U n i v e r z i t e t u B e o g r a d u

G. Lazović G. Vorotović

Č. Mitrović I. Aranđelović A. Bengin

NAPREDNI ALATI ZA UPRAVLJANJE BAZAMA PODATAKA

M a š i n s k i f a k u l t e t

Beograd

U n i v e r z i t e t u B e o g r a d u

G. Lazović G. Vorotović

Č. Mitrović I. Aranđelović A. Bengin

NAPREDNI ALATI ZA UPRAVLJANJE BAZAMA PODATAKA

M a š i n s k i f a k u l t e t

Beograd, 2017

Autori:
dr Goran Lazović, docent,
dr Goran Vorotović, docent,
dr Časlav Mitrović, redovni profesor,
dr Ivan Aranđelović, redovni profesor,
dr Aleksandar Bengin, redovni profesor.

NAPREDNI ALATI ZA UPRAVLJANJE BAZAMA PODATAKA

Prvo izdanje

Recenzenti:
Prof. dr Zlatko Petrović, Mašinski fakultet Univerziteta u Beogradu
Prof. dr Dragan Cvetković, Fakultet za informatiku i računarstvo, Univerzitet Singidunum

Izdavač:
MAŠINSKI FAKUTET
Univerziteta u Beogradu
Ul. Kraljice Marije 16, Beograd
tel. (011) 3370 760
faks. (011) 3370 364

Za izdavača:
Prof. dr Radivoje Mitrović, dekan

Glavni i odgovorni urednik:
Prof. dr Milan Lečić

Odobreno za štampu odlukom Dekana Mašinskog fakulteta u
Beogradu br. 22/17 od 08.09.2017. godine

Tiraž: 100 primeraka

ISBN-978-86-7083-953-3

Štampa:
PLANETA – print
Ruzveltova 10, Beograd
tel/faks:(011)3088 129

PREDGOVOR

Knjiga koja je pred Vama je plod pedagoškog iskustva. Poslednjih 20 godina držali smo više univerzitetskih kurseva iz oblasti Relacionih baza podataka i njihovih primena, studentima Mašinskog fakulteta Univerziteta u Beogradu i odseka za informatiku Vojne akademije. Pored toga u okviru međunarodnog AQUIT projekta, formirali smo kompletne nastavne i ispitne materijale za specijalistički kurs Relational database systems, namenjen diplomiranim inženjerima mašinstva, za potrebe nemačke Steinbeis fondacije. Ovaj međunarodno verifikovani kurs je u našoj državi održan na Univerzitetima u Beogradu, Novom Sadu, Kragujevcu i Nišu, školske 2004/05 godine.

Evidentan eksponencijalni rast informacionih tehnologija sa, prvenstveno, aspekta centralizacije baza znanja, uslovio je veoma neobičnu pojavu. Naime, u modernim IoT (Internet of Things) tehnologijama baze podataka su dobile fundamentalnu ulogu, što je dovelo do pojave novih zahteva prilikom projektovanja i eksploatacije baza podataka. Kao što je poznato SQL (Structured Query Language) je definisan sedamdestih godina prošlog veka. Međutim programska okruženja novije generacije isključivo forsiraju upotrebu SQL-a kao primarnog jezika u okvirima interfejsa, kako korisničkih, tako i na primarnim nivoima kontrole aplikativnih softvera.

Praktični problemi sa kojima su se autori ove knjige susretali, ukazali su na neophodnost razdvajanja postupaka upisivanja podataka, od učitavanja i obrade istih. Naime, mašinska nauka kao najveća oblast tehnike, u bezbroj situacija je ukazala na potrebu kvalitetne obrade podataka koji u velikim količinama, pa čak i na nivou BLOB-a (Binary Large Objects) pristižu (telemetrija vazduhoplova, CAN magistrala na motornim vozilima ili u CERN-u, verifikacija toplotnih ciklusa, ispitivanje mašinskih konstrukcija, kontrola saobraćaja, itd.). Evidentni nedostaci modernih spreadsheet softvera u oblastima multidimenzionalnih problema, odziva u realnim uslovima eksploatacije i standardizacije rasporeda i distribucije podataka, uslovili su da autori u svom radu sa bazama podataka izdvoje upravljanje bazom podataka, kroz primitive upita u SQL-u, u jednu celinu.

