Usporedba Apache Cassandra i relacijske baze podataka na primjeru srednje škole

Kramar, Gabrijela

Undergraduate thesis / Završni rad

2016

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:112390

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-03-29**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

zir.nsk.hr

Image not found or type unknown

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
ELEKTROTEHNIČKI FAKULTET

Sveučilišni studij

USPOREDBA APACHE CASSANDRE I RELACIJSKE BAZE PODATAKA NA PRIMJERU SREDNJE ŠKOLE

Završni rad

Gabrijela Kramar

Osijek, 2016.

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Obrazac Z1P - Obrazac za ocjenu završnog rada na preddiplomskom sveučilišnom studiju

Osijek, 02.09.2016.

Odboru za završne i diplomske ispite

Prijedlog ocjene završnog rada

Ime i prezime studenta:	Gabrijela Kramar
Studij, smjer:	Preddiplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	R3569, 01.08.2013.
OIB studenta:	66951002343
Mentor:	Doc.dr.sc. Ivica Lukić
Sumentor:	
Naslov završnog rada:	Usporedba Apache Cassandra i relacijske baze podataka na primjeru srednje škole
Znanstvena grana rada:	**Programsko inženjerstvo (zn. polje računarstvo)**
Predložena ocjena završnog rada:	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 3 Postignuti rezultati u odnosu na složenost zadatka: 3 Jasnoća pismenog izražavanja: 3 Razina samostalnosti: 3
Datum prijedloga ocjene mentora:	02.09.2016.
Datum potvrde ocjene Odbora:	12.09.2016.

Potpis mentora za predaju konačne verzije rada u Studentsku službu pri završetku studija:	Potpis:
	Datum:

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

IZJAVA O ORIGINALNOSTI RADA

Osijek, 13.09.2016.

Ime i prezime studenta:	Gabrijela Kramar
Studij:	Preddiplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	R3569, 01.08.2013.
Ephorus podudaranje [%]:	1

Ovom izjavom izjavljujem da je rad pod nazivom**: Usporedba Apache Cassandra i relacijske baze podataka na primjeru srednje škole**

izrađen pod vodstvom mentora Doc.dr.sc. Ivica Lukić

i sumentora

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

IZJAVA

Ja, Gabrijela Kramar,OIB:66951002343, student/ica na studiju: Preddiplomski sveučilišni studij Računarstvo, dajem suglasnost Elektrotehničkom fakultetu Osijek da pohrani i javno objavi moj **završni rad**:

Usporedba Apache Cassandra i relacijske baze podataka na primjeru srednje škole

u javno dostupnom fakultetskom, sveučilišnom i nacionalnom repozitoriju.

Osijek, 13.09.2016. 2016.

_____

potpis

Sadržaj

1. UVOD 1

1.1. Zadatak završnog rada 1

2. APACHE CASSANDRA 2

2.1. NoSQL baze podataka 2

2.2. Što je Apache Cassandra i koje su joj osobine 3

2.3. Arhitektura Apache Cassandre 4

2.3.1. Dijelovi Cassandre 4

2.3.2. Prikaz prstena 5

2.3.3. Zapisivanje podataka 5

2.3.4. Čitanje podataka 6

2.3.5. Protokol tračanja (eng. *Gossip protocol*) 7

2.4. Model podataka 7

2.4.1. Stupac 7

2.4.2. Superstupac 7

2.4.3. Porodica stupaca 8

2.4.4. Eng. Keyspace 8

2.5. Tipovi podataka 8

3. BAZA PODATAKA ZA SREDNJE ŠKOLE 10

3.1. Relacijska baza podataka 10

3.1.1. Model relacijske baze podataka 10

3.1.2. Fizička shema relacijske baze podataka 13

3.2. Cassandra baza podataka 15

3.2.1. Apache Cassandra model baze i fizička shema 16

4. USPOREDBA RELACIJSKIH BAZA PODATAKA S APACHE CASSANDROM 19

5. ZAKLJUČAK 21
LITERATURA 22
SAŽETAK 23
ABSTRACT 23
ŽIVOTOPIS 24
PRILOZI 25

1. UVOD

Cassandra je distribuirana baza podataka koju je razvio Apache. Ona je NoSQL baza podataka koja je vrlo skalabilna, dizajnirana za upravljanje vrlo velikim strukturama podataka preko mnogih poslužitelja te je otporna na pogreške.Također, nudi snažnu podršku za nakupinepreko višestrukih podatkovnih centara uz asinkrone kopijekoje omogućuju nisku latentnost poslovanja za sve klijente. Cassandra je moćna baza podataka s dobrim temeljima koju su testirale tvrtke kao što suFacebook, Twitter i Netflix.Ona je ključ-vrijednost i stupac-orijentirana baza podataka čime je otporna na greške i brzo dohvaćanje velikog broja podataka. Osmišljena je za pokretanje na jeftinom hardveru, što znači da izvršava jako brza pisanja i može uskladištiti stotine terabajtapodataka bez žrtvovanja efikasnosti čitanja.

Glavni zadatak završnog rada je usporedba Apache Cassandrom s najčešće korištenim bazama podataka, relacijskim bazama podataka.

U drugom poglavlju dana je teorijska osnova o NoSQL bazama podataka i Apache Cassandri, njenim osobinama, arhitekturi, modelu podataka i tipovima podataka.

U trećem poglavlju napravljeni su relacijski model baze podataka srednje škole i Apache Cassandra model, prikazani su primjeri naredbi za stvaranje baze (*keyspace-a* u slučaju Cassandre), stvaranje tablica (porodice stupaca) te unos, brisanje i mijenjanje podataka.

Nadalje, četvrto poglavlje sažima sva saznanja iz prijašnjih poglavlja te prikazuje usporedbu relacijskih baza podataka i Apache Cassandre, nakon čega slijedi zaključak.

1.1. Zadatak završnog rada

Modelirati Apache Cassandra bazu podataka za evidenciju djelatnika i učenika u srednjoj školi. Dati teorijsku osnovu Apache Cassandra baze podataka, usporediti ju sa drugim modelima baza podataka. Navesti prednosti i nedostatke Apache Cassandra baze podataka u odnosu na najčešće korištene baze podataka.

