универзитет у бањој луци

електротехнички факултет

**Александар Бјелошевић**

ДОКУМЕНТ БАЗИРАНЕ БАЗЕ ПОДАТАКА

**дипломски рад**

**Бања Лука, септембар 2015.**

**Тема: ДОКУМЕНТ БАЗИРАНЕ БАЗЕ ПОДАТАКА**

**Кључне ријечи:**

**NоSQL сервери базе података**

**Документ базиране базе података**

**Комисија: проф. др Ратко Дејановић, предсједник**

**проф. др Славко Марић, ментор**

**мр Михајло Савић, члан**

**Кандидат:**

**Александар Бјелошевић**

УНИВЕРЗИТЕТ У БАЊОЈ ЛУЦИ

ЕЛЕКТРОТЕХНИЧКИ ФАКУЛТЕТ

КАТЕДРА ЗА РАЧУНАРСКУ ТЕХНИКУ

Предмет: БАЗЕ ПОДАТАКА

Тема: ДОКУМЕНТ БАЗИРАНЕ БАЗЕ ПОДАТАКА

Задатак: Кратак преглед врста NоSQL сервера база података. Детаљан опис документ базираних база података уз

анализу предности и ограничења у односу на SQL

базе података. За практичан рад реализовати веб апликацију употребом документ базираних база података.

Ментор: проф. др Славко Марић

Кандидат: Александар Бјелошевић (45/07)

Бања Лука, септембар 2015.

1. УВОД 1
2. ПРЕГЛЕД NоSQL СЕРВЕРА БАЗА ПОДАТАКА 2

2.1 Историјат NоSQL сервера база података 2

2.2 Типови NоSQL сервера база података 4

2.3 Специфичности NоSQL и SQL сервера база података 7

1. ДОКУМЕНТ БАЗИРАНЕ БАЗЕ ПОДАТАКА 8

3.1 Преглед документ базираних база података 8

3.2 MongoDB модел података 12

3.3 Документи у MongoDB базама података 12

3.4 MongoDB агрегација 14

3.5 Сигурност у MongoDB базама података 16

3.6 MongoDB операције индексирања 18

3.7 Операција репликације у MongoDB базама података 22

3.8 MongoDB *sharding* 26

1. ПРАКТИЧАН РАД 27

4.1 MongoDB CRUD операције 27

4.2 Практична апликација 37

4.3 Дискусија 45

1. ЗАКЉУЧАК 46
2. ЛИТЕРАТУРА 47

Уз рад је приложен CD

# 2. ПРЕГЛЕД NоSQL СЕРВЕРА БАЗА ПОДАТАКА

## 2.1 Историјат NоSQL сервера база података

## NоSQL базе података пружају механизме за складиштење и добављање података кориштењем средстава која се значајно разликују од табеларних веза које су кориштене у релационим базама података. Континуалан развој веб и мобилних апликација као и константне промјене корисничких захтјева допринјели су развоју нових рјешења за чување и одржавање података.[1]

## NоSQL сервери база података обухватају широки спектар технологија које су развијене као одговор на пораст обима података о корисницима, предметима и производима, учесталости приступа овим подацима као и потребама учинка и обраде података. Релационе базе података, са друге стране, нису пројектоване да се носе са величином и агилности са којма се сусрећу модерне апликације, нити су направљене да искористе јефтиније складиштење и обрадиву моћ хардвера која је доступна данас.[2] Структура података која је кориштена у NоSQL серверима базе података(граф, документ) незнатно се разликује од оне кориштене у релационим базама података, што ће неке операције чинити бржим у NоSQL базама података, док ће остале чинити брже у релационим базама података.

