Зборник радова Факултет техничких наука, Нови Сад

UDK: 004.9
DOI: https://doi.org/10.24867/18BE24Arsenijevic

АНАЛИЗА ПЕРФОРМАНСИ УПИТА У РЕЛАЦИОНИМ БАЗАМА ПОДАТАКА У ЗАВИСНОСТИ ОД УПОТРЕБЕ ИНДЕКСА

QUERY PERFORMANCE ANALYSIS IN RELATIONAL DATABASES DEPENDING ON THE USE OF THE INDEX

Никола Арсенијевић, *Факултет техничких наука, Нови Сад*

Област – ЕЛЕКТРОТЕХНИКА И РАЧУНАРСТВО

Кратак садржај – *Креирана је једноставна релациона база података и мерено је време потребно да се изврше различити типови упита над базом података, користећи различите индексе. База података се састоји из 20 табела исте шеме релације али са променљивим бројем насумично генерисаних торки. Над сваком од њих су креирани индекси типа Б-стабло и бинарни индекс. Са свим релацијама и свим типовима индекса вршило се 7 одабраних, фреквентних упита и мерено је просечно време одзива. На крају су сва добијена мерења приказана графички у трослојној веб апликацији.*

Кључне речи: *релациона база података, брзина одзива, Б-стабло, бинарни индекс, веб апликација*

Abstract *–A simple relational database was created and time needed for different queries to complete was measured, while using different indexes. The database consisted of 20 tables with the same underlying scheme but with different numbers of randomly generated rows. Subsequently, B-tree and binary indexes on each of them were created. Seven frequently used queries were executed on each table and each type of index and the average execution time was measured. The results of measurements were presented in a 3-layered web application and commented.*

Keywords: *relational database, query execution time, B-tree, binary index, web application*

1. УВОД

Савремене базе података, тј. складишта података, постале су незаобилазни део свакодневице у готово свим областима људског деловања. Користе се у науци, индустрији, банкарству, администрацији, видео играма и др. Развојем рачунарства расте и број и величина база података које су у употреби. Савремен хардвер нуди све већи и већи капацитет за складиштење података, али више меморијског капацитета није увек решење. У зависности од случаја употребе, већа база података са више података може бити веома корисна, али може изазвати и нове потешкоће, попут хардверске подршке и брзине рада са базом података [2]. Тема овог рада јесте примена индекса код релационих база података како би се убрзало извршавање упита.

НАПОМЕНА:
Овај рад проистекао је из мастер рада чији ментор је био др Милан Челиковић, доцент.

2. ТЕОРИЈСКА ОСНОВА РАДА

У овом одељку дат је кратак преглед теорије на којој је рад заснован.

2.1 Релационе базе података

Релационе базе података први пут је описао Тед Код 1970. године. Основно начело рада са релационим базама података јесте концепт релација, тј. табеле, описане својим обележјима, тј. колонама, са редовима попуњеним торкама, тј. подацима. Релацију описује скуп обележја и прикључених ограничења.

Из перспективе корисника базе података, сви подаци из релационе базе података се у меморији чувају у виду табела. Корисник може да креира табеле, може да их мења, брише и попуњава подацима, и једина брига коју при раду мора да има јесте начин на који су подаци логички организовани. О физичкој организацији података корисник не мора да размишља; она се може препустити систему за управљање базом података - СУБП. СУБП може да извршава комплексне процесе ради складиштења података, оптимизације итд, али корисник не мора да их буде свестан [1].

2.2 Организација рачунарске меморије

Меморијски уређаји имају коначан меморијски капацитет, који је подељен на меморијске локације. Ове локације су обично недељиве, све су најчешће једнаког капацитета на истом уређају, који их третира као уређену секвенцу меморијских локација. Уколико је неки податак, тј. меморијски објекат, превише велик да би се цео могао сместити у једну меморијску локацију, биће смештен у више локација, које могу али не морају бити узастопне локације у секвенци.

