Baze podataka

Količina podataka se duplira svakih 9 meseci, pa se količina prostora na diskovima (i broj prodatih diskova) duplira u istom periodu.

Modifikacija Murovog zakona: Vreme potrebno za obradu podataka se duplira svakih 18 meseci.

Modifikacija Parkinsonovog zakona: Podaci teže da ispune sav slobodan prostor u memoriji.

Pošto se mogućnosti korisnika ne povećavaju istom brzinom potrebne su efikasnije tehnike za čuvanje i obradu podataka.

SBP (Sistem baza podataka) je u osnovi sistem za računarsko zapisivanje i čuvanje slogova, tj. sistem čija je svrha da čuva informacije i dozvoli korisniku da te informacije dobije i ažurira po želji.

Glavne komponente SBP

- Podaci
  - Integrisani
  - Deljivi
- Hadver
  - Spoljašnji memorijski uređaj
  - Procesor i glavna memorija
- Softver
  - SUPB (Sistem za Upravljanje Bazama Podataka) - nivo softvera koji se nalazi između korisnika i fizičkih podataka u bazi, štiti korisnika od detalja na hardverskom nivou i upravlja svim zahtevima za direktan pristup bazi.
  - Alati za razvoj aplikacija, pisanje izveštaja, pomoćni(utility) programi, program za upravljanje transakcijama (TP monitor).
- Korisnici
  - Aplikativni programeri
  - Krajnji korisnici
  - Administratori
    - Administrator baze podataka (eng. database administrator, DBA)
      - profesionalac u IT
      - formira bazu i implementira kontrolne strukture
      - odgovoran za implementaciju odluka DA
      - odgovoran za rad sistema, performanse, ...
    - Administrator podataka (eng. data administrator, DA)
      - razume postojeće podatke
      - odlučuje koji podaci će biti čuvani u bazi
      - ustanovljava pravila za održavanje i rad sa podacima po njihovom čuvanju u bazi
      - nije tehničko lice, već pripada upravljačkim strukturama

Baza podataka je skup postojanih podataka koji se koriste od strane aplikativnih sistema u nekom okruženju. Postojani podaci, kada se jednom nađu u bazi, ne mogu da budu uklonjeni iz baze bez eksplicitnog zahteva SUBP.

Entitet možemo definisati kao objekat o kome želimo da skupljamo informacije (osoba, mesto, stvar,...). *Odnos* je veza između dva ili više entiteta.

Model podataka je apstraktna, samostalna, definicija objekata, operatora, ... koji zajedno čine apstraktnu mašinu sa kojom korisnik komunicira.

- *Objekti* dopuštaju modeliranje strukture podataka.
- *Operatori* dopuštaju modeliranje ponašanja.

Implementacija datog modela je fizička realizacija na stvarnoj mašini komponenata apstraktne mašine koje zajedno čine model.

Zašto koristiti baze podataka?

- Kompaktnost – troši se manje prostora
- Brzina – brže je raditi sa bazom nego sa pojedinačnim datotekama
- Manji napor – svi podacima se isto pristupa
- Aktuelnost podataka – kod baze menjam podatke na jednom mestu, a kod pojedinačnih datoteka moram taj isti podatak da menjam svuda gde se pojavljuje
- Centralizovana kontrola(u višestrukom okruženju)

Prednosti rada sa bazom

- Podaci mogu biti deljeni – ne treba ponavljati iste podatke na više mesta
- Smanjenje redudantnosti podataka
- Izbegavanje nekonzistentosti – npr. postoji podatak da je neko položio ispit, a nije upisan na fakultet
- Podrška za transakcioni rad – karte za avion, transakcije u bankama...
- Održavanje integriteta
- Primena zaštite podataka – koji korisnik sme da pristupa kojim podacima
- Balansiranje konfliktnih zahteva
- Primena standarda

Aplikacije implementirane na starim sistemima su bile zavisne od podataka. Nije pogodno da aplikacije budu zavisne od podataka iz baza

- Različite aplikacije zahtevaju različite poglede nad istim podacima.
- DBA mora da ima slobodu da prmeni fizičku reprezentacijuili pristupne tehnike radi performansi.

Nezavisnost podataka je otpornost aplikacije na promene fizičke reprezentacije podatka i pristupnih tehnika

Baza treba da bude otporna na promene kao što su dodavanje podataka, menjanje načina zapisa podataka.

Pojmovi

- sačuvano polje (eng. stored field) je najmanja jedinica podataka koja može da se čuva
- sačuvani slog (eng. stored record) je skup sačuvanih polja
- sačuvana datoteka (eng. stored file) je skup svih trenutno postojećih pojava sačuvanih slogova istog tipa

Aspekti sačuvanih reprezentacija koji mogu da budu predmet promena od strane DBA

- reprezentacija brojčanih podataka
- reprezentacija znakovnih podataka
- jedinice za brojčane podatke
- kodiranje podataka

Baza treba da bude sposobna da se širi bez promene postojećih aplikacija

- materijalizacija podataka
- struktura sačuvanih slogova
- struktura sačuvanih datoteka

Baza treba da bude sposobna da se širi bez negativnog uticaja na postojeće aplikacije.

Istorijat

Kasne 60-te

- IBM
- Information Management Systems (IMS)
- Hijerarhijski model podatka

70-te

- Edgar Codd, IBM (Tjuringova nagrada 1981)
- Relacioni model podataka

80-te

- Dominacija relacionog modela
- SQL
- Upravljanje transakcijama (James Gray, Tjuringova nagrada 1999)

Danas

- Objektno-orijentisani model podataka
- Objektno-relacioni model podataka
- Skladištenje podataka (eng. Data warehousing) i istraživanje podataka (eng. data mining)
- Pristup bazama podataka preko veba/Interneta
- Multimedijalni podaci
- Tekstualni podaci (eng. information retrieval)
- Struktura podataka (XML)

ANSI/SPARC arhitektura

ANSI – American National Standards Institute

SPARC – System Planning and Requirements Committee

Ovu arhitekturu čine tri nivoa:

- Spoljašnji nivo (eksterni nivo, individualni korisnički izgled)
- Konceptualni nivo (zajednički logički izgled)
- Unutrašnji nivo (interni nivo, fizički izgled)

Spoljašnji nivo

Svaki korisnik za izražavanje zahteva ima na raspolaganju matični (eng. host) jezik u koji se ugrađuje jezik podataka (eng. data sublanguage, DSL)

- matični jezici: Java, C, PL/I, COBOL, ...
- DSL: SQL, DB2, QUEL, DL/I, ...

Ako matični jezik ne može jasno da se odvoji od jezika podataka tada se za njih kaze da su **čvrsto vezani**. Ako mogu jasno i lako da se razdvoje tada se za njih kaže da su **labavo vezani**.

Jezik podataka je kombinacija najmanje dva podjezika.

1. Jezika za definiciju podataka (eng. Data Definition Language, DDL) koji se koristi za definisanje ili deklarisanje objekata u bazi.
2. Jezika za rad sa podacima (eng. Data Manipulation Language, DML) koji se koristi pri radu i obradi objekata iz baze
3. DCL ?!

Pojedinačni korisnik

1. Pojedinačnog korisnika interesuje samo jedan deo ukupne baze (spoljašnji izgled)
2. Korisnik vidi spoljašnje slogove (ne odgovaraju nužno sačuvanim slogovima)
3. Spoljašnja shema sadrži definicije svakog od različitih tipova slogova u spoljašnjem izgledu.

Konceptualni nivo

- Predstavlja informacioni kontekst celokupne baze podataka
- Podaci su nezavisni od jezika i hardvera
- Konceptualni izgled je definisan konceptualnom shemom
- Konceptualna shema uključuje definicije svakog od tipova konceptualnih slogova
- Zapisuje se pomoću konceptualnog DDL-a

Unutrašnji nivo

Reprezentacija baze podataka na niskom nivou

- Sastoji se od pojava različitih tipova unutrašnjih slogova (ANSI/SPARC termin za sačuvani slog)
- Njihove karakteristike su definisane unutrašnjom shemom i zapisane pomoću unutrašnjeg DDL-a.

Još uvek je iznad fizičkog nivoa (ne radi sa adresama, blokovima podataka ili stranicama u memoriji)!

Termini

- sačuvana baza podataka = unutrašnji izgled
- definicija sačuvanih struktura = unutrašnja shema

Neki (aplikativni) programi mogu da rade nad unutrašnjim izgledom baze. To se ne preporucuje zbog sigurnosti i integriteta. Ovaj način obično koriste *utility* programi.

Preslikavanje nivoa

- Preslikavanje je opis povezanosti dva nivoa.
- Jedno konceptualno/unutrašnje preslikavanje
  - Kako su konceptualni slogovi i polja predstavljeni na unutrašnjem nivou.
  - Ključno za nezavisnost podataka od promene fizičke strukture.
- Više spoljašnje/konceptualnih preslikavanja
  - U nekim sistemima je moguće definisati jedan spoljašnji pogled prek ostalih (**spoljašnje/spoljašnje preslikavanje**). Čest slučaj u relacionim sistemima.
  - Spoljašnje/konceptualno preslikavanje je ključno za nezavisnost podataka od promene logičke strukture.

Poslovi DBA

- Definisanje konceptualne sheme (logičko projektovanje baze)
- Definisanje unutrašnje sheme (fizičko projektovanje baze)
- Komunikacija sa korisnicima
  - da li su im obezbedjeni svi željeni podaci
  - konsultacija pri projektovanju aplikacija
  - pomoć pri rešavanju problema, ...

Sistem za upravljanje bazom podataka

SUPB (Sistem za Upravljanje Bazama Podataka) - nivo softvera koji se nalazi između korisnika i fizičkih podataka u bazi, štiti korisnika od detalja na hardverskom nivou i upravlja svim zahtevima za direktan pristup bazi.

Koraci u pristupu bazi

- Korisnik ispostavlja zahtev (npr. SQL upit).
- SUBP prihvata zahtev i analizira ga.
- Da bi odredio potrebne operacije SUBP proverava spoljašnju shemu korisnika, odgovarajuće spoljašnje/konceptualno preslikavanje, konceptualnu shemu, konceptualno/unutrašnje preslikavanje i definicije memorijskih struktura.
- SUBP izvrsava potrebne operacije (tj. zahtev korisnika) nad bazom.

Funkcije SUBP-a

- Definisanje podataka (preko DDL procesora)
- Obrada podataka (preko DML procesora)
  - planska (zahtev poznat unapred)
  - neplanska (zahtev nepoznat unapred, ad-hoc zahtev)
- Optimizacija izvršavanja upita
- Obezbeđenje zaštite i integriteta podataka Obezbedjivanje konkurentnog pristupa podacima i oporavka
- Formiranje rečnika podataka (repozitorijuma podataka, kataloga)
  - Sadrži informacije o definiciji SVIH objekata (shema, preslikavanja, ograničenja, zaštite, ...)
  - rečnik sadrži metapodatke (podatke o podacima)
- Obezbedjivanje što efikasnijeg rada
- SUBP takodje predstavlja korisnički interfejs ka sistemu baza podataka

Upravljanje prenosom podataka

- Zahtevi korisnika se prenose od mesta nastanka do SUBP-a preko komunikacionih poruka; rezultati se takođe vraćaju istim putem.
- Prenosom poruka upravlja deo softvera koji se naziva komunikacioni upravljač podataka (eng. data communication manager, DC manager).
- DC manager nije deo SUBP-a; rade zajedno formirajući DB/DC sistem.

Klijent-server arhitektura

- Sistem baza podataka moze da se posmatra kao da ima dve komponente
  - server (u suštini SUBP)
  - klijent (ispostavlja zahteve serveru)
- Klijent-server arhitektura je omogu´cena i u distribuiranoj obradi

Utility programi

- Koriste se za različite administratorske poslove
  - spoljašnji - aplikacije specijalne namene
  - unutrašnji - deo servera
- Primeri
  - LOAD/UNLOAD/RELOAD
  - REORG
  - programi za statistiku, analizu, ...

Distribuirana obrada

- Isti posao se (delom) izvršava na različitim računarima koji su spojeni u mrežu.
- Skoro da se upotrebljava kao sinonim za klijent/server.
- Podaci iz jedne baze mogu da budu smešteni na dva ili više servera.
- Klijent može da pristupi do više servera.

Relaciona algebra

Algebra je formalni matematički sistem koji se sastoji od skupa objekata i operacija nad tim objektima. (Bulova algebra, algebra skupova, ...)

