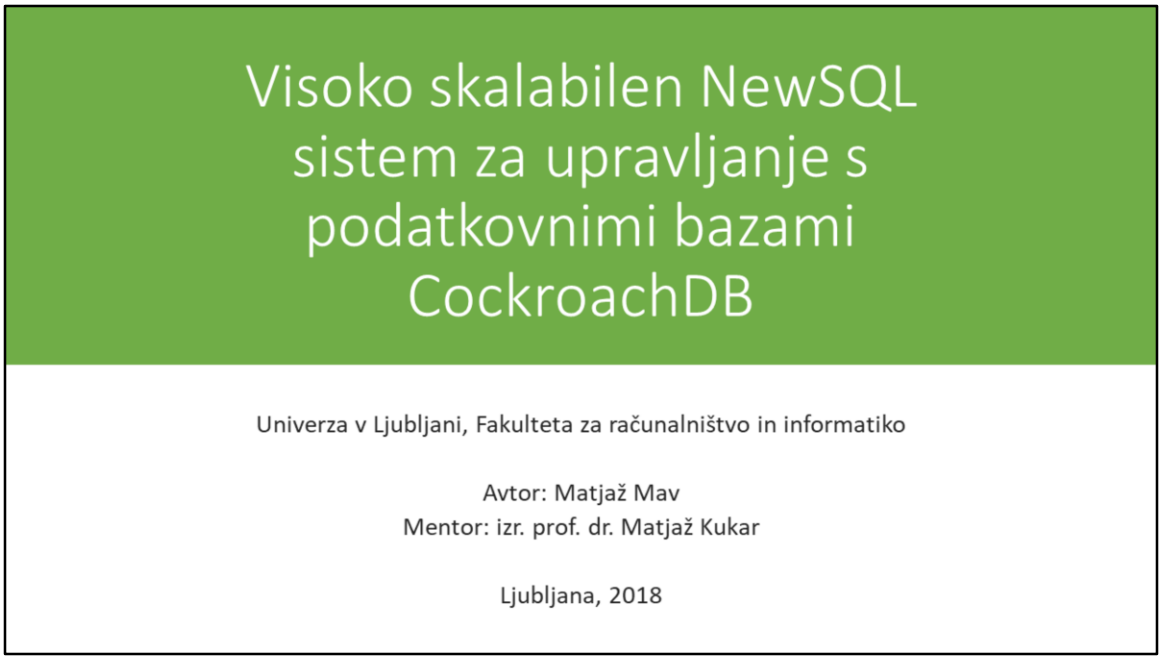
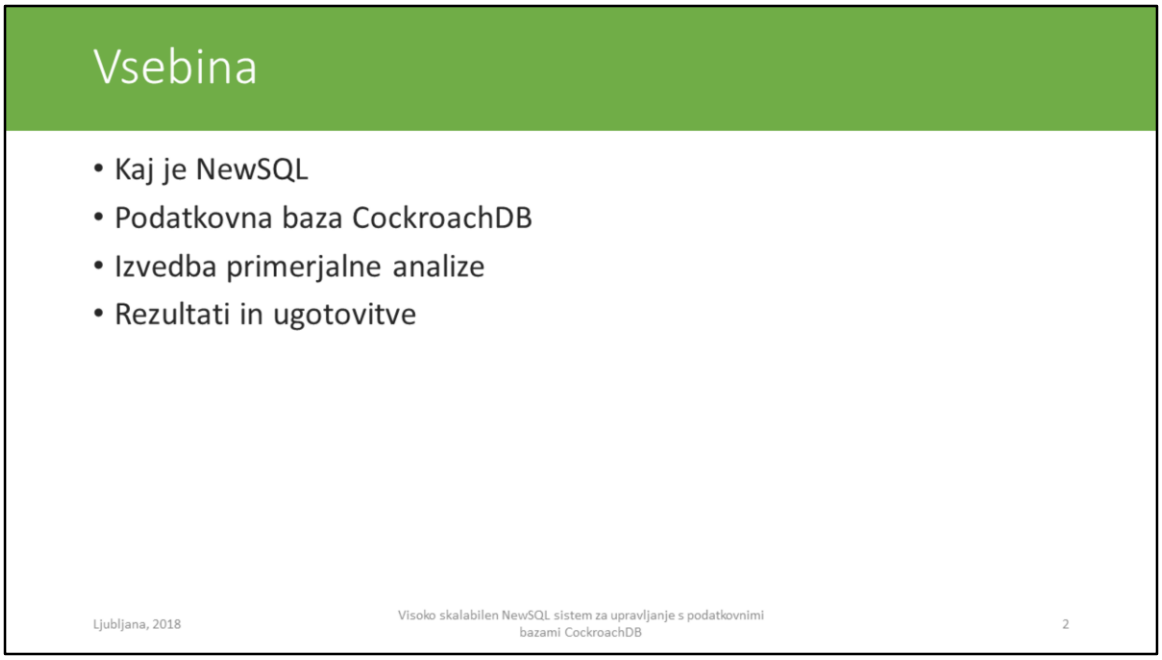