Materijal koji je pred Vama sadrži suštinu generisanja upita u modernim bazama podataka i kao takav ga mogu koristiti studenti u procesu analiza baza podataka, a svojom “pitkošću” i demonstracijom modernih algoritama je aktuelan i za profesionalce u ovoj oblasti. U knjizi su data odgovarajuća objašnjenja kao i uputstva i pitanja, a u cilju popularizacije korišćenja baza podataka u svim sferama tehnike, prvenstveno u mašinstvu, kao celini.

Beograd, 2017

Autori

Sadržaj

Skraćenice i pojmovi

IoT (engl. *Internet of Things*) – najšešće prevođeno kao *Internet stvari*. Predstavlja sistem, putem interneta umreženih objekata (stvari) iz svakodnevnog života - kompjutera, telefona, mehaničkih i digitalnih mašina, zgrada, životinja, ljudi, ... Objekti imaju jedinstvenu identifikaciju, ugrađenu elektroniku, softver, senzore i komunikacione uređaje koji im omogućavaju da samostalno prikupljaju i razmenjuju podatke putem mreže, bez potrebe za interakcijom čovek-čovek ili čovek-kompjuter. Na primer, omogućava uređajima da samostalno razmenjuju podatke sa proizvođačem, operaterom i drugim povezanim uređajima.
SQL (engl. *Structured Query Language*) - je struktuirani upitni jezik (ANSI i ISO standard). Omogućava korisnicima pristup, definisanje i manipulaciju podacima u sistemima za upravljanje relacionim bazama podataka.
BLOB (engl. *Binary Large Objects*) - tip podatka koji predstavlja mogući element strukture za smeštaj podataka. BLOB su veliki promenljivi nizovi bajtova kojima se predstavlja slika, zvuk ili video zapis (multimedijalni objekti).
CAN (engl. *Controller Area Network*) – serijski komunikacijski protokol projektovan za primenu u automobilskoj industriji. Danas je dominantni protokol u mrežnim sistemima motornih vozila. Omogućava efikasan i pouzdan način komunikacije između senzora, aktuatora, kontrolera i čvorišta u realnom vremenu. Osim u motornim vozilima, široku primenu je našao i u industrijskim automatizovanim i drugim sistemima, sa primenom u različitim proizvodima kao što su mašine, medicinska oprema, pametne zgrade itd.
RDBMS (engl. *Relational Database Management System*) - sistem za upravljanje relacionim bazama podataka. Najpopularniji komercijalni i slobodni sistemi za upravljanje bazama podataka trenutno u upotrebi su bazirani upravo na ovom modelu. RDBMS je osnova za SQL i za sve moderne baze podataka kao što su: MS SQL Server, IBM DB2, Oracle, MySQL, Sybase i Microsoft Access.
DDL (engl. *Data Definition Language*) – računarski programski jezik za definisanje podataka koji omogućava formiranje strukture baze podataka. Omogućava kreiranje i brisanje tabela baze, definiše indekse (ključeve), definiše veze između tabela i ograničenja između tabela.
DML (engl. *Data Manipulation Language*) - računarski programski jezik za manipulaciju sa podacima u bazi podataka. Omogućava dodavanje (umetanje), brisanje i menjanje podataka.
INGRES (engl. *INteractive Graphics and Retrieval System*) – RDBMS namenjen za podršku velikim komercijalnim i vladinim aplikacijama. Razvijen je, primarno, kao projekat na Kalifornijskom univerzitetu u Berkliju. Izvorno je slobodni (*open source*) RDBMS, a dodatno je nadograđen u komercijalnu varijantu u kompaniji RTI, Kalifornija.
QUEL (engl. *QUEry Language*) - primarni (i originalni) upitni jezik sistema INGRES. Komercijalni INGRES (razvijen u kompaniji RTI, Kalifornija) podržava i relacioni upitni jezik SQL. Mada daleko manje prisutan od SQL-a, ovaj jezik predstavlja, po mišljenju većine stručnjaka, superiorniji upitni jezik od SQL-a po nizu svojstava.
SYSTEM R - baza podataka razvijena kao istraživački projekat IBM-ove laboratorije iz San Hozea u Kaliforniji. System R je bio prototipski sitem relacione baza podataka. On je prvi implementirao SQL, koji je kasnije postao standardni upitni jezik relacionih baza podataka. Metoda optimizacije sistema SYSTEM R sa manjim izmenama primenjena je i u svim komercijalnim proizvodima razvijenim iz njega, kao što su DB2 i SQL/DS.
DB2 – sistem za upravljanje relacionim bazama podataka (RDBMS) koji je IBM predstavio 1983. godine. Ime treba da asocira na prelazak sa tada prevladavajućeg hijerarhijskog modela baze podataka u novi relacioni model. Inicijalno dizajniran da radi