Враћајући се 20 година уназад када су архитекти и програмери апликација, за потребе складиштења и обрађивања података, имали избор различитих релационих база података. Релационе базе су представљале једину опцију за девелопере и инфраструкционе тимове. За ове апликације перформансе и њихова доступност сз биле од пресудног значаја. Нови проблеми у високој доступноски на високој скали покренули су велико компаније као *Google*, *Facebook* и *Amazon* да креирају нова ријешења и нове технологије. Једно ријешење које су ове компаније развиле и унаприједиле су не-релационе базе података које су једним именом назване NоSQL базе података.[1]

Једна од великих заблуда је да термин NоSQL значи „Nо SQL“, као и да NоSQL базе података не користе SQL(прев. Структуирани Упитни Језик) као упитни језик. Али многе NоSQL базе података користе SQL, као један од многих упитних језика. Чињеница је да NоSQL боље описује оно што није, него оно што заправо јесте, израз је и даље веома користан за опис широког спектра класа база података идеалних за рјешавање проблема којима се бавимо данас.[1]

NоSQL базе података веома добро процесуирају: величину, разноврсност и брзину протока великих података.

## 2.1 Типови NоSQL сервера база података

Постоје различити приступи за класификацију NоSQL база података, од којих сваки садржи различите категорије и поткатегорије. Због различитих приступа и преклапања тешко је да се одреди преглед не-релационих база података. Ипак, основна класификација се заснива на разлици у моделу података.

Према моделу података не-релационе базе података можемо подијелити у следеће категорије[2]:

1. документ базиране базе података (енг. *Document model databases*);

2. граф базиране базе података (енг. *Graph model databases*);

3. кључ-вриједност базиране базе података (енг. *Key-value model databases*);

4. базе оријентисане ка колонама (енг. *Column model databases*).

Документ базиране базе података, за разлику од релационих база података које податке складиште у табеле, користе документе за складиштење података. Подаци се у документе типично смјештају у JSON (енг. *JavaScript Object Notation*) формату. JSON представља текстуални стандард који служи за размјену података у формату који је разумљив и читљив од стране човјека и машине. JSON формат изведен је из JavaScript језика, но сам формат је независан од било ког конкретног програмског језика. Документи пружају интуитиван и природан начин за модел складиштење података који је блиско везан са објектно-орјентисаним програмирањем – сваки документ ефикасно представља објекат. Документи могу садржавати једно или више поља, при томе свако поље представља дефинисану вриједност као што су: текстуални подаци, датуми, бинарни подаци или низови података. За разлику од релационих база података гдје су подаци распоређени у више табела и колона повезаних страним кључевима з документ базираним базама података сви записи и њихови повезани (или сродни) подаци се чувају у једном документу. Овакав приступ моделу података поједноставлјује сам приступ подацима и, у многим случајевима, елимише потребу за скупим JOIN операцијама и другим комплексним упитима.[2]

Граф базиране базе података користе графовске структуре заједно са чворовима, гранама и својствима за предстабљање и чување података. У суштини, подаци су моделовани као мрежа односа између специфичних елемената. Граф базиране базе података могу се чинити контра-интуитивне што може бити корисно за специфичне класе упита. Из дефиниције, граф базе података представља сваки систем код којега сви елементи садрже директан показивач на сусједни елемент и при чени није потребан никакав преглед индекса. Слабост граф базираних база података огледа се у томе што оне обично захтијевају да се сви подаци на једном мјесту (једном серверу) ограничајући тиме њихову скалабилност.[1]

Кључ-вриједност базиране базе података имају најједноставнији модел података међу NоSQL базама података. Оне користе претраживи индекс кључ који је повезан са вриједношћу. Ове базе података користе упарене вриједности кључева са вриједностима. Аналогију можемо повући са фајл системом, путања представља кључ док вриједност на путањи (сам фајл) представља вриједност. Релационе кључ-бриједност базе података постоје већ дуги низ година, али новије кључ-вриједност базе података спадају у NоSQL категорију база података јер су намјенски направљене да би подржале велике брзине и скалабилност чиме су одређене функционалности ускраћене (непостојање страних кључева, немогућност претраживања по вриједности и др.). Кључ-вриједност базе података се обично користе за складиштење корисничких подешавања или складиштење великих токова нетрансакционих података.[1]