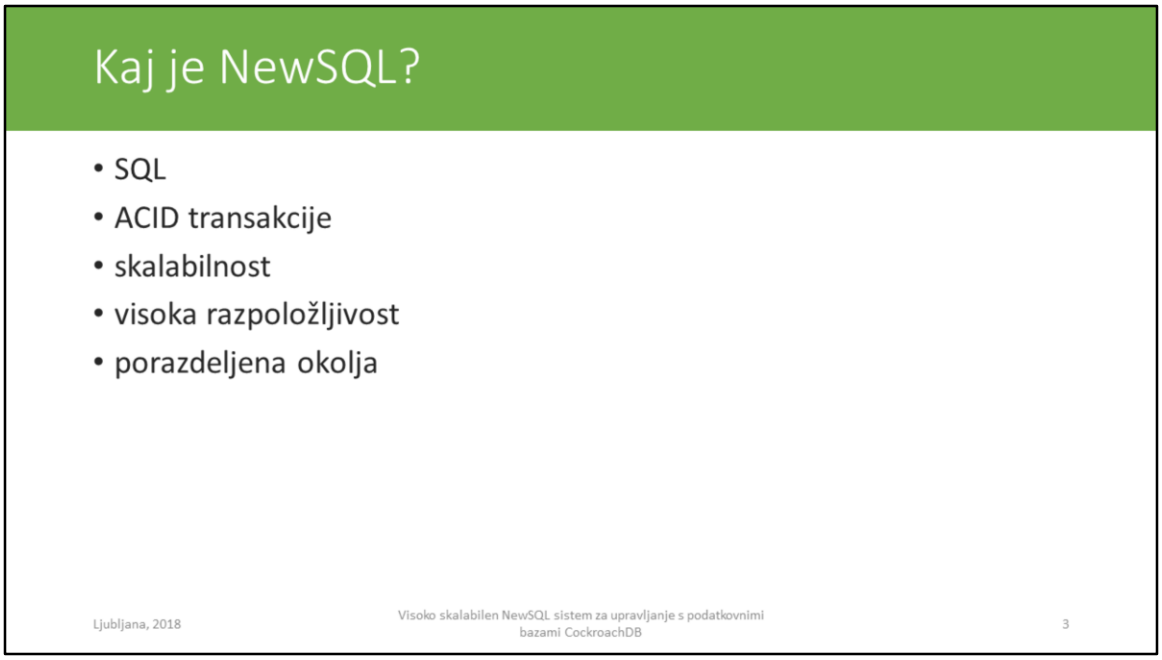
Pozdravljeni,
sem **Matjaž Mav** in sedaj vam bom predstavil diplomsko delo z naslovom „**Visoko skalabilen NewSQL sistem za upravljanje s podatkovnimi bazami CockroachDB“**.