2. APACHE CASSANDRA

2.1. NoSQL baze podataka

NoSQL baza podataka (eng. *NotOnly SQL* – „ne samo SQL“) je baza podataka koja pruža mehanizam za spremanje i dohvaćanje podataka koji je drugačiji od tabličnih relacija korištenih u relacijskim bazama podataka. Te baze nemaju standardiziranu shemu, podržavaju jednostavno umnažanje, imaju jednostavan API (eng. *ApplicationProgrammingInterface*), u konačnici sudosljedne i mogu upravljati velikom količinom podataka.
Primarni cilj NoSQL baze podataka je:

- Horizontalna skalabilnost – dodavanje novih čvorova
- Jednostavan dizajn
- Ljepše upravljanje dostupnosti podataka

NoSQL baze koriste različite strukture podataka za razliku od relacijskih baza podataka zbog čega su neke operacije brže u NoSQL-u. Prikladnost NoSQL baze ovisi o problemu koji mora riješiti.
Vrste NoSQL baze podataka neformalno se dijele na:

- Ključ-vrijednost (npr. Apache Cassandra, Amazon Dynamo, …)
- Dokument (npr. MongoDB, CouchDB, SimpleDB, …)
- Graf (npr. GraphDB, InfoGrid, DEX, Neo4j …)
- Stupčasti model pohrane podataka (Google BigTable, , HBase, …)
- Objektno orijentirane baze podataka (JADE, ObjectDB, ObjectDatabase++, …)

Za razliku od relacijski baza podataka, ne-relacijske baze podataka su BASE.

- eng. ***B****asically****A****vailable* – ako se dogode neki kvarovi, aplikacija je i dalje dostupna
- eng. ***S****oft-state* – s obzirom da se sustav stalno mijenja, baza ne mora biti dosljedna
- eng. ***E****ventualconsistency* – sve izmjene koje se provedu će u konačnici svima biti dostupne

2.2. Što je Apache Cassandra i koje su joj osobine

Apache Cassandra je otvorenog koda, distribuiran i decentraliziran sustav za pohranu podataka te se koristi za upravljanje vrlo velikim količinama podataka raširenih diljem svijeta. Ona pruža učinkovite usluge bez ikakve greške.

Apache Cassandra je:

- Skalabilna, otporna na greške i dosljedna,
- Ključ-vrijednost i stupac-orijentirana baza podataka
- Distribucijski dizajn temeljen je na AmazonovomDynamo i GoogleovomBigTable modelu podataka
- Stvorio ga je Facebook te joj se sustav upravljanja značajno razlikuje od relacijskih baza podataka
- Implementira Dynamov stil replikacijskog modela koji je otporan na greške, ali nadodaje model podataka „porodica stupaca“ (eng. *columnfamily*)

Zbog svojih izvrsnih tehničkih osobina, Cassandra je postala vrlo popularna, a neka od tih osobina su:

- Elastična skalabilnost – Cassandra dopušta dodavanje hardvera kako bi se zadovoljile potrebe korisnika i njihovim sve većim zahtjevima za podacima.
- Brze i linearno proporcionalne performanse – Cassandra je linearno skalabilna, što znači da se povećanjem broja čvorova u nakupinama, povećava i propusnost podatak. Dakle, održava brzo vrijeme odziva.
- Prilagodljivo skladište podataka –Cassandra podržava sve moguće oblike podataka, strukturirane, polu-strukturirane i nestrukturirane. Ona se dinamički prilagođava promjenama strukture podataka prema korisnikovim potrebama.
- Jednostavna distribucija podataka – Cassandra omogućuje prilagodljiv način distribuiranja podataka repliciranjem tih podataka preko mnogih podatkovnih centara.
- Podržavanje transakcije–Cassandra podržava transakcije, no ne omogućava nužno svojstva nedjeljivosti (eng. *Atomicity*), dosljednosti (eng. *Consistency*), izolacije (eng. *Isolation*) i izdržljivosti (eng. *Durability*)
- Brzo pisanje – Cassandra je osmišljena za pokretanje na jeftinom hardveru. Ona izvršava jako brza pisanja i može uskladištiti stotine terabajta podataka bez žrtvovanja efikasnosti čitanja.

2.3. Arhitektura Apache Cassandre

Cassandra koristi prednosti od dva dokazana i vrlo slična mehanizma pohrane podataka, Google BigTable-ov distribuirani sustav pohrane za strukturirane podatke i Amazon Dynamo-vu vrlo dostupnu ključ-vrijednost pohranu.Kod Cassandre, podaci su prikazani u obliku tablice, no ne u najstrožem smislu. Prikaz je sličniji strukturi rječnika gdje svaki unos sadrži drugi, sortirani rječnik. Takav model prikaza podataka naziva se porodica stupaca (eng. *columnfamily*) te je snažniji od klasičnog ključ-vrijednost modela. Porodica stupaca može se zamisliti kao velika proračunska tablica, no za razliku od proračunske tablice, svaki red je identificiran pomoću broja ključa (žetona) te svaka ćelija može imati jedinstveno ime unutar reda. Stupci su poredani po jedinstvenom imenu stupca unutar reda. Također, zbog dopuštenog vrlo velikog broja redova, redovi su ravnomjerno distribuirani preko dostupnih strojeva, dijeljenjem redova u jednake grupe žetona. Ovi redovi stvaraju eng. *keyspace*-ove, dakleeng. *keyspace* je set svih žetona (ID redova). Cassandra ima eng. *peer-to-peer* distribuirani sustav preko svojih čvorova i podatak je distribuiran preko svih čvorova u nakupinama. Svi čvorovi u nakupinama imaju jednaku ulogu te je svaki čvor neovisan i u isto vrijeme međusobno povezan s drugim čvorovima. Također, svaki čvor u nakupini može prihvatiti zahtjeve za pisanje i čitanje bez obzira na mjesto podatak u nakupini. Kada jedan čvor ispadne iz nakupine, zahtjevi za pisanje i čitanje obrađuje drugi čvor u mreži.

2.3.1. Dijelovi Cassandre

Glavni dijelovi Cassandre su:

- Čvor (eng. *Node*) – mjesto gdje su podaci spremljeni.
- Središnjica podataka (eng. *Data center*) – kolekcija povezanih čvorova.
- Nakupina (eng. *Cluster*)–komponenta koja sadrži jedan ili više središnjica podataka.
- Dnevnik izvršenja (eng. *Commit log*) –mehanizam za oporavak od pada sustava, svaka operacija zapisivanja je zapisana u dnevnik izvršenja.
- Memorijska tablica (eng. *Mem-table*) – tablica u kojoj su smještene strukture podataka. Nakon dnevnika izvršenja, podatak će biti zapisan u memorijsku tablicu. Ponekad će biti višestrukih memorijskih tablica za jednostupčanu porodicu.
- SSTablica (eng. *SortedStructure Table*) –datoteka na disku u koju je podatak prebačen iz memorijske tablice kada njezin sadržaj dosegne najvišu dopuštenu vrijednost.