Za definiciju formalnog sistema je potrebno:

- prikazati sintaksu
- dati semantiku
- dati pravila izvođenja dokaza

Relaciona algebra je familija algebri sa dobro zasnovanom semantikom koja se koristi za modeliranje relacija (objekata) smeštenih u relacionoj bazi podataka i za definisanje upita nad njima.

U suštini predstavlja skup operatora čiji su operandi i rezultati relacije. Prvu verziju je dao Codd 1972. godine, kasnije je proširivana od strane raznih autora.

Operatori koje je Codd originalno predložio	Operatori koji su kasnije dodati	Minimanli skup operatora
Restrikcija(selekcija) Projekcija Proizvod Unija Presek Razliku (Prirodno) Spajanje Deljenje	Rename Semijoin Extend Summarize	Restrikcija Projekcija Proizvod Unija Razlika

Sintaksa

Relacioni izraz je izraz oblika $ROP\ arg_1\ arg_2\ ...\ arg_n$ gde su

- ROP – relacioni operator
- arg_i – relacije koje su argumenti relacionog operatora

Semantika

U opisu semantike se koristi

- da su relacije matematički zasnovane i da predstavljaju skupove torki
- u pitanju su operacije nad skupovima koje predstavljaju preslikavanje domena relacija u novi domen

Relaciono zatvorenje

- Osobina da su i argumenti i rezultat primene bilo kog relacionog operatora takđe relacije se naziva relaciono zatvorenje.
- Zatvorenje znači da mogu da se pišu ugneždeni relacioni izrazi, tj. relacioni izrazi čiji su operandi takođe relacioni izrazi
- Treba obezbediti da i novodobijene relacije imaju odgovarajuće zaglavlje (sa jedinstvenim nazivima atributa) i odogovarajuće telo, bez obzira da li su u pitanju osnovne ili izvedene relacije.

Relacioni operatori

Restrikcija (selekcija)

Traženje torki koje zadovoljavaju postavljeni uslov

- Neka relacija A ima bar atribute X i Y i neka je Θ operator (obično $'=', '<', itd.$) takav da je uslov $X\Theta Y$ dobro definisan i da se izračunava kao istinitosna vrednost (tačna ili netačna).
- Tada je Θ restrikcija relacije A na atribute X i Y relacija koja ima isto zaglavlje kao i A i telo koje sadrži sve torke t iz A za koje je vrednost uslova $X\Theta Y$ tačno.
- Primer: Prikazati sve torke iz tabele dosijea za koje je vrednost atributa prezime jednaka 'Petrović'
  dosije WHERE prezime = 'Petrović'

Projekcija

Izdvajanje željenih atributa

- Neka relacija A ima bar atribute $X, Y, \ldots, Z$. Tada se projekcija relacije A na $X, Y, \ldots, Z$ označava sa $AX, Y, \ldots, Z$ i predstavlja relaciju čije
  - zaglavlje je izvedeno iz A uklanjanjem svih atributa koji se ne nalaze u skupu $X, Y, \ldots, Z$
  - telo se sastoji od svih torki $X : x, Y; y, \ldots, Z; z$ pri čemu se svaka torka javlja u A sa X vrednošću x,Y vrednošću $y, \ldots, Z$ vrednošću z.
- Ako su u listi navedeni svi atributi relacije A tada je projekcija *identitet*.
- Primer: Prikazati imena studenata i nazive mesta u kojima su rođeni:
  dosije{ime, mesto_rođenja}

Prirodno spajanje

- Vraća relaciju koja se sastoji od svih mogućih torki koje su kombinacija dve torke, jedne iz svake od dve navedene relacije, takve da dve torke koje učestvuju u svakoj kombinaciji imaju zajedničku vrednost za zajednički atribut(e) u datim relacijama (i zajednička vrednost se javlja samo jednom, ne dvaput, u rezultujućoj torki).
- Semantika: Neka relacije A i B imaju sledeća zaglavlja
  $A : \{X_1, X_2, \ldots, X_m, Y_1, Y_2, \ldots, Y_n\}$
  $B : \{Y_1, Y_2, \ldots, Y_n, Z_1, Z_2, \ldots, Z_p\}$
  Tada je prirodno spajanje relacija A i B definisano sa
  $A\ JOIN\ B = \{\{X : x, Y : y, Z : z\} | \{X : x, Y : y\} \in A \wedge \{Y : y, Z : z\} \in B\}$
- Primer: Prirodno spajanje relacija dosije i ispit
  dosije JOIN ispit
- Postoji i Θ spajanje za torke čiji atributi zadovoljavaju uslov $X\Theta Y$. Ako je $\Theta = '='$ tada se ovo spajanje naziva jednakosno spajanje (ako se jedan od atributa X ili Y eliminiše dobija se prirodno spajanje)

Deljenje

- Uzima dve relacije, jednu binarnu i jednu unarnu i vraća relaciju koja se sastoji od svih vrednosti jednog atributa binarne relacije koja odgovara (u drugom atributu) svim vrednostima u unarnoj relaciji.
- Sintaksa: A DIVIDEBY B PER C
- Semantika: pretpostavimo da naredne tri relacije imaju sledeća zaglavlja:
  - $A : \{X_1, X_2, \ldots, X_m\}$
  - $B : \{Y_1, Y_2, \ldots, Y_n\}$

  - $C : \{X_1, X_2, \ldots, X_m, Y_1, Y_2, \ldots, Y_n\}$
- Tada je delenje A sa B po C definisano kao
  $DIV(A, B, C) = \{\{X : x\} \in A | \forall \{Y : y\} \in B : \{X : x, Y : y\} \in C\}$

Unija

- Vraća relaciju koja se sastoji od svih torki se pojavljuju u jednoj ili obe od navedenih relacija
- Sintaksa: A UNION B
- Semantika: $A \cup B = \{t | t \in A \vee t \in B\}$

Presek

- Vraća relaciju koja se sastoji od svih torki koje se pojavljuju u obe navedene relacije.
- Sintaksa: A INTERSECT B
- Semantika: $A \cap B = \{t | t \in A \wedge t \in B\}$

Razlika

- Vraća relaciju koja se sastoji od svih torki koje se pojavljuju u prvoj a ne u drugoj navedenoj relaciji.
- Sintaksa: A DIFFERENCE B
- Semantika: $A - B = \{t | t \in A \wedge t \in B\}$

Dekartov (Kartezijev) proizvod

- Vraća relaciju koja se sastoji od svih mogućih torki koje su kombinacija dve torke, jedne iz svake od dve navedene relacije.
- Sintaksa: A TIMES B
- Semantika: neka relacije A i B imaju sledeća zaglavlja
  $\{A_1, A_2, \ldots, A_m\}$
  $\{B_1, B_2, \ldots, B_n\}$
  Tada proizvod A TIMES B ima zaglavlje $\{A_1, A_2, \ldots, A_m, B_1, B_2, \ldots, B_n\}$
  i važi A TIMES B $= \{t \cup t' | t \in A \wedge t' \in B\}$

Svrha relacione algebre

- Pisanje relacionih izraza koji se koriste za
  - definisanje prostora za dohvatanje podataka
  - definisnje prostora za ažuriranje podataka
  - definisanje pravila integriteta
  - definisanje izvedenih relacija
  - definisanje pravila zaštite
  - ...
- Osnova za optimizaciju upita

Relaciona kompletnost

- Jezik je relaciono kompletan ako je moćan isto kao i algebra, tj. ako bilo koja relacija predstavljiva u algebri može da se predstavi i u jeziku.
- SQL je relaciono kompletan jer postoje SQL izrazi za svaki od 5 primitivnih operatora relacione algebre

Algebarski zakoni

- Zakon komutacije
  A UNION B = B UNION A
  A INERSECT B = B INTERSECT A
  A TIMES B = B TIMES A
  A JOIN B = B JOIN

Dodatni operatori

RENAME – promena naziva relacije
SEMIJOIN – Spajanje relacija A i B projektovano na atribute relacije A
EXTEND – proširivanje relacije novim atributom
SUMMARIZE – omogućava sabiranje po kolonama

Relacioni račun

Relacioni račun

- Opisan, neproceduralni jezik
- Logički ekvivalent relacione algebre ako se posmatra deo relacionog modela podataka za obradu podataka
- Zasnovan na predikatskom računu
- Dve varijante:
  - Račun orijentisan ka torkama
  - Račun orijentisan ka domenima – osnova za QBE

Predikatski račun

- Predikat
  - je istinitosno vrednosna funkcija sa argumentima
  - kada se argumenti zamene vrednostima funkcija daje izraz koji se naziva predlog koji može da bude tačan ili netačan
- Opseg promenljivih
  - Promenljiva torki ima opseg iz skupa navedenih relacija i dopuštene vrednosti koje pripadaju torkama iz tih relacija
  - Promenljiva domena ima opseg iz skupa navedenih domena i dopuštene vrednosti koje pripadaju tim domenima
- Neka je x predikat. Tada se skup svih x takav da je P tačno za x označava sa $\{x|p(x)\}$
- Postoje dva kvantifikatora:
  - $\forall$: "za svaki"
  - $\exists$: "postoji"

Relacioni račun torki

- Slobodne i vezane promenljive
- Kvantifikatori
  - FORALL V (p)
  - EXISTS V (p)
  - Kvantifikacija i rad sa slobodnim i vezanim promenljivim su u skladu sa pravilima predikatskog računa
- Korišćenjem promenljivih torki traže se torke za koje je predikat tačan
- Sintaksa ovog računa se u literaturi prikazuje na različite načine

- RANGE OF <promenljiva> IS <tabele>
  RETRIEVE <promenljiva>.<imeatributa>
  [WHERE<uslovni izraz>]
  ili
  RANGEVAR <promenljiva> RANGES OVER <tabela>
  <promenljiva>.<imeatributa>
  [WHERE<uslovni izraz>]

Primer 1:

- prikazati imena i godine rođenja svih studenata koji su rođeni u Beogradu, i upisani na fakultet školske 2011-2012 godine, a imaju broj indeksa veći od 456
  RANGEVAR DOSIJEX RANGES OVER DOSIJE
  {DOSIJEX.IME, DOSIJEX.GOD_RODJENJA}
  WHERE DOSIJEX.MESTO_RODJENJA='Beograd'
  AND DOSIJEX.INDEKS¿20110456

Primer 2:

- Prikazati imena studenata koji su položili najmanje jedan predmet čija je šifra "P270"
  RANGEVAR DX RANGES OVER DOSIJE
  RANGEVAR PX RANGES OVER PREDMET
  RANGEVAR IX RANGES OVER ISPIT
  DX.SIME
  WHERE EXISTS IX (DX.INDEKS = IX.INDEKS AND EXISTS
  PX (PX.IS?PREDMETA = IX.ID_PREDMETA AND
  PX.SIFRA = "P270")))

Relacioni račun domena

- Opseg važenja promenljivih su domeni a ne relacije
- Moguće je definisati *uslov pripadnosti.*
- Oblik R (lista_parova)
  - R je naziv relvara
  - svaki par u listi je oblika A x gde je A naziv atributa u R, a x ili ime promenljive torki ili poziv selektora
- Uslov je tačan akko postoji torka u relaciji R takva da je za svaki konkretan par poredjenje $A = x$ tačno
  Na primer,
  ISPIT {INDEKS INDEKS(20110456),
  ID_PREDMETA ID_PREDMETA(1001)}
  ima vrednost tačno akko postoji torka u ispitu koja ima vrednost 20110456 za atribut indeks i 1001 za atribut id_predmeta

Algebra ili račun

- Algebra i račun su semantički ekvivalentni
- Kodov algoritam redukcije (prikaz da je algebra moćna bar koliko i račun i obratno)
- Neki upitni jezici su više zasnovani na algebri, a neki na računu
- SQL ima osobine i algebre i računa

Uvod u relacione baze podataka

Neformalni pogled na relacioni model

- Teorijska osnova relacioni model podataka. [Codd 1970g.]

- Intuitivno, relacioni model predstavlja jedan nacin gledanja na podatke. Sadrži pravila za predstavljanje podataka (preko tabela) i pravila za rad sa tim podacima (izdvajanje, spajanje, ...)