00:15



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

00:35



Kratica **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

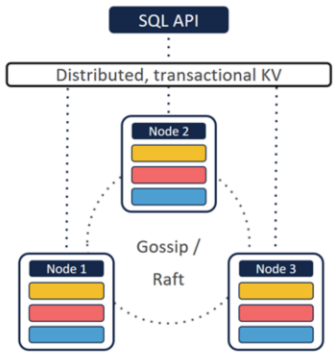
Slika 1: Celostna grafična podoba podatkovne baze CockroachDB (vir: https://github.com/cockroachdb/cockroach)

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 postavitev 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**:

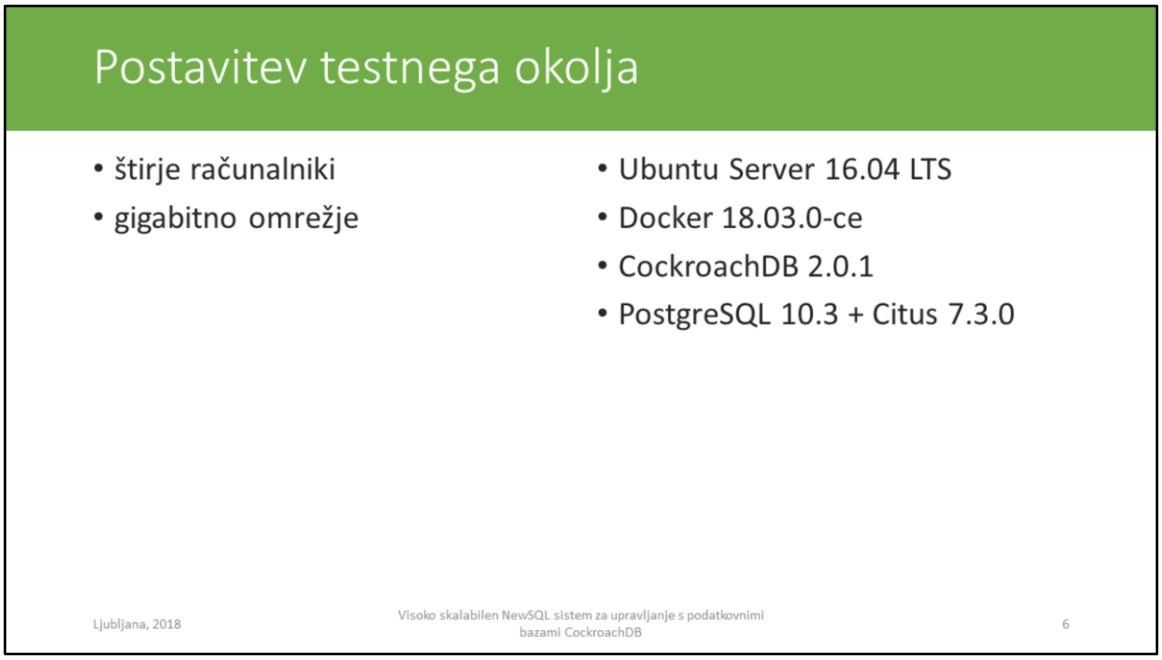
Ker so vozlišča v gruči **simetrična,** se lahko odjemalec preko **PostgreSQL žičnega protokola** poveže do katerega koli vozliša in izvrši **SQL poizvedbo**. Plast SQL to poizvedbo prevede v **množico KV operacij**, s katerimi noto operira naprej.

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 2^60B).

Podatki so razdeljeni v **obsege** (angl. ranges) kateri nosijo do 64MiB (2^26B). Vsak obseg je **repliciran 3 krat** in shranjen na 3 različnih vozliščih. Le eno od vozlišč lahko za določen obseg obdela zahtevek, temu obsegu pravijo **najemnik** (angl. leaseholder).