- eng. *Bloom filter*–brz, neodređeni algoritam za provjeru je li element dio nekog kompleta. On je posebna vrsta predmemorije kojoj se pristupa poslije svakog upita

2.3.2. Prikaz prstena

Raspodjela žetona (eng. *tokens*) unutar nakupina prikazuje oblik prstena.Npr., djelitelj (eng. *partitioner*) generira žetone od 0 do 255 i u nakupini je 5 Cassandra strojeva. Kako bi se dodijelio jednak teret, svaki čvor trebao bi snositi jednak broj žetona. Dakle, prvi čvor bit će odgovoran za žetone od 1 do 51, drugi će posjedovati žetone od 52 do 102, treći od 103 do 153, četvrti od 154 do 204 i peti od 205 do 255. Ako se svaki čvor označi s brojem žetona kojeg može posjedovati, onda nakupina dobiva izgled prstena. Djelitelj je eng. *hash* funkcija koja određuje opseg mogućih ključeva redova (eng. *rowkey, row ID*). Pri konfiguriranju Cassandre, poželjno je postaviti najveći broj žetona koje neki čvor može posjedovati, no treba imati na umu da dodavanjem novih čvorova ili odbacivanjem starih čvorova, jednaka podjela žetona može biti neuravnotežena.

2.3.3. Zapisivanje podataka

Klijenti trebaju biti spojeni na bilo koji Cassandrin čvor i poslati zahtjev za pisanje ako žele zapisati neki podatak. Taj čvor se naziva koordinacijski čvor (eng. *coordinatornode*) i on je možda pravo mjesto za zapisivanje tog podatka. Kada čvor u nakupini primi zahtjev za pisanje, tada taj zahtjev obrađuje usluga*StorageProxy*. Zadatak *StorageProxy-a* je dohvatiti sve čvorove (sve kopije) koje su odgovorne za držanje podatka koji će biti zapisan te optimizirati strategiju kopiranja. Kada su čvorovi za kopiranje identificirani, on šalje *RowMutation* poruku i čeka odgovore, no ne čeka sve odgovore već samo onoliko koliko je definirano u *ConsistencyLevel* opciji (dovoljno da zadovolji korisnikov najmanji broj uspješnih zapisa). Nakon toga ima tri toka odvijanja radnje:

- *FailureDetector* otkriva ima li dovoljno čvorova za zadovoljavanje *ConsistencyLevel-a*. Ako nema, zahtjev za pisanje se ne izvršava.
- Ako *FailureDetector* detektira dovoljno čvorova, no ispiše pauzu (eng. *time-out*) poslije zahtjeva zbog problema u infrastrukturi ili prevelikog opterećenja, *StorageProxy* zabilježi

lokalni savjet (eng. *hint*) za ponovno zapisivanje kada se podbačeni čvorovi vrate nazad umrežu. Taj postupak se naziva uručeni savjet (eng. *hintedhandoff*)

- Ako je sve uredu, izvršava se zapisivanje podatka.

Ako su čvorovi koji primaju kopiju, distribuirani preko središnjica podataka, preporučljivo je slati poruke jednoj kopiji u svakoj središnjici podataka s zaglavljem koji naređuje prosljeđivanje zahtjeva drugim čvorovima koji primaju kopiju u toj središnjici podataka.
Unutar čvora, prvo je podatak dodan kao prilog u dnevniku izvršenja (eng. *Commit log*) te je proslijeđen memorijskoj tablici (eng. *Mem-table*) u odgovarajuću porodicu stupaca u memoriji. Kada se memorijska tablica napuni, njeni podaci se premjeste (eng. *flush*) na disk u sortiranu strukturu koja se naziva SSTablica. S mnogo premještanja eng. (*flush*), disk dobiva puno SSTablica. Za upravljanje tim tablicama, odvija se proces zbijanja (eng. *compactionprocess*). Taj proces spaja podatke iz malih SSTablica u jednu veliku i sortiranu datoteku.

2.3.4. Čitanje podataka

Kada *StorageProxy* čvora na koji je klijent spojen, dobije zahtjev za čitanje, on dobije popis čvorova koji sadrže taj ključ temeljeno na strategiji kopiranja (eng. *ReplicationStrategy*). Tada *StorageProxy* sortira čvorove prema udaljenosti od sebe. Ta udaljenost određuje se *Snitch* funkcijom koja je postavljena za tu nakupinu. Nakon nabavljanja željenih ključeva dolazi čitanje podatka. Koordinatorski čvor (onaj na kojeg je korisnik spojen) šalje naredbu za čitanje najbližem čvoru i vraća podatak. Na osnovi *ConsistencyLevel-a*, drugi čvorovi će poslati naredbu za izvršavanje operacije čitanja i poslatiskraćen pregled rezultata. Ako je omogućena opcija*ReadRepair*, ostalim čvorovi koji sadrže kopiju bit će poslana poruka za izračunavanje skraćenog pregleda rezultata. Taj zahtjev se izvršava u pozadini, nakon što su vraćeni rezultati. Zahvaljujući tome, svaki čvor je ažuriran, što kopije čini dosljednima.
Unutar čvora, podatak se pretražuje u memorijskoj tablici i ta je pretraga jako brza zato što postoji samo jedna kopija željenog podatka. Ako se podatak ne nalazi u memorijskoj tablici, pretražuje se SSTablica. Prije je spomenuto da svakim premještanjem (eng. *flush*) podataka iz memorijske tablice, stvara se nova SSTablica te tako postoji nekoliko SSTablica koje mogu sadržavati željeni podatak. Zato je svaka SSTablica povezana sa svojim eng. *Bloom Filterom* za ključeve redova koji je u memoriji i koristi se za otkrivanje željenog podatka u određenoj SSTablici. Podaci u SSTablici su sortirani kronološki od najnovijeg podatka do najstarijeg podatka. Cassandra pretražuje podatke od najnovijeg do najstarijeg i ako naiđe na željeni

podatak, korisniku vrati vrijednost te prestaje s pretraživanjem jer korisniku je potrebna najnovija vrijednost. Osim eng. *Bloom Filtera* za ključeve redova, postoji eng. *Bloom Filter* za svaki red SSTablice. Drugi eng. *Bloom Filter* otkriva postoji li željeno ime stupca u SSTablici.

2.3.5. Protokol tračanja (eng. *Gossipprotocol*)

Za komunikaciju unutar čvorova, Cassandra koristi protokol tračanja. Kao što naziv govori, protokol širi informacije kao što se prenosi neka glasina. Također, širenje informacija može se usporediti sa širenjem virusa. Dakle, nema podatkovnog centrakoja dalje širi informacije, nego se informacija prenosi kroz sve čvorove. Na taj način čvorovi izgrađuju globalnu mapu sustava s malim brojem lokalnih međudjelovanja. Cassandra koristi protokol tračanja za ažuriranje stanja čvorova i njihovih lokacija unutar prstena (nakupine). Proces tračanja pokreće se svake sekunde i izmjenjuje informacija s najviše tri čvora unutar nakupine. Čvorovi izmjenjuju informacije o sebi i drugim čvorovima o kojima mogu nešto saznati preko drugih sesija tračanja. U konačnici, svi čvorovi znaju sve o svim čvorovima. Kao sve u Cassandri, svaki trač ima svoj broj verzije, što znači da kad god dva čvora tračaju, starija informacija o čvoru se prepiše s novijom informacijom.

