ODGOVORI NA PITANJA - RBP

/* U prva tri dela su pitanja i odgovori koja su bila na svim prethodnim rokovima koji su okaceni kod Mitica na sajtu, a u poslednjem delu pitanja koja vecinom obuhvataju ostatak teorije i za koja smatram da bi eventualno mogla doci na ispitu (mada mislim da su sanse za to male). Odgovori na teorijska pitanja su uglavnom sa slajdova uz mala prosirenja iz literature koju je Mitic preporucio. Fale jos poslednji zadatak sa rokova januar 2013., jun 2013., septembar 2013., jun 2014. i septembar 2015. */

I Teorija

1. Opisati ANSI/SPARC arhitekturu baze podataka.

Prema ANSI/SPARC grupi arhitektura baze podataka sadrze tri nivoa

1) Spoljasnji nivo koji definise nacin na koji individualni korisnik vidi podatke iz baze. Ovaj nivo je najblizi korisniku, jer korisnika zanima samo jedan deo cele baze gde on vidi samo spoljasnje slogove, ali ne i njihovu fizicku reprezentaciju u bazi. Svaki spoljasnji izgled je opisan spoljasnjom shemom koje se sastoji iz definicija svakog od razlicitih tipova slogova. Svaki korisnik ima na raspolaganju maticni jezik (Java, C, PL1, Cobol) u koji se ugradjuje jezik podataka (SQL, QUEL, ...) pomocu koga moze da vrsi operacije nad svojim delom podataka iz baze. Ovi jezici mogu biti cvrsto vezani, kada maticni jezik ne moze da se odvoji od jezika podataka ili labavo vezani kada oni mogu lako i jasno da se razdvoje. Jezik podataka je kombinacija jezika za definiciju podataka (DDL) koji se koristi za deklarisanje ili definisanje objekata u bazi i jezika za rad sa podacima (DML) koji se koristi pri radu i obradi objekata iz baze.

2) Konceptualni nivo predstavlja ukupni informacioni kontekst baze podataka u obliku koji je na nesto visem nivou u poredjenju sa nacinom kako su podaci fizicki smesteni. Podaci se predstavljaju nezavisno i od upitnog jezika i od hardvera na kome se nalaze. Konceptualni izgled je definisan konceptualnom shemom koja sadrzi definicije svakog od tipova konceptualnih slogova i zapisuje se pomocu konceptualnog DDL-a, bez ikakve veze sa fizickom reprezentacijom tih slogova ili pristupa njima. Definicije u konceptualnoj shemi mogu sadrzati i dodatne funkcionalnosti vezane za bezbednost i integritet podataka.

3) Unutrasnji nivo koji predstavlja celokupnu bazu podataka na niskom nivou. Sastoji se od velikog broja razlicitih unutrasnjih slogova. Unutrasnji izgled je definisan preko unutrasnje sheme napisane na unutrasnjem DDL-u koja sadrzi ne samo definicije razlicitih slogova vec sadrzi i informacije o postojanju indeksa, reprezentaciji sacuvanih polja, kako su fizicki smesteni sacuvani slogovi, itd. Neki programi mogu da rade nad

unutrasnjim izgledom baze sto donosi bezbednosni rizik.

Osim ova 3 nivoa arhitektura ukljucuje odredjene vrste preslikavanja:
- Konceptualno/unutrasnje: Definise preslikavanje izmedju konceptualnog nivoa i baze tj. kako su konceptualna polja i slogovi predstavljeni na unutrasnjem nivou. Ako se promeni definicija sacuvanje baze mora se promeniti i konceptualno/unutrasnje preslikavanje. Ono je kljucno za nezavisnost podataka od promene fizicke strukture.
- Spoljasnje/konceptualno: Definise vezu izmedju spoljasnjeg i konceptualnog nivoa. Kljucno je za nezavisnost podataka od promene logicke strukture.
- Spoljasnje/spoljasnje: Definise jedan spoljasnji pogled preko ostalih, cesto je u relacionim sistemima.

2. Nabrojati i ukratko opisati najvaznije funkcije SUBP-a

- definisanje podataka: SUBP treba da primi podatke u izvornom formatu i pretvori ih u objekte tako da on zapravo mora da ima DDL procesor ili kompajler da razume DDL definicije
- obrada podataka: SUBP treba da resava zahteve dohvatanja, menjanja, dodavanja ili brisanja podataka, zapravo mora da ima DML kompajler ili procesor. DML zahtevi mogu biti planirani (zahtev je poznat unapred) i neplanirani (zahtev nije poznat unapred)
- optimizacija izvrsavanja upita: DML zahtevi bilo da su planirani ili ne moraju proci kroz optimizaciju i potom tako optimizovani se izvrsavaju pod kontrolom runtime menadzera.
- obezbedjivanje zastite i integriteta podataka: SUBP mora da kontrolise zahteve korisnika i odbije svaki zahtev koji bi narusio integritet i bezbednost baze.
- obezbedjivanje konkuretnog pristupa podacima i oporavka podataka: Pomocu TP monitora.
- formiranje kataloga podataka: informacije o definiciji svih objekata.
-obezbedjivanje korinsnickog interfejsa
-izvodjenje drugih akcija u svrhu obezbedjivanja sto efikasnijeg rada

3. Sta znaci princip relacionog zatvorenja? Navedite njegove posledice. Sta je relaciona kompletnost?

Osobina da su argumenti i rezultati primene bilo kog relacionog operatora takodje relacije se naziva relaciono zatvorenje. Posledica relacionog zatvorenja je mogucnost pisanja ugnjezdenih relacionih izraza tj. relacionih izraza ciji su operandi takodje relacioni izrazi.
Jezik je relaciono kompletan ako je mocan isto kao i algebra, tj. ako bilo koja relacija predstavljiva u algebri moze da se predstavi i u tom jeziku. SQL je relaciono kompletan, jer postoje SQL izrazi za svaki od 5 primitivnih operatora relacione algebre.

