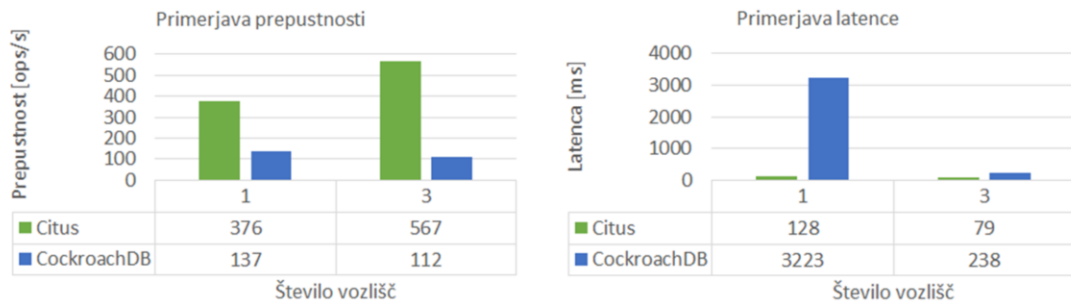
Za izvedbo primerjalne analize smo **pregledali več različnih orodij**. Na koncu pa smo se odločili za enostavno orodje Yahoo Cloud Storage Benchmarks oziroma **YCSB**. To orodje je enostavno in **namenjeno primerjavi med različnimi podatkovnimi bazami**, preko **JDBC** vmesnika podpira tudi podatkovni bazi **CockroachDB** in **PostgreSQL**. Orodje deluje nad **eno samo tabelo** in vrši različne vrste obremenitev.

Najprej smo ročno **pripravili podatke**, z orodjem smo generirali **5M vrstic**, kar na disku zavzame približno **~6GB prostora**.

Vsak test smo ponovili 3x, z različnimi parametri. Te parametri so **št. vozlišč** (1 in 3), **podatkovna baza** (CRDB in Postgres + Citus), **št. niti** (3:9:66) in **različne YCSB obremenitve** (A, B, C, D, F). Kar pomeni, da smo izvedli **480 meritev**. Za to smo si pripravili orodje s katerim smo to dokaj dobro avtomatizirali. Sama **izvedba avtoamtskih meritev** pa je trajala potem **slab teden**.

5:30

Rezultati primerjalne analize zmogljivosti



Ljubljana, 2018

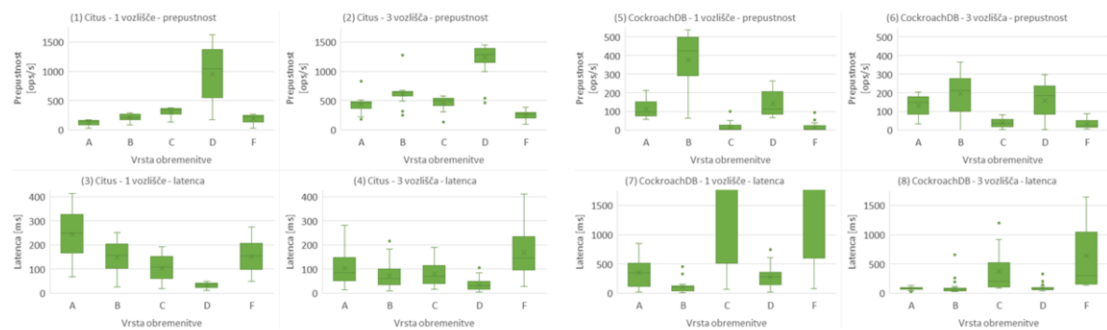
Visoko skalabilen NewSQL sistem za upravljanje s podatkovnimi bazami CockroachDB

8

Če pogledamo **grobe rezultate** primerjalne analize zmogljivosti, vidimo da je podatkovna baza **CockroachDB bistveno slabša** od podatkovne baze PostgreSQL. Na zgornjih grafih prikazujemo agregirane povprečne vrednosti za obe podatkovni bazi na enem in treh vozliščih. Na desni za prepustnost in na levi za latenco.

CockroachDB na treh vozliščih dosega kar **5x manjšo prepustnost** in **3x večjo latenco** kot podatkovna baza PostgreSQL. Opazimo tudi, da z skaliranjem podatkovne baze CockroachDB na 3 vozlišča **nismo povečali prepustnosti**.

06:10



Na zgornjih grafih **prikazujemo porazdelitve** prepustnosti in povprečnih latenc pri posameznih obremenitvah. Na desni za podatkovno bazo PostgreSQL in na levi za CockroachDB. Z zgornjih grafov opazimo predvsem to, da so **rezultati meritve** pri podatkovni bazi CockroachDB **veliko bolj razpršeni**.

