NERELACIONE BAZE PODATAKA: MOGUĆNOSTI I OGRANIČENJA

Slađana Janković, Snežana Mladenović, Ana Uzelac, Stefan Zdravković
Univerzitet u Beogradu - Saobraćajni fakultet,
s.jankovic@sf.bg.ac.rs, snezanam@sf.bg.ac.rs,
ana.uzelac@sf.bg.ac.rs, s.zdravkovic@sf.bg.ac.rs

Sadržaj: *Modeli podataka i tehnologije baza podataka koje su ih pratile doživeli su do sada tri revolucije. Prvu revoluciju donelo je pojavljivanje hijerarhijskog i mrežnog modela podataka, drugu - relacioni model podataka, a treću -nerelacione (NoSQL) baze podataka. Prve NoSQL baze podataka nastale su sa ciljem da zadovolje potrebe Web 2.0, zbog čega su distribuirane, open-source i horizontalno skalabilne. Danas su ove baze jedan od nosećih stubova Big Data tehnologija. U poređenju sa relacionim bazama podataka, one omogućavaju veću raspoloživost, ali se kod ovih baza podataka, ne može garantovati konzistentnost svih podataka u svakom trenutku. Do danas su razvijeni sledeći nerelacioni modeli podataka: ključ-vrednost, kolonski, grafovski i model zasnovan na dokumentima. Ovaj rad opisuje kontekst u kom su nastale nerelacione baze podataka, predstavlja njihove zajedničke karakteristike, prednosti i ograničenja u korišćenju, a na primerima demonstrira osobenosti svakog od četiri u ovom trenutku prisutna nerelaciona modela podataka.*

Ključne reči: *NoSQL baze podataka, ključ-vrednost baze podataka, kolonske baze podataka, baze dokumenata, grafovske baze podataka*

1. Uvod

Dosadašnji razvoj tehnologija baza podataka jasno je podeljen na tri epohe. Prva epoha bila je period od nastanka elektronskih računara do pojave relacionog modela podataka, tj. od 1950-1970. godine. Tokom prve epohe za dominaciju su se borili hijerarhijski i mrežni modeli podataka i njima odgovarajući sistemi za upravljanje bazama podataka (SUBP). Engleski matematičar i programer *Edgar Frank Codd*, zaposlen u istraživačkoj laboratoriji IBM-a, 1970. godine objavio je naučni rad u kojem je dao osnove relacionog modela podataka [1]. Objavljivanje ovog rada predstavlja početak druge, do sada najznačajnije epohe u razvoju baza podataka. Tokom naredne tri decenije dominirale su relacione baze podataka bez značajnije konkurencije. Prve godine XXI veka donele su nove modele podataka i nove tehnologije baza podataka, što označava početak treće epohe u razvoju baza. Baze podataka treće generacije imaju jednu zajedničku karakteristiku – nisu relacione. Stoga su dobile zajedničko ime – nerelacione baze podataka. Kao sinonim za nerelacione baze podataka, 2009. godine pojavio se termin *NoSQL* baze podataka.

Suprotno očekivanom, *NoSQL* ne znači "*Not SQL*" već ima značenje "*Not only SQL*". Termin *NoSQL* ne predstavlja jedan proizvod ili tehnologiju, već klasu proizvoda i kolekciju različitih, ponekad povezanih koncepata o načinima skladištenja i manipulacije podacima.

Druga sekcija rada sadrži uporednu analizu relacionog i osnovnih nerelacionih modela podataka i njima odgovarajućih predstavnika sistema za upravljanje bazama podataka. Treća sekcija rada ističe prednosti, ali prepoznaje i ograničenja u korišćenju nerelacionih baza podataka. U poslednjoj sekciji rada dati su zaključci o najistaknutijim vrstama baza podataka danas, kao i pravci budućih istraživanja.

2. Nerelacione *vs.* relacione baze podataka

Savremeni internet i njegovi servisi neprekidno generišu ogromne količine nestrukturiranih ili polustrukturiranih podataka, kao što su: senzorski podaci, podaci koji se dele putem društvenih mreža, lična podešavanja vezana za različite naloge, fotografije, informacije o lokaciji uređaja, *online* aktivnosti i dr. Pokušaji da se skladište, obrade i analiziraju ovakvi podaci, uz pomoć relacionih baza podataka, otkrili su ograničenja ove vrste baza.