Торке релације могу се посматрати као независни меморијски објекти. Они се смештају у меморију рачунара ради њиховог чувања или даље обраде. Због начина на који је већина ових уређаја имплементирана, у општем случају не постоји гаранција да ће једне релације бити смештене у исту, или чак више узастопних, меморијских локација. СУБП треба да на неки начин води евиденцију о томе како су на меморијском уређају распоређени подаци из свих торки свих релација који припадају бази података. То се, опет, реализује помоћу специјализованих дескриптора СУБП.

Када је у питању рад са базама података, један од главних узрока успорења рада базе података може бити спора комуникација између радне и трајне меморије. Приликом упита, подразумевани начин рада јесте да се, део по део, цела релација која је предмет упита учита у радну меморију, након чега се избацују оне торке које не одговарају критеријуму селекције а кориснику приказују оне које остану. Како количина података са којом база података мора да рукује расте, било због растућег броја корисника, или веће количине података у бази података које је потребно обрадити или из оба разлога, рад са базом података може бити значајно успорен, што је веома непожељно.

2.3 Индекс

Једно од могућих решења за споро извршавање упита може бити увођење индекса. Индекс представља помоћну структуру података која за неку релацију памти вредности свих торки за унапред дефинисан скуп обележја те торке, који је готово увек прави подскуп, и меморише сваку меморијску локацију на трајном меморијском уређају на којој се налази торка са том вредношћу. Ово је корисно зато што уколико имамо упит селекције према неком од индексираних обележја на основу индекса можемо неке торке да одбацимо из селекције и пре него што буду учитане у радну меморију ради провере њихове вредности на том обележју. То значи да ће бити потребно да мања количина података буде учитана са трајног меморијског уређаја, што у неким случајевима може да вишеструко скрати време упита.

2.3.1 Б-стабло

Б-стабло, тј. балансирано стабло (енг. *B-tree, Balanced tree*), представља посебну, помоћну структуру података која се придружује тебелама ради убрзања операција читања података. Б-стабло чува податке уређеним према неком критеријуму и поседује механизме помоћу којих може да врши ребалансирање, како би у сваком тренутку било што боље балансирано, ради бржег рада. Пошто су подаци у Б-стаблу сортирани, стабла се често користе за претрагу. Са порастом количине података у стаблу, време тражења једног елемента расте логаритамски [1].

2.3.2 Бинарни индекс

Бинарни индекс (енг. *Bitmap Index*) је помоћна структура која се придружује релацији, а састоји се од једног или више бинарног низа. Везује се за једно или више обележја релације. Бинарни низ има тачно онлико елемената колико је торки у релацији за коју је везан, и сваки низ се везује за једну конкретну вредност обележја (или комбинацију вредности обележја, уколико их прати више). Пошто је број елемената низа једнак броју торки, а торке су нужно у меморију смештене у неком редоследу, сваки елемент низа логички се поклапа са по једном торком, и то баш оном чији редни број дели. Уколико торка неког низа садржи управо ону вредност/оне вредности за које је низ везан, одговарајући елемент низа имаће на том положају логичко тачно, а у супротном ће имати логичко нетачно. На тај начин, уколико се врши претрага табеле по вредности бинарног низа, на основу самог низа могу да се открију све торке где се та вредност у меморији налази, без учитавања података из торке у радну меморију и поређења вредности из торке са оном која се тражи, и тако брже приступи и осталим елементима торке без потребе за учитавањем вредности које не испуњавају критеријум претраге [2].

3. ПРОЦЕС МЕРЕЊА И ПРИКАЗА ПОДАТАКА

Циљ овог рада био је да се направи алат за мерење брзине одзива базе података у зависности од количине података које база података садржи и употребе различитих типова индекса.

У првој фази креирано је 20 релација исте шеме релација, која описује раднике неке компаније. Потом је генерисано 20 скрипта са по n*1,000,000, n < {1, 2, 3, 4, 5, ..., 20}, инсерт наредби са случајно генерисаним вредностима, које по типу вредности одговарају атрибутима релација радника. На крају су тих 20 скрипта покренуте, и напуниле су табеле за мерење подацима. Оволико података је било потребно зато што кориштени софтвер мери време са презизношћу од до 10 ms, а операције са мањим табелама на модерном хардверу често трају краће од тога.