isključivo na IBM mainframe platformama, na operativnom sistemu MVS, kasnije je prebačen na druge široko korišćene operativne sisteme kao što su UNIX, Windows i Linux.

ORACLE - sistem za upravljanje relacionim bazama podataka (RDBMS), proizvod kompanije Oracle Corporation. Sistem karakteriše rad u dvoslojnoj, troslojnoj i višeslojnoj klijent/server arhitekturi. Omogućava rad sa distribuiranim bazama podataka - pristup podacima sa bilo kog mesta, bez obzira gde i na kojoj platformi se nalaze. Uključuje upotrebu jezika SQL kao struktuiranog upitnog jezika.

SQL/DS (engl. *Structured Query Language/Data System*) – prvi komercijalni RDBMS kompanije IBM, predstavljen 1981. godine. Razvijen je za operativne sisteme DOS/VSE i VM/CMS. Dve godine kasnije IBM je predstavio DB2, koji je postao vodeći IBM-ov RDBMS, a SQL/DS koegzistira i danas, pod nazivom „DB2 for VM&VSE“.

XML (engl. *eXtensible Markup Language*) - definiše opštu sintaksu za označavanje podataka pomoću odgovarajućih etiketa koje imaju poznato ili lako razumljivo značenje. Format koji obezbeđuje XML za računarske elemente može se prilagoditi najrazličitijim oblastima, kao što su elektronska razmena podataka, čuvanje podataka, vektorska grafika, sistemi govorne pošte, izrada novih specijalizovanih jezika za označavanje... Osnovna svrha mu je da olakša deljenje podataka kroz različite informacione sisteme, posebno kroz one sisteme koji su povezani sa Internetom.

Xquery (XML Query) – XML upitni jezik. Omogućava pronalaženje i izvoz (ekstrakovanje) elemenata i atribua iz XML dokumenata.

Xpath (XML Path Language) **-** jezik za identifikovanje delova XML dokumenta. Sintaksno opisuje logičku strukturu XML dokumenta.

QBE (engl. *Query By Example*) – jednostavni, grafički upitni jezik za relacione baze podataka.Upiti su sređeni u obliku tabele i izraženi kao primeri. Umesto procedure za dobijanje željenih odgovora, korisnik daje primer onoga što zahteva. Korisnik je u mogućnosti da specificira polja koja će biti prikazana, međusobne veze i kriterijume za pretraživanje nad obrascima koji se prikazuju na ekranu. Ovi obrasci su direktni slikoviti prikazi tabele koja definiše bazu podataka.

1. Relacioni model

Sistemi baza podataka su između ostalog odgovorni za organizovano smeštanje velike količine podataka, pa su prisutni u svakom realnom poslovnom i naučno-tehničkom okruženju. Od velikih baza podataka za praćenje kao što su sistemi za rezervaciju avionskih karata pa do dečijih kolekcija bejzbol sličica, sistemi baza podataka održavaju i distribuiraju podatke od kojih zavisimo. Do skoro su se veliki sistemi baza podataka mogli implementirati samo na velikim računarskim sistemima. Ove mašine su tradicionalno bile skupe za projektovanje, kupovinu i održavanje. Međutim, današnja generacija snažnih, jeftinih računara tipa radnih stanica omogućava programeru izradu softvera koji brzo i jeftino održava i distribuira podatke.