Osnovni razlozi koji su inicirali razvoj nerelacionih baza podataka su: internetom izazvana eksplozija količine i vrsta podataka, kao i broja istovremenih zahteva nad bazom podataka i nepredvidiva dužina trajanja *ACID* (*Atomicity* - atomičnost, *Consistency* - konzistentnost, *Isolation* - izolacija, *Durability* - trajnost) transakcije u *Big Data* okruženju. *ACID* osobine transakcija definisao je američki naučnik *Jim Gray* krajem 1970-ih godina [2]. One karakterišu transakcije koje se izvršavaju u relacionim bazama podataka. Kod nerelacionih baza podataka, da bi se transakcija izvršila u razumnom vremenu, nije uvek moguće obezbediti da pre početka i posle okončanja transakcije stanje baze podataka zadovoljava uslove konzistentnosti. Osobine transakcija kod *NoSQL* sistema biće detaljno razmatrane u trećoj sekciji rada.

Iako postoje na desetine nerelacionih baza, prema modelu podataka sve se mogu svrstati u jednu od četiri kategorije: ključ-vrednost baze podataka, kolonske baze podataka, baze podataka bazirane na dokumentima i grafovske baze podataka. Sledi kratak opis relacionih i sve četiri kategorije nerelacionih baza podataka.

2.1. Relacione baze podataka

Relacione baze podataka (RBP) predstavljaju poseban tip baza podataka čija je organizacija podataka zasnovana na relacionom modelu. U ovakvim bazama podataka podaci se organizuju u skup relacionih tabela (relacija) između kojih mogu biti kreirani određeni tipovi veza. U RBP svaka relacija mora da ima primarni ključ, odnosno skup atributa pomoću kojeg se jedinstveno identifikuje svaka *n*-torka. Skup atributa preko kojih se ostvaruje veza sa nekom drugom relacijom naziva se spoljni ključ. U relacionim sistemima za upravljanje bazama podataka (RSUBP) uglavnom se koristi najmoćniji upitni jezik danas - *SQL* (*Structured Query Language*) ili neki od njegovih dijalekata. Važna prednost RSUBP ogleda se u tome što pružaju izvanrednu sigurnost. Administrator baze podataka korisnicima može dodeliti određene privilegije kojima ih ovlašćuje za odgovarajuće operacije u bazi podataka.

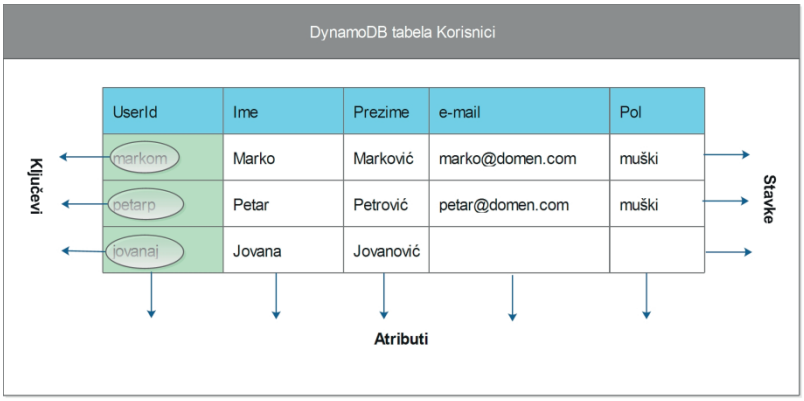
Transakcije RBP često su kompleksne jer zahtevaju spajanje tabela i izvršavanje operacija nad više zapisa. Ukoliko je potrebno izvršiti veliki broj transakcija u jedinici vremena, kao kod zadataka *Big Data* analize, kompleksnost transakcija može biti značajan nedostatak. RBP projektovane su tako da nisu horizontalno već samo vertikalno skalabilne. U slučaju povećanja opterećenja baze podataka, kod relacionih baza, bolji rezultati se postižu pojačavanjem resursa servera (hard diska, memorije i procesora), nego deljenjem opterećenja na više servera. Međutim, ovakva rešenja mogu da budu izuzetno skupa i u slučaju primene kod *Big Data* ekonomski su neisplativa. *Big Data* okruženje podrazumeva i česta pojavljivanja novih izvora i formata podataka. Nefleksibilan relacioni model podataka nije odgovarajući izbor u takvim slučajevima.