4. Navesti SQL ekvivalente operatora relacione algebre na osnovu kojih se vidi da je SQL relaciono kompletan jezik.

SQL je relaciono kompletan, jer je mocan koliko i relaciona algebra. SQL ekvivalenti osnovnih operatore relacione algebre su:

A WHERE P - SELECT * FROM A WHERE P
A {x, y, ... , z} - SELECT DISTINCT x, y, ..., z FROM A
A TIMES B - A CROSS JOIN B
A UNION B - SELECT * FROM A UNION SELECT * FROM B
A MINUS B - SELECT * FROM A EXCEPT SELECT * FROM B
A RENAME x AS y - SELECT x AS y FROM A

5. Sta su utility programi? Navedite neke od programa koji rade sa unutrasnjim izgledom baze podataka.
To su programi koji sluze da pomognu DBA sa razlicitim administrativnim poslovima. Oni mogu da rade na spoljasnjem nivou i to su aplikacije specijalne namene i unutrasnjem nivou i deo su servera. U praksi su nam potrebni ovakvi programi najcesce za kreiranje inicijalne verzije baze od regularnih podataka, za reorganizovanje podataka u bazi, za statisticke rutine (racunaju statistiku performansi baze: velicina fajlova, I/O brojaci, ...) i analizu statistickih podataka.
Utility programi: load, copy, import, reorg, recover, runstats, check, ...

6. Sta u bazama podataka znaci termin nezavisnost podataka? Sta su glavne karakteristike nezavisnosti podataka.

U bazama podataka, nezavisnost podataka predstavlja otpornost aplikacije na promene fizicke reprezentacije podataka i pristupnih tehnika. Glavne karakteristike su da baza podataka treba da bude sposobna da se siri bez promene postojecih aplikacija kao i da u slucaju sirenja baze ne sme da bude negativnih uticaja na postojece aplikacije. Pojmovi vezani za nezavisnost podataka:
- sacuvano polje: najmanja jedinica podataka koja moze da se cuva
- sacuvani slog: skup sacuvanih polja
- sacuvana datoteka: skup svih trenutno postojecih pojava sacuvanih slogova istog tipa
Aspekti sacuvanih reprezentacija koji mogu da budu predmet promena od strane DBA:
- reprezentacija brojcanih podataka
- reprezentacija znakovnih podataka
- jednice za brojcane podatke
- kodiranje podataka

7. Ko je prvi i kada (navesti godinu) definisao pojam relacionog modela podataka? Koji modeli podataka su bili korisceni u prethodnom periodu? Navesti primere baze koja

implementira svaki od navedenih modela.

Edgar F. Codd, 1970. godine. U prethodnom periodu su korisceni mrezni (primer baze je IDMS) i hijerarhijski model (primer baze je IMS)

8. Definisati bar 5 razlicitih specijalnih registara koji postoje u sistemu DB2 i opisite sta oni sadrze.

- current date: To je tekuci datum koji se uzima iz biosa racunara i predstavlja se u nekom formatu. Npr. select current date from dosije dobija se tekuci datum za svaki red u tabeli dosije. To nije zgodno ako hocemo nesto dalje da radimo s tim, to mozemo da izbegnemo ako koristimo distinct, ali to moze da bude relativno skupa operacija, jer distinct uvek ima sortiranje i onda izbacivanje duplikata, ako je tabela velika to nije zgodno. Da bismo to izbegli koristimo tabelu sysibm.sysdummy1 koja uvek daje tacno jedan podatak, npr. select current date from sysibm.sysdummy1. Sto se tice oblika datuma on nije jedinstven, moze postojati razlika u prikazu datuma na komandnoj liniji i u datastudio-u kao okruzenju.
- current decfloat rounding mode: Definise na koji nacin se vrsi zaokruzivanje. Predefinisan nacin zaokruzivanja je na parnu cifru.
- current degree: Govori koliki je stepen paralelizma moguc na racunaru, taj stepen zavisi od broja fizickih procesora na racunaru, ne od broja jezgara.
- current explain mode: Pomocu njega se moze definisati da li se opisuje ceo postupak odredjene naredbe i kroz sta ona prolazi, posto je moguce dobiti ceo postupak odredjene naredbe koja se izvrsava.
- current isolation: Daje tekuci nivo izolacije, nivo izolacije govori kako nesto sto se radi u okviru neke transakcije utice na neke druge transakcije.
- current sqlid: Daje identifikaciju sql naloga sa kojim se ispituje autorizacija za pristup. Ako zelimo da promenimo current sqlid sa naredbom set current sqlid = ... Menja se autorizacija sa kojom radimo.
- user: Najcesce se koristi, to je identifikacija korisnika koji se logovao.

Kod nekih specijalnih registara je moguce da nemamo razmak nego podvlaku izmedju i one su tek kasnije ukljucene u standard. Isti je rezultat i ako se ne koristi i ako se koristi podvlaka. Moze se koristiti jos jedan oblik, select value(...), dobijamo trenutnu vrednost.

9. Opisite osobine koje mora da poseduje svaka transakcija.

1) Atomicnost: garantuje se da ce se izvrsiti ili sve sto se nalazi u transakciji ili nista od toga.

2) Trajnost: garantuje se da ce po uspesnom izvrsavanju (COMMIT-a) sve promene ostati trajno zapamcene u bazi, bez obzira na kasnije eventualne padove sistema.

3) Izolovanost: transakcije su medjusobno izolovane u smislu da su efekti izvrsavanja jedne transakcije nevidljivi za drugu transakciju sve do (uspesnog) izvrsavanja COMMIT naredbe.