2.4. Model podataka

Postoje tri osnovne strukture podataka: stupac (eng. *column*), porodica stupaca (eng. *columnfamily*) i superstupac (eng. *super column*). Te tri strukture se nalaze u eng. *keyspace-u*.

2.4.1. Stupac

Stupac je atomska jedinica Cassandrinog modela podataka i najmanja komponenta kojom se može upravljati. On je sadržan u porodici stupaca. Stupci se prikazuju kao trojka naziva/ključa, vrijednosti i vremenske oznake. Vremenska oznaka se koristi za ažuriranje vrijednosti stupca, što znači da vrijednost koja je zapisana kasnije je bitnija, dok se starija vrijednost briše.

2.4.2. Superstupac

Superstupac je stupac koji sadržava još stupaca. On sadržava poredanu mapu stupaca i podstupaca. Često se koristi za denormalizaciju dodavanjem višestrukih redova neke porodice stupaca u jedan red superstupca.

2.4.3. Porodica stupaca

Porodica stupaca je kolekcija redova u kojem je svaki red par ključ-vrijednost. Dakle, porodica stupaca je mapa sa svojim ključevima koji su ključevi redova, i vrijednostima koje su poredana kolekcija stupaca. Svaka porodica stupaca spremljena je u svoju datoteku i ne postoji relacijski integritet između dvije porodice stupaca. U porodici stupaca, ključ reda je jedinstven i služi kao primarni ključ. Koristi se za identificiranje, dohvaćanje i postavljanje zapisa u određeni red porodice stupaca. Postoje dvije vrste porodice stupaca: dinamička ili široka i statička ili uska porodica stupaca.

Dinamička porodica stupaca iskorištava sposobnost porodice stupaca za spremanje proizvoljnog broja stupaca (par ključ-vrijednost). Tipičan primjer dinamičke porodice stupaca je za statističko skupljanje podataka.

Statička porodica stupaca slična je tablicama u relacijskim sustavima baze podataka. Statička porodica stupaca ima predefinirana imena stupaca i njihov potvrdnik (eng. *validator*), ali dodavanje stupaca je dozvoljeno. Tipičan primjer statičke porodice stupaca je porodica stupaca koja predstavlja korisnike. Velika prednost statičke porodice stupaca je potvrđivanje (eng. *validation*), zato što se definiranjem klase (eng. *validation_class*) na nekom stupcu može kontrolirati uneseni tip podatka.

2.4.4. Eng. Keyspace

Eng. *keyspace* je najveći dio Cassandre i sadrži sve porodice stupaca, superstupce i stupce. Ugrubo, može se promatrati kao baza podataka u sustavima relacijskih baza podataka isvrha mu je grupirati porodice stupaca.Generalno gledajući, jedna aplikacija koristi jedan eng. *keyspace*. Eng. *keyspace* sadrži replikacijski faktor (eng. *replicationfactor*) i strategiju za razmještanje kopija (eng. *replicaplacementstrategies*) koji se globalno primjenjuju na svaku porodicu stupaca u tom eng. *keyspace-u*.

2.5. Tipovi podataka

CQL (eng. *Cassandra QueryLanguage*) omogućuje bogati set ugrađenih tipova podataka (neki su prikazani u tablici 2.1.), uključujući i kolekcijske tipove podataka.

Tab.2.1. Neki od ugrađenih tipova podataka

Tip podataka	Opis
Boolean	Tip podatka koji ima dva stanja: istina ili laž
counter	Brojač stupaca
int	32-bit cjeli broj.
double	Decimalni broj dvostruke preciznosti (64-bit IEEE-754)
float	Decimalni broj (32-bit IEEE-754)
inet	Predstavlja IP adresu (IPv4 ili IPv6)
timestamp	Predstavlja vremensku oznaku
uuid	Predstavlja univerzalni, jedinstveni identifikator
varchar	Predstavlja uTF8 kodiran niz znakova

Kolekcijski tipovi podataka su:

- Lista – kolekcija jednog ili više poredanih elemenata
- Mapa –kolekcija parova ključ-vrijednost
- Set – kolekcija jednog ili više elemenata

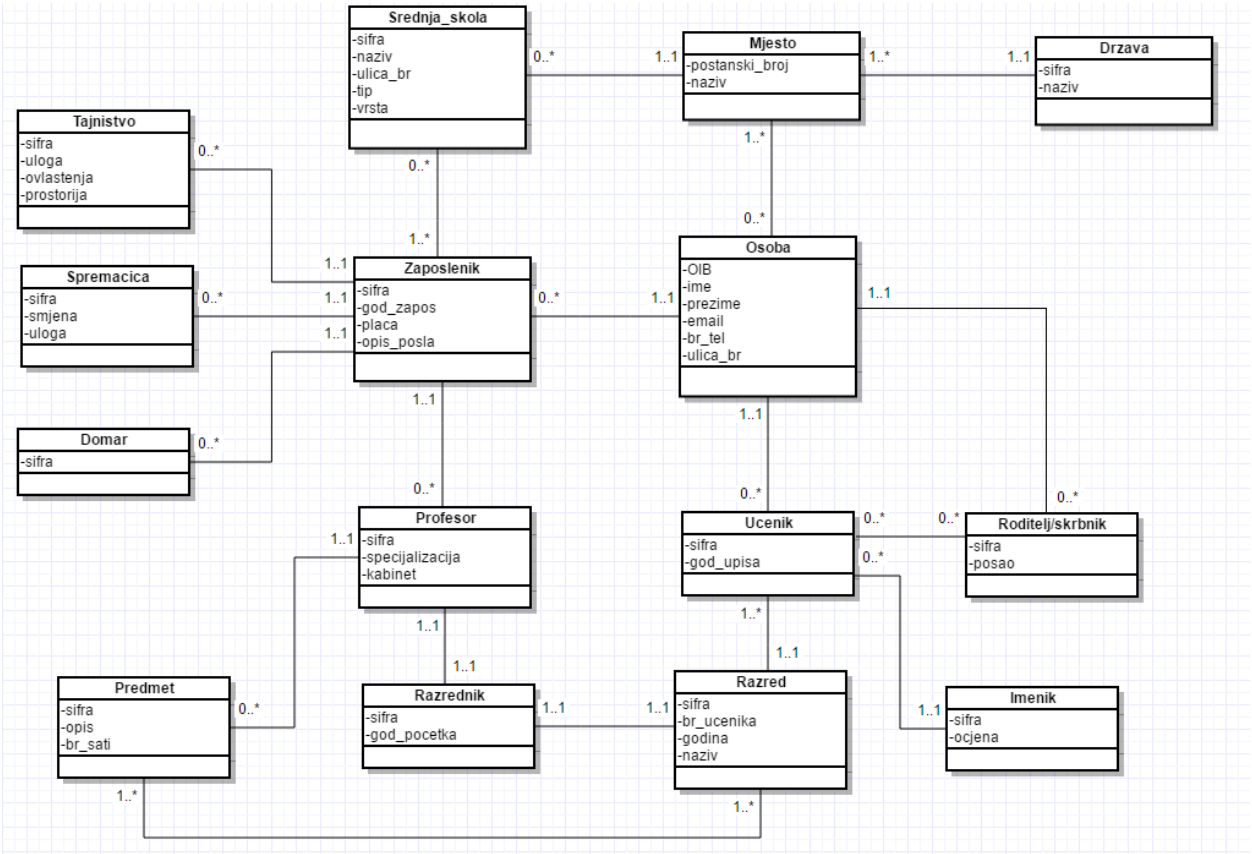
Uz te tipove podataka, korisnik može stvoriti, mijenjati i obrisati svoj tip podatka.