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

03:00

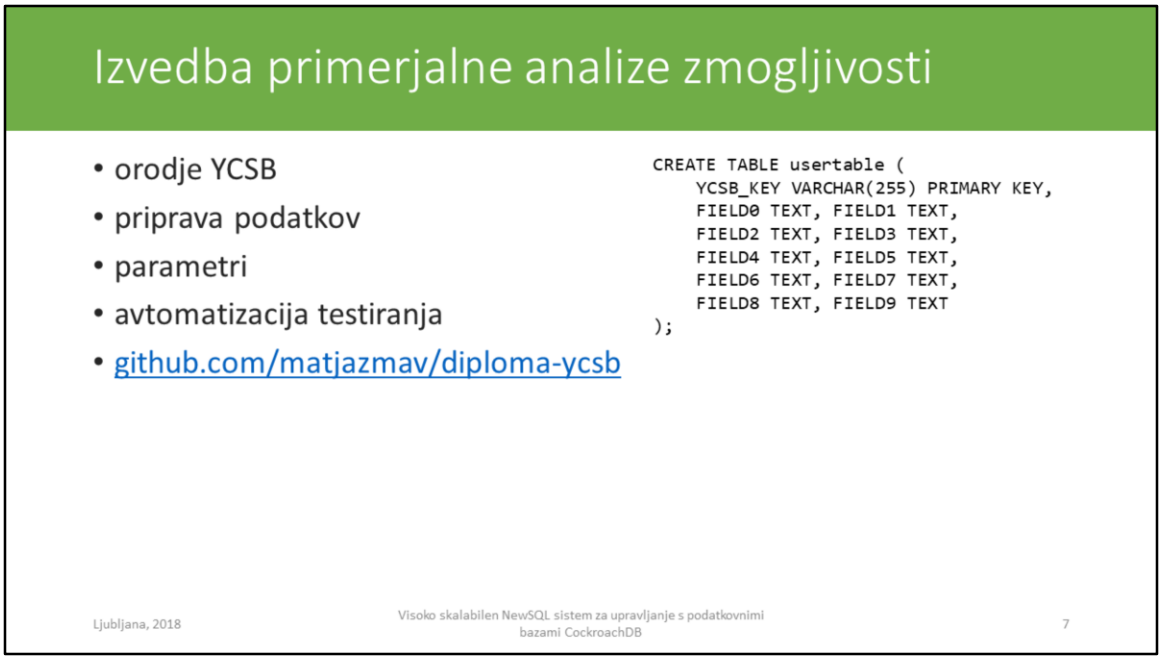


Za izvedbo primerjalne analize smo najprej **postavili testno okolje**. Uporabili smo **4 starejše računalnike**. En v vlogi odjemalca in tirje v vlogi strežnika. Vsi računalniki so bili med seboj povezani v **gigabitno ethernet omrežje**.

Na njih je tekel operacijski sistem **Ubuntu Server**. Zaradi lažjega upravlanja, ponovljivosti in primerljivosti razultatov pa smo uporabili **Docker**. Tehnologije Docker v začetku še nismo doboro 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



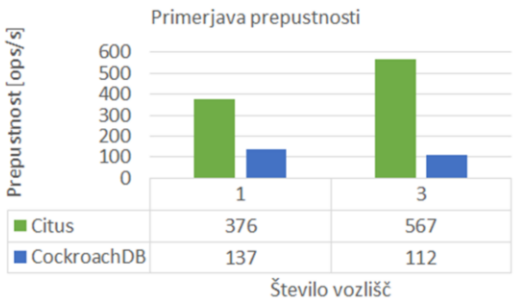
Za izvedbo primerjalne analize smo **pregledali več različnih orodij**. Na koncu pa smo se odločili za enostavno orodje Yahoo Cloud Storage Banchmarks oziroma **YCSB**. To orodje je enostavno in **namenjeno primerjavi med različnimi podatkovnimi bazami**, preko **JDBC** vmesnika podpira tudi podatkovni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

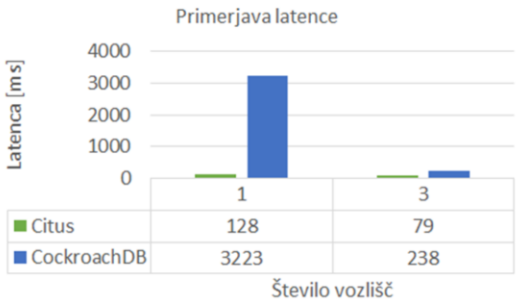
Najprej smo ročno **pripravili podatke**, z orodjem smo generirali **5M vrstic**, kar na disku zavzame približno **~6GB prostora**.