Aspekti relacionog modela

- Aspekt strukture: svi podaci u bazi se korisniku prikazuju isključivo u obliku tabela
- Aspekt integriteta: tabele zadovoljavaju izvesna ograničenja (primarni i spoljašnji ključevi, ...)
- Aspekt obrade: operatori koji su na raspolaganju korisnicima za obradu tabela su takvi da izvode tabele iz tabela
  - relacioni račun
  - relaciona algebra

Primer relacionih operatora

- Restrikcija (selekcija) izdvaja pojedinačne redove iz tabele
  DOSIJE WHERE God_rodjenja>1992
- Projekcija izdvaja pojedinačne slogove iz tabele
  DOSIJE over Indeks, Ime, Prezime
- Spajanje kombinuje dve tabele na osnovu zajedničkih vrednosti u zajedničkoj koloni (u primeru je prikazano prirodno spajanje)
  DOSIJE AND ISPIT OVER Indeks
  - Kod spajanja se u rezultujućoj tabeli zajedničke vrednosti javljaju samo jednom u redu.
  - Javljaju sve kombinacije uparenih vrednosti (npr. 20100021 vrednost za Indeks)
  - ne javljaju se redovi sa vrednostima koje se ne nalaze u obe tabele koje učestvuju u spajanju (npr. 20100027 vrednost za Indeks)

Karakteristike relacione baze

- Relacioni sistemi zahtevaju samo **da se baza prikaže** korisniku u obliku tabele (način smeštanja i čuvanja na medijumima nije specificiran)
- **Informacioni princip**: Celokupan informacioni kontekst baze se prikazuje na tačno jedan način kao eksplicitne vrednosti u pozicijama vrsta i kolona tabele
- Posledica Informacionog principa: nema pokazivača koji medjusobno povezuju tabele
  - pokazivači mogu da postoje na fizičkom nivou
  - razlika u odnosu na nerelacione modele gde su pokazivači deo struktura koje se prikazuju korisnicima i koje korisnik koristi u obradi

Efekat relacionih operatora

- Rezultat primene svakog od operatora je tabela - osobina zatvorenja relacionih sistema
- Rezultat primene operatora je istog tipa kao i njegov argument $\Rightarrow$ mogu da se pišu ugneždeni relacioni izrazi
- Sve operacije ce primenjuju na ceo skup istovremeno a ne samo na pojedinačni red
- Rezultat operacije nije nikada pojedinačni red već je uvek kompletna tabela koja sadrži skup redova

Formalni pogled na relacioni model

Relacioni model se sastoji od

- Otvorenog skupa **skalarnih tipova** (koji uključuje i tip logičkih vrednosti boolean)
- **Generatora relacionih tipova** i njihove odgovarajuće interpretacije
- Mogućnosti definisanja **relacionih promenljivih** za generisane relacione tipove
- Operacije **relacione dodele** kojom se dodeljuju relacione vrednosti definisanim relacionim promenljivim

- Otvorenog skupa opštih **relacionih operatora** ("relaciona algebra") za izvođenje relacionih vrednosti iz drugih relacionih vrednosti

Terminologija

Codd je pri formulisanju principa relacionih baza uveo novu terminologiju koja se razlikovala od neodređene terminologije tog vremena

Relacije i relacione promenljive

- U definiciji baze studenata, predmeta i ispita, DOSIJE i ISPIT su relacione promenljive, odnosno promenljive čije su vrednosti relacione vrednosti
- Npr. neka DOSIJE ima tekuću vrednost iz koje treba da se izbriše red za studenta sa brojem indeksa 20100027:
  DELETE DOSIJE WHERE Indeks = Indeks(20100027);
- Stara vrednost relacione promenljive DOSIJE je zamenjena novom relacionom vrednošću
- Operator brisanja je skra´cenica za relacioni operator dodele
  DOSIJE := DOSIJE WHERE NOT (Indeks = Indeks(20100027));
- Terminologija
  - relvar = relaciona promenljiva
  - relacija = specifična relaciona vrednost
- Skup otvorenih tipova znači da korisnik može da definiše svoje sopstvene tipove

Optimizacija

- Relacioni jezici su neproceduralni
- Za relacioni sistem često se kaze da vrši automatsku 'navigaciju' (nad sačuvanim podacima)
- U nerelacionim sistemima korisnik je odgovoran za 'navigaciju'
- Npr. za upit
  (DOSIJE WHERE Indeks = Indeks(20100027) {Ime}
  Rezultat se dobija
  - restrikcijom tekuće vrednost relvar-a DOSIJE na redove sa u kojima je broj indeksa jednak 20100027
  - projekcijom dobijenih vrednosti na kolonu Ime
- Postoje bar dva načina pristupa obrade ovog upita:
  - vršenjem fizičkog skeniranja (sačuvane vrednosti) relvar-a DOSIJE dok se ne nađe traženi podatak
  - ako postoji indeks nad (sačuvanom verzijom) kolone Indeks (a obično postoji jer je Indeks primarni ključ) tada se on koristi za direktan pristup traženim podacima
- Odluka kako vršiti automatsku navigaciju spada u domen komponente SUBP nazvane optimizator. U pštem slučaju optimizator određuje strategiju na osnovu razmatranja:
  - koji relvari-i su referencirani u zahtevu i koliko su oni veliki
  - koji indeksi postoje
  - kako su podaci fizički smešteni na disku
  - ...

Katalog

- Katalog ('rečnik') se i sam sastoji od relvar-a
  $\Rightarrow$ katalogu se pristupa na isti način kao i drugim relacionim promenljivim
- Informacije o samom katalogu se nalaze u relvar-ima iz kataloga

Osnovni relvar-i i pogledi

- Originalni relvar-i = **osnovni relvar-i**
- Njihove vrednosti = **osnovne relacije**

- **Izvedene relacije** = relacije koje nisu osnovne ali mogu da se dobiju iz osnovnih putem relacionih izraza
- Osnovni relvar-i moraju da imaju ime
- Relacione baze imaju mehanizam za formiranje osnovnih relvar-a
- U SQL-u su osnovni relvar-i tabele. Njihovo formiranje se vrši naredbom
  CREATE TABLE <ime_osnovnog_relvara> ...
- Relacioni sistemi podržavaju i imenovane relvar-e koji su izvedene relacije - poglede
- Vrednost pogleda je (trenutni) rezultat izvršavanja odredjenog relacionog izraza koji se navodi pri formiranju pogleda
  CREATE VIEW Studenti_iz_Kraljeva
  AS (DOSIJE WHERE Mesto_rodjenja='Kraljevo') {Indeks, Ime, Prezime}
- Nad pogledima mogu da se vrše operacije kao nad osnovnim tabelama
  (Studenti_iz_Kraljeva WHERE Indeks > 20100023){Ime, Prezime}
- Pri izvršavanju sistem modifikuje upit u
  ((DOSIJE WHERE Mesto_rodjenja='Kraljevo') {Indeks, Ime, Prezime})
  WHERE Indeks > 20100023){Ime, Prezime}
- Odnosno uprošćava u
  (DOSIJE WHERE Mesto_rodjenja='Kraljevo' AND Indeks > 20100023){Ime, Prezime}
- Osnovni relvar-i predstavljaju podatke koji su fizički smešteni u bazi podataka ('zaista postoje')
- Pogledi predstavljaju različit način gledanja na 'realne podatke' (ne postoje, virtualni relvar-i)

Transakcije

Transakcija je osnovna 'logička jedinica posla' koja obično uključuje više operacija nad bazom

Svojstva transakcije

- **Atomičnost**: garantuje se da će se izvršiti ili sve što se nalazi u transakciji ili ništa od toga
- **Trajnost**: garantuje se da će po uspešnom izvršavanju (COMMIT-a) sve promene ostati trajno zapamćene u bazi, bez obzira na kasnije eventualne padove sistema
- **Izolovanost**: transakcije su međusobno izolovane u smislu da su efekti izvršavanja jedne transakcije nevidljivi za drugu transakciju sve do (uspešnog) izvršavanja COMMIT naredbe
- Izvršavanje isprepletanog (u smislu početka i kraja) skupa transakcija će obično biti **serijalizovano** u smislu da se dobija isti rezultat kao da se te iste transakcije izvršavaju jedna po jedna u unapred neodredjenom redosledu izvršavanja

Domeni i relacije

Domeni

Domen = tip podataka

- ugradjen (sistemski predefinisan)
  - INTEGER
  - CHAR
  - ...
- korisnički definisan
  - INDEKS
  - IME
  - GOD_RODJENJA
  - ...
- Provera tipova. Stroga tipiziranost

  - Dosije.Indeks + Ispit.ocena → pogrešno!
  - Ispit.Godina_roka * Ispit.Ocena → ispravno!
- Definisanje novog tipa
  TYPE <ime tipa> <moguće reprezentacije>
- TYPE INDEKS REPREZENT(INTEGER);
- TYPE IME REPREZENT (CHAR);

Domeni - SQL podrška

- Naredbe
  - CREATE DOMAIN
  - ALTER DOMAIN
  - DROP DOMAIN (RESTRICT, CASCADE)
- U DB2 ove naredbe ne postoje. Postoje
  - CREATE TYPE
  - ALTER TYPE
  - DROP TYPE
- Istovremeno se definišu i odgovarajuće funkcije i operatori poredjenja
- Nema stroge tipiziranosti ali postoji osnovni oblik provere tipova
- SQL podržava osam relacionih domena
  1. Brojevi (numbers)
  2. Niske karaktera (character strings)
  3. Niske bitova (bit strings)
  4. Datumi (dates)
  5. Vremena (times)
  6. Kombinacija datuma i vremena (timestamps)
  7. Intervali godina/mesec (year/month intervals)
  8. Intervali dan/vreme (day/time intervals)

Relacije

Neka je dat skup od n tipova ili domena T_i $(i = 1, 2, ..., n)$, pri čemu ne moraju svi tipovi da budu medjusobno različiti. R je relacija nad tim tipovima ako se sastoji od dva dela, zaglavlja i tela gde važi:

- **Zaglavlje** je skup od n atributa oblika $A_i : T_i$ gde su A_i (koji svi moraju da budu različiti) imena atributa relacije R, a T_i odgovarajuća imena tipova $(i = 1, 2, ..., n)$
- **Telo** je skup od m **torki** t gde je t skup komponenti oblika $A_i : v_i$ u kojima je v_i vrednost tipa T_i
  m se naziva kardinalnost, a n stepen (arnost) relacije R

Osobine relacije

- Nema ponovljenih (duplih) torki
- Torke su neuredjene, od vrha ka dnu
- Atributi su neuredjeni, sa leva u desno
- Svaka torka sadrži tačno jednu vrednost za svaki atribut. Za relaciju koja zadovoljava ovu osobinu se kaže da je normalizovana, odnosno da je u prvoj normalnoj formi.

Relacija = tabela ?

Nisu jednake jer

- Tabela moze da sadrži duplirane redove dok relacija ne može da sadrži duplirane torke
- Redovi u tabeli su uredjeni u redosledu od vrha ka dnu, dok za relaciju to ne važi
- Kolone u tabeli su uredjene u redosledu sa leva u desno, dok za relaciju to ne važi

Relacije i prazan skup

- Relacija koja ima prazan skup torki
  - neprazno zaglavlje
  - telo je prazno
- Relacija koja ima praznu torku
  - prazno zaglavlje
  - telo sa jednom torkom bez komponenti

Tipovi relacije

- definicija tipa relacije ima sledeći oblik
  RELATION {<lista atributa razdvojenih zarezima>}
- atribut je uređen par oblika
  <ime atributa> <ime tipa>
- Primer:
  RELATION {Indeks INDEKS, Ime IME, Prezime PREZIME, God_rodjenja GOD_RODJENJA, Mesto_rodjenja MESTO_RODJENJA}

Atributi i tipovi podataka

- Svaki atribut ima neki tip
- Svaki tip može da bude ugradjen ili korisnički definisan
- Svaki tip može da bude atomski (skalarni) ili učauren (nije skalaran)
- Atributi relacije mogu da budu proizvoljnog tipa

Nedostajuće vrednosti

U svakodnevnoj praksi se često javlja problem nedostatka podataka:

- "...datum rodjenja nepoznat..."
- "...iz izborne jedinice XX nedostaju podaci..."
- ...

Postoji potreba da se indikator o nedostatku vrednosti čuva u bazi i na odgovarajući način vrši obrada takvih podataka.
Najčešći pristup prihvaćen i u praksi je korišćenje "nedostajuće vrednosti" (NULL) odnosno trovalentne (3VL) logike. Codd jer predlozio korišćenje 4-valentne logike jer postoje dve vrste nedostajućih vrednosti:

- vrednost je nepoznata
- vrednost nije primenljiva, vrednost ne postoji, ...