3. BAZA PODATAKA ZA SREDNJE ŠKOLE

3.1. Relacijska baza podataka

3.1.1. Model relacijske baze podataka

Na slici 3.1. prikazan je ER (eng. Entity-Relationship) model relacijske baze zapisan UML notacijom (eng. UnifiedModelingLanguage). ER model prikazuje entitete s odgovarajućim atributima i veze između entiteta. Na slici 3.2. prikazan je relacijski model dobiven prevođenjem ER modela sa slike 3.1, gdje podvučeni atributi predstavljaju primarne ključeve. Rječnik podataka za relacijski model sa slike 3.1. dan je u tablici 3.1.



Sl.3.1. ER dijagram za bazu podataka srednje škole

DRZAVA (<u>SIFRA</u>, NAZIV)
MJESTO (<u>POSTANSKI_BROJ</u>, NAZIV, SIF_DRZAVE)

MJESTO_OSOBE (OIB, POSTANSKI_BROJ)
OSOBA (OIB, IME, PREZIME, EMAIL, BR_TEL, ULICA_BR)
UCENIK (SIFRA, GOD_UPISA, OIB, SIF_IMENIK, SIF_RAZRED)
UCENIKOV_RODITELJ (SIF_UCENIK, SIF_RODITELJ)
RODITELJ/SKRBNIK (SIFRA, POSAO, OIB)
IMENIK (SIFRA, OCJENA)
RAZRED (SIFRA, BR_UCENIKA, GODINA, NAZIV, SIF_RAZREDNIKA)
PREDMET_RAZREDA (SIF_RAZRED, SIF_PREDMET)
PREDMET (SIFRA, OPIS, BR_SATI, SIF_PROF)
RAZREDNIK (SIFRA, GOD_POCETKA, SIF_RAZRED, SIF_PROF)
PROFESOR (SIFRA, SPECIJALIZACIJA, KABINET, SIF_RAZREDNIK, SIF_ZAPOSL)
ZAPOSLENIK (SIFRA, GOD_ZAPOS, PLACA, OPIS_POSLA, OIB)
ZAPOSLEN_U (SIF_ZAPOS, SIF_SREDNJA_SKOLA)
SREDNJA_SKOLA (SIFRA, NAZIV, ULICA_BR, TIP, VRSTA, POST_BR)
TAJNISTVO (SIFRA, ULOGA, OVLASTENJE, PROSTORIJA, SIF_ZAPOS)
SPREMACICA (SIFRA, SMJENA, ULOGA, SIF_ZAPOS)
DOMAR (SIFRA, SIF_ZAPOS)

Sl.3.2. Relacijska shema baze podataka srednje škole

Tab.3.1. Rječnik podataka za bazu podataka srednje škole.

Ime atributa	Tip	Opis
SIFRA SREDNJE_ŠKOLE	Niz od točno 5 znamenki	Broj koji jednoznačno određuje srednju školu
NAZIV (SREDNJE ŠKOLE)	Niz znakova	Naziv srednje škole
ULICA_BR	Niz znakova	Ulica i broj škole
TIP	Niz znakova	Privatna ili javna škola
VRSTA	Niz znakova	Gimnazija ili strukovna škola
SIFRA ZAPOSLENIKA	Niz od točno 5 znamenki	Broj koji jednoznačno određuje zaposlenika
GOD_ZAPOS	Niz od točno 4 znamenke	Godina kada je djelatnik zaposlen
PLACA	Decimalni broj	Iznos plaće zaposlenika
OPIS_POSLA	Niz znakova	Opis posla pojedinog radnika
SIFRA DOMARA	Niz od točno 3 znamenki	Broj koji jednoznačno određuje domara
SIFRA SPREMACICE	Niz od točno 3 znamenki	Broj koji jednoznačno određuje spremačicu
SMJENA	Niz znakova	Smjena u kojoj spremačica radi
ULOGA (SPREMACICE)	Niz znakova	Privremena ili stalna spremačica.

SIFRA TAJNISTVO	Niz od točno 5 znamenki	Broj koji jednoznačno određuje zaposlene u tajništvu.
ULOGA (TAJNISTVO)	Niz znakova	Opis uloge zaposlenika u školi (ravnatelj, tajnica, pedagoginja i sl.)
OVLASTENJA	Niz znakova	Opis ovlaštenja zaposlenika u školi
PROSTORIJA	Niz znakova	Na kojem katu i u kojoj prostoriji se nalazi tajništvo.
ŠIFRA PROFESOR	Niz od točno 5 znamenki	Broj koji jednoznačno određuje profesora
SPECIJALIZACIJA	Niz znakova	Koje predmete profesor može držati
KABINET	Niz znakova	Kat i prostorija gdje se nalazi kabinet profesora
ŠIFRA PREDMET	Niz od točno 5 znamenki	Broj koji jednoznačno određuje predmet
OPIS	Niz znakova	Opis tematike i nastavnog plana predmeta
BR_SATI	Niz znamenki	Broj sati koliko bi predmet trebao trajati u godini
ŠIFRA RAZREDNIK	Niz od točno 5 znamenki	Broj koji jednoznačno određuje razrednika
GOD_POCETKA	Niz od točno 4 znamenke	Godina od kad je profesor postao razrednik
ŠIFRA_RAZREDA	Niz od točno 5 znamenki	Broj koji jednoznačno određuje razred
BR_UCENIKA	Pozitivan cijeli broj	Broj učenika u razredu
GODINA	Niz znakova	Trenutna godina razreda
NAZIV (RAZREDA)	Niz znakova	Jedinstven naziv razreda
ŠIFRA UČENIK	Niz od točno 5 znamenki	Broj koji jednoznačno određuje učenika
GOD_UPISA	Niz od točno 5 znamenki	Godina upisa u prvi razred
SIFRA RODITELJ	Niz od točno 5 znamenki	Broj koji jednoznačno određuje roditelja/skrbnika
POSAO	Niz znakova	Roditeljevo zanimanje
ŠIFRA IMENIK	Niz od točno 5 znamenki	Broj koji jednoznačno određuje imenik
OCJENA	Pozitivan cijeli broj	Ocjena koju je učenik dobio
OIB	Niz od točno 11 znamenki	Broj koji jednoznačno određuje osobu
IME	Niz znakova	Ime osobe
PREZIME	Niz znakova	Prezime osobe
EMAIL	Niz znakova	E-mail osobe
BR_TEL	Niz znakova	Broj telefona osobe