Базе оријентисане ка колонама су теоретски јако слични табеларном складиштењу података у релационим базама података, једна разлика представља промјенљиви број колокна у редовима. Ове базе података углавном серијализују све вриједности одређене колоне заједно на диску, чиме све добављање специфичног атрибута у великим базама података чини веома брзо. Свака колона се може посматрати као скуп кључ-вриједност, гдје се свака колона идентификује са јединственим примарним идентификатором. Базе оријентисане ка колонама су постале познате након што је *Google* објавио *BigTable*, те је им је подстактнута популарност након *Cassandra-e* и *HBase-a.*[2]

На Табела 1.1 је табеларно представљена поређења различитих NоSQL база података:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Модел података** | **Перформансе** | **Скалабилност** | **Флексибилност** | **Комплексност** |
| Кључ-бриједност | високе | висока | висока | нема |
| Базе оријенстисане ка колонама | високе | висока | средња | мала |
| Документ базиране | високе | промјенљива (висока) | висока | мала |
| Граф базиране | промјенљиве | промјенљива | висока | висока |
| Релационе | промјенљиве | промјенљива | мала | средња |

*Табела 1.1 Поређење различитих NоSQL база података*

## 2.3 Специфичности SQL и NoSQL сервера база података

SQL базе података представљају релационе базе података. Релационе базе података користе релације (табеле) за складиштење података. Релација (табела) у релационим базама података се састоји од колона и редова. За приступ подацима у SQL базама података користи се структуирани упитни језик (енг. *Structured Query Language*).[4]

Основне караткеристике SQL сервера база података су: [2]

* релационе базе података се састоје од релација (табела) које садрже податке;
* свака табела садржи једну или више категорија података садржаних у колонама;
* свака ред у табели садржи један јединствени кључ – примарни кључ;
* корисници могу приступити подацима унутар базе података без познавања саме структуре табеле базе података.

Нека од ограничења код SQL сервера база података огледају се у комплексностии скалабилности. Комплексност се односи на ситуације у којима одређени подаци не могу да се смјесте у тренутну структуру базе података. У том случају поновно дизајнирање структуре базе података је неопходно чиме се повећава и њена комплексност. Приликом скалирања релациона база података мора бити расподјељена на више сервера. Обрађивање података из табела које се налазе на више сервера представља јако комплексан задатак.[4]

NoSQL базе података не захтјевају фиксну шему табеле, избјегавајући тиме операције придруживања (енг. *join operations*) чиме се типично хоризонтално скалирају. Код NoSQL подаци се могу смјештати без претходно дефинисане шеме базе података. Формат података се може промјенити у сваком тренутку без страха да ће се пореметити постојеће апликације. Ово омогућава огромну флексибилност апликација, која на крају испоручује значајну пословну флексибилност.[2]

Разлике NoSQL и SQL база података: [4]

* NoSQL базе података генерално процесуирају податке знатно брже у односу на релационе базе података;
* NoSQL базе података су обично и брже у односу на релационе из разлога што им је модел података знатно једноставнији;
* шема базе података је флексибилна што представља значајну предност у односу на релационе базе података;
* у апликацијама гдје се користе комплексни упити SQL база података знатно предњачи у односу на NoSQL;
* SQL базе података су вертикално скалабилне док су NoSQL базе хоризонтално скалабилне;
* SQL база истиче ACID (Atomicity, Consistency, Isolation, Durability) svojstva dok је NoSQL база заснована на CAP теореми (Consistency,Availability Partition tolerance).

# 3. ДОКУМЕНТ БАЗИРАНЕ БАЗЕ ПОДАТАКА

## 3.1 Преглед документ базираних база података

Као што смо раније навели, документ базиране базе података користе документе за склатиштење података за разлику од релационих које користе табеле. Подаци у документима могу бити сериализовани у различитим форматима. Неки од формата су: XML, YAML, JSON и BSON. Документима се приступа преко јединственог кључа који је везан за специфични документ. Овај кључ је престављен преко јединственог низа слова, јединственог идентификатора ресурса (енг. *Uniform resource identifier*) или јединствне путање. Типично, база података индексира овај кључ како би се повећала брзина приступа документима. [4]

Неки од познатијих представника документ базираних база података су[4]:

1. MongoDB

2. CouchDB

3. Couchbase

4. Clusterpoint

5. MarkLogic

MongoDB представља документ базу података са високим перформансама, високе доступности и аутоматског скалирања. Документи имају динамичку шему. Подаци се смјештају у JSON формату (BSON у званичној MongoDB документацији). MongoDB ће бити детаљније објашњен у дањем тексту.[4]

CouchDB представља базу података која је у потпуности окренута ка вебу. Подаци су такође сачувани у JSON формату. CouchDB омогућава приступ подацима и извршавању упита преко HTTP протокола кориштењем веб читача. Омогућава индексирање, комбиновање и трансформацију докумената кориштењем JavaScript-а. CouchDB је оптимизована и добро ради са модерним веб и мобилним апликацијама. Такође, CouchDB омогућава ефикасно дистрибуирање података кориштењем CouchDB инкременталне репликације. CouchDB долази са пакетом функција за трансформације докумената и промјенама обавјештења у реалном времену, чиме се знатно олакшава развој веб апликација. Такође, омогућена је и једноставна административна конзола доступна преко веба.[5]

Couchbase представља дистрибуирану NoSQL документ оријентисану базу података оптимизовану за интерактивне апликације. Couchbase базе података су дизајниране да знатно повећају брзину приступа документима уз висок и одржив проток. Неке од ствари које Couchbase сервери база података дизајнирани да омогуће су: флексибилни модел података, снажан упитни језик, скалабилност, перформансе, једноставну администрацију и високу доступност. Документи су форматирани у JSON формату. За разлику од релационих база података шема је у потпуности дефинисана у коду апликације и у дефиницији самих докумената. Couchbase базе података користе N1QL као упитни језик за приступ, терансформацију и манипулацију JSON подацима. N1QL представља комплетан SQL диалект чиме је знатно једноставан за учење од стране програмера који познају SQL. За разлику од других NoSQL база података, Couchbase подржава операције придруживања (енг. *JOIN operations*) што омогућава широк спектар модела података. N1QL подржава богату структуру индексирања. Такође, N1QL подржава додатке помоћу којих је омогућено руковање хетерогеним документима са угњежденим структурама. Репликација представља једну од основних карактеристика Couchbase сервера база података. Сервери су дизајнирани да аутоматски дистрибуирају податке кроз све чворове у кластеру.[6]

Clusterpoint представља документ оријентисану базу података која обрађивање и процесуирање XML и JSON типова докумената у дистрибуираним окружењима. Clusterpoint базе података обављају своје трансакције на исти начин као и SQL базе података. Корисници обављају сигурне упите у реалном времену, текстуалне претраге, аналитичке SQL упите и извјештаје у великој брзини у великим дистрибуираним системима. Главна компаративна предност Clusterpoint база података огледа се у томе: што су оне ACID (енг. *Atomicy, Consistency, Isolation, Durability*) компатибилне, обављају дистрибуиране трансакције међу документ оријетисаним моделима, гарантује сигурниост и конзистентност података.[7]

MarkLogic сервери база података су: документ оријентисани, трансакциони, клустер оријентисани, високих перформанси, шема-агностични и оријенстисани ка претрагама. Документи су обично у XML формату. MarkLogic базе података у потпуности подржавају ACID својства: Atomicy (скуп измјена или се одвија у цијелини или се не одвија уоште), Consistency (pravila sistema se sprovode, нпр. не постоје два документа који имају исти идентификатор), Isolation (недовршене трансакције нису иначе видљиве) и Durability (подаци се неће изгубити при комуникацији са базом). MarkLogic омогућава индексирање како самог текста тако и структуре докумената чиме се омогућава извшавање упита међу њима.[3]

Неке од осталих документ оријентисаних база података које је вриједно поменути су: BaseX, Virtuoso Universal Server, TokuMX, Terrastore, RavenDB, OrientDB, ThruDB, SisoDB, RaptorDB, CloudKit, Perservere.