2.2. Ključ-vrednost baze podataka

Ključ-vrednost baze podataka predstavljaju najjednostavniji tip *NoSQL* baza podataka. Svaki zapis se čuva kao par sastavljen od ključa i njemu odgovarajuće vrednosti, pri čemu je vrednost potpuno nepoznata sistemu i može joj se pristupiti samo putem ključa. Koristi se za prikaz polimorfnih i nestrukturiranih podataka jer omogućuje skladištenje podataka bez prethodno definisane šeme, korišćenjem ključ-vrednost parova. Imaju mogućnost primanja i ažuriranja podataka zasnovanih samo na ograničenom skupu vrednosti dok je za potrebe upita nad drugim vrednostima neophodno kreirati sopstvene indekse. Ažuriranje ovakvih sistema zahteva više prolaza kroz bazu: prvi kojim se pronalazi zapis, drugi kojim se ažurira zapis i treći kojim se ažurira indeks. Koriste se u slučajevima kada postoji potreba da se podatku pristupi pomoću jedne ključne reči. Ključ-vrednost baze podataka dozvoljavaju horizontalno skaliranje u meri u kojoj to neke druge baze ne omogućavaju.

DynamoDB je prva u nizu ključ-vrednost baza podataka i po uzoru na nju su nastale mnoge druge baze ovog tipa, mada postoji i jedan broj ključ-vrednost baza podataka koje se dosta razlikuju od nje kao što su *Redis* i *Oracle NoSQL*. *DynamoDB* je napravljen od strane Amazona sa ciljem da podrži njihovu veliku *online* prodavnicu. Infrastruktura *DynamoDB* se sastoji od desetina hiljada servera i mrežnih komponentni lociranih u mnogim centrima podataka širom sveta [3]. *DynamoDB* tabele nemaju fiksnu šemu dok njeni objekti mogu imati proizvoljan broj atributa. Prednost ove baze podataka je što raste paralelno sa podacima, a za poboljšanje performansi je dovoljno dodati nove servere. U *DynamoDB*, tabela predstavlja kolekciju stavki a svaka stavka predstavlja kolekciju ključ-vrednost parova koji predstavljaju atribute. U *DynamoDB* postoji podrška i za primarne ključeve koji su unutar tabele jedinstveni i služe da jedinstveno odrede stavku. Primarni ključ je predstavljen putem heša ili putem heša sa opsegom. Heš ključ koristi jedan atribut od koga se kreira heš indeks. Da bi se pronašla stavka sa ovim ključem, neophodno je znati njegovu tačnu vrednost. Na primer, ako bi postojala tabela korisnika koja koristi korisnički *UserId* kao primarni heš ključ, do podataka o korisniku bi bilo moguće doći korišćenjem *UserId* vrednosti. Da bi se kreirali robusniji indeksi koriste se dva atributa: heš i opseg vrednosti. Pomoću ova dva atributa moguće je obaviti pretragu opsega od neke početne tačke. Na primer u tabeli gde se čuvaju poruke korisnika može se koristiti *UserId* kao heš ključ, a za opseg vremenska oznaka pa bi na taj način bilo moguće doći do poruka datog korisnika koje su novije od zadate vremenske tačke [4]. Na Slici 1 dat je prikaz tabele *Korisnici* u *DynamoDB*. Tip podataka *list* u *DynamoDB* može da čuva uređenu kolekciju

vrednosti različitih tipova. Pogledu u RBP odgovara globalni sekundarni indeks u *DynamoDB*.



Slika 1. Primer tabele u DynamoDB

2.3. Kolonske baze podataka

Za razliku od ključ-vrednost baza podataka, u kolonskim bazama podataka podacima se pristupa po kolonama, a ne po vrednostima. Obrada po kolonama prilikom paralelnog procesiranja doprinosi boljim performansama. Za čuvanje podataka se koriste kolonske familije, odnosno multidimenzione sortirane mape. Kolone imaju ime i pamte veći broj vrednosti po vrsti koje su identifikovane vremenskom oznakom, tako da predstavljaju trodimenzionalnu strukturu i do svake vrednosti se može doći korišćenjem ključ-vrste, ključ-kolone i vremenske oznake. Primer ovakve baze podataka je *Cassandra* koja predstavlja distribuiranu bazu podataka i koristi se prilikom obrade velike količine brzorastućih podataka. Na Slici 2 dat je primer tabele u kolonskoj bazi *Cassandra*. Podaci su organizovani u particije, svaka particija sadrži više kolona, a particije se skladište u čvorove [5]. Čvorovi predstavljaju deo klastera gde je svaki čvor zadužen za delić particije. Svaki ključ particionisanja se sastoji od jedne ili više kolona. Druge popularne baze ovog tipa su: *MariaDB*, *CrateDB*, *ClickHouse*, *Apache HBase*, *Hypertable*, itd.