4) Izvrsavanje isprepletanog (u smislu pocetka i kraja) skupa transakcija ce obicno biti serijalizovano u smislu da se dobija isti rezultat kao da se te iste transakcije izvrsavaju jedna po jedna u unapred neodredjenom redosledu izvrsavanja.

10. Opisite aspekte relacionog modela podataka.

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

To su:

1) Aspekt strukture: svi podaci u bazi se korisniku prikazuju iskljucivo u obliku tabela

2) Aspekt integriteta: tabele zadovoljavaju izvesna ogranicenja (primarni i spoljasnji kljucevi, ...)

3) Aspekt obrade: operatori koji su na raspolaganju korisnicima za obradu tabela su takvi da izvode tabele iz tabele.

11. Navesti i ukratko opisati glavne komponente SBP.

Glavne komponente SBP-a su:

1) Podaci: mogu biti integrisani i deljivi. Kod deljivih podataka razliciti korisnici pristupaju istim podacima cesto i u isto vreme. Kod integrisanih podataka bazu cini skup inace nepovazanih fajlova koji medjusobno gotovo da nemaju viskova (suvisnih podataka ili onih koji se ponavljaju).

2) Hardver: spoljasnji memorijski uredjaji i procesori i glavna memorija.

3) Softver:
- SUBP (Sistem za upravljanje bazom podataka) i on predstavlja nivo softvera koji se nalazi izmedju korisnika i fizickih podataka u bazi, stiti korisnike od detalja na hardverskom nivou i upravlja svim zahtevima za direktan pristup bazi.
- Alati za razvoj aplikacija, pisanje izvestaja, pomocni (utility) programi, program za upravljanje transakcijama (TP monitor)

4) Korisnici:
- Aplikativni programeri: Pisu programe na visim programskim jezicima (Cobol, PL1, C++, Java, ...) koji sluze za pristup bazi.
- Krajnji korinsnici: Interaktivno prisupaju bazi pomocu programu koji pisu aplikativni progameri.
- Administratori
    - Administrator baze podataka: on je profesionalac u IT, formira i implementira kontrolne strukture, odgovoran je za implementaciju odluke administratora podataka kao i za rad sistema, performanse, itd. Njegovi poslovi su definisanje konceptualne sheme (logicko projektovanje baze), definisanje unutrasnje sheme (fizicko projektovanje baze) i komunikacija sa korisnicima (da li su im obezbedjeni svi zeljeni podaci, konsultacija pri projektovanju aplikacija, pomoc pri resavanju problema, itd.)
    - Administrator podataka: On razume postojece podatke i odlucuje koji podaci ce biti cuvani u bazi. On takodje ustanovljava pravila za odrzavanje i rad sa podacima po njihovom cuvanju u bazi, nije tehnicko lice vec pripada upravljackim strukturama.

12. Nabrojati relacione operatore. Koji od njih cine minimalni skup operatora?

Codd je originalno predlozio 8 operatora:
restrikcija (selekcija), projekcija, proizvod, unija, presek, razlika, (prirodno) spajanje, deljenje
Kasnije su dodati operatori:
promena imena, poluspajanje, polurazlika, ekskluzivna unija, prosirenje, slika relacije, operatori agregata, sumarizacija, ...
Minimalni skup operatora cine:
restrikcija (selekcija), projekcija, prozivod, unija, razlika.

13. Definisati pojam arhitektura sistema baza podataka.

Arhitektura sistema baza podataka je apstraktni opis njegovih komponenti i njihovih interakcija.

14. Dati formalnu definiciju relacionog operatora spajanja.

Za definicije ovog kao i svih ostalih operatora spajanja pogledati 3. Miticev slajd, strane 12. do 25.

15. Napisati prednosti relacionog modela u odnosu na hijerarhijski.

Osnovna prednost relacionog modela u odnosu na hijerarhijski i mrezni je u tome sto se

u potpunosti oslanja na matematiku, konkretnije na relacionu algebru cime je omogucena racunarska podrzka, razvoj specificnog softvera i obrada uz zagarantovanu konzistentnost podataka i rezultata.

16. Detaljno opisati bar 5 razlicitih prednosti rada sa bazom podataka u odnosu na rad sa podacima koji se nalaze u datotekama. Primedba: navodjenje prednosti bez opisa nece biti priznato kao delimicno uradjen zadatak.

- Podaci mogu biti deljeni: Deljeni znaci da ne samo da aplikacije mogu da dele podatke u okviru baze vec i da mogu biti pisane nove aplikacije koje ce raditi sa istim podacima.
- Smanjenje redundantnosti podataka: U sistemima koji nisu baze podataka svaka aplikacija ima svoje privatne fajlove sto moze dovesti do mnogo ponavljanja medju podacima i bespotrebnog trosenja memorije.
- Izbegavanje nekonzistentnosti: povezano sa ovom stavkom iznad.
- Podrska za transakcioni rad: Ako korisnik trazi da se izvrse dve transakcije odjednom sistem moze da garantuje da su se ili obe izvrsile ili nijedna.
- Odrzavanje integriteta: ako imamo redundantnost podataka, moze da se desi da prilikom reazuiranja azuriramo jedan podataka, a ne drugi, cime baza nece vracati validne podatke kada joj se pristupi. Resenje je da kada reazuriramo jedan, automatski ce biti reazuirana i sva ostala ponavljanja i SUBP garantuje da korisnik nece videti redundantnost podataka.
- Bezbednost: konfliktnim zahtevima i standardima se bavi DBA koji bira na koji nacin ce resiti ove slucajeve u cilju sto optimalnijeg sistema.

17. Navedite upitne jezike koji su zasnovani samo na relacionom racunu.

QBE i QUEL

18. Sta je Kodov algoritam redukcije?

To je prikaz da je relaciona algebra mocna koliko i relacioni i racun i obratno.