3VL logika Tri vrednosti

- Tačno
- Netačno
- Nepoznato (kod DATE-a UNK od unknown)

Operatori: I, ILI, NE, MOŽDA (AND, OR, NOT, MAYBE)

AND	t	u	f		**OR**	t	u	f		**NOT**			**MAYBE**	
t	t	u	f		t	t	t	t		t	f		t	f
u	u	u	f		u	t	u	u		u	u		u	t
f	f	f	f		f	t	u	f		f	t		f	f

Operator MAYBE je potreban npr. zbog ovakvih upita:
Prikazati sve zaposlene koji su možda bili, ali za koje nije sasvim pouzdano da su bili, programeri rođeni pre 25. januara 1991. godine sa platom manjom od 50.000 na dan 30.09.2011.

Uvod u SQL

Razvoj SQL-a

- Čita se "S-KU-EL" ili "SEKUEL"
- Danas je standardni upitni jezik na RSUBP
- Prva verzija je implementirana na System R
- Standardi:
  - SQL/89
  - SQL/92 (SQL2)
  - SQL/99 (SQL3)
  - SQL:2003 (dodati OLAP, XML, ...)
  - SQL:2006 (dodato još XML-a)
  - SQL:2008

	System R	INGRES
Autor	IBM San Jose Res. Lab 1974 - 1979	UC Berkeley kasne 1970 - rane 1980
Računar	IBM System 370	DEC PDP
Operativni sistem	VM/CMS	UNIX
Upitni jezik	SQL	QUEL
Host jezik	COBOL, PL/1	COBOL, PASCAL, C, FORTRAN, BASIC
Komercijalni proizvod	DB2, SQL/DS	Komercijalni INGRES
Distribuirana OB	R*	Distribuirani INGRES
OO proširenja	Starburst	POSTGRES

DDL podskup SQL-a - podskup za definisanje objekata
Najvažnije naredbe:

1. CREATE
2. ALTER
3. DROP
4. DECLARE

DML podskup SQL-a - podskup za obradu upita
Najvažnije naredbe

1. SELECT
2. INSERT
3. UPDATE
4. DELETE
5. MERGE

SQL podrška za nedostajuće vrednosti

- SQL pri primeni WHERE klauzule na tabelu T eliminiše sve redove za koje izraz u WHERE ima vrednost netačno ili nedefinisano
- Test za null: IS [NOT] NULL
- Ne važi ekvivalencija: *r IS NOT NULL* i *NOT (r IS NULL)*
- Ostali načini obrade NULL biće dati kroz primere

Neki tipovi podataka u SQL/DB2

- niske karaktera
- numerički podaci
- datumsko-vremenski podaci
- konstante
- grafičke niske
- korisnički definisani tipovi podatka
- rowid vrednosti
- binarne niske
- LOB
- XML
- ...

Objekti u DB2

- Memorijske grupe
- Baze podataka
- Sheme
- Prostori za čuvanje tabela
- Tabele
- Pogledi
- Indeksi
- Korisnički definisani tipovi
- Korisnički definisane funkcije
- Pul bafera
- Aliasi i sinonimi
- Okidači (trigeri)
- Uskladištene procedure
- Planovi i paketi
- Serveri
- Sekvence
- ...

Specijalni registri u DB2

- current application encoding scheme
- current date
- current degree
- current locale lc_ctype
- current optimization hint
- current packageset
- current path
- current precision
- current rules
- current server
- current sqlid
- current time
- current timestamp
- current timezone
- user

Integritet u relacionim bazama podataka

Osnovni pojmovi

- Pojam integritet se u kontekstu baza podataka odnosi na preciznost, punovažnost i korektnost podataka u bazi
- Održavanje integriteta podataka je od najveće važnosti za RSUBP. Zbog toga se u sistemu definišu pravila (tzv. ograničenja integriteta) koja se primenjuju na podatke
- Intuitivno, ograničenje integriteta je logički izraz pridružen bazi za koga se zahteva da njegovo izračunavanje uvek daje vrednost tačno
- Ograničenja se proveravaju pri formiranju objekata u bazi ili menjanju njihovog sadržaja

Zlatno pravilo : Ni jednoj operaciji ažuriranja nije dozvoljeno da ostavi bilo koji relvar u stanju koje narušava bilo koje od ograničenja tog relvar-a.

- Verzija 1: Ni jednoj operaciji ažuriranja nije dozvoljeno da ostavi bilo koju bazu podataka u stanju u kome se neki od atributa baze izračunava kao netačno (posledica: pre bilo kakvog stvarnog ažuriranja proverava se važenje ograničenja)
- Primer ograničenja integriteta: Ocena dobijena na ispitu mora da bude u intervalu od 5 do 10.
  CONSTRAINT OCENA1 IS EMPTY (ISPIT WHERE OCENA¡5 OR OCENA¿10)

Klasifikacija ograničenja integriteta : Klasifikacija prema DATE-AIDB

- Ograničenja stanja: definišu prihvatljiva stanja u bazi

- Ograničenja prelaza: definišu prihvatljiva stanja prelaza u bazi

Ograničenja stanja

- Ograničenja baze: ograničenja koja se odnose na vrednosti koje je dozvoljeno čuvati u bazi (tj. koje se odnose na dve ili više različitih relacija)
- Ograničenja relacija (relvar-a): zadaje se ograničenje na vrednost pojedinačne relacije (relvar-a) koje se proverava pri ažuriranju te relacije
- Ograničenja atributa: ograničenja na skup dozvoljenih vrednosti datog atributa
- Ograničenja tipa: definicija skupova vrednosti koji čine dati tip

Ograničenja prelaza Primer: ako baza sadrži podatke o osobama tada su važeca sledeća ograničenja:

- Nije dozvoljeno venčanje već venčanih osoba
- Dozvoljeno je venčati se sa razvedeno osobom
- Osobe koje više nisu žive ne mogu da primaju platu (penziju, ...)
- ...

Klasifikacija ograničenja integriteta - drugi pogled : Klasifikacija prema tipu ograničenja koje mora da bude ispoštovano u bazi

- Referencijalni integritet
- Integritet domena
- Integritet redundatnosti
- Integritet (poslovnih) ograničenja

Ključevi

Kandidat za ključ

- Kandidat za ključ relacije R predstavlja podskup atributa X te relacije, ako važi:
  - Pravilo jedinstvenosti: ne postoje dve torke u relaciji R koje imaju iste vrednosti za X, i
  - Pravilo minimalnosti: ne postoji pravi podskup skupa X koji zadovoljava pravilo jedinstvenosti.
- Svaka relacija ima bar jednog kandidata za ključ (skup svih atributa ili neki njegov pravi podskup)

Vrste ključeva

- Primarni ključ - jedan od kandidata za ključ
- Alternativni ključevi - ostali kandidati
- Spoljašnji (strani) ključ - skup atributa jednog relvar-a R2 čije vrednosti treba da odgovaraju vrednostima nekog kandidata za ključ nekog relvar-a R1
- Superključ - nadskup kandidata za ključ; poseduje jedinstvenost ali ne i minimalnost

Primer ključeva

- Relvar DOSIJE - primarni ključ je INDEKS
- Relvar PREDMET - primarni ključ je ID_PREDMETA
- Relvar ISPITNI_ROK - primarni ključ je par atributa (GODINA_ROKA, OZNAKA_ROKA)
- Relvar ISPIT
  - primarni ključ je (INDEKS, ID_PREDMETA, GODINA_ROKA, OZNAKA_ROKA)
  - spoljašnji ključevi su
    - (GODINA_ROKA, OZNAKA_ROKA)
    - INDEKS
    - ID_PREDMETA

Referncijalni integritet

Osnovna ideja očuvanja integriteta u ovom slučaju je da sve vrednosti u tabelama treba da budu usaglašene
Primer: ako tabela ispit sadrži podatke o studentu za koga ne postoje informacije u tabeli dosije tada je došlo da narušavanja integriteta baze

Spoljašnji ključ predstavlja referencu na torku koji sadrži odgovarajući primarni ljuc. Odatle je problem osiguravanja da baza podataka ne sadrži pogrešne spoljašnje ključeve poznat kao problem **referencijalnog integriteta**, a ograničenja koja to omogućuju se nazivaju **referencijalna ograničenja**.

Relacija koja sadrži primarne ključeve se naziva *roditelj relacija*, a relacija koja sadrži spoljašnje ključeve koji se referišu na roditelj relaciju se naziva *dete relacija*.

Referencijalni integritet: Baza ne sme da sadrži neuparene vrednosti spoljašnjih ključeva

Relvar-i koji nemaju kandidate za ključ (tj. sadrže duple slogove) se ponašaju nepredvidivo u pojedinim situacijama (videti primer1a.sql iz 5.primeri.sql)
Sistem koji ne poseduje znanje o kandidatima za ključ ponekad pokazuje karakteristike koje nisu "čisto relacione".

Definicija spoljasnjih kljuceva: FOREIGN KEY lista atributa, REFERENCES ime relvar-a

- Pravilo brisanja
  - CASCADE
  - RESTRICT
  - NO ACTION
  - SET NULL
- Pravilo ažuriranja
  - RESTRICT
  - NO ACTION

Referencijalni ciklus

$$T_n \to T_{n-1} \to T_{n-2} \to \cdots \to T_1 \to T_n \tag{1}$$

Roditelj tabela i dete tabela ne moraju da budu različite tabele.

Ograničenja osnovnih tabela

- Definicija kandidata za ključeve
  - UNIQUE (lista naziva atributa)
  - PRIMARY KEY (lista naziva atributa)
  - NOT NULL
- Definicija spoljašnjih ključeva
  - FOREIGN KEY (lista naziva atributa)
  - REFERENCES osnovna tabela [(lista naziva atributa)]
  - [ON DELETE referencijalna akcija]
  - [ON UPDATE referencijalna akcija]
- Definicija ograničenja provere: CHECK (uslovni izraz)

Ograničenja u opštem smislu

- Tvrdnja (eng. assertion) je logička vrednost koja mora uvek da bude ispunjena
- Okidač (eng. trigger) je niz akcija koje su pridružene odredjenim dogadjajima, i koji se izvršavaju svaki put kada se takav dogadjaj dogodi

Implementacije RSUBP ne podržavaju tvrdnje ali podržavaju okidače.

Jezik za definisanje podataka (DDL)

- Memorijske grupe
- Baze podataka
- Tabele
- Sheme
- Prostori za cuvanje tabela
- Pogledi
- Indeksi
- Planovi i paketi
- Funkcije
- Pul bafera
- Aliasi i sinonimi
- Okidaci (trigeri)
- Katanci
- Uskladistene procedure
- Log datoteke
- ...

Naredbe za definisanje podataka

CREATE naredba

- Sintaksa: CREATE <objekat> ...
- Formira nove objekte.
- Referentna sintaksa: DB2 V10 SQL reference
- Ilustracija u primerima: primer1 - primer10
- Objekti na koje može da se primeni su:
  - Memorijske grupe
  - Baze podataka
  - Sheme
  - Prostori za čuvanje tabela
  - Tabele
  - Pogledi
  - Indeksi
  - Korisnički definisani tipovi
  - Korisnički definisane funkcije
  - Pul bafera
  - Aliasi i sinonimi
  - Okidači (trigeri)
  - Uskladištene procedure
  - Planovi i paketi
  - Serveri
  - Sekvence
  - ...

ALTER naredba

- Sintaksa: ALTER <objekat> ...
- Menja karakteristike postojećih objekata.
- Referentna sintaksa: DB2 V10 SQL reference
- Ilustracija u primerima: primer11 - primer12
- Objekti na koje može da se primeni su:
  - Memorijske grupe
  - Particije baze podataka
  - Prostori za čuvanje tabela
  - Tabele
  - Pogledi
  - Korisnički definisani tipovi
  - Korisnički definisane funkcije
  - Pul bafera
  - Nadimci
  - Serveri
  - Sekvence
  - Indeksi
  - ...