Slika 2. Primer tabele u kolonskoj bazi Cassandra

2.4. Baze podataka bazirane na dokumentima

Za razliku od prethodno opisanih vrsta baza podataka, ova baza podatke skladišti u dokumentima. Ovi dokumenti imaju strukturu nalik *JSON* (*JavaScript Object Notation*) formatu. Svaki dokument opisuje jedan objekat, a sadrži jedno ili više polja. Svako polje sadrži jednu vrednost nekog od predefinisanih tipova: tekstualnog, binarnog, decimalnog, datumskog ili tipa niza. Ovaj model podiže produktivnost u razvoju baza podataka jer se podaci mogu unositi onakvi kakvi jesu, bez prethodnih priprema. Model podataka je takav da obezbeđuje jednostavan i brz pristup podacima.

Najpopularnija među bazama ove vrste danas je *MongoDB*. *MongoDB* baza podataka sastoji se od kolekcija (*Collections)*. U kolekcije se dodaju dokumenti. Ono što su kod relacionih baza tabele, to su kod *MongoDB* baze kolekcije, dok zapisima u relacionim bazama odgovaraju dokumenti kod *MongoDB* baze. Ključna razlika između ovih koncepata je u tome što svi zapisi jedne relacione tabele moraju imati istu strukturu, dok dokumenti jedne kolekcije mogu imati različita polja. Osim toga, istoimena polja u različitim dokumenta mogu biti različitih tipova podataka, čak i kada modeliraju istu vrstu objekata. U ovim osobinama ogleda se fleksibilnost modela podataka baziranog na dokumentima. Između *MongoDB* dokumenata mogu se kreirati veze tipa *1:1* ili *1:N*. Obe vrste veza mogu se ostvariti na dva načina: uz pomoć ugnježdenih dokumenata (Slika 3) ili referenciranjem dokumenata (Slika 4). Implementiranjem veza između dokumenata u *MongoDB* bazi obezbeđuje se referencijalni integritet podataka.

```
{
  _id: "marko",
  ime: "Marko Marković",
  adrese: [
            {
             ulica: "Petra Lekovića 74",
             grad: "Užice",
             država: "Srbija"
            },
            {
             ulica: "Kostolačka 123",
             grad: "Beograd",
             država: " Srbija "
            }
           ]
}
```

Slika 3. Veza 1:N ostvarena uz pomoć ugnježdenih dokumenata

```
{
  _id: "orm",
  naziv: "O'Reilly Media",
  godina_osnivanja: 1980,
  država: "SAD",
  knjige: [123456789, 234567890]
}

{
  _id: 123456789,
  naslov: "MongoDB: The Definitive
  Guide",
  autor: [ "Kristina Chodorow", "Mike
  Dirolf" ],
  broj_strana: 216,
  jezik: "Engleski"
}

{
  _id: 234567890,
  naslov: "50 Tips and Tricks for MongoDB
Developer",
  autor: "Kristina Chodorow",
  broj_strana: 68,
  jezik: "Engleski "
}
```

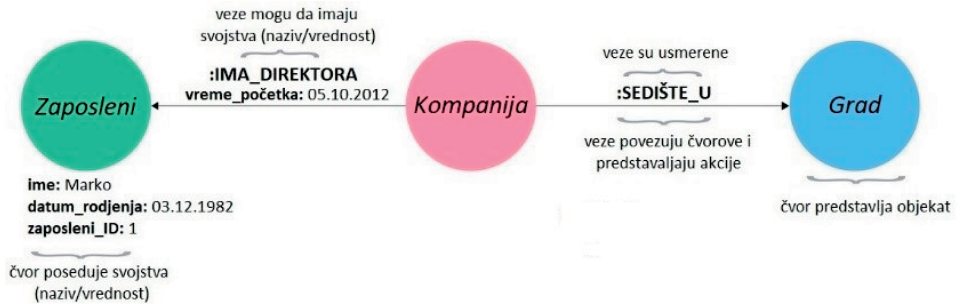
Slika 4. Veza 1:N ostvarena referenciranjem dokumenata