Више података у релацијама успорава брзину одзива, што значи да ће разлика у времену одзива бити већа под различитим условима извршавања упита, што је пожељно, пошто је главни циљ пројекта био упоређивање ефикасности одзива у зависности од параметара, а не мерење тачног времена одзива.

Наредна фаза била је мерење. У фази мерења написане су функције које за свих 20 табела радника врше по један од 7 одабраних упита у по три серије; прво без индекса, потом са индексом типа Б-стабло, и на крају са бинарним индексом. Сваки упит се вршио по 20 пута. На крају су се просечне вредности времена одзива бележиле у посебну, за то осмишљену, табелу.

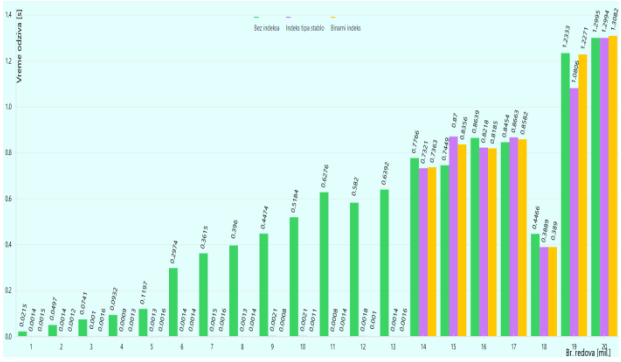
Трећа фаза припреме података било је креирање графичког приказа добијених података. Креирана је трослојна веб апликација која крајњем кориснику укратко описује како су подаци добијени, и графички приказује резултате свих мерења. Најнижи слој представља иста база података која је кориштена у претходна два корака. Фронт енд на захтев корисника може да прикаже текст о самом пројекту или неки од графика мереног времена потребног за одзив, подељене према упиту. Средњи слој нема много пословне логике и служи да податке из базе података форматира на начин који је предњем слоју читак.

4. АНАЛИЗА

Анализа брзине одзива неких од коришћених упита приказана је графички. Сви графици су дизајнирани, тако да сена x-оси налазе бројеви од 1 до 20, који представљају број милиона редова у табели над којом се врши мерење, а на y-оси просечно време које је било потребно да се упит изврши у секундама. Свакој x-вредности одговарају по три стубића. Зелени стубићи представљају одзив при упитима над неиндексираним табелама, љубичасти стубићи представљају упите над табелама са индексом типа Б-

стабло, а жути стубићи табеле са бинарним индексима.

4.1 Упит на основу вредности примарног кључа



Слика 1: *Приказ брзине одзива за упите са претрагом по примарном кључу*

Ако се погледа брзина одзива на упит за табеле без индекса [Слика 1], уочава се приближно линеарни пораст времена одзива у зависности од броја редова у табели. Обзиром да подаци у табели нису сортирани, овакав исход био је и очекиван. Приликом позива упита СУБП почиње да чита податке из торки случајним редоследом и за сваку учитану торку проверава да ли садржи тражену вредност кључа. Више података значи да ће сваки покушај имати мању вероватноћу проналажења тражене вредности. Порастом количине података та вероватноћа експоненцијално опада, а потребан број провера да би се торка пронашла расте линеарно.
Међутим, са графика се такође види да тај линеарни тренд има неколико јасних скокова, након којих наставља свој приближно линеарни раст. То вероватно може да се објасни начином на који *Оракл* [3] складишти податке; иако СУБП-у није експлицитно задато да користи стабло, он ће то сам према предефинисаној поставци табеле урадити. Како број података у стаблу расте, потребно га је периодично ребалансирати, што чини приступ случајном податку према вредности примарног кључа у општем случају бржим, али такође повећава број корака које у општем случају треба проћи да би се стигло од корена до жељеног листа стабла, пошто је дубина целог стабла постала већа.