19. Pokazati da se pomocu operatora iz minimalnog skupa operatora mogu izvesti bar dva od preostalih Kodovih operatora.

Presek moze trivijalno iz unije i razlike:
A INTERSECT B = (A UNION B) MINUS ((A MINUS B) UNION (B MINUS A))

Prirodno spajanje:
Neka su date dve relacije A i B koje imaju sledeca zaglavlja:
A: {X1, X2, ..., Xm, Y1, Y2, ..., Yn}
B: {Y1, Y2, ..., Yn, Z1, Z2, ..., Zp}

Definicija prirodnog spajanja preko osnovnih Kodovih operatora bi onda bila:
A TIMES B WHERE Y1a = Y1b and Y2a = Y2b and ... Yna = Ynb

Posto poluspajanje i polurazlika mogu da se izvedu iz prirodnog spajanja i projekcije, a prirodno spajanje moze iz osnovnih onda deljenje mozemo izraziti kao:
A{X} NOT MATCHING ((A{X} JOIN B) NOT MATCHING A), gde su A i B relacije takve da zaglavlje relacije A cine unija skupova atributa {X} i {Y}, pri cemu je {Y} skup atributa relacije B.

20. Formalno dokazati: A INTERSECT (A UNION B) = A

Koristim oznaku e za pripada i c za podskup.

=> pretpostavimo: x e A ∩ (A ⋃ B), odatle
x e A i x e A ⋃ B, tj
x e A i (x e A ili x e B), sledi
A ∩ (A ⋃ B) c A (1)
<= pretpostavimo: x e A, onda je tacno i
x e A i x e A ⋃ B, pa
x e A ∩ (A ⋃ B), sledi
A c A∩ (A ⋃ B) (2)
iz (1) i (2) => A ∩ (A ⋃ B) = A sto je trebalo dokazati.

21. Formalno dokazati: da je unija distributivna preko preseka.

A UNION (B INTERSECT C) = (A UNION B) INTERSECT (A UNION C)

=> pretpostavimo: x e A ⋃ (B ∩ C), sledi
x e A ili (x e B i x e C), sledi
(x e A ili x e B) i (x e A ili x e C), dakle
x e (A ⋃ B) ∩ (A ⋃ C)
<= pretpostavimo: x e (A ⋃ B) ∩ (A ⋃ C)
(x e A ili x e B) i (x e A ili x e C)
1) x e A => x e A ⋃ (B ∩ C)
2) x nije e A => x e B i x e C => x e B ∩ C => x e A ⋃ (B ∩ C)

22. Formalno dokazati: da je projekcija distributivna preko unije.

(A UNION B){X} = A{X} UNION B{X}

=> pretpostavimo: x e (A ⋃ B){X}, sledi
x e A{X} ili x e B{x} => x e A{X} UNION B{X}

<= pretpostavimo: x e A{X} ⋃ B{X}, sledi
x e A{X} ili x e B{X} => x e (A ⋃ B){X}

23. Formalno dokazati: da su spajanje i unija asocijativne operacije, ali da to nije i razlika.

Unija:

pretpostavimo: x e A ⋃ (B ⋃ C), sledi
x e A ili (x e B ili x e C), sledi
(x e A ili x e B) ili x e C, dakle
x e (A ⋃ B) ⋃ C

Spajanje:
Neka su dati A, B i C, pretpostavimo da ih je moguce spojiti tj. da imaju zajednicki atribut po kom se mogu spojiti. U tom slucaju ovaj dokaz se svodi na dokaz da je presek asocijativna operacija.

pretpostavimo: x e A ∩ (B ∩ C), sledi
x e A i (x e B i x e C), sledi
x e A i x e B i x e C, dakle
x e (A ∩ B) ∩ C

I nacin: Dokaz kontradikcijom:

A \ (B \ C) != (A \ B) \ C ?
neka su A, B, C c X, tada je
A\B = A ∩ B' gde je B' = X \ B, sledi
(1) (A \ B) \ C = (A ∩ B') ∩ C' = A ∩ B' ∩ C'
(2) A \ (B \ C) = A∩ (B ∩ C')' = A ∩ (B' ⋃ C'') = (A ∩ B') ⋃ (A ∩ C) =
(A ∩ B' ∩ (C ⋃ C')) ⋃ (A ∩ (B ⋃ B') ∩ C) =
(A ∩ B' ∩ C) ⋃ (A ∩ B' ∩ C') ⋃ (A ∩ B ∩ C) =
(A ∩ B' ∩ C') u (A ∩ (B ⋃ B') ∩ C) = (A ∩ B' ∩ C') ⋃ (A ∩ C)
iz (1) i (2) => (A ∩ B' ∩ C') = (A ∩ B' ∩ C') ⋃ (A ∩ C) sto je u opstem slucaju kontradikcija!

II nacin(znatno laksi): Dokaz kontraprimerom:
Neka je A = {1, 2, 3}, B = {1, 5}, C = {3, 4}. Tada je:
A \ (B \ C) = A \ {1, 5} = {2, 3},
(A \ B) \ C = {2, 3} \ C = {2}.
S obzirom da tvrdjenje treba da vazi za svako x e N, dokazali smo da u opstem slucaju ne vazi A \ (B \ C) = (A \ B) \ C.

24. Izraziti relacione operatore poluspajanja i polurazlike pomocu osnovnih Kodovih operatora.

Operator poluspajanja mozemo izraziti preko projekcije i prirodnog spajanja:
Neka su date relacije r1 i r2, tada poluspajanje relacija r1 i r2 u oznaci r1 MATCHING r2 definisemo kao (r1 JOIN r2){H1} gde je {H1} zaglavlje relacije r1.