ULICA_BR	Niz znakova	Ulica i kućni broj osobe
POSTANSKI_BROJ	Niz od točno 5 znamenki	Broj koji jednoznačno određuje mjesto
NAZIV (MJESTA)	Niz znakova	Naziv grada u kojem osoba živi
SIFRA DRZVA	Niz od točno 5 znamenki	Broj koji jednoznačno određuje državu
DRŽAVA	Niz znakova	Naziv države

3.1.2. Fizička shema relacijske baze podataka

Fizička shema (SQL naredbe za implementaciju baze podataka) baze podataka srednje škole prikazana je detaljno u prilogu 3.1., 3.2. te je dobivena na osnovu relacijske sheme baze podataka sa slike 3.2.Primjer naredbe za stvaranje baze dana je na slici 3.3.

```
CREATE DATABASE srednja_skola;
USE srednja_skola;
```

Sl. 3.3. Stvaranje baze podataka srednjih škola

Primjer stvaranja tablice s primarnim i stranim ključevima te željenim atributima prikazan je na slici 3.4. Ostale naredbe za stvaranje tablice dostupne su u prilogu 3.1.

```
CREATE TABLE drzava(
        sifra NUMERIC(5) PRIMARY KEY,
        naziv VARCHAR(20)
);

CREATE TABLE mjesto(
        postanski_broj NUMERIC(5) PRIMARY KEY,
        naziv VARCHAR(30),
        sif_drzave NUMERIC(5),
        CONSTRAINT fk_mjesto_drzava FOREIGN KEY(sif_drzave) REFERENCES drzava(sifra) ON
        DELETE CASCADE
);
```

Sl. 3.4. Stvaranje tablica država i mjesto te povezivanje tablica preko stranih ključeva

U nastavku slijede okidač (Sl.3.5.), pogled (Sl.3.6.) i procedura (Sl.3.7.) koji su također sastavni dio fizičke sheme baze podataka. Okidač se pokrene kad god netko pokuša obrisati ocjenu iz imenika te tu obrisanu ocjenu vraća nazad u imenik. Pogled omogućuje uvid u ime, prezime i

ocjene učenika poredanih prezimena po abecedi. Procedura *upis_ocjene* unosi ocjenu učenika pod šifrom @sif_ucenik u određeni imenik @sifra s određenom ocjenom @ocj.

```
CREATE TRIGGER
ON imenik
FOR DELETE
AS
        DECLARE @sifra,@ocjena;
        SELECT @sifra=sifra, @ocjena=ocjena FROM DELETED;
        UPDATE imenik SET ocjena=@ocjena WHERE sifra=@sifra;
```

Sl. 3.5. Stvaranje okidača koji će spriječiti brisanje ocjene iz imenika

```
CREATE VIEW ocjene
AS
SELECT osoba.ime, osoba.prezime, imenik.ocjena
FROM ucenik u, imenik i, osoba o
WHERE i.sifra=u.sif_imenik AND u.OIB=o.OIB;
ORDER BY o.prezime;
```

Sl. 3.6. Pogled na ime, prezime i ocjene učenika

```
CREATE PROCEDURE upis_ocjene @sifra, @sif_ucenik,@ocj SMALLINT
        AS
        UPDATE imenik SET imenik.ocjene = @ocj
        WHERE imenik.sifra = @sifra AND ucenik.sifra= @sif_ucenik
END
```

Sl. 3.7. Procedura za unos ocjene

Na slici 3.8. prikazan je jedan primjer unosa podataka u tablicu osoba. Svi unosi u bazu podataka za srednje škole prikazani su u prilogu 3.2.

```
INSERT INTO osoba VALUES (11111111117, 'Mia',
'Mijic','mia.m@gmail.com','+38592666652','Vukovarska 10c');
```

Sl. 3.8. Primjer unosa podataka u tablicu

Slika 3.9. prikazuje naredbu za brisanje podatka iz tablice, tj. brisanje podataka o učeniku kojemu je šifra u bazi jednaka *31310.*

```
DELETE ucenik WHERE sifra=31310;
```

Sl. 3.9. Primjer brisanja podataka iz tablice

Naredba za mijenjanje podataka u tablici, prikazana je na slici 3.10. Svakom zaposleniku koji je imao plaću manju od 3300 kn, plaća je povećana na 3400 kn.

```
UPDATE zaposlenik SET placa=3400.00 WHERE placa<3300.00;
```

Sl. 3.10. Primjer naredbe za ažuriranje tablice

3.2. Cassandra baza podataka

Prije početka modeliranja Cassandra baze podataka, potrebno je upoznati se s najčešćim pogreškama pri prelasku s relacijskih baza podataka na Apache Cassandru i ograničenjima. Prvo, cijeli entitet se ne bi trebao spremiti u jedan stupac ili porodicu stupaca. Osim gubitka direktnog pristupa jednom podatku iz reda, dolazi do slabe čitljivosti u CLI (eng. *Commmand-line Interface*), ne mogu se stvarati sekundarni indeksi (eng. *secondaryindex*) i teže čitanje, mijenjanje i pisanje podataka.

Nadalje, trebali bi se izbjegavati superstupci jer velikim brojem podstupaca usporavaju se sve operacije, podstupci nisu poredani (za razliku od stupaca u porodici stupaca) i na podstupce se ne može napraviti indeks.

Transakcije su moguće u Cassandri, no ne osigurava ACID svojstva (nedjeljivost, dosljednost, izolacija i izdržljivost) kao što je slučaj u relacijskim bazama, ali zbog toga su joj performanse puno brže i opseg operacija je puno veći nego u bilo kojoj relacijskoj bazi podataka.

Primarni ključ (ključ reda, primarni indeks) je jedinstveni identifikator reda kao i u relacijskim bazama podataka te omogućava brz pristup redovima. Kako su redovi podijeljeni između servera u prstenu, svaki server ima samo dio redova,a time su i ključevi distribuirani među njima. Cassandra koristi djelitelj (eng. *partitioner*) i strategiju za razmještanje kopija (eng. *replicaplacementstrategy*) za lociranje traženih čvorova u prstenu za pristup određenom redu. Problem kod primarnih ključeva je u tome što je njihova lokacija određena djeliteljem. Djelitelj primjenjuje eng. *hash* funkciju za pretvaranje ključ reda (eng. *rowkey*) u jedinstveni broj (koji se

naziva žeton) i tada zapiše/pročita taj ključ u/iz čvora koji posjeduje taj žeton. Dakle, koristeći djelitelj koji ne primjenjuje eng. *hash* koji prati poredak ključeva, velike su šanse da se čitanje ključeva ne može obavljati po redu, a time i pristupanju sljedećem žetonu na čvoru.
SELECT naredba ima sintaksu jednaku kao u relacijskim bazama podataka, no funkcionalnost je dosta slabija. Za razliku do relacijskih baza podataka, *WHERE* dio podržava samo I operaciju (eng. *AND conjunction*), nema ILI operacije. Također, mogu se koristiti samo stupci koji čine primarni ključ ili imaju sekundarni indeks.
Indeksi koji su napravljeni nad stupcima nazivaju se sekundarni indeksi. Oni su slični eng. *hash* funkciji, no iako se čine sličnima indeksu relacijskih baza podataka, on je inferiorni od njih. Indeksi su spremljeni u svojoj porodici stupaca i korisnik im ne može pristupiti. Sinkronizirani su s lokalnim podacima u čvoru, što znači da će indeks unutar porodice stupaca uvijek biti lokalno dosljedan (no ne i izvan porodice stupaca).