DROP naredba

- Sintaksa: DROP <objekat> ...
- Koristi se za fizičko brisanje objekata na tekućem serveru. Svi objekti koji direktno ili indirektno zavise od obrisanih objekata se takodje brišu

- Referentna sintaksa: DB2 V10 SQL reference
- Ilustracija u primerima: većina primera

DECLARE naredba

- Sintaksa: DECLARE <objekat> ....
- Slična je CREATE naredbi sem što se koristi za formiranje privremenih tabela koje se koriste jedino za vreme tekuće sesije.
- Jedini objekat koji može da se deklariše je tabela koja se upisuje u privremeni prostor za čuvanje tabela korisnika.
- Referentna sintaksa: DB2 V10 SQL reference
- Ilustracija u primerima: primer13

Naredbe za obradu podataka

SELECT naredba

- Funkcija: prikazuje rezultat SQL upita
- Sintaksa: videti sintaksu SQL upita
- Referentna sintaksa: DB2 V10 SQL reference
- Ilustracija u primerima:
  - primer1 - ...
  - primer14 - primer21

INSERT naredba

- Sintaksa:
  - INSERT INTO <objekat> VALUES...
  - INSERT INTO <objekat> WITH ...
  - INSERT INTO <objekat> <puna select naredba> ...
- Funkcija: unosi vrednosti u tabelu (pogled)
- Referentna sintaksa: DB2 V10 SQL reference
- Ilustracija u prethodnim primerima

UPDATE naredba

- Sintaksa: UPDATE <objekat> SET ...
- Funkcija: ažurira postojeće vrednosti
- Referentna sintaksa: DB2 V10 SQLreference
- Ilustracija u prethodnim primerima

DELETE naredba

- intaksa: DELETE <objekat> [WHERE uslov]
- Funkcija: prazni sadržaj objekata
- Referentna sintaksa: DB2 V10 SQL reference
- Ilustracija u prethodnim primerima

MERGE naredba

- Sintaksa: MERGE <objekat> USING ...
- Funkcija: ažurira ciljni objekat na osnovu podataka iz izvornih objekata
- Referentna sintaksa: DB2 V10 SQL reference
- Ilustracija: primer21

Pogledi

- Relacioni sistemi podržavaju poglede – imenovane relvar-e
- Vrednost pogleda je rezultat izvršavanja određenog relacionog izraza u tom trenutku
- Relacioni izraz se navodi pri formiranju pogleda
- Sistem pretvara upit naveden pri formiranju pogleda u ekvivalentan upit nad osnovnim relvar-ima
- Pretvaranje se vrši supstitucijom
- Susptitucija je moguća na osnovu osobine relacionog zatvorenja

Primeri

```
VAR UPISANI2010 VIEW
(DOSIJE WHERE INDEKS/10000=2010)
{ALL BUT MESTO_RODJENJA} RENAME
GOD_RODJENJA AS GODINA
```

Pogled upisani2010 ima atribute: Indeks, ime, prezime

Pogled koji sadrži parove naziva roka i predmeta u kojima je predmet polagan

```
VAR PAR_NAZIVA VIEW
((PREDMET RENAME NAZIV AS PNAZIV)
JOIN ISPIT JOIN
(ISPITNI_ROK RENAME NAZIV AS INAZIV))
{PNAZIV, INAZIV}
```

Pogled može da bude definisan i nad drugim pogledom a ne samo nad osnovnim relavarom.
Primer: definisati pogled koji sadrži studente upisane 2010 godine koji su rodjeni posle 1992. godine.

```
VAR UPISANI_2010_1992 VIEW
UPISANI2010 WHERE GODINA>1992
```

Formiranje i brisanje pogleda

Sintaksa naredbe za formiranje pogleda:

```
VAR <ime relvar-a> VIEW <relacioni izraz>
<lista kandidata za ključeve>
[RESTRICT/CASCADE]
```

<lista kandidata za ključeve> može biti i prazna ako pogled može da nasledi kandidate za ključeve RESTRICT/CASCADE može ali ne mora da postoji.

Sintaksa naredbe za brisanje pogleda:

```
DROP VAR <ime relvar-a>
```

Definicija pogleda kombinuje:

- spoljašnju šemu
- preslikavanje između spoljašnjeg i konceptualnog nivoa (sadrži izgled spoljašnjeg objekta i opis kako se on preslikava na konceptualni nivo)
- spoljašnje/spoljašnje preslikavanje (npr. pogled UPISANI_2010_1992)

Funkcije pogleda

- Obezbeđuju automatsku zaštitu za skrivene podatke
- Omogućuju da različiti korisnici (istovremeno) vide iste podatke na različite načine
- Uprošćavaju slošene operacije (slično makroima u programskim jezicima)
- Omogućuju logičku nezavisnost podataka

Logička nezavisnost podataka

- **širenje** relacija baze ne sme da ima efekta na izvršavanje aplikativnih programa
- Restruktuiranje baze ne sme da ima efekta na postojeće aplikativne programe. Nova i stara baza treba da budu **informaciono ekvivalentne**.
- Primer restrukturiranja baze: ako se relacija DOSIJE razbije na dve osnovne relacije
  - `VAR DOSIJE1 BASE RELATION {INDEKS INDEKS, IME IME, PREZIME PREZIME, GOD_RODJENJA INTEGER} PRIMARY KEY {INDEKS}`
  - `VAR DOSIJE2 BASE RELATION {INDEKS INDEKS, MESTO_RODJENJA CHAR} PRIMARY KEY {INDEKS}`
- S može da se formira kao pogled
  - `VAR DOSIJE VIEW DOSIJE1 JOIN DOSIJE2`
- Aplikativni programi koji su radili sa osnovnim relvar-om DOSIJE bi trebalo da mogu bez izmena da se referišu na pogled DOSIJE

Dohvatanje rezultata iz pogleda

- Operacija čitanja podataka nad pogledom se konvertuje u ekvivalentnu operaciju nad osnovnim relvar-ima
  - materijalizacija relacije koja je trenutna vrednost pogleda V
  - supstitucija relacionog izraza u drugom relacionom izrazu
- Semantika pogleda se definiše preko materijalizacije relacija!

Ograničenja integriteta kod pogleda

- Zlatno pravilo se primenjuje i na poglede, tj. ažuriranje pogleda ne sme da naruši ograničenja integriteta nad pogledima
- Ograničenja integriteta nad pogledima su izvedena iz ograničenja integriteta osnovnih relvar-a.

Problem ažuriranja pogleda

Za neko ažuriranje nekog pogleda kakve vrste ažuriranje treba izvesti nad osnovnim relacijama da bi se implementiralo originalno ažuriranje pogleda?
Preciznije, ako je D baza, V pogled nad D i X funkcija nad D kojom se definiše pogled V, tada za dati pogled $V = X(D)$ i operaciju ažuriranja U nad V, potrebno je odrediti operaciju ažuriranja U' nad D tako da važi $U(X(D)) = X(U'(D))$

Moguće je da postoji više operacija U'. Koju odabrati?

R_1	R_2	$V = (R1 JOIN R2)XZ$
X Y	Y Z	X Z
x_1 y_1	y_1 z_1	x_1 z_1
x_2 y_2	y_2 z_2	x_2 z_2

Kako implementirati naredbu brisanja torke (x_2, y_2) iz pogleda V?
Korektno je brisanjem odgovarajućih torki iz relacija R_1 i R_2.

- CODD–ov pristup: definisanje pogleda koji mogu da se ažuriraju
- Date–ov pristup: svi pogledi mogu da se ažuriraju. Operacije ažuriranja se izvode izbegavanjem ograničenja integriteta u međukoracima ažuriranja
  - ažuriranje se izvodi kao brisanje postojećih i unošenje novih podataka

SQL podrška

- CREATE VIEW naredba

```
CREATE VIEW <ime pogleda> AS <izraz nad tabelom> [WITH [<kvalifikator>] CHECK OPTION]
```

- DROP VIEW <ime pogleda>

Primeri definicije pogleda u sistemu DB2 su prikazani u 10.primer.sql.zip

Dohvatanje podataka

Po standardu bi sve operacije sa čitanjem podataka preko pogleda trebalo da rade korektno.
U praksi to nije slučaj, pogotovo ako se pogledi ne materijalizuju.
Npr. upit:

```
select avg(ispit.zbir) from ispit
```

ukoliko se ne materijalizuje se implementira kao

```
select avg(sum(ispit.ocena))
from ispit
group by id_predmeta
```

koji je nekorektan jer sadrži ugneždenu agregatnu funkciju.

Ažuriranje pogleda

SQL/92 podrška za ažuriranje pogleda je vrlo ograničena.
Jedini pogledi koji se smatraju mogućim za ažuriranje su pogledi koji su izvedeni iz jedne osnovne tabele preko kombinacija restrikcije i projekcije.
U SQL/92 pogled može da se ažurira ako važi:

- Izraz kojim se definiše pogled je select izraz koji ne sadrži JOIN, UNION, INTERSECT ili EXCEPT.
- Select klauzula ne sadrži ključnu reč DISTINCT
- Svaka select stavka sadrži (kvalifikovano) ime koje predstavlja referncu na kolonu osnovne tabele
- from kluzula sadrži referencu na tačno jednu tabelu koja je ili osnovna tabela ili pogled koji može da se ažurira
- where klauzula select izraza ne sadrži podupit u kome se from klauzula referiše na istu tabelu kao i from klauzula u select izrazu na najvišem nivou
- select izraz ne sadrži group by klauzulu
- select izraz ne sadrži having klauzulu

Pogledi u DB2 V10

Za diskusiju o pogledima koji mogu da se ažuriraju, u koje mogu da se unose podaci i iz kojih mogu da se brišu podaci u DB2 videti SQL Reference

Funkcionalne zavisnosti

Neka je R relacija i neka su X i Y proizvoljni podskupovi atributa iz R. Tada Y funkcionalno zavisi od X, u oznaci $X \rightarrow Y$ ako i samo ako je svakoj važećoj vrednosti torke X u R pridružena tačno jedna vrednost Y iz R.

Koristi se još i termin X funkcionalno određuje Y.
Funkcionalna zavisnost: $\exists f : f(X) = Y$

Intuitivno:
Funkcionalna zavisnost u relaciji R je tvrđenje oblika "Ako su dve torke od R identične na svim atributima $A_1, A_2, \ldots, A_n$ tada moraju da imaju iste vrednosti i u ostalim atributima $B_1, B_2, ..., B_m$"
Dve torke su identične ako su im identične vrednosti respektivnih komponenata.
Oznaka: $A_1, A_2, ..., A_n \rightarrow B_1, B_2, ..., B_m$

Primeri - funkcionalne zavisnosti u bazi STUD2011

Relacija DOSIJE

- {Indeks}→{Ime}
- {Indeks}→{Prezime}
- {Indeks}→{God_rodjenja}
- {Indeks}→{Mesto_rodjenja}

Relacija PREDMET

- {Id_predmeta}→{Sifra}
- {Id_predmeta}→{Naziv}
- {Id_predmeta}→{Bodovi}

Relacija ISPIT

- {Indeks, Id_predmeta, Godina_roka, Oznaka_roka}→{Ocena}
- {Indeks, Id_predmeta, Godina_roka, Oznaka_roka}→{Datum_ispita}

GODINA_ROKA	OZNAKA_ROKA	NAZIV
2011	jan	Januar 2011
2011	feb	Februar 2011
2011	apr	April 2011
2011	jun	Jun 2011
2011	sep	Septembar 2011
2011	okt	Oktobar 2011

{Godina_roka, Oznaka_roka}→{Naziv}

Ključevi relacije i funkcionalne zavisnosti

Redefinicija ključeva:
Kandidat za ključ relacije R predstavlja podskup atributa $X \subseteq R$, ako važi:

- $\forall Y : Y = R \backslash X \Rightarrow X \rightarrow Y$
- $\nexists Z, W : (Z \in Y) \wedge (W = R \backslash Z) \wedge (Z \rightarrow W)$

Nadključ (superključ) relacije R je skup atributa koji uključuje kao podskup bar jedan kandidat za ključ relacije R.

Napomene

- Svaka FZ predstavlja ograničenje integriteta
- Posledica: svaki atribut relacije funkcionalno zavisi od nekog kandidata za ključ
- FZ ne zavisi od trenutne vrednosti relvar-a i zbog toga nisu FZ {Oznaka_roka}→{Naziv} i {Naziv}→{Oznaka_roka}
- Dva skupa funkcionalnih zavisnosti S i T nad relacijom R su ekvivalentni ako skup instanci relacije R koji zadovoljava S je jednak skupu instanci relacije R koji zadovoljava T.
- U opštem slučaju, skup S FZ se izvodi iz skupa T FZ ako svaka instanca relacije koja zadovoljava sve FZ iz T takodje zadovoljava sve FZ iz S.
- Posledica: dva skupa FZ S i T su ekvivalentna akko se svaka FZ u S izvodi iz FZ u T i svaka FZ u T izvodi iz FZ u S.