Sada iz poluspajanja i razlike mozemo izvesti polurazliku kao:
r1 NOT MATCHING r2 = r1 MINUS (r1 MATCHING r2)

25. Formalno dokazati: da je presek distributivan preko unije.

A INTERSECT (B UNION C) = (A INTERSECT B) UNION (A INTERSECT C)

=> pretpostavimo: x e A ∩ (B ⋃ C), sledi
x e A i (x e B ili x e C), sledi
(x e A i x e B) ili (x e A i x e C), dakle
x e (A ∩ B) ⋃ (A∩ C)
<= pretpostavimo: x e (A ∩ B) ⋃ (A ∩ C)
(x e A i x e B) ili (x e A i x e C)
1) (x e A i x e B) => x e A i (B ⋃ C) => x e A ∩ (B ⋃ C)
2) (x e A i x e B) => x e A i (B ⋃ C) => x e A ∩ (B ⋃ C)

26. Formalno dokazati da za operatore relacione algebre vazi da je restrikcija distributivna preko unije.

A Θ (B ⋃ C) = (A Θ B) ⋃ (A Θ C)
=> pretpostavimo: x e A Θ (B ⋃ C), sledi
x e A Θ B ili A Θ C => x e (A Θ B) ⋃ (A Θ C)
<= pretpostavimo: x e (A Θ B) ⋃ (A Θ C), sledi
x e A Θ B ili A Θ C => x e A Θ (B ⋃ C)

27. Ukratko opisati ogranicenja integriteta. Koje vrste postoje? Dati primere.

Postoje ogranicenja stanja i ogranicenja prelaza. Ogranicenja stanja definisu prihvatljiva stanja u bazi i ona se dele na ogranicenja baze koja se odnose na vrednosti koje je dozvoljeno cuvati u bazi (tj. koje se odnose na dve ili vise razlicitih relacija), ogranicenja relacija (relvar-a) kojima se zadaje ogranicenje na vrednost pojedinacne relacije (relvar-a) koje se proverava pri azuriranju te relacije, ogranicenja atributa koja predstavljaju ogranicenja na skup dozvoljenih vrednosti datog atributa i ogranicenja tipa

koja predstavljaju definiciju skupova vrednosti koji cine dati tip.
Primer ogranicenja baze:

```
CONSTRAINT BAZA1
FORALL DOSIJE D FORALL ISPIT I
IS_EMPTY (( D JOIN I )
WHERE I.INDEKS > 20150000
AND   I.INDEKS = D.INDEKS
AND   GODINA_ROKA=GODINA_ROKA(2015);
```

Primer ogranicenja relacija:

```
CONSTRAINT REL1
IF NOT ( IS_EMPTY ( PREDMET ) ) THEN
COUNT ( PREDMET
WHERE SIFRA= SIFRA ('R270')) > 0
END IF;
```

Primer ogranicenja atributa:

```
VAR PREDMET BASE RELATION
{
ID_PREDMETA    INTEGER,
SIFRA          SIFRA  ,
NAZIV          NAZIV  ,
BODOVI         SMALLINT
}
```

Primer ogranicenja tipa:

```
TYPE POINT POSSREP
CARTESIAN (X RATIONAL, Y RATIONAL)
CONSTRAINT ABS
(THE_X (POINT)) <= 100.0 AND
ABS(THE_Y (POINT)) <= 100.0  ;
```

28. Dati primer ogranicenja prelaza na primeru studentske baze.

Student ne moze da slusa neki predmet koji nije na tom smeru...

29. Detaljno opisati pravila referencijalne akcije koja se odnose na pravila brisanja azuriranja i unosa u DB2. Kojom sql naredbom mozete da postavite ova pravila?

Ova pravila mozemo da primenjujemo pri create table ili alter table navodeci opcije on delete [pravilo] ili on update [pravilo]. Ova pravila ce biti primenjivana na redove dete tabele koja je spoljasnjim kljucem povezana sa roditelj tabelom.

Pri brisanju postoje pravila:
- NO ACTION(default)
- RESTRICT

- CASCADE
- SET NULL

NO ACTION i RESTRICT:
Ako je specificirano neko od ova dva pravila, prijavi se greska i nijedan red nije obrisan.

CASCADE:
Kada se obrise red u roditelj tabeli, svi redovi dete tabele povezani sa roditelj tabelom se takodje brisu.
SET NULL:
Kada se obrise red u roditelj tabeli, svi redovi dete tabele povezane sa roditelj tabelom su postavljeni na NULL (ako je moguce njihove vrednosti postaviti na NULL).

Pravlila za azuriranje su:
- NO ACTION(default)
- RESTRICT

NO ACTION je default, a jedin alternativa je RESTRICT.

Razlika izmedju RESTRICT i NO ACTION je u tome sto se RESTRICT primenjuje pre bilo kojih drugih referencijalnih ogranicenja za menjanje kao sto su CASCADE i SET NULL dok se NO ACTION primenjuje posle svih drugih referencijalnih ogranicenja. Pri prijavljivanju greske za NO ACTION i RESTRICT bice drugaciji SQLSTATE.

Kod unosenja nemamo dodatne naredbe, pravilo za unosenje je takvo da ako se nesto unosi u roditelj tabelu, nikada nece biti uneseno u dete tabelu koja je povezana sa roditelj tabelom osim ako vec postoji vrednost u odgovarajucim kolonama povezanim sa roditelj tabelom.

30. Navesti bar 5 objekata u DB2 nad kojima se primenjuje DDL kao i naredbe za definisanje i za obradu podataka.

Objekti su: Memorijske grupe, particije baze podataka, prostori za cuvanje tabela, tabele, korisnicki definisane funkcije, pul bafera...
Naredbe za definisanje podataka: CREATE, ALTER, DROP, DECLARE
Naredbe za obradu podataka: SELECT, INSERT, UPDATE, DELETE, MERGE.

31. Navedite objekte u DB2 bazi na koje moze da se primeni dodela/oduzimanje dozvola.

Tabele, indeksi, seme, prostori za tabele, funkcije...