Дубље стабло и више корака у његовом обиласку значе и више времена потребног за одзив. Ако се упореде вредности у табелама од 1 до 5 милиона редова, са вредностима у опсегу од 6 до 15 милиона редова, види се скок у времену одзива између табеле са 5 и табеле са 6 милиона редова. Претпоставка је да је то успорење присутно управо због употребе структуре попут стабла од стране СУБП, које реструктурирањем стабла повећава његову дубину и нагло успорава упите у региону који гледамо.

Следећа значајна промена линеарног тока јавља се у регији између 17 и 19 милиона редова у табели. Претпоставља се да је и ту повод реструктурирање стабла, које прво учини одзиве бржим при преласку са 17 на 18 милиона редова, а затим настави приближно линеарни тренд при преласку са 18 на 19 милиона редова.

Реструктурирање стабла га прво избалансира, што чини упите бржим, а дубљи слојеви стабла у почетку остају непопуњени. Ипак, како број торки расте и стабло се попуњава, расте и његова дубина и самим тим и потребан број корака да се одређени чвор пронађе, те се враћа претходни линеарни шаблон. Поред тога, већа помоћна структура при раду мора да се учитава у меморију, што дотатно успорава рад са порастом броја редова. Вероватно је да би се такав тренд наставио ако би се и даље повећавао број редова у табели са којима се ради; приближно линеаран раст времена одзива у одређеним зонама са периодичним наглим падовима праћеним још већим скоковима када се предефинисано *Оракл* [3] стабло реструктурира.

4.2 Упит на основу минималне вредности

Слика 2: *Приказ брзине одзива за упите са претрагом по минималној вредности*

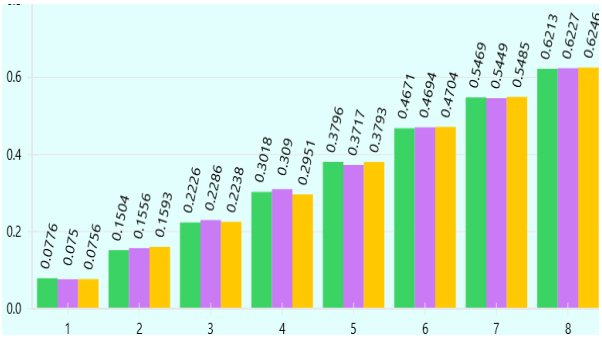
Време одзива неиндексираних табела расте приближно линеарно, уз неколико скокова, са порастом броја редова у табелама. Понашање личи на оно у првом упиту за неиндексиране табеле [Слика 2].
Време одзива на овај упит при употреби индекса типа Б-стабло готово је занемарљиво за табеле са мање од 14 милиона редова података. Претпоставља се да је то управо зато што цео индекс заузима довољно мало меморијског простора да га СУБП стално држи у радној меморији, те по потреби може веома брзо да прође кроз индекс и врати одговор. Код табела са >14 милона редова долази до наглог успоравања одзива. Вероватно је то последица раста индекса заједно са количином података, што доводи до тога да цео индекс више није смештен у радну меморију, већ га треба учитати са диска. Брзина одзива почиње да прати брзину одзива код неиндексираних табела, када количина података у табелама пређе одређени праг.

4.3 Упит према делу саадржаја текста

Код упита са делом садржаја текста критеријум селекције је постојање одређеног подниза знакова у низу знакова бираног обележја. За овај упит је изабран један низ знакова са којим су вршени сви упити. Упит чији је одзив посматран био је:

```
SELECT * FROM radnik WHERE
 SUBSTR(ime, 3, 2) = 'ra'
```

Упит враћа све податке о оним радницима чије име садржи знакове 'ra' на трећем и четвртом положају.



Слика 3: *Приказ дела графика брзине одзива за упите са претрагом по вредности дела садржаја текста*

Слично упиту где је критеријум селекције био мање од неке граничне вредности, и овај упит мора да прође кроз све торке у табели и провери да ли њихова вредност за торку одговара критеријуму. За све три варијанте табела што се тиче индекса, време потребно за упит расло је приближно линеарно са порастом броја редова у табелама [Слика 3]. Делује као да врста индекса није имала готово никакав утицај на потребно време одзива.