06:40

Izvedba analize stičnih operacij

Priprava podatkov

```
CREATE TABLE ext (  
  ycsb_key VARCHAR(255) PRIMARY KEY,  
  value int NOT NULL);  
  
INSERT INTO ext  
SELECT  
  ycsb_key,  
  LTRIM(RIGHT(ycsb_key,5),0)::int % 10 value  
FROM usertable  
ORDER BY ycsb_key  
LIMIT 100;
```

Poizvedba

```
\timing  
  
SELECT u.ycsb_key, u.field4  
FROM usertable u  
INNER JOIN ext e  
  ON e.ycsb_key = u.ycsb_key  
WHERE e.value = 4;
```

Ker orodje **YCSB** izvaja teste nad samo **eno tabelo**, smo sami **ročno** preverili kako je s podporo **stičnih operacij** pri podatkovni bazi CockroachDB. Za osnovo smo vzeli podatke katere smo uporabili za izvedbo zmogljivostne analize. Dodali smo še eno tabelo in jo napolnili z 100 vrsticami.

Poizvedba katero smo testirali, je bila enostavna stična operacija INNER JOIN, katera združuje obe tabeli preko njunih primarnih ključev. Omejitev value = 4 pa omeji velikost rezultata na 11 vrstic.

Rezultati za podatkovno bazo CockroachDB so bili zelo slabi. Po 1 minuti smo poizvedbo prekinili, tako na enem kot tudi na treh vozliščih.

Podobno smo poizkusili tudi na podatkovni bazi PostgreSQL. Rezultate na enem vozlišču smo dobili po približno ~70ms in na treh po približno ~160ms.

07:40

Optimizacija stičnih operacij

- eksperimentalna zastavica
- vrstni red stičnih operacij
- / → ~800ms

```
SET experimental_force_lookup_join = true;
```

```
\timing
```

```
SELECT u.ycsb_key, u.field4  
FROM ext e  
INNER JOIN usertable u  
    ON e.ycsb_key = u.ycsb_key  
WHERE e.value = 4;
```

Zaradi res slabih rezultatov smo se **obrnili na razvijalce** podatkovne baze CockroachDB. Kateri so nam pomagali pri optimizaciji poizvedbe. Kakor prikazuje zgornja poizvedba, dodali smo **eksperimentalno zastavico** in **obrnili vrstni red stičnih operacij**.

Rezultate smo dobili po približno **~800ms**, ko smo poizvedbo **izvršili ponovno** smo dobili rezultate v dobrih **~5ms**. Medtem ko je **PostgreSQL** ob ponovni izvedbi poizvedbe rezultate vrnil **le malenkost**.

08:20

Ugotovitve

- slabši rezultati zmogljivostne analize
- slaba podpora stičnim operacijam
- slaba podpora obstoječim orodjem
- + zelo enostavno upravljanje
- + zadovojljiva kompatibilnost s PostgreSQL
- + aktivna skupnost
- ? druge obremenitve (TPC-C, ...)
- ? več vozlišč (3+)
- ? naslednje verzije (2.0.1+)

Podatkovna baza CockroachDB je na trgu šele **dobri 2 leti**. V tem diplomskem delu smo analizirali **verzijo 2.0.1** z aprila 2018. V primerjavi z dobro uveljavljeno podatkovno bazo **PostgreSQL** dosega CockroachDB bistveno **nižjo prepustnost in večjo latenco**. Poleg tega pa v ti verziji **implementacija stičnih operacij ni najbolj učinkovita**. Razvijalci obljubljajo **izboljšave** le teh v naslednjih verzijah.

Od prednosti bi izpostavil predvsem **enostavnost** z upravljanjem, dobro **dokumentacijo**, aktivno **skupnost** in relativno zadovoljivo **kompatibilnost** s podatkovno bazo PostgreSQL.

Poleg tega podjetje **CockroachLabs** **aktivno sodeluje z strankami**, podatkovno bazo CockroachDB pa uporablja že nekaj podjetji.

V prihodnje menim, da bi bilo zanimivo **slediti razvoju** te podatkovne baze. Poleg tega pa bi bilo smiselno izvesti primerjalno analizo zmogljivosti na **bolj realnih obremenitvah** in opazovati, kaj se dogaja ko v gručo povežemo **več vozlišč**.

Priporočila

- min 2 GB in 1 jedro
- CPU > RAM
- več vozlišč
- Google NTP storitev
- SSD
- hitro omrežje
- porazdeljene in replicirane podatke
- transakcijske obremenitve
- globalne aplikacije
- oblak

Hvala za pozornost!

YCSB vrste obremenitve

- izbira ključev po Zipfovi distribuciji (A, B, C, F)
- izbira vedno zadnjega ključa (D)

Vrsta obremenitve	SELECT	UPDATE	INSERT
A	50%	50%	
B	95%	5%	
C	100%		
D	95%		5%
F	50%	50%	

Izolacijski nivoji

- Serializable
- zastareli Snapshot

Izolacijski nivo	Dirty reads	Non-repeatable reads	Phantoms
Read Uncommitted	X	X	X
Read Committed		X	X
Repeatable Read			X
Serializable			

CockroachDB omogoča le dve stopnji izolacije. Privzeto Serializable omogoča pa tudi zastareli ne standardni Snapshot izolacijski nivo.

Snapshot prav tako kot Serializabe ne dovoljuje nobenih anomalij, omogoča pa da podatke katere je ena transakcija že prebrala druga spremeni.