Pravila i zatvorenje

Dekompozicija: FZ
$A_1, A_2, \ldots, A_n \to B_1, B_2, \ldots, B_m$
je ekvivalentna sa skupom FZ
$A_1, A_2, \ldots, A_n \to B_1$
$A_1, A_2, \ldots, A_n \to B_2$
...
$A_1, A_2, \ldots, A_n \to B_m$

Kompozicija: skup FZ
$A_1, A_2, \ldots, A_n \to B_1$
$A_1, A_2, \ldots, A_n \to B_2$
...
$A_1, A_2, \ldots, A_n \to B_m$
je ekvivalentna sa FZ $A_1, A_2, \ldots, A_n \to B_1, B_2, \ldots, B_m$

Trivijalne i netrivijalne FZ

- Trivijalna zavisnost je FZ koja ne može a da ne bude zadovoljena za bilo koji skup vrednosti u relaciji.
- $Y \subseteq X \Rightarrow FZ$ $X \to Y$ je trivijalna
- FZ koja nije trivijalna je netrivijalna

Neke trivijalne zavisnosti:

- {Indeks}→{Indeks}
- {Godina_roka, Oznaka_roka}→{Godina_roka, Oznaka_roka}
- {Godina_roka, Oznaka_roka}→{Godina_roka}
- {Godina_roka, Oznaka_roka}→{Oznaka_roka}
- {Id_predmeta}→{Id_predmeta}

Zatvorenje skupa FZ

Neka je S skup FZ nad relacijom R. Skup svih FZ koje mogu da se izvedu iz skupa S FZ se naziva zatvorenje od S i označava sa S^+.

Posledica: dva skupa funkcionalnih zavisnosti S i T nad relacijom R su ekvivalentni ako važi $S^+ = T^+$.

Određivanje zatvorenja skupa FZ

Zatvorenje S^+ skupa FZ S može da se odredi primenom pravila - Armstrongovim aksiomama kojima se nove FZ izvode iz postojećih.
Neka su A, B, i C proizvoljni podskupovi atributa relacije R. Tada važe pravila:

Refleksivnost:	$B \subseteq A$	$\Rightarrow$	$A \rightarrow B$
Proširenje:	$A \rightarrow B$	$\Rightarrow$	$AC \rightarrow BC$
Tranzitivnost:	$A \rightarrow B \wedge B \rightarrow C$	$\Rightarrow$	$A \rightarrow$

Iz prethodna tri mogu da se izvedu dodatna pravila:

- Samo-određenje: $A \rightarrow$
- Dekompozicija: $A \rightarrow BC \Rightarrow A \rightarrow B \wedge A \rightarrow C$
- Unija: $A \rightarrow B \wedge A \rightarrow C \Rightarrow A \rightarrow BC$
- Kompozicija: $A \rightarrow B \wedge C \rightarrow D \Rightarrow AC \rightarrow BD$
- Opšta teorema unifikacije: $A \rightarrow B \wedge C \rightarrow D \Rightarrow A \cup (C - B) \rightarrow BD$

Algoritam za računanje zatvorenja skupa FZ F:

```
Inicijalno F+=F;
repeat
za svaku FZ f ∈ F^+
primeniti refleksivnost i proširenje na f
dodati dobijene FZ u F^+
za svaki par FZ (f_1, f_2) ∈ F^+
ako f_1 i f_2 mogu da se kombinuju pomoću tranzitivnosti
tada dodati dobijenu FZ u F^+
until više ne bude promena u F^+
```

Određivanje zatvorenja skupa FZ - primer

Neka su A,B,C,D i F atributi relacije R, i neka je S skup FZ dat sa:
$A \rightarrow BC$
$B \rightarrow E$
$CD \rightarrow EF$
Ispitati da li je FZ $AD \rightarrow F$ u S^+
Primer: Dokaz da je $AD \rightarrow F$ u S^+

1. $A \rightarrow BC$ (dato)
2. $A \rightarrow C$ (dekompozicija)
3. $AD \rightarrow CD$ (proširenje)
4. $CD \rightarrow EF$ (dato)
5. $AD \rightarrow EF$ (tranzitivnost)
6. $AD \rightarrow F$ (dekompozicija)

Zatvorenje skupa atributa

- često treba izračunati da li data FZ pripada zatvorenju skupa FZ
- U praksi se određivanje zatvorenja FZ relativno retko radi (algoritam je neefikasan!)
- Da bi odredili da li je K nadključ treba odrediti skup atributa relacije R koji funkcionalno zavisi od K
- Ako je skup atributa K nadkljuc tada K funkcionalno određuje sve atribute u R
- K je nadključ akko je K^+ od K (u odnosu na dati skup FZ) jednako skupu svih atributa relacije R

Definicija:

Neka je $A = \{A_1, A_2, \ldots, A_n\}$ skup atributa relacije R i S skup FZ nad R. Zatvorenje skupa atributa A u odnosu na skup S FZ je skup atributa B takvih da svaka relacija koja zadovoljava sve FZ u skupu S zadovoljava i FZ $\{A_1, A_2, \ldots, A_n\} \to B$

Zatvorenje skupa atributa A se označava sa A^+. Uvek važi da su Svi pojedinačni atributi iz A u A^+

Određivanje zatvorenja skupa atributa

Neka je $A = \{A_1, A_2, \ldots, A_n\}$ skup atributa i S skup FZ. Algoritam za odredjivanje zatvorenja A^+ je

1. Izvršiti dekompoziciju svih FZ tako da imaju samo jedan atribut na desnoj strani
2. Neka je X skup atributa koji predstavlja zatvorenje. Inicijalno $X = \{A_1, A_2, \ldots, A_n\}$
3. Tražiti FZ oblika $\{B_1, B_2, \ldots, B_m\} \to C$ takve da $\forall i : B_i \subset X \wedge C \not\subseteq X$. Dodati C u X. Ponavljati pretragu sve dok ima promena skupa X.
4. Ukoliko ne postoji atribut koji bi mogao da se doda skupu X tada je $X = \{A_1, A_2, \ldots, A_n\}^+$

Pokrivač skupa FZ

- Ako su S_1 i S_2 dva skupa FZ i ako je svaka FZ iz S_1 uključena u S_2, tj. ako je S_1^+ podskup od S_2^+ tada je S_2 pokrivač za S_1.
- Ako je S_1 pokrivač od S_2 i S_2 pokrivač od S_1 tada su S_1 i S_2 ekvivalentni
- Ako RSUBP obezbedi FZ u S_2 tada će automatski biti obezbeđene i FZ u S_1

Nereducibilni skup FZ

Definicija:

FZ $X \to Y$ u skupu S funkcionalnih zavisnosti je levo-nereducibilna ako iz X ne može da se ukloni ni jedan atribut bez promene zatvorenja S^+

Zašto nereducibilni skup FZ: Nadključ K je kandidat za ključ relacije R akko je njen nereducibilni nadključ

- Desna strana svake FZ u S sadrži tačno jedan atribut
- Leva strana svake FZ je levo nereducibilna
- Ni jedna FZ ne može da se ukloni iz S bez promene S^+

skup FZ koji zadovoljava prethodna pravila se naziva i *minimalan* ili *kanonički*.
Algoritam za nalaženje nereducibilnog skupa S FZ:

- Koristeći pravilo dekompozicije prepisati sve FZ u S tako da desna strana sadrži tačno jedan atribut
- Eliminisati sve redundantne FZ iz skupa funkcionalnih zavisnosti dobijenih prethodnim postupkom

Nereducibilni skup FZ - primer

Neka relvar R ima atribute A,B,C,D i skup FZ:

- $A \to BC$
- $B \to C$
- $A \to B$
- $AB \to C$
- $AC \to D$

Naći nereducibilni skup FZ ekvivalentan datom skupu.

Prvi korak: prepisati skup FZ tako da je na desnoj strani samo jedan atribut

- $A \to B$
- $A \to C$
- $B \to C$
- $A \to B$
- $AB \to C$
- $AC \to D$

Kako se FZ $A \to B$ javlja dva puta, jedno pojavljivanje se eliminiše

Drugi korak: atribut C se eliminiše sa leve strane FZ $AC \to D$ jer važi

- $A \to C$ (dobijeno u prvom koraku)
- $A \to AC$ (proširenje)
- Kako važi i $AC \to D \Rightarrow A \to D$ (tranzitivnost)

Na osnovu prethodnog, dobija se da je C na levoj strani $AC \to DC$ redundantno

Treći korak: FZ $AB \to C$ se eliminiše jer važi

- $A \to C$ (dobijeno u prvom koraku)
- $AB \to CB$ (proširenje)
- $AB \to C$ (dekompozicija)

Na osnovu prethodnog, dobija se da je $AB \to C$ redundantno

Četvrti korak: FZ $A \to C$ se eliminiše jer važi

- FZ $A \to C$ (dobijeno u prvom koraku)
- $A \to B$ (dato)
- $B \to C$ (dato)
- $A \to C$ (tranzitivnost)

Na osnovu prethodnog, dobija se da je $A \to C$ redundantno
Dobijeni nereducibilni skup FZ je $A \to B$ $B \to C \to D$

Projekcija FZ

Neka se relacija R sa skupom S FZ projektuje na relaciju $R_1 = \pi\{R\}$. Koje FZ su važeće u R_1? Odredjuje se se projekcija FZ S koja sadrži sve FZ takve da

- Mogu da se izvedu iz S
- Uključuju jedino atribute iz R_1

Algoritam:

- Neka je T skup FZ u R_1. Inicijalno je $T = \emptyset \forall X \subseteq R_1$ odrediti X^+ u odnosu na FZ u S.
- Atributi koji su u R ali ne i u R_1 mogu da se koriste u izračunavanju X^+. Dodati u T sve netrivijalne FZ $X \to A$ takve da $A \subset X^+ \wedge A \subset R_1$

Odredjuje se minimalni skup T na sledeći način:

- Ako postoji $F \subset T$ koja može da se izvede iz drugih FZ u T, ukloniti FZ F
- Neka je $Y \to B$ FD u T sa najmanje dva atributa u Y i neka je Z dobijeno iz Y uklanjanjem jednog od atributa. Ako $Z \to B$ može da se izvede iz FZ u T (uključujući $Y \to B$), tada se $Y \to B$ zamenjuje sa $Z \to B$
- Ponoviti prethodne korake sve dok ima promena u T

Normalizacija

Uvod

Projektovanje baze podataka:

- Logičko projektovanje/fizičko projektovanje baze
- Logičko projektovanje baze
  - Normalizacija - korišćenje ideja o normalizaciji radi razbijanja "velikih" u "male" relacije
  - Semantičko modeliranje - upotreba modela entiteta i odnosa radi formiranja "velikih" relacija

Normalizacija je proces zamene relacija skupom relacija koje su u pogodnijem obliku. Svrha normalizacije je izbegavanje redundantnosti i pojedinih anomalija ažuriranja.

U procesu normalizacije operator projekcije se više puta primenjuje na datu relaciju na takav način da spajanjem projekcija može da se dođe do početne relacije. Na taj način, proces normalizacije je reverzibilan i čuva informacije, tj. uvek je moguće da se uzme izlaz iz procesa i preslika unatrag do ulaza.

Normalne forme

Anomalije

Primer: relacija PredmetIspit koja je dobijena "mešanjem" relacija Predmet i Ispit. Relacija sadrži atribute {Id_predmeta, Sifra, Naziv, Bodovi, Indeks, Ocena}.
Primarni ključ je Id_predmeta, Indeks, a važi i dodatna FZ Naziv $\longrightarrow$ Bodovi.

ID_PREDMETA	SIFRA	NAZIV	BODOVI	Indeks	Ocena
1001	M111	Analiza 1	6	20100021	7
1002	M112	Analiza 2	6	20100021	8
1001	M111	Analiza 1	6	20100022	7
1021	M131	Geometrija	6	20100021	8
1101	M105	Diskretne strukture 1	5	20100021	6
1101	M105	Diskretne strukture 1	5	20100023	6
2002	P102	Programiranje 2	8	20100024	9
2002	P102	Programiranje 2	8	20100025	9
2003	P103	Objektno orijentisano programiranje	8	20100021	7
2004	P104	Algoritmi i strukture podataka	8	20100021	6
4001	R101	Uvod u organizaciju racunara	7	20100021	10
4002	R102	Uvod u Veb i Internet tehnologije	7	20100021	10

Postoji redundatnost: svaki Id_predmeta 1001 kao naziv pokazuje Analiza 1. Takodje, svaka Analiza 1 za broj bodova ima vrednost 6, itd.