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

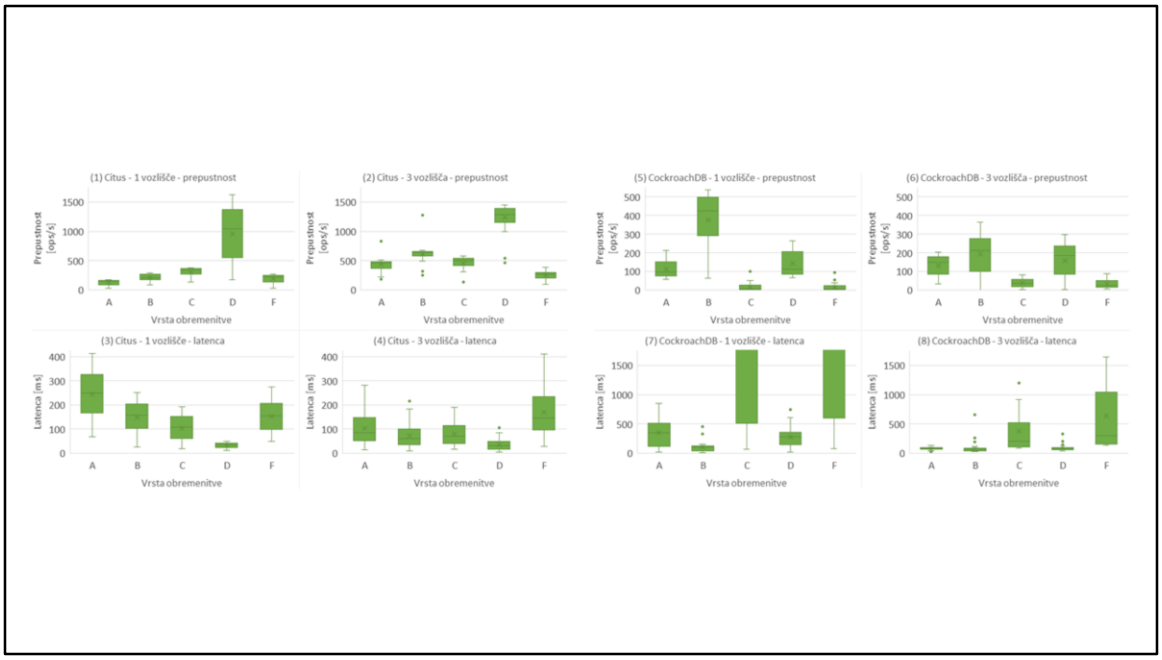




Če pogledamo **grobe razultate** primerjalne analize zmogljivosti, vidimo da je podatkovna baza **CockroachDB bistveno slabša** od podatkovne baze PostgreSQL. Na zgornjih grafih prikazujemo agregirane povprečne vrednosti za obe podatkovni bazi na enem in treh vozliščih. Na desni za prepustnost in na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