32. Opisati efekat izvrsavanja MERGE naredbe SQL-a.

Merge naredba azurira tabelu ili pogled na osnovu nekih ulaznih podataka. Redovi koji se poklapaju sa ulaznim podacima su azurirani kao sto je naznaceno, a redovi koji ne postoje u toj tabeli ili pogledu su uneseni. Azuriranje i unosenje reda u pogled azurira ili unosi red u tabelu na kojoj je zasvnovan pogled, ako nije definisan INSTEAD OF triger nad pogledom.

33. Sta je referencijalni ciklus? Kojim SQL naredbama moze da se napravi? Navesti primer referencijalnog ciklusa. Da li je moguce napraviti referencijalni ciklus koristeci tabele u studentskoj bazi? Ako je odgovor potvrdan navedite SQL naredbe kojima se formira ciklus, a ako je odrican objasnite zasto nije moguce napraviti ciklus sa postojecim tabelama.

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
• Sistem koji ne poseduje znanje o kandidatima za ključ ponekad pokazuje karakteristike koje nisu “čisto relacione”.

Referencijalni ciklus je specijalni slučaj referencijalnog integriteta kod koga roditelj tabela i dete tabela predstavljaju istu tabelu, ili se, u slučaju da u referencijalnom nizu postoji više tabela, poslednja tabela referencijalnog niza referise na prvu tabelu. Ako u ciklusu učestvuje samo jedna tabela tada se on može formirati naredbom CREATE TABLE. U suprotnom, naredba koja je neophodna za formiranje referncijalnog ciklusa je ALTER TABLE pomoću koje se definišu spoljašnji ključevi kojima se zatvara referencijalni ciklus.

34. Dati definiciju BCNF. Da li je relacija dosije (iz prosirene studentske baze) u BCNF? Dati obrazlozenje odgovora.

Relvar je u BCNF ako i samo ako svaka netrivijalna levo-nereducibilna FZ ima kandidat za kljuc kao svoju levu stranu, manje formalno, relvar je u BCNF ako i samo ako su jedini kandidati za kljuc leve strane FZ.
Dosije nije u BCNF, jer cak nije ni u 3NF, jer postoji tranzitivna zavisnosti
Indeks → Jmbg, Jmbg → Datum_rodjenja => Indeks → Datum_rodjenja

35. Dati definiciju viseznacne zavisnosti dva podskupa A i B relacije R.

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

36. Dati formalne definicije levog, desnog i potpunog spoljasnjeg spajanja.

Levo spoljasnje spajanje:
Neka su date relacije R i S neka su R1, R2, ... , Rn atributi relacije R i neka tabela S1 sadrzi isto zaglavlje kao i tabela S s tim sto za svaki atribut sadrzi vrednost null.
Definicija levog spoljasnjeg spajanja bi u tom slucaju bila:
(R JOIN S) UNION ((R MINUS (R JOIN S){R1, R2, ..., Rn}) TIMES S1)
Naravno, pretpostavlja se da znamo formalne definicije prirodnog spajanja, unije, razlike, dekartovog proizvoda i projekcije.

Desno spoljasnje spajanje:
Neka su date relacije R i S i neka su S1, S2, ... , Sn atributi relacije S i neka tabela R1 sadrzi isto zaglavlje kao i tabela R s tim sto za svaki atribut sadrzi vrednost null.
Definicija desnog spoljasnjeg spajanja bi u tom slucaju bila:
(R JOIN S) UNION (R1 TIMES (S MINUS ((R JOIN S){S1, S2, ..., Sn}))

Potpuno spoljasnje spajanje:
Kada znamo definicije levog i desnog jednostavno je videti da je definicija potpunog spoljasnjeg spajanja:
R FULL OUTER JOIN S = (R LEFT OUTER JOIN S) UNION (R RIGHT OUTER JOIN S)

37. Dati definicije I, II, III, IV, V normalne forme.

Relvar je u 1NF ako i samo ako u svakoj vazecoj vrednosti tog relvar-a svaka torka sadrzi tacno jednu vrednost za svaki atribut.
Relvar je u 2NF ako i samo ako je svaki nekljucni atribut nereducibilno zavisan od primarnog kljuca.
Relvar je u 3NF ako i samo ako je u 2NF i svaki nekljucni atribut je netranzitivno zavisan od primarnog kljuca. Prethodna definicija podrazumeva postojanje samo jednog kandidata za kljuc koji je istovremeno i primarni kljuc, a posledica toga je da su nekljucni atributi uzajamno nezavisni.
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 viseznacna zavisnost $A \rightarrow\rightarrow B$, tada su svi atributi funkcionalno zavisni od A. Primedba: VZ $A \rightarrow\rightarrow B$ je trivijalna ako je ili A nadskup od B ili $A \cap B$ sadrzi sve atribute od R.

5NF: 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 svaka moguca vazeca vrednost u R je jednaka spajanju njenih projekcija na A, B, ..., Z.
Relvar R je u 5NF (projekcija-spajanje-NF) ako i samo ako R je u 4NF i svaka netrivijalna zavisnost spajanja koja vazi u R je posledica kandidata za kljuc u R, gde:
- Zavisnost spajanja *{A, B, ..., Z} u R je trivijalna akko je najmanje jedan od A, B, ..., Z skup svih atributa R.
- Zavisnost spajanja *{A, B, ..., Z} u R je posledica kandidata za kljuc relvara R akko je svaki od A, B, ..., Z nadkljuc za R.

38. Definisati pojam domena i njegovu vezu sa tipovima podataka u SQL-u?

Domen je u SQL-u prost, korisnički definisan imenovan objekat koji se može koristiti kao alternativa za predefinisan tip podatka nad kojim se definiše. Može imati default vrednost i jedno ili više ograničenja.