MongoDB pruža bogat skup opcija indeksiranja radi optimizovanja različitih vrsta upita. Osim standardnih indeksa nad jednim poljem, ovaj proizvod nudi i sledeće vrste indeksa: složene (*Compound*), tekstualne (*Text*), geoprostorne (*Geospatial*), indekse za polja koja sadrže nizove (*Multikey*), indekse koji omogućavaju upravljanje životnim vekom podataka (*Time to Live Indexes*), jedinstvene indekse (*Unique Indexes*) i druge. Uz pomoć svog *Aggregation Framework*-a *MongoDB* omogućava analiziranje podataka u samoj bazi, bez potrebe za kopiranjem podataka u analitičke softverske sisteme. U *Big Data* slučajevima upotrebe ovo je veoma važna osobina baze podataka. Ono što *MongoDB* bazu takođe kandiduje za *Big Data* skladište jeste njena sposobnost skladištenja fajlova bilo koje vrste i veličine. Fajlovi koji se čuvaju u bazi mogu se povezati sa dokumentima iz *MongoDB* kolekcija.

Osim *MongoDB* baze, trenutno najpopularnije baze ovog tipa su: *Amazon WorkDocs*, *ArangoDB*, *Amazon DynamoDB*, *Azure Cosmos DB*, *MarkLogic*, itd.

2.5. Grafovske baze podataka

Grafovske baze podataka zasnovane su na teoriji grafova uzimajući u obzir veze između podataka i tipove podataka. Grafovi se sastoje od čvorova (entiteta), koji mogu da budu međusobno povezani neodređenim brojem veza (grana) i svojstava koja predstavljaju atribute čvorova ili veza. Cilj projektovanja ovakve baze je čuvanje podataka bez ograničenja nametnutih unapred definisanim modelom. Umesto toga, podaci se organizuju na način koji diktiraju priroda podataka i njihovih međusobnih veza. Organizacija podataka u bazi pokazuje kako se svaki pojedinačni entitet povezuje, odnosno u kakvoj je relaciji sa drugim entitetima. Samoreferencirajući podaci predstavljaju jedan od značajnih problema RBP. U grafovksim bazama podataka kao što je npr. *Neo4j* ovaj problem je prevaziđen brzim prolaskom kroz čvorove i njihove veze, kako bi se pronašli relevantni podaci [6]. *Neo4j* predstavlja tipičnu grafovsku bazu podataka koja čuva podatke u formatu koji je optimizovan za grafovsku obradu u realnom vremenu. To istovremeno znači da ovo nije baza podataka opšte namene i da ima delimičnu primenu u druge svrhe. Na Slici 5 može se videti primer čvorova, usmerenih veza i svojstava u grafovskoj bazi podataka.



Slika 5. Primer grafovskog modela podataka [2]

Posebna prednost ovakve baze podataka je čuvanje podataka u obliku u kom su uneti, kao i korišćenje pokazivača za navigaciju i prolazak kroz graf. U najpopularnije grafovske baze podataka spadaju još: *AllegroGraph*, *GraphDB Lite*, *Amazon Neptune*, *AnzoGraph*, *ArangoDB*, *InfiniteGraph*, *InfoGrid*, *HyperGraphDB*, itd. Ove baze nastale su

popularizacijom društvenih mreža gde postoji veliki broj korisnika koji poseduje određene veze sa drugim korisnicima, komentarima, slikama i statusima.

Grafovske baze podataka ne podržavaju deklarativni upitni jezik visokog nivoa poput *SQL*-a, kako bi se izbeglo prekovremeno izvršavanje upita. Umesto toga upiti nad ovakvim bazama zavise od konkretnog modela podataka [7]. Jezici *SPARQL*, *Cypher* i *Gremlin* su optimizovani za operacije prolaska kroz graf. Oni imaju sintaksu koja omogućava prolazak kroz graf bez potrebe za rekurzivnim spojevima koji su karakteristčni za RBP. Retko se koriste u opštoj operativnoj primeni. Umesto toga povezuju se sa dokumentima ili relacionim bazama podataka u vidu multimodalne baze podatka kako bi ostvarili specifičnu grafovsku strukturu podataka. Grafovske baze podataka imaju specifičnu primenu i od njih se ne očekuje da zamene RBP. One predstavljaju dobru alternativu relacionim bazama u situacijama kada je pored samih objekata od velikog značaja i analiza odnosa između njih.