ID_PREDMETA	SIFRA	NAZIV	BODOVI	Indeks	Ocena
1001	M111	Analiza 1	6	20100021	7
1001	M111	Analiza 1	6	20100022	7

Anomalije ažuriranja:

- Unošenje: ne može jednostavno da se unese podatak da neki Id odgovara pojedinom nazivu predmeta dok neko nije polagao taj predmet. (jer ne postoji odgovarjaući broj indeksa koji je deo p.k)
- Ako se izbrišu svi podaci za Id_predmeta, takođe se brišu i podaci o nazivu tog predmeta, njegovoj šifri, broju bodova, ...
- Pošto se isti naziv javlja na više mesta, to može da dovede do problema pri promeni naziva.
- Zbog toga se vrši dekompozicija (preko projekcija) ovakvih relacija

Prva normalna forma

Relvar je u 1NF ako i samo ako u svakoj važećoj vrednosti tog relvar-a svaka torka sadrži tačno jednu vrednost za svaki atribut.

Nereducibilna funkcionalna zavisnost (FZ): Ako $A \longrightarrow B$ i uklanjanje bilo kog atributa iz A povlači da $A \longrightarrow B$ postaje netačno, tada je B **nereducibilno** zavisno od A.

Druga normalna forma

Relvar je u 2NF ako i samo ako je u 1NF i svaki neključni atribut je nereducibilno zavisan od primarnog ključa.

Prethodna definicija podrazumeva postojanje **samo jednog** kandidata za ključ koji je istovremeno i primarni ključ.

Treća normalna forma

Relvar je u 3NF ako i samo ako je u 2NF i svaki neključni atribut je netranzitivno zavisan od primarnog ključa.

Prethodna definicija podrazumeva postojanje **samo jednog** kandidata za ključ koji je istovremeno i primarni ključ.
Posledica: neključni atributi su uzajamno nezavisni.

Nedostaci treće normalne forme
Codd-ova originalna definicija 3NF nije uzimala u obzir slučajeve kada relacija

1. ima više od jednog kandidata za ključ
2. kandidat za ključ je kompozitan
3. kompozitni kandidati za ključeve se preklapaju

Ovi slučajevi su obuhvaćeni Bojs-Kodovom normalnom formom.

Primer: relacija SPN (Student-Predmet-Nastavnik)

S	P	N
Lazić	matematika	Petrović
Lazić	računarstvo	Marković
Perić	matematika	Petrović
Perić	računarstvo	Marković

Značenje torke: student S sluša predmet P kod nastavnika N.

Pretpostavke:

1. za svaki predmet svaki student sluša nastavu samo kod jednog nastavnika, tj. {S,P}⟶N
2. Kandidati za ključ su {S,P} i {S,N}
3. FZ {S,P}⟶N
4. Svaki nastavnik predaje samo jedan predmet, tj. N⟶P

Bojs-Kodova normalna forma

Relvar je u BCNF ako i samo ako svaka netrivijalna levo-nereducibilna FZ ima kandidat za ključ kao svoju levu stranu.

Manje formalno: Relvar je u BCNF ako i samo ako su jedini kandidati za ključ leve strane FZ.

Primer redukcije u BCNF;
Neka su nazivi predmeta jedinstveni. Relacija PredmetDosijeId_predmeta, Naziv, Indeks, Ocena
Kandidati za ključ su

- {Id_predmeta, Indeks}⟶Ocena
- {Naziv, Indeks}⟶Ocena

Naziv predmeta je jedinstven:

- Id_predmeta⟶Naziv
- Naziv⟶Id_predmeta

Id_predmeta i Naziv su na levoj strani FZ, ali nisu kandidati za ključ $\Longrightarrow$ PredmetDosije nije u BCNF.
Rešenje: razbijanje PredmetDosije na dve relacije:

1. IN{Id_predmeta,Naziv} i PD{Id_predmeta, Indeks, Ocena}, ili
2. IN{Id_predmeta,Naziv} i PD{Naziv, Indeks, Ocena}

Moguće redukcije:
U relaciji Dosije {Indeks, Jmbg, Datum_rodjenja, ...} kandidati za ključ su Indeks i Jmbg. Neka važe sledeće FZ:

- Indeks ⟶Jmbg
- Jmbg ⟶Datum_rodjenja
- ... ⟶ ...

Relacija nije u 3NF jer postoji tranzitivna zavisnost Indeks ⟶Datum_rodjenja.

Razbijanja relacije Dosije:

- Dosije_1a {Indeks, Jmbg, ...}
  Dosije_1b {Jmbg, Datum_rodjenja, ...}
- Dosije_2a {Indeks, Jmbg, ...}
  Dosije_2b {Indeks, Datum_rodjenja, ...}
- Dosije_3a {Jmbg, Datum_rodjenja, ...}
  Dosije_3b {Indeks, Datum_rodjenja, ...}
- Sva tri para relacija jesu u 3NF i u BCNF.
- Dekompozicije [1] i [2] ne dovode do gubitka informacija
- Pri dekompoziciji [2] javlja se anomalija pri unosu $\Longrightarrow$ korektno razbijanje je kao u slučaju [1].

Pravila:

1. Sve FZ polaznog skupa moraju da budu očuvane (direktno ili mogućim izvođenjem iz skupa relacija dobijenih dekompozicijom).
2. Ako u novodobijenim projekcijama nastalim razbijanjem osnovne relacije postoji zajednički atribut, on mora da bude ključ u bar jednoj od novodobijenih relacija.

Višeznačne zavisnosti:

Relacija PNU (Predmet-Nastavnik-Udžbenik):

P	N	U
matematika	{Petrović, Marković}	{Analiza, Linearna algebra}
računarstvo	Petrović	{Linearna algebra, Uvod u programiranje, Strukture podataka}

Značenje torke: predmet P može da predaje bilo koji nastavnk N i da koristi bilo koji udžbenik U.

Pretpostavke:

1. za dati kurs postoji proizvoljan broj nastavnika i udžbenika
2. nastavnici i tekstovi su nezavisni
3. nastavnik ili tekst može da se pridruži bilo kom kursu

U ovoj relaciji ne postoje FZ.

- Primarni ključ: {Predmet, Nastavnik, Udžbenik}
- Relvar je u 1NF, 2NF, 3NF i BCNF
- Relvar PNU poseduje redundantost
  - ako obe torke (p, n_1, u_1) i (p, n_2, u_2) postoje tada moraju da postoje i torke (p, n_1, u_2) i (p, n_2, u_1)
- Posledica je anomaliju pri ažuriranju:
  - da bi se uneo podatak da novi nastavnik predaje matematiku moraju da se unesu dve torke, po jedna za svaki udžbenik

Intuitivno može da se izvrši dekompozicija

- {predmet, nastavnik} i
- {predmet, udžbenik}

Dekompozicija ne sledi iz FZ (kojih i nema) već iz postojanja višeznačnih zavisnosti.

Neka je R relvar i neka su A, B i C podskupovi atributa od R. Kaze se da je B višeznačno zavisno (VZ) od A, u oznaci A$\longrightarrow\!\!\!\!\!\rightarrow$B, ako i samo ako u svakoj mogućoj važećoj vrednosti od R, skup vrednosti B koji se uparuje sa parom (vrednost A, vrednost C) zavisi jedino od vrednosti A i nezavisan je od vrednosti C.

Četvrta normalna forma

Relvar R je u 4NF ako i samo ako je u BCNF i svaki put kada postoje podskupovi A i B atributa od R takvi da je zadovoljena netrivijalna višeznačna zavisnost A$\longrightarrow\!\!\!\!\!\rightarrow$B, tada su svi atributi od R takođe funkcionalno zavisni od A vskip 2ex

Primedba: VZ A$\longrightarrow\!\!\!\!\!\rightarrow$B je trivijalna ako je ili A nadskup od B ili je A $\cap$ B sadrži sve atribute od R.

Zavisnost spajanja
Neka je R relvar i neka su A, B, ..., Z podskupovi atributa od R. Tada R zadovoljava zavisnost spajanja (ZS) *{A,B,..., Z} ako i samo ako je R u 4NF i svaka moguća važeća vrednost u R je jednaka spajanju njenih projekcija na A,B,..., Z

Peta normalna forma

Relvar R je u 5NF (projekcija-spajanje-NF) ako i samo ako R je u 4NF i svaka netrivijalna zavisnost spajanja koja važi u R je posledica kandidata za ključ u R, gde

- Zavisnost spajanja *{A,B,..., Z} u R je trivijalna akko je najmanje jedan od A,B,..., Z skup svi atributa R

- Zavisnost spajanja *{A,B,..., Z} u R je posledica kandidata za ključ relvara R akko je svaki od A,B,..., Z nadključ za R

Proces normalizacije

Proces normalizacije

1. Uzeti projekcije originalnog relvara u 1NF radi eliminisanja FZ koje nisu nereducibilne. Dobijeni skup relvara je u 2NF.
2. Uzeti projekcije relvara u 2NF radi eliminisanja tranzitivnih zavisnosti. Dobijeni skup relvara je u 3NF.
3. Uzeti projekcije relvara u 3NF radi eliminisanja preostalih FZ u kojima na levoj strani nije kandidat za ključ. Dobijeni skup relvara je u BCNF
4. Uzeti projekcije relvara u BCNF radi eliminisanja VZ koje nisu i FZ. Dobijeni skup relvara je u 4NF.
5. Uzeti projekcije relvara koji su u 4NF i eliminisati ZS koje ne slede iz kandidata za ključ(eve). Dobijeni skup relvara je u 5NF.

Denormalizacija

U praksi se često ne sprovodi puna normalizacija zbog dobrih performansi

- puna normalizacija dovodi do velikog broja logički razdvojenih relvar-a
- veliki broj razdvojenih relvar-a znači veliki broj razdvojenih datoteka u kojima se čuvaju
- veliki broj datoteka znači veliki broj U/I operacija

U praksi se normalizacija najčešće sprovodi do 3NF.

Primer 1
Neka je dat relvar R = {A,B,C,D} i skup F FZ:

1. AB $\longrightarrow$ C
2. C $\longrightarrow$ D
3. D $\longrightarrow$ A

A) Navesti neke netrivijalne FZ koje mogu da budu izvedene iz F

B) Odrediti kandidate za ključ relvara R

C) Navesti sve FZ koje sprečavaju da relavar R bude u BCNF

D) Dekomponovati R tako da dobijene relacije budu u BCNF

Nastavak A
Neke netrivijalne FZ su:

1. C $\longrightarrow$ ACD
2. D $\longrightarrow$ AD
3. AB $\longrightarrow$ ABCD
4. AC $\longrightarrow$ ACD
5. BC $\longrightarrow$ ABCD
6. BD $\longrightarrow$ ABCD
7. CD $\longrightarrow$ ACD
8. ABC $\longrightarrow$ ABCD
9. ABD $\longrightarrow$ ABCD
10. BCD $\longrightarrow$ ABCD

Nastavak B
Iz zatvorenja skupa atributa

1. A+=A
2. B+=B
3. C+=ACD
4. D+=AD
5. AB+=ABCD
6. AC+=ACD
7. AD+=AD
8. BC+=ABCD
9. BD+=ABCD
10. CD+=ACD
11. ABC+=ABCD
12. ABD+=ABCD
13. ACD+=ACD
14. BCD+=ABCD

dobija se da su kandidati za ključ AB, BC i BD.

Nastavak C
FZ koje onemogućavaju da relacija R bude u BCNF su

1. C $\longrightarrow$ D (FZ 2) iz početnog skupa
2. D $\longrightarrow$ A (FZ 3) iz početnog skupa

jer atributi koji se nalaze na njihovim levim stranama nisu kandidati za ključ.

Nastavak D
Neformalni pristup:

1. Ako se relacija R = {A,B,C,D} razbija na osnovu FZ 2. dobija se
   R_1 = {C,D} i R_x = A,B,C (isključuje se atribut koji je na desnoj strani FZ) koja se zatim razbija na osnovu FZ C $\longrightarrow$ A (tranzitivnost!), tako da je krajnji rezultat R_1 = C,D i R_2 = C,A i R_3 = B,C
2. Ako se relacija R = A,B,C,D razbija na osnovu FZ 3. dobija se R_1 = D,A i R_x = B,C,D (isključuje se atribut koji je na desnoj strani FZ) koja se zatim razbija na osnovu FZ 2), tako da je krajnji rezultat R_1 = D,A i R_2 = C,D i R_3 = B,C

Primer 2

Neka je dat relvar R = {A,B,C,D,E,F} i skup FZ:

1. AB $\longrightarrow$ D

2. B $\longrightarrow$ C
3. AE $\longrightarrow$ B
4. A $\longrightarrow$ D
5. D $\longrightarrow$ EF

Transformisati relaciju R tako da novodobijena relacija bude u BCNF.
Uputstvo: odrediti kandidate za ključ za nereduciblnog skupa FZ.