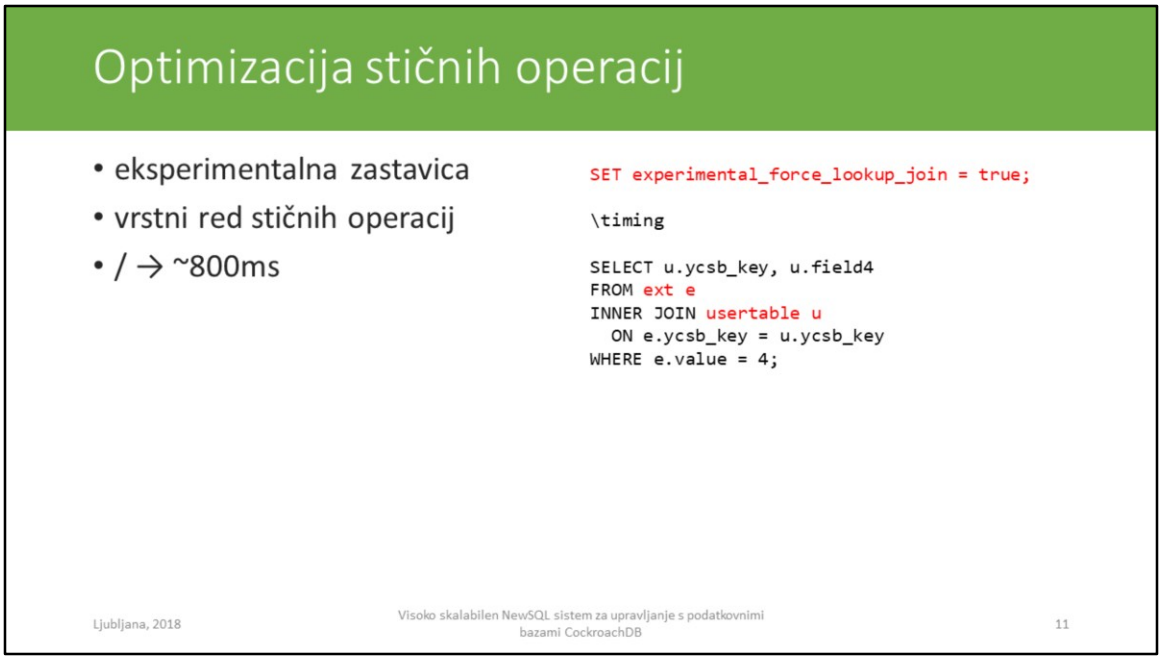

Ker orodje **YCSB** izvaja teste nad samo **eno tebelo**, smo sami **ročno** preverili kako je s podporo **stičnih operacij** pri podatkovni bazi CockroachDB. Za osnovo smo vzeli podatke katere smo uporabili za izvedbo zmogljivostne analize. Dodali smo še eno tabelo in jo napolnili z 100 vrsticami.

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 CockraochDB 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.

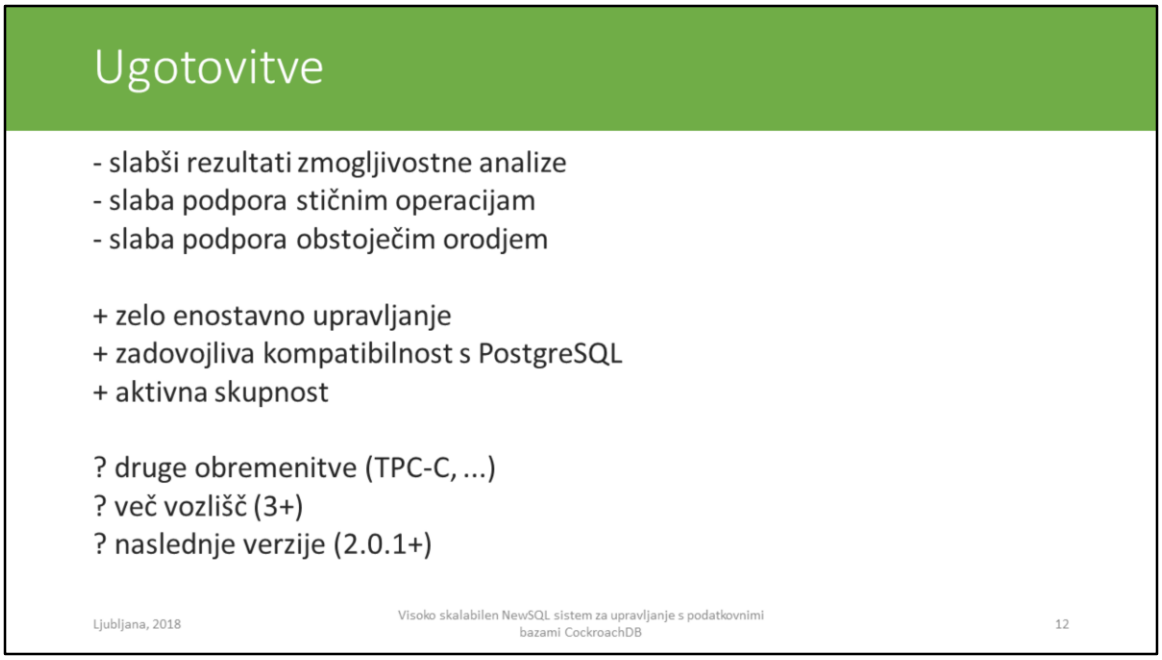
07:40



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



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 nalsednih verzijah.

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

Poleg tega podjetje **CockraochLabs aktivno sodeluje z strankami**, podatkovno bazo CockraochDB 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**.

09:30