3. Mogućnosti i ograničenja u korišćenju nerelacionih baza podataka

Nerelacione baze podataka nastale su kao odgovor na rastuće potrebe *Web 2.0* aplikacija, kao što su: društvene mreže, vikipedije, blogovi, folksonomije, *online* deljenje video zapisa, *web* i mobilne aplikacije ("*Apss*"), itd. U skladu sa njihovom osnovnom namenom, sposobne su da prihvate velike količine nestrukturiranih, heterogenih podataka i efikasno se skaliraju po potrebi, ali ne mogu uvek garantovati konzistentnost podataka.

3.1. Mogućnosti

Najvažnije prednosti nerelacionih baza podataka, u poređenju sa relacionim, su: elastično, horizontalno skaliranje, mnogo bolje odgovaraju zahtevima *Big Data*, manje su zahtevne u pogledu održavanja, ekonomičnije su i nude fleksibilnije modele podataka.

Decenijama unazad administratori baza podataka radije su se oslanjali na vertikalno skaliranje (*scale vertically or scale up/down*), nego na horizontalno skaliranje (*scale horizontally or scale out/in*). To znači da su radije kupovali moćnije servere kako je opterećenje baza raslo, nego što su distribuirali baze podataka na više čvorova. Jedan od razloga za to bio je što relacione baze podataka nisu dizajnirane za horizontalno skaliranje. Ipak, kako su brzine transakcija i zahtevi u pogledu dostupnosti rasli, a posebno kako su baze podataka preseljene u oblak ili virtuelizovano okruženje, ekonomske prednosti horizontalnog skaliranja postale su neodoljive. Zbog toga su *NoSQL* baze projektovane za horizontalno skaliranje.

Big Data era donela je ogromno povećanje količina podataka koje treba skladištiti i obraditi. Kapaciteti RSUBP rasli su da bi zadovoljili nove zahteve, ali količina podataka kojom se praktično može upravljati uz pomoć RSUBP za mnoge organizacije ipak je bila nedovoljna. Danas, količina podataka kojom može rukovati neki *NoSQL* sistem, kao što je *Hadoop*, daleko prevazilazi mogućnosti najvećih RSUBP.

Uprkos mnogim poboljšanjima upravljivosti koje su proizvođači RSUBP postigli tokom godina, vrhunski RSUBP mogu se održavati samo uz pomoć skupih, visoko stručno obučenih administratora baza podataka. Administratori su uključeni u projektovanje, instalaciju i tekuća podešavanja sistema. *NoSQL* baze podataka od samog početka dizajnirane su tako da im je potrebno manje upravljanja. Automatsko popravljanje i

distribucija podataka i jednostavniji modeli podataka dovode do smanjenja potreba za administracijom i podešavanjem.

NoSQL baze podataka obično koriste klastere jeftinih servera za upravljanje brzo rastućim količinama podataka i transakcija, dok se RSUBP oslanjaju na skupe vlasničke servere i sisteme za skladištenje podataka. Rezultat je da cena po GB ili po broju transakcija u sekundi, kod *NoSQL* sistema, može biti mnogo puta manja nego kod relacionih, što omogućava skladištenje i obradu većih količina podataka po nižoj ceni.