Razvoj relacione organizacije podataka je započeo *E. F. Codd* koji je predložio skup od 13 pravila za identifikaciju relacija između pojedinih skupova podataka. *Codd*-ova pravila su postala osnova za razvoj sistema za upravljanje podacima. Današnji relacioni sistemi za upravljanje bazama podataka (RDBMS) su rezultat *Codd*-ove vizije. Podaci u RDBMS-u su organizovani kao redovi u tabelama. Struktuirani jezik se koristi za upite (pretragu), smeštanje i izmenu podataka. Struktuirani jezik upita SQL (*Structured Query Language*) je standardni relacioni jezik upita (ANSI standard). Njegov tvorac je *Chamberlin*, a nastao je u IBM-ovoj istraživačkoj laboratoriji (*IBM Research Laboratory*) u *San Jose*-u, Kalifornija 1974. godine, dakle na istom mestu gde je *E.F. Codd* 1970 definisao osnovne koncepte relacionog modela podataka. SQL je razvijen prema konceptima relacionog modela baze podataka. *Codd* je definisao 13 pravila, koja se neobično zovu **13 *Codd*-ovih** pravila za relacioni model:

1. **Pravilo zasnovanosti** - upravljanje bazom podataka u RDBMS-u mora biti u potpunosti zasnovano na relacionim mogućnostima.

2. **Pravilo informacija** – sve informacije u relacionoj bazi podataka (uključujući i samu strukturu baze) su eksplicitno predstavljene vrednostima u tabelama.

3. **Garantovan pristup** – garantuje se pristup svakoj vrednosti u relacionoj bazi podataka korišćenjem kombinacije imena tabele, vrednosti primarnog ključa i imena kolone.

4. **Sistematska podrška NULL vrednosti** – RDBMS pruža sistematsku podršku NULL vrednosti (nepoznat ili neprimenljiv podatak) koja se razlikuje od regularnih vrednosti i nezavisna je od tipa podataka.

5. **Aktivni, uvek dostupan relacioni katalog** – podaci o strukturi same baze podataka (katalog) su organizovani na isti način kao obični podaci, pristupa im se jezikom upita na isti način kao i oičnim podacima.

6. **Sveobuhvatan podjezik podataka** – sistem mora da podržava barem jedan relacioni jezik koji dalje, između ostalog, podržava: definisanje podataka, njihovu manipulaciju, pravila integriteta, ovlašćenja i transakcije.
7. **Pravilo ažuriranja pogleda** – svi pogledi koji se teorijski mogu ažurirati, ažuriraju se kroz sistem.

9. Praktični primer integrisanog SQL-a u stored proceduru MS SQL Servera

Kao što je poznato, distribucija podataka putem interneta je ograničena propusnom moći same mreže. Bez obzira na činjenicu da nove tehnologije donose sve veće brzine protoka informacija, uzevši u obzir da veliki nizovi podataka jesu BLOB, to je od vitalnog značaja da povratni setovi podataka budu paketne forme.

Uzevši u obzir i intencije prema univerzalnom multiplatforskom korišćenju web browsera (orijentacija ka web aplikacijama) postaje jasno da neprihvatljivo dugački nizovi podataka (tabela na primer) nisu praktični za prikazivanje na jednom ekranu browsera. Prethodno se odnosi i na sam protok mreže. Ovo praktično znači da je prikazivanje podataka diktirano uslovima paginacije koja se i praktično primenjuje u savremenom webu.

Rešenje koje je ovde prikazano se naslanja isključivo na SQL serversku platformu. Osnovna ideja je da se uslovi paginacije diktiraju sa klijentske strane, dok serverska strana standardnom SELECT naredbom uz korišćenje lokalnih memorisjkih prostora generiše izabrani paket podataka diktiran sa klijentske strane. Prethodno se postiže upotrebom dve stored procedure:

Prva stored procedura generiše paket podataka na osnovu diktiranih uslova:

```
USE [IoT]
GO

/****** Object:  StoredProcedure [dbo].[Pagination]    Script
Date: YY/YY/YYYY YY:YY:YY ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO
-- =============================================
-- Author:        IoT
-- Create date:   Today
-- Description:   -
-- =============================================
CREATE PROCEDURE [dbo].[Pagination]
      -- Add the parameters for the stored procedure here
      @Page int,
      @RecsPerPage int,
      @Col int,
      @AscDesc nvarchar(10),
      @Question nvarchar(50)
AS

DECLARE

@FirstRec int,
```

```
@LastRec int,

@Kolona nvarchar(100),
@cmd nvarchar(4000),
@REZ datetime

BEGIN
	-- SET NOCOUNT ON added to prevent extra result sets from
	-- interfering with SELECT statements.
	SET NOCOUNT ON;


SELECT @FirstRec = (@Page - 1) * @RecsPerPage
SELECT @LastRec = (@Page * @RecsPerPage + 1)


CREATE TABLE #TempItems
(
employee nvarchar(50))

CREATE TABLE #TempItems1
(RowNum int IDENTITY PRIMARY KEY,
employee nvarchar(50))

INSERT INTO #TempItems (employee)


SELECT     TOP (100) PERCENT dbo.employee.string
FROM         dbo.employee
WHERE

	(string Like '%' + @question + '%')

SET   @cmd   =   'Select   *   from   #TempItems   ORDER   BY   '   +
CONVERT(varchar(20),@col)+ ' ' +@AscDesc

INSERT INTO #TempItems1 EXEC(@cmd)

SET @cmd = '      SELECT TOP   ('+CONVERT(varchar(20),@LastRec -
1)+') employee
	FROM #TempItems1
	WHERE RowNum >'+ CONVERT(varchar(20),@FirstRec) +'
	AND RowNum <'+ CONVERT(varchar(20),@LastRec)


EXEC(@cmd)

---- Drop the temp table
    DROP TABLE #TempItems
    DROP TABLE #TempItems1
END
GO
```

Iz prethodnog koda se vidi da su početni uslovi diktirani stranicom koju je neophodno prikazati, brojem vrsta po stranici, brojem kolone po kojoj će se vršiti sortiranje (ASC ili DESC), kao i upitom na osnovu čega će se izvšiti kompletno pretraživanje baze.

Na osnovu početnih uslova vrši se generisanje prve privremene tabele sa rezultatima pretraživanja osnovnog upita (*question*). Druga privremena tabela sadrži rezultate dodatnog filtera sa uslovima koji opisuju samu paginaciju (broj stranice, broj vrsta, kolona po kojoj se vrši sortiranje). Konačno rezultati se vraćaju klijentu.

Kako je iz prve stored procedure ostalo nejasno sa kolikim uzorkom raspolaže klijent, za potrebe kompletiranja logike paginacije druga stored procedura vraća ukupan broj vrsta koje je neophodno prikazati, odnosno koliko je velik uzorak CLOB:

```
USE [IoT]
GO

/******  Object:    StoredProcedure  [dbo].[  PaginationCounter]
Script Date: YY/YY/YYYY YY:YY:YY ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO




-- =============================================
-- Author:        IoT
-- Create date:   Today
-- Description:   -
-- =============================================
CREATE PROCEDURE [dbo].[ PaginationCounter]
      -- Add the parameters for the stored procedure here

      @Page int,
      @RecsPerPage int,
      @Col int,
      @AscDesc nvarchar(10),
      @question nvarchar(50)



AS

DECLARE


@FirstRec int,
@LastRec int,
```

```
@Kolona nvarchar(100),
@cmd nvarchar(4000),
@REZ datetime



BEGIN
      -- SET NOCOUNT ON added to prevent extra result sets from
      -- interfering with SELECT statements.
      SET NOCOUNT ON;
SELECT @FirstRec = (@Page - 1) * @RecsPerPage
SELECT @LastRec = (@Page * @RecsPerPage + 1)

SELECT      TOP (100) PERCENT COUNT(dbo.employee.employeeId) as
Countt
FROM         dbo.employee
WHERE
      (string Like '%' + @question + '%')
END

GO
```

Praktično, klijent dobija dva seta informacija:
- veličinu uzorka sa kojim raspolaže i
- set podataka za predviđenu veličinu uzorka.

Ovo praktično znači da server vraća samo ograničen set podataka, čime je problematika nekontrolisanog protoka velike količine informacija prevaziđena, što rezultuje rasterećenjem resursa mreže, a to je upravo i vrhunski inženjerski cilj u oblasti baza podataka.