II SQL upiti i naredbe

39. Napisati SQL upit koji za svakog studenta prikazuje:
- broj indeksa, ime, prezime, sifru predmeta i ocenu na predmetu koje je student polagao, ako su poznati godina njegovog rodjenja i datum kada je polagao ispit
- broj indeksa, ime, prezime, sifru predmeta i tekst 'Nepoznati godina rodjenja i datum polaganja ispita', u slucaju da su nepoznati i godina rodjenja i datum polaganja predmeta, i
- broj indeksa, ime, prezime, sifru predmeta i negativnu vrednost ocene u ostalim slucajevima

```
select indeks, ime, prezime, sifra, char(ocena)
from dosije join ispit on ispit.indeks=dosije.indeks
join predmet on ispit.id_predmeta=predmet.id_predmeta
where year(dosije.datum_rodjenja) is not Null and ispit.datum_ispita is not Null

union

select indeks, ime, prezime, sifra, 'Nepoznati godina rodjenja i datum polaganja ispita'
from dosije join ispit on ispit.indeks=dosije.indeks
join predmet on ispit.id_predmeta=predmet.id_predmeta
where year(dosije.datum_rodjenja) is Null and ispit.datum_ispita is Null

union

select indeks, ime, prezime, sifra, char(ocena*(-1))
from dosije join ispit on ispit.indeks=dosije.indeks
join predmet on ispit.id_predmeta=predmet.id_predmeta
where (year(dosije.datum_rodjenja) is Null and ispit.datum_ispita is not Null) or
(year(dosije.datum_rodjenja) is not Null and ispit.datum_ispita is Null)
```

40. Napisati SQL upit koji za sve vrednosti atributa bodovi iz tabele predmeta prikazuje brojeve bodova i nazive predmeta koji nose taj broj bodova razdvojene zarezima. Atribute u rezultujucoj tabeli nazvati "Broj bodova" i "Predmeti", i izvestaj sortirati opadajuci po redosledu broja bodova.

```
select listagg(naziv,',') "Predmeti", bodovi "Broj bodova"
from predmet
group by bodovi
order by bodovi desc
```

41. Napisati SQL upit koji za svakog studenta prikazuje broj indeksa , ime i prezime, i naziv predmeta koga je student polozio, ako vazi da je dan kad je ispit polagan bio jednak poslednjem danu u godini u kojoj je student rodjen. Pri tome, u prikazu imena i prezimena treba sva velika slova pretvoriti u mala i obrnuto.

```
select indeks, translate(ime,
'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ',
'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz'),
translate(prezime,
'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ',
'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz'), naziv
from dosije join ispit on ispit.indeks=dosije.indeks
join predmet on predmet.id_predmeta=ispit.id_predmeta
where ocena > 5 and dayname(ispit.datum_ispita) = dayname(last_day('1.12.' ||
char(year(datum_rodjenja))))
```

42. Napisati SQL upit koji prikazuje imena i prezimena studenata cija je srednja ocena ili veca od srednje ocene svih studenata uvecane za standardnu devijaciju svih ocena studenata, ili manje od srednje ocene svih studenata umanjene za standardnu devijaciju svih studenata.

```
select ime, prezime
from dosije d join ispit i on d.indeks = i.indeks
where datum_ispita = (select max(datum_ispita)
                      from ispit i1
                      where i1.indeks = i.indeks and i1.id_predmeta = i.id_predmeta
                      and i.ocena > 5)
group by ime, prezimena
        having avg(ocena * 1.0) >
        (select avg(ocena * 1.0) from ispit i)   + (select stddev(ocena) from ispit i)

union

select ime, prezime
from dosije d join ispit i on d.indeks = i.indeks
where datum_ispita = (select max(datum_ispita)
                      from ispit i1
                      where i1.indeks = i.indeks and i1.id_predmeta = i.id_predmeta
```

```
                    and i.ocena > 5)
group by ime, prezimena
        having avg(ocena * 1.0) <
        (select avg(ocena * 1.0) from ispit i)   - (select stddev(ocena) from ispit i)
```

43. Napisati SQL upit koji prikazuje imena i prezimena studenata i broj godina, dana i meseci koji je protekao od njihovog rodjenja do danas. Pri tome, u prikazu imena i prezimena i treba sva velika slova pretvoriti u mala i obrnuto.

```
select translate(ime,
'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ',
'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz'),
translate(prezime,
'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ',
'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz'),
year(current_date-datum_rodjenja), months_between(current_date, datum_rodjenja),
days(current_date) - days(datum_rodjenja)
from dosije
```

44. Napisati SQL upit koji za sve studente rodjene u istom mestu rodjenja prikazuje njihov broj, prosecnu ocenu i standardnu devijaciju ocena. Izvestaj treba da se odnosi na sve studente iz tog mesta, bez obzira da li su do sada polozili neki ispit ili ne. Pri izracunavanju uzeti u obzir ocene samo iz predmeta koji su polozeni.

```
select count(d.indeks), coalesce(avg(ocena * 1.0), 0) prosek, coalesce(stddev(ocena), 0)
from dosije d left outer join ispit i on d.indeks = i.indeks
and ocena > 5 and status_prijave = 'o'
group by d.mesto_rodjenja;
```

45. Napisati SQL upit koji prikazuje imena i prezimena studenata i broj godina, dana i meseci koji je protekao od njihovog rodjenja do danas. Pri tome, u prikazu imena i prezimena treba sve niske karaktera 'ra' zameniti sa 'arny'.

```
select replace(ime, 'ra', 'arny'), replace(prezime, 'ra', 'arny'), year(current date -
datum_rodjenja), months_between(current date, datum_rodjenja),
days(current date) - days(datum_rodjenja)
from dosije
```