Претпоставља се да томе доприносе два фактора: 1) чињеница да приликом читања свих вредности индекси не значе много јер су осмишљени тако да брже пронађу одређену вредност или скуп вредности у табелама, и 2) чињеница да су операције са стринговима у принципу релативно споре у рачунарству, те је разлика коју индекси могу да направе у односу на време потребно за саму операцију провере стринга занемарљиво мала.

4.4 Општи закључци

На основу мерења, јасно је да под одређеним условима неки индекси имају потенцијал да значајно убрзају одзив на упите, али то убрзање над неком табелом нипошто не важи у свим условима и за произвољни упит. Подаци са којима се радило у овом раду су били високе селективности, што у општем случају више одговара индексу типа Б-стабло него бинарном индексу, те је можда зато Б-стабло готово увек имало боље резултате од бинарног индекса, а често су се и веома слично понашали.

Такође се може закључити да је индекс највише убрзао одзив при оним упитима који не морају да проваравају све торке, већ само нађу неке редове по вредности њихових обележја. Надаље, индекси су најзначајније смањили време одзива када су били довољно малог меморијског капацитета тако да цели могу да се чувају кеширани у радној меморији.

У Оракл СУБП [3] је могуће ручно подесити максималну дозвољену количину кешираног индекса. Стога би у пракси при решавању проблема спорог одзива упита прво требало да се уведе индекс, а потом да се провери величина тог индекса у односу на максималну дозвољену количину меморије за кеширани индекс, и обезбеди да дозвољени меморијски кеш за индекс буде већи од величине самог индекса.

Индекси су боље резултате показали при убрзању процесорски мање захтевних радњи, попут провере исте бројевне вредности, него при убрзавању процесорски захтевнијих радњи, попут претраге дела садржаја текста.

Ако је у неком систему упит где је део садржаја текста критеријум селекције критичан фактор који успорава рад, на основу овог истраживања, препоручује се испитивање других решења, а не употреба индекса. Индекси заузимају више меморије и успоравају измене података, а нису показали измерљиво убрзање рада. Овде треба напоменути и да је индекс прављен према целој вредности стринг обележја, а не према делу садржаја текста који је упит користио као критеријум селекције.

5 ДАЉИ ПРАВЦИ ИСТРАЖИВАЊА

Након мерења и анализе података, може се закључити да индекси у неким случајевима, попут претраге по најмањој вредности обележја, стварно могу да убрзају одзив упита за више редова величине у односу на неиндексиране табеле. Такође се види да у другим случајевима исти индекси уопше не морају значајно утицати на време одзива, као приликом претраге по делу садржаја текста. Поред тога се види разлика у одзиву између индекса који су довољно малог меморијског капацитета да цели стају у радну меморију и оних који су превише велики, те их је потребно прво учитавати у радну меморију.

Најједноставнији смер за даља истраживања било би просто даље повећавање броја редова у табелама. Поред тога, вредело би променити податке, тако да вредности неких обележја буду из мањих скупова предефинисаних вредности уместо да у сваком реду буду насумично генерисани бројеви или стрингови. Тако би се можда више истакле разлике између бинарних индекса и индекса типа Б-стабло. На крају, оглед би вредело вршити и на потпуно различитим табелама, са више или чак мање обележја, другачијим типовима обележја и сложеним упитима који користе више различитих табела да добаве податке.

ЛИТЕРАТУРА

[1] D. Comer, (1979), "The Ubiquitous B-Tree", Computing Surveys
[2] https://www.geeksforgeeks.org/bitmap-indexing-in-dbms (27.12.2021)
[3] https://www.java.com/en/ * (09.02.2022)

Кратка биографија:

Никола Арсенијевић рођен је 1997. у Новом Саду. Дипломски рад из области рачунарства и аутоматике одбранио је 2020. године на Ф.Т.Н. у Новом Саду. Мастер студије на Ф.Т.Н. у Новом Саду је уписао 2020. године.