3.2.1. Apache Cassandra model baze i fizička shema

Naredbom za stvaranje eng. *keyspace-a* stvoren je „kontejner“ koji će sadržavati sve stupce, porodice stupaca i željene podatke (slično bazi podataka u relacijskom modelu). Nakupina sadržava jedan eng. *keyspace* po čvoru. Uz definiranje eng. *keyspace-a*, potrebno je definirati replikaciju (eng. *replication*) podataka na čvoru i trajno pisanje (eng. *durable_writes*).
Opcijama replikacijeodređuje se eng. *ReplicaPlacement* strategija i broj željenih kopija podataka.
Strategije za repliciranje su:

- eng. *SimpleStrategy* – definira jednostavan replikacijski faktor za nakupinu
- eng. *Network TopologyStrategy* – replikacijski faktor postavlja se za svaki podatkovni centar neovisno
- eng. *Old Network TopologyStrategy* – nasljeđuje se prijašnja replikacijska strategija

Svojstvo trajnog pisanja tablice je postavljeno na istina (eng. *true*), no može se postaviti i na laž (eng. *false*). U eng. *SimpleStrategy*, ovo svojstvo se ne može postavljati.
Na slici 3.11. prikazano je stvaranje eng. *keyspace-a* s replikacijskom strategijom eng. *SimpleStrategy* i replikacijskim faktorom 3 (unesen podatak bit će kopiran na 3 čvora).

```
CREATE KEYSPACE srednja_skola
        WITH replication = {'class':'SimpleStrategy', 'replication_factor' : 3};
USE srednja_skola;
```

Sl.3.11. Stvaranje i korištenje *keyspace-a* srednje škole

Model u Apache Cassandri sličan je relacijskom modelu, samo su entiteti mjesto i država stavljeni u jednu porodicu stupaca jer se ti podaci zajedno ažuriraju. Sintaksa za stvaranje porodice stupaca u Cassandri vrlo je slična relacijskoj sintaksi za stvaranje tablice (primjer prikazan na slici 3.12.). Cijeli kod za stvaranje porodice stupaca dan je u prilogu 3.3.

```
CREATE TABLE imenik(
        sifra INT PRIMARY KEY,
        ocjena SMALLINT
);
```

Sl.3.12. Primjer stvaranja porodice stupaca u Apache Cassandri

Zbog ograničenja SELECT funkcije koja može u WHERE dijelu koristiti samo primarne ključeve ili sekundarne indekse, nad nekim stupcima koji bi se mogli dodatno pretraživati ili poredati stvoreni su sekundarni indeksi (P.3.4.). Primjer za stvaranje jednog indeksa dan je u slici 3.13.

```
CREATE INDEXosoba_prezime ON osoba(prezime);
```

Sl.3.13. Primjer naredbe za stvaranje sekundarnog indeksa nad porodicom stupaca osoba

Nakon stvaranja porodice stupaca dolazi popunjavanje s podacima (kod je prikazan u prilogu 3.5.). Primjer naredbe za unos podataka dan je na slici 3.14. Razlika od relacijskih baza podataka je u točnom definiranju porodice stupaca i stupaca u koje se unose podaci. Kod relacijskih baza podataka se ne mora navoditi svaki stupac u koji se unose podaci zato što se u svaku ćeliju mora unijeti neki podatak (ako se želi ostaviti „prazna“ ćelija, onda se zapisuje NULL vrijednost), dok se kod Cassandre može ostaviti prazan stupac, čime se memorija koja je bila namijenjena tom stupcu oslobađa za neki drugi podatak.

```
INSERT INTOsrednja_skola(sifra, naziv, adresa, tip, vrsta,post_br) VALUES (33335, 'Strojarska', 'Licka
```

```
10', 'javna', 'strukovna',11114);
```

Sl.3.14. Primjer unosa podataka u Apache Cassandra bazu podataka srednje škole

Apache Cassandra nema opciju okidača (eng. *trigger*), procedure i funkcija, alipodržava transakcije, no ne garantira ACID svojstva. Naredbe za mijenjanje i brisanje podataka, tablice i eng. *keyspace-a* (eng. *database*) iste su sintakse kao i u relacijskim bazama podataka.

4. USPOREDBA RELACIJSKIH BAZA PODATAKA S APACHE CASSANDROM

Sintaksa naredbi u Apache Cassandri jako je slična relacijskim bazama podataka, no funkcionalnost i arhitektura baze je drugačija (Tab.4.1.)

Tab.4.1. Usporedba Apache Cassandre i relacijskih baza podataka

Apache Cassandra	**Relacijske baze podataka**
Podržava jednostavan upitni jezik.	Podržavaju moćan upitni jezik.
Ima prilagodljivu shemu.	Imajunepromjenjivu shemu.
U konačnici je dosljedna. Ima BASE svojstva (eng. ***B****asically****A****vailable,* ***S****oft-state i* ***E****ventualconsistency*).	Imaju ACID svojstva (nedjeljivost (eng. *Atomicity*), dosljednost (eng. *Consistency*), izolacija (eng. *Isolation*) i izdržljivost (eng. *Durability*)).
Podržava transakcije, no ne zadržava ACID svojstva baze podataka.	Podržavaju transakcije i zadržavaju ACID svojstva baze podataka.
Tablica sadržava stupce ili može biti definirana kao porodica stupaca.	Relacijske tablice jedino definiraju stupce, a korisnik popunjava tablicu s vrijednostima.
Ima sve tipove podataka (strukturirane, polu-strukturirane i nestrukturirane)	Imaju strukturirane podatke.
Tablica je lista parova ključ-vrijednost (RED x KLJUČ STUPCA x VRIJEDNOST STUPCA)	Tablica se sastoji od polja koje sadrži druga polja. (RED x STUPAC)
Eng. keyspace je vanjski kontejner koji sadrži podatke potrebne za aplikaciju.	Baza podataka je vanjski kontejner koji sadrži podatke potrebne za aplikaciju.
Tablice ili porodice stupaca su entiteti eng. *keyspace-a*.	Tablice su entiteti baze podataka
Red je jedinica replikacije.	Red je individualan zapis.
Stupac je spremnik podataka.	Stupac predstavlja atribut relacije.
Veze su predstavljene kolekcijom (porodicom stupaca, kolekcijski tipovi podataka).	Podržavaju koncepte stranih ključeva i spajanja.
Svaki red ne mora imati popunjene sve definirane stupce.	Pri unosu podataka svaki red mora biti popunjen (barem s NULL vrijednosti).