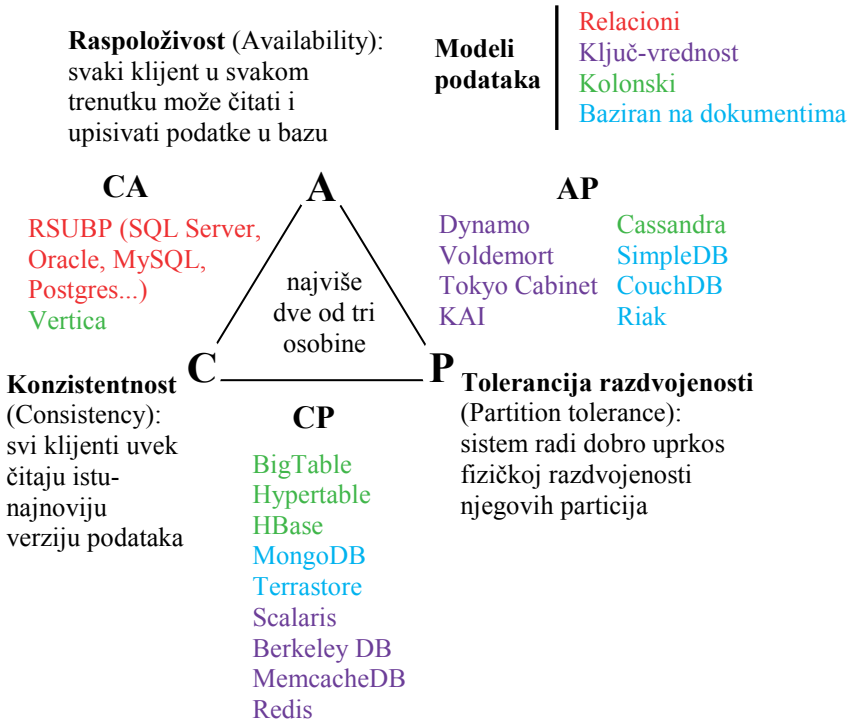
Upravljanje promenama je veliki problem za RSUBP. Čak i manje promene u relacionom modelu podataka moraju biti pažljivo sprovedene i gotovo uvek zahtevaju privremeno zaustavljanje rada sistema ili smanjenje nivoa usluga. *NoSQL* baze podataka značajno su relaksirane od problema koje nosi menjanje modela podataka – većina gotovo da i nema ograničenja tog tipa. Ključ-vrednost baze podataka i baze zasnovane na dokumentima dozvoljavaju aplikaciji da čuva bilo koju strukturu koju želi u elementu podataka. Čak i strože definisane baze podataka bazirane na *BigTable* modelu, kao što su *Cassandra* i *HBase*, dozvoljavaju da se kreiraju nove kolone bez dodatnih komplikacija. Zahvaljujući fleksibilnim modelima podataka *NoSQL* baze podataka olakšavaju praćenje evolucije aplikacije kroz njen životni ciklus.

3.2. Ograničenja

Prema *CAP* (*Consistency* - konzistentnost*, Availability* - raspoloživost*, Partition tolerance* - tolerancija razdvojenosti) teoremi, primenljivoj na sisteme zasnovane na distribuiranoj arhitekturi, sistem koji skladišti deljene podatke može obezbediti istovremeno zadovoljenje najviše dva od sledeća tri uslova: konzistentnost, raspoloživost i tolerancija razdvojenosti [8]. Sistem poseduje osobinu konzistentnosti ako svako čitanje iz baze podataka kao rezultat ima najnoviju verziju podataka. Raspoloživost se odnosi na odziv sistema u garantovanim vremenskim okvirima. Razdvojenost nastupa kada je sistem podeljen na dve ili više particija između kojih ne postoji komunikacija. Tolerancija razdvojenosti znači da nijedan skup otkaza, osim potpunog otkazivanja, ne sme da proizvede neispravan odziv sistema baze podataka. To znači da sistem mora da prihvata delimične otkaze i nastavlja ispravan rad. Tvrđenje *CAP* teoreme, primenjeno na savremene relacione i nerelacione baze podataka, grafički je prikazano na Slici 6.

SUBP dizajnirani tako da ispunjavaju *ACID* garancije, u *Big Data* eri ne mogu uvek istovremeno da obezbede i konzistentnost i dostupnost. U takvim situacijama *ACID* sistemi prednost daju konzistentnosti, na račun dostupnosti. Nasuprot njima, *NoSQL* sistemi ne ispunjavaju *ACID* garancije, već su kreirani tako da ispunjavaju *BASE* (*Basically Available, Soft state, Eventual consistency*) garancije:

- suštinski raspoloživ (*Basically available*): većina podataka je dostupna veći deo vremena,
- nekonzistentno stanje (*Soft state*): baza podataka ne mora biti konzistentna u svakom trenutku,
- konvergentna konzistencija (*Eventual consistency*): ne postoji garancija da će u svakom trenutku svi čvorovi sadržati identične kopije podataka. Teži se vremenskoj tački u kojoj će svi čvorovi sadržati konzistentne podatke.