Rešenje
Važi $\{A\}^+ = \{ABCDEF\}$ pa ostale 3 FZ narušavaju BCNF. Razbijanje

1. Iz B $\longrightarrow$ C dobija se R1(B,C), R1a{ABDEF}
2. Iz D $\longrightarrow$ E dobija se R2(D,E), R2a{ABDF}
3. Iz D $\longrightarrow$ F dobija se R3(D,F), R4{ABD}

Sigurnost i integritet

Sigurnost: zaštita podataka od neautorizovanih korisnika (zaštita protiv neautorizovanog pristupa, promene ili uništenja)

Integritet: zaštita podataka protiv autorizovanih korisnika (obezbedenje ispravnosti i korektnosti podataka) - već razmatrano

Sličnosti između sigurnosti i integriteta:

- sistem mora da bude svestan izvesnih ograničenja koje korisnici ne smeju da prekrše
- ograničenja moraju da budu zadata (od strane DBA) u nekom jeziku
- ograničenja moraju da budu evidentirana u sistemskom katalogu (rečniku podataka)
- SUBP mora da vrši nadzor nad operacijama korisnika

Sigurnost

Aspekti problema sigurnosti

- pravni, socijalni i etički (npr. uvid u stanje računa korisnika)
- fizička kontrola (npr. fizičko obezbeđenje računarske sale)
- politička pitanja (odlučivanje ko i čemu sme da pristupi)
- operativni problemi (npr. kako obezbediti tajnost lozinki)
- hardverska kontrola (npr. da li CPU ima mogućnost hardverske zaštite programa)
- podrška operativnog sistema (npr. da li OS briše sadržaj memorije i diskova po završetku rada programa)
- problemi vezani za same baze podataka (npr. da li postoji koncept vlasnika podataka)

Jedinica podataka na koja se osigurava

- baza podataka
- relvar
- pojedinačna torka ili vrednost atributa
- aliasi
- šeme
- indeksi
- paketi
- prostori za čuvanje tabela
- ...

Sigurnost u SQL-u

Sigurnost se obično zapisuje preko kontrolne matrice pristupa

- predstavljanje po korisnicima
- predstavljanje po objektima
- predstavljanje po dozvoli za pristup

Primer: videti autorizacije u DB2 preko DB2 kontrolnog centra.

Postoje dva mehanizma koji su, nezavisno jedan od drugog, uključeni u sistem zaštite

- Pogledi koji mogu da se koriste za sakrivanje osetljivih podataka od neautorizovanih korisnika
- Podsistem za autorizaciju, koji dopušta korisniku sa odredjenim pravima pristupa da ta prava selektivno i dinamički prenosi na druge korisnike, i/ili da preneta prava povuče

Sigurnost u SQL-u - podsistem za autorizaciju

Da bi se korisnik izvršio bilo kakavu operaciju nad nekim objektom on mora da poseduje dozvolu (ili autorizaciju) za tu operaciju nad tim objektom

- Tipovi i vrste dozvola nisu isti kod svih SUBP
- Sistemski administrator (SYSADM nivo autorizacije) je inicijalni vlasnik svih dozvola
- Davanje dozvola se vrši GRANT naredbom, a povlačenje REVOKE naredbom
- Kompletna sintaksa u DB2 SQL Reference

GRANT naredba

Sintaksa za dozvole nad tabelama ili pogledima:

GRANT dozvola [ON [tip] objekat] TO korisnik;

- *dozvola* je lista jedne ili vise vrsta dozvola, razdvojenim zarezima ili fraza ALL PRIVILEGES ili ALL (koja označava sve privilegije koje se mogu dati GRANT naredbom
- *korisnik* je lista korisnika razdvojenih zarezima ili PUBLIC (svi korisnici)
- *objekat* je lista imena jednog ili više objekata (koji su svi istog tipa) razdvojenih zarezima
- *tip* označava tip objekta - ako se izostavi podrazumeva se TABLE ON se ne upotrebljava kada se daje dozvola sa sistemske privilegije

Dozvole koje se odnose na osnovne tabele i poglede:

GRANT dozvola [ON [tip] objekat] TO korisnik;

- CONTROL
- DELETE
- INSERT
- SELECT
- UPDATE (mogu da se navedu pojedinačne kolone)

Dozvole koje se odnose samo na osnovne tabele

- ALTER (dozvola za izvršavanje ALTER TABLE nad tabelom)
- INDEX (dozvola za izvršavanje CREATE INDEX nad tabelom)
- REFERENCES (dozvola za formiranje/brisanje spoljašnjeg ključa koji referiše tu tabelu kao roditelj tabelu)

Primeri dozvola za operacije nad tabelama

- GRANT SELECT ON TABLE DOSIJE TO KORISNIK01;
- GRANT SELECT, UPDATE (SIFRA, NAZIV) ON TABLE PREDMET TO KORISNIK02, KORISNIK03, KORISNIK09;
- GRANT ALL PRIVILEGES ON TABLE DOSIJE, PREDMET, ISPIT TO KORISNIK76, KORISNIK77;
- GRANT SELECT ON TABLE DOSIJE TO PUBLIC;
- GRANT DELETE ON ISPITNI_ROK TO KORISNIK99;

REVOKE naredba

Sintaksa za dozvole nad tabelama ili pogledima

REVOKE dozvola [ON [tip] objekat] FROM korisnik [BY ALL];

Povlačenje dozvole za nekog korisnika uzrokuje da svi planovi/paketi zasnovani na toj dozvoli postanu neispravni i uzrokuju automatsko vezivanje/ponovno vezivanje prilikom pozivanja takvog plana/paketa. Nije moguće ukinuti UPDATE dozvolu samo za pojedine kolone.

GRANT naredba - WITH GRANT OPTION

Ako korisnik K1 želi da prenese dozvolu D korisniku K2, to može da uradi

- naredbom GRANT dozvola ....
- naredbom GRANT dozvola ... WITH GRANT OPTION čime omogućuje korisniku K2 da dalje distribuira dozvolu koja mu je preneta

WITH GRANT OPTION - primeri

Neka je dato
Korisnik K1:
GRANT SELECT ON TABLE DOSIJE TO K2 WITH GRANT OPTION;

Korisnik K2:
GRANT SELECT ON TABLE DOSIJE TO K3 WITH GRANT OPTION;

Korisnik K3:
GRANT SELECT ON TABLE DOSIJE TO K4 WITH GRANT OPTION;
.....
Tada povlačenje dozvole REVOKE SELECT ON TABLE KVOTA FROM K2; prouzrokuje lančano povlačenje dozvole za korisnike K3, K4, ...

Zadatak: Proveriti da li u prethodnom slučaju ako je korisnik K3 (K4) imao i dozvolu dobijenu od nekog drugog korisnika i ta dozvola povučena ili je ostala važeća.

Ostali aspekti autorizacije

- Kompletan sistem treba da bude zaštićen.
- Ne pretpostavljati da je sistem zaštite savršen.
- Voditi evidenciju o prijavljivanju na bazu.

Autorizacija u DB2

- Nivoi autorizacije (videti SQL Reference)
- Primeri

OLAP

OLAP (eng. Online Analytical Processing) se može definisati kao "interaktivni proces formiranja, upravljanja, analiziranja i prikaza podataka". Obično se podaci sa kojima se radi posmatraju i sa njima se upravlja kao da se čuvaju u višedimenzionalnom nizu.

Agregacija podataka

1. Proces analize zahteva odredjenu agregaciju podataka, obično na različite načine i prema različitim grupisanjima
2. Primer: posmatrajmo sledece upite nad bazom STUD2011
    1. Naći prosečne ocene na položenim ispitima
    2. Naći prosečne ocene na položenim ispitima po predmetima
    3. Naći prosečne ocene na položenim ispitima po ispitnom roku (bez obzira na godinu)
    4. Naći prosečne ocene na položenim ispitima po predmetima i roku (bez obzira na godinu roka)

Odgovarajući upiti koji su rešenje su

1. olap.primer1.sql
2. olap.primer2.sql
3. olap.primer3.sql
4. olap.primer4.sql

Problemi:

- pravljenje više sličnih ali neznatno različitih upita je dosadno i zamorno za korisnika
- Izvršavanje svih upita moze da bude jako skupa operacija

Sa više nivoa agregacije u jednom upitu

- olakšava se posao korisniku

- nudi se mogućnost da se sve agregacije izra˘cunaju mnogo efikasnije (u jednom prolazu)

Opcija GROUPING SETS

olap.primer5.sql - kombinacija upita iz olap.primer2.sql i olap.primer3.sql

```
select naziv, oznaka_roka,
dec(avg(ocena*1.0),4,2) as prosek
from ispit a, predmet b
where ocena >5
and a.id_predmeta=b.id_predmeta
group by grouping sets ((naziv), (oznaka_roka))
```

Rezultat se prikazuje u obliku jedne tabele (koja se vrlo teško može nazvati relacijom).

Razlikovanje NULL-ova u prethodnoj tabeli se postiže upitom:

```
select case grouping (naziv)
                      when 1 then '??'
                      else naziv
            end as naziv,
            case grouping (oznaka_roka)
                      when 1 then '!!'
                      else oznaka_roka
            end as oznaka_roka,
            dec(avg(ocena*1.0),4,2) as prosek
from   ispit a, predmet b
where  ocena >5
and    a.id_predmeta=b.id_predmeta
group  by grouping sets ((naziv), (oznaka_roka))
```

ROLLUP

```
select naziv, oznaka_roka,dec(avg(ocena*1.0),4,2) as prosek
from   ispit a, predmet b
where  ocena >5
and    a.id_predmeta=b.id_predmeta
group  by rollup ((naziv), (oznaka_roka))
```

rollup = grouping sets ((naziv,oznaka_roka),(naziv),())

```
group by rollup (A,B,....,Z)=
   grupisanje preko(A,B,...,Z)
                   (A, B, ...)
                    ........
                   (A, B)
                   (A)
                   ()
```

CUBE

```
select naziv, oznaka_roka,
       dec(avg(ocena*1.0),4,2) as prosek
from   ispit a, predmet b
where  ocena >5
and    a.id_predmeta=b.id_predmeta
group  by cube ((naziv), (oznaka_roka))
```

CUBE - u OLAP-u se podaci posmatraju kao da se nalaze u ćelijama višedimenzionalnog niza odnosno

hiperkocke.

GROUP BY CUBE(A,B, ...,Z) znači grupisanje po svakom mogućem podskupu skupa A,B,...,Z (ovim upitom se dobijaju rezultati sva 4 početna upita)

ROLAP i MOLAP

- **ROLAP** podrazumeva da su podaci sačuvani u konvencionalnoj SQL bazi (ROLAP, eng. Relational OLAP)
- **MOLAP** (eng. Multidimesional OLAP) uključuje višedimenzionu bazu u kojoj su podaci konceptualno smešteni kao ćelije u višedimenzionalnom nizu
- Nezavisne (dimenzione) promenljive
- Zavisne (nedimenzione) promenljive

Ukrštene tabele

- OLAP softverski paketi često prikazuju rezultate ne u obliku SQL tabela vec u obliku ukrštenih tabela.
- Ukrštena tabela je višedimenziona tabela koja sadrži vrednosti zavisnih atributa i koja je indeksirana sa ključnim atributima SQL tabela.

CUBE

	apr	feb	jan	jun
Algebarska geometrija	9		10	
Algebarska topologija	10		10	9,7
Algebarska topologija 2			10	
Algebra 1	7,47		8,3	8,31
Algebra 1A	6,64	6,33	9	6,81
Algebra 1B	7,32	6,66	7,12	8,12
Algebra 2			8,77	7,88
Algebra 2A	8,33	10	9,5	9,6
Algebra 2B			8	9,33
Algebra 3	9,33		10	
Algebra 4	10		10	
Analiza 1	6,05		6,8	7,16

Analitičke funkcije

- Poglavlje 3 u priručniku SQL Fundamentals for IBM DB2 Scholars
- Drugi DB2 priručnici (SQL reference, ...)

Rekurzivni SQL, LOB, MQT

Primeri:

- Rekurzivni SQL 11.primeri.sqln {primer1, primer2, primer3}
- LOB 11.primeri.sqlnLOB
- MQT 11.primeri.sqlnMQT