Osim predefinirani tipova podataka, korisnik može stvarati svoje tipove podataka.	Mogu koristiti samo predefinirane tipove podataka.

5. ZAKLJUČAK

Apache Cassandra je ključ-vrijednost i stupac-orijentirana baza podataka koja je otporna na greške i može brzo dohvatiti velik broj podataka. Osmišljena je za pokretanje na jeftinom hardveru, što znači da izvršava jako brza pisanja i može uskladištiti stotine terabajtapodataka bez žrtvovanja efikasnosti čitanja. Svi čvorovi u nakupinama imaju jednaku ulogu te je svaki čvor neovisan i u isto vrijeme međusobno povezan s drugim čvorovima. Također, svaki čvor u nakupini može prihvatiti zahtjeve za pisanje i čitanje bez obzira na mjesto podatak u nakupini.
Glavni zadatak završnog rada je usporedba Apache Cassandrom s najčešće korištenim bazama podataka, relacijskim bazama podataka. Prema rezultatima zaključuje se dasvaka baza ima svoje prednosti i nedostatke. Glavne prednosti Apache Cassandre su fleksibilnost sheme, repliciranje na više čvorova čime se ne gube podaci ako neki od čvorova ispadne iz mreže, može imati sve tipove podataka (strukturirane, polu-strukturirane i nestrukturirane), svaki stupac ne mora imati neku vrijednost (u relacijskim bazama mora biti barem NULL vrijednost) i jednostavnost dohvaćanja podataka (zbog para ključ-vrijednost). No relacijske baze podataka podržavaju moćan upitni jezik, zadržavaju ACID svojstva (nedjeljivost, dosljednost, izolacija i izdržljivost) te su prilagođenija za manje baze podataka. Ovisno o korisnikovim potrebama određuje se koja baza podataka je bolji alat. Ako je potrebno napraviti manju bazu podataka u kojoj su jako bitna ACID svojstva i dohvaćanje podataka iz baze u raznim oblicima, onda se preporuča relacijska baza podataka, no ako je potrebna baza podataka koja može brzo čitati i pisati velike količine podataka i koja neće izgubiti podatke padom servera, onda se preporuča Apache Cassandra.

LITERATURA

[1]http://cassandra.apache.org/ - svojstva Cassandre, arhitektura, 26.-28. 06. 2016.

[2] M. Baranović, S. Zakošek, Baze podataka-Predavanja, veljača 2012., NoSQL baze podataka, 26.06.2016.

[3]https://docs.datastax.com/en/cassandra/2.0/cassandra/architecture/architectureIntro_c.html, arhitektura Apache Cassandre, 01.08.2016.

[4]http://www.bug.hr/molex/apache-cassandra/95207.aspx, osobine, arhitektura i dijelovi Apache Cassandre, 01.- 05.08.2016.

[5]TutorialsPoint (I) Pvt. Ltd., Cassandra querylanguage, 2015., osobine Apache Cassandre, dijelovi Cassandre, model podataka (stupac, superstupac, porodica stupaca, *keyspace*), tipovi podataka, usporedba Apache Cassandre i relacijske baze podataka, naredbe Apache Cassandre,26.06.2016, 01.- 05.08.2016., 20.-30.08.2016.

[6] N, Neeraj, Mastering Apache Cassandra, PacktPublishingLtd., UK, 2013., prikaz prstena, zapisivanje podataka, čitanje podataka, protokol tračanja, model podataka (stupac, superstupac, porodica stupaca, *keyspace*), naredbe Apache Cassandre, 26.06.2016, 20.-25. 07.2016., 01.- 05.08.2016., 20.-30.08.2016

[7]E, Hewitt, Cassandra TheDefinitiveGuide, O'Reilly Media Inc., USA, 2011., prikaz prstena, zapisivanje podataka, čitanje podataka, protokol tračanja, model podataka (stupac, superstupac, porodica stupaca, *keyspace*), 20.-25. 07.2016., 01.- 05.08.2016.

SAŽETAK

Usporedba Apache Cassandre i relacijske baze podataka na primjeru srednje škole

Glavni zadatak završnog rada je usporedba Apache Cassandrom s najčešće korištenim bazama podataka, relacijskim bazama podataka. U radu je dana teorijska osnova Apache Cassandre, napravljen je relacijski model i model Apache Cassandre baze podataka za srednju školu te na temelju teorijskih osnova i modela, napravljena je usporedba baza podataka.

Ključne riječi: Apache Cassandra, NoSQL, relacijska baza podataka, ključ-vrijednost, stupčasti model

ABSTRACT

Title: Comparisonof Apache Cassandra andrelationaldatabase on exampleofhighschool

Themaintaskinthispaperisthecomparisonof Apache Cassandra withthe most useddatabases - relationaldatabases. Thispaperconsistsofthetheoreticalbackgroundof Apache Cassandra, relationaland Apache Cassandra model ofthedatabaseandcomparisonoftheresults.

Keywords: Apache Cassandra, NoSQL, relationaldatabase, key-value, columnbased model

ŽIVOTOPIS

Gabrijela Kramar rođena je 25.01.1994. godine u Osijeku. Od rođenja živi u Višnjevcu gdje stječe osnovnoškolsko obrazovanje u Osnovnoj školi Višnjevac. U 4. razredu osnovne sudjelovala je u natjecanju iz matematike gdje je osvojila 6. mjesto na državnom natjecanju. 7. i 8. razred osnovne škole sudjelovala je u županijskim natjecanjima iz hrvatskog jezika. Upisala se u 3. gimnaziju Osijek 2009. godine. U srednjoj školi sudjelovala je u županijskim natjecanjima iz kemije i informatike. 2011. preselila se u Topolje gdje i danas još živi. Godine 2013. završava srednju školu i državnu maturu, te upisuje Elektrotehnički fakultet, preddiplomski studij računarstva, u Osijeku. Godine 2016. sudjeluje na Elektrijadi u Riminiju iz kolegija Matematika I gdje ekipno postiže mjesto u sredini. Hobi joj je pjevanje u vokalnom ansamblu Brevis s kojim je putovala na brojna svjetska natjecanja i osvajala prva mjesta te dodatne nagrade žirija. Od početka školovanja pa do danas, sve je razrede prošla s odličnim uspjehom.

PRILOZI

Svi prilozi su u datoteci kod.docx.