Slika 6. CAP teorema i savremene baze podataka

4. Zaključak

Nestrukturirani podaci ili česte potrebe za promenom strukture podataka zahtevaju fleksibilan model podataka, što relacioni model nije. S druge strane, svi nerelacioni modeli podataka veoma su fleksibilni. Skladištenje podataka u oblaku (*Cloud-based storage*) je odlično rešenje koje smanjuje troškove, ali samo ako je obezbeđena horizontalna skalabilnost sistema, tj. ako se sistem skalira dodavanjem novih servera. Za razliku od *NoSQL* sistema koji su horizontalno skalabilni, relacione baze podataka su vertikalno skalabilne, što znači da njihovo skaliranje zahteva kupovinu moćnijih i skupljih servera. Velike količine heterogenih podataka zahtevaju efikasne algoritme obrade, kao što je *Apache MapReduce*. Obrada podataka u relacionim bazama zasnovana je na zahtevnim operacijama spajanja. Kompleksni *SQL* upiti u slučaju sistema distribuiranih na hiljade servera ne pružaju zadovoljavajuće performanse. Relacioni model podataka i *ACID* transakcije sprečavaju brojne anomalije, garantuju konzistentnost i integritet baze podataka. Nerelacione baze podataka podržavaju *BASE* transakcije, pa se za njih kaže da su eventualno konzistentne. Za mnoge aplikacije, kao što su aplikacije elektronske trgovine i finansijske aplikacije, *ACID* garancije su neophodne.

Iz svega gore navedenog proizilazi zaključak da ne postoji jedna tehnologija baza podataka koja zadovoljava sve potrebe. Različiti zadaci zahtevaju da koristimo kako relacione, tako i nerelacione baze podataka. Integracija nerelacionih i relacionih baza podataka nameće se kao izazov za neka buduća istraživanja.

Zahvalnost

Ovaj rad delimično je podržan od strane Ministarstva prosvete, nauke i tehnološkog razvoja Republike Srbije, u okviru projekta pod brojem 032025.

Literatura

[1] E. F. Codd, "A relational model of data for large shared data banks", *Communications of the ACM*, vol. 13 (6), pp. 377-387, 1970.
[2] J. Gray, "The Transaction Concept: Virtues and Limitations", *In Proceedings of 7th International Conference on Very Large Databases*, pp. 144-154, 1981.
[3] G. Harrison, *Next Generation Databases NoSQL and Big Data*, New York, NC: Apress, 2015.
[4] Amazon DynamoDB. (2018, October 10). [Online]. Available at: https://aws.amazon.com/dynamodb/
[5] C. Sherman (2017, April 16). Designing a Cassandra Data Model [online]. Available at: https://shermandigital.com/blog/designing-a-cassandra-data-model/
[6] Neo4j. (2018, October 2). [online]. Available at: https://neo4j.com/developer/graph-database/
[7] 3pillar global. (2018, October 5). A short history of databases: From RDBMS to NoSQL & beyond [online]. Available at: https://www.3pillarglobal.com/insights/short-history-databases-rdbms-nosql-beyond
[8] E. Brewer, "Towards Robust Distributed Systems", *In Proceedings of 19th Annual ACM Symposium on Principles of Distributed Computing*, pp. 7-10, July 2000.

Abstract: *Data models and the technologies that follow databases development went through three revolutions. The first revolution delivered a hierarchical and network data model, the second – a relational data model, while the third brought non-relational (NoSQL) databases. The first NoSQL databases were developed in order to meet the needs of Web 2.0 which determined them to be distributed, open-source and horizontally scalable. Today, these databases represent the main pillar of Big Data technology. Comparing to relational databases, non-relational databases enable high availability while sacrificing data consistency through time. Till now, the following four non-relational data models have been developed: key-value, wide column, graph, and document model. This paper defines a context in which non-relational databases were appear, presents their common features, demonstrates their advantages and limitations, and describes characteristics of each data model through examples.*

Keywords: *NoSQL databases, key-value databases, wide column databases, document databases, graph databases*

NON-RELATIONAL DATABASES: OPPORTUNITIES AND LIMITATIONS

Slađana Janković, Snežana Mladenović, Ana Uzelac, Stefan Zdravković