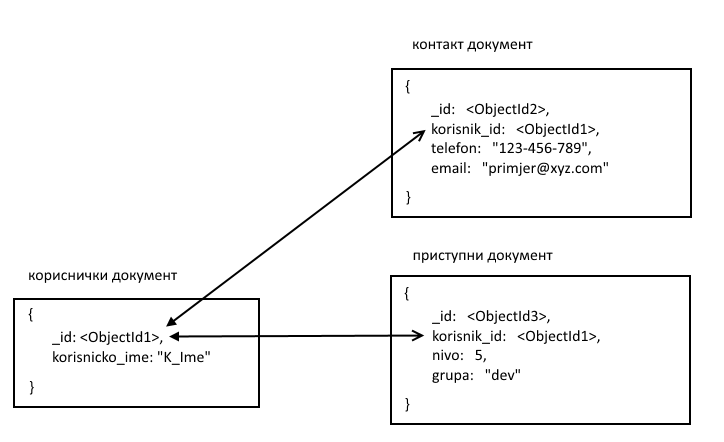
## 3.2 MongoDB модел података

Документи у MongoDB бази података садрже флексибилну шему. За разлику од SQL база података, гдје се прво мора утврдити шема табеле прије смјештања података, колекције код MongoDB базе не захтјевају предефинисане шеме података. Ова флексибилност омогућава мапирање докумената на ентитете или објекте. Сваки документ може се упоредити са пољима представљеног ентитета, чак и ако подаци значајно варирају. У пракси, међутим, документи у једној колекцији дијеле сличну структуру.

Кључни изазов у моделовању података представља балансирање између потреба саме апликације, перформаси карактеристика самих база података, као и образаца за проналажење самих података. При пројектовању модела, потребно је размотрити употребу података саме апликације (нпр. упити, ажурирање и обрада података), као и својствену структуру самих података.

Кључна одлука у пројектовању модела података у MongoDB базама података врти се око структуре докумената и како апликација престравља однос између самих података. Постоје два алата који омогућавају апликацијама да представљају ове односе: референце и уграђени документи.

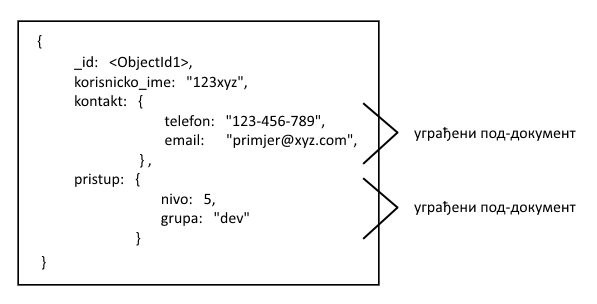
Референце складиште релације између података преко веза или референци са једног документа ка другом. Референце представљају нормализоване моделе података. Референце прижају већу флексибилност него уграђени документи. Међутим, апликације морају извршити и по неколико упита ка бази података како би разријешили референце.



*Слика 1.1 Референце у MongoDB*

У пракси нормализоване моделе података потребно је користити: када би уграђени документи доводили до дуплирања података при чему се не би добила довољна брзина података која би оправдала дуплирања, да би се представиле комплексне више-ка-више релације и при моделовању великих хијерархијских скупова података.

Уграђени документи складиште све референте податке (документе) у један документ. Овакве шеме су познате као „денормализовани“ модели. MongoDB документи омогућавају да се уграде структуре докумената у поље или у нуз унутар самог документа. Ови денормализовани модели података омогућавају апликацијама да прикупе и манипулишу подацима у само једном обраћању бази података. Слика 1.2 даје као примјер уграђене документе у MongoDB базама података.



*Слика 1.2. Уграђени документи у MongoDB*

Уграђени документи обезбјеђују боље перформансе за операције читања, као и способност да захтјевају и проналазе податке у једном упиту ка бази података. Међутим, документи могу знатно расти што може утицати на перформансе и довести до фрагментације података.

У пракси уграђене документе препоручљиво је користити: уколико имамо један-на-један оресликавање и један-на-више пресликавање између ентитета.

У MongoDB базама података, атомност се задржава на нивоу документа, и ниједна операција на нивоу атомности не може утицати на више докумената или на више од једне колекције. Код денормализованих модела података уграђени документи комбинују све репрезентативне податке у једном документу. То олакшава атомност операција уписа јер само једна операција је довољна за ажурирање или упис података. Међутим, нормализовањем подаци ће бити подјељени у више колекција чиме би се број уписа и ажурирања повећао и тиме би утицали на атомност.

## 3.3 Документи у MongoDB базама података

MongoDB чува податке на диску у BSON серијализованом формату. BSON представља бинарну репрезентацију JSON доккумента, иако садржи више врста података него JSON. Сама структура документа се састоји од парова поље-вриједности:

{ “predmet” : “olovka”, “broj” : 500, “tip” : “pribor” }

BSON подржава следеће типове података као вриједности у документима. Сваки тип податка има одговарајући индекс који се може искористити са **$type** оператором при генерисању упита.

|  |  |  |
| --- | --- | --- |
| **Тип** | **Индекс** | **Напомена** |
| Double | 1 |  |
| String | 2 |  |
| Object | 3 |  |
| Array | 4 |  |
| Binary data | 5 |  |
| Undefined | 6 | Застаријело. |
| Object id | 7 |  |
| Date | 9 |  |
| Null | 10 |  |
| Regular Expression | 11 |  |
| JavaScript | 13 |  |
| Symbol | 14 |  |
| JavaScript (with scope) | 15 |  |
| 32-bit integer | 16 |  |
| Timestamp | 17 |  |
| 64-bit integer | 18 |  |
| Min key | 255 | Код упита користи се -1 |
| Max key | 127 |  |

Примјер кориштења **$type** оператора:

db.predmeti.find( { ime: { $type : 2 } } );

Претходни примјер ће да прикаже листу свих докумената који садрже поље „ime“ које је или текстуални податак (енг. *String*) или низ података (енг. *Array*) који садржи бар један текстуални податак. Детаљнији опис MongoDB CRUD (енг. *Create, Read, Update, Delete*) операција биће приказан у наредној секцији.

Као што је већ раније поменуто, документи се садрже податке типа поље-вриједност. Вриједност може да буде било који BSON тип податка, укључујући и друге документе, низове података, као и низове докумената. Следећи примјер представља документ сложеног типа:

var dokument = {

**\_id** : ObjectId(“5099803df3f4948bd2f98391”),

**puno\_ime**: { ime: “Alan”, prezime: “Turing” },

**datum\_rodjenja**: new Date(“Jun 23, 1912”),

**datum\_smrti**: new Date(“Jun 07, 1954”),

**prilozi**: [ “Tjuringova masina”, “Tjuringov test” ],

**broj\_pregleda**: NumberLong(1250000)

}

6. **ЛИТЕРАТУРА**

[1] IBM Coloudant, *Why NoSQL?*, <https://cloudant.com/wp-content/uploads/Why_NoSQL_IBM_Cloudant.pdf>, посјећено: 16.07.2015.године.

[2] MongoDB, *Top 5 Considerations When Evaluating NoSQL Databases*, <https://s3.amazonaws.com/info-mongodb-com/10gen_Top_5_NoSQL_Considerations.pdf>, посјећено: 17.07.2015.године.

[3] MarkLogic, *The NoSQL Generation: Embracing the Document Mode*, <http://www.marklogic.com/wp-content/uploads/2014/12/nosql-generation-embracing-document-model.pdf>, посјећено: 17.07.2015.године.

[4] MongoDB документација, <http://docs.mongodb.org/manual/>, посљедња посјета септембар 2015.

[5] CouchDB документација, <https://cwiki.apache.org/confluence/display/COUCHDB/>, посљедња посјета септембар 2015.

[6] Couchbase документација, <http://developer.couchbase.com/documentation/server/4.0/introduction/intro.html>, посљедња посјета септембар 2015.

[7] Clusterpoint документација, [https://www.clusterpoint.com/docs/,](https://www.clusterpoint.com/docs/,%20) посљедња посјета септембар 2015.