46. Napisati SQL upit koji za studente rodjene u petak 13. prikazuje broj indeksa, koliko

je dana proteklo od njihovog rodjenja do danas, kao i trenutni prosek ocena. Ako student nije polozio nijedan predmet, kao prosek ocena treba prkazati -1, a ako nije polagao nijedan predmet kao prosek ocena prikazati poruku 'Do sada nije polagao ispit'.

```
select d.indeks, days(current_date) - days(datum_rodjenja),
case
        when d.indeks not in(select indeks from ispit) then 'Do sada nije polagao nijedan
ispit'
        when avg(ocena * 1.0) is null then '-1'
        else char(avg(ocena * 1.0))
end
from dosije d join ispit i on d.indeks = i.indeks
where ocena > 5 and status_prijave = 'o' and
day(datum_rodjenja) = 13 and dayname(datum_rodjenja) = 'Friday'
group by d.indeks, datum_rodjenja;
```

47.Napisati SQL upit koji prikazuje imena i prezimena svih studenata koji su rodjeni u mestu ciji naziv poseduje najvise tri znaka '%' i zavrsava se malim slovom.

```
select ime, prezime, substr(mesto_rodjenja, length(trim(mesto_rodjenja)), 1),
mesto_rodjenja,length(trim(mesto_rodjenja))
from dosije
where mesto_rodjenja not like '%\%%\%%\%%\%%' escape '\' and mesto_rodjenja is
not null
and mesto_rodjenja <> '' and
substr(mesto_rodjenja, length(trim(mesto_rodjenja)), 1) in
('a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z');
```

48. Na nivou fakulteta doneta je odluka da se prebrojavaju studenti koji se upisuju na studije. Pri tome, prvi naredni upisani sudent treba da ima redni broj 1001, student upisan posle njega 1002, itd. Napisati SQL naredbu kojom se formira tabela dosije1 koja pored istih atributa kao i tabela dosije ima i atribut rbr koji sadrzi redni broj novoupisanih studenata, i zabraniti brisanje tako formirane tabele.

```
create table dosije_nov like dosije;

alter table dosije_nov
add rbr integer
constraint uslov check(rbr >= 1001);

alter table dosije_nov add restrict on drop;
```

```
create trigger dosije_nov
before insert on dosije_nov
referencing new as nova
for each row
set nova.rbr = (select coalesce(max(rbr), 1001)
                        from dosije_nov) + 1;
```

49. Ne koristeci WITH naredbu napisati pogled koji sadrzi naziv smera, godinu upisa, broj studenata na tom smeru koji su upisani te godine, i maksimalnu vrednost trenutne prosecne ocene svakog od studenata upisanih u toj godini na tom smeru. Da li je preko prethodnog pogleda moguce izvrsiti unos podataka u osnovnu tabelu? Obrazloziti odgovor. Nabrojati bar pet slucajeva kada definisani pogled ne moze da se koristi za azuriranje osnovnih tabela.

```
create view pogled (naziv, godina, broj, prosek) as
(
select s.naziv, year(datum_upisa), count(d.indeks),
max
(
        (select avg(ocena * 1.0) prosek
        from ispit i join dosije d1 on i.indeks = d1.indeks
        where status_prijave = 'o' and ocena > 5 and year(d1.datum_upisa) =
        year(d.datum_upisa)
        and d1.id_smera = d.id_smera and d.indeks = d1.indeks
        group by i.indeks
        )
)
from smer s join dosije d on d.id_smera = s.id_smera
group by s.id_smera, s.naziv, year(datum_upisa)
);
```

Preko ovog pogleda nije moguce izvrsiti unos podataka u osnovnu tabelu, jer sadrzi join.

Osnovna tabela preko pogleda moze da se azurira ako:
- Izraz kojim se definise pogled je select izraz koji ne saddrzi JOIN, UNION, INTERSECT ili EXCEPT
- Select klauzula ne sadrzi kljucnu rec DISTINCT
- Svaka select stavka sadrzi (kvalifikovano) ime koje predstavlja referencu na kolonu osnovne tabele
- from klauzula sadrzi referencu na tacno jednu tabelu koja je ili osnovna tabela ili pogled koji moze da se azurrira

- where klauzula select izraza ne sadrzi podupit u kome se from klauzula referise na istu tabelu kao i from klauzula u select izrazu na najvisem nivou
- select izraz ne sadrzi group by niti having klauzulu

50. Formirati tabelu Dosije_nov koja sadrzi iste atribute kao i tabela dosije. Definisati jmbg kao primarni kljuc tabele Dosije_nov. Prosiriti tabelu Dosije_nov atributom srednje_slovo koji treba da sadrzi srednje slovo (ako postoji) studenta iz tabele. U slucaju da srednje slovo ne postoji, vrednost atributa treba da bude "nepoznato". Napuniti tabelu Dosije_nov podacima iz tabele dosije, pri cemu se analizira sadrzaj atributa prezime, i ako njemu postoji srednje slovo tada se taj sadrzaj razdvaja na dva dela: srednje slovo i ostatak prezimena koji se upisuju u atribute srednje_slovo i prezime tabele Dosije_nov. Srednje slovo je karakter za kojim sledi ".".

```
create table dosije_nov like dosije;

alter table dosije_nov
   add constraint primarni_kljuc
   primary key(jmbg);

alter table dosije_nov
   add srednje_slovo char;

insert into dosije_nov( indeks, id_smera, status, ime, pol, jmbg, dat_rodjenja,
                        mesto_rodjenja, drzava_rodjenja, ime_oca, ime_majke,
                        adr_ulica, adr_broj, adr_grad, adr_postbroj, adr_drzava,
                        adr_telefon, adr_mobilnitel, adr_email, adr_wwwuri,
                        dat_upisa, prezime, srednje_slovo)
select  indeks, id_smera, status, ime, pol, jmbg, dat_rodjenja, mesto_rodjenja,
        drzava_rodjenja, ime_oca, ime_majke, adr_ulica, adr_broj, adr_grad,
        adr_postbroj, adr_drzava, adr_telefon, adr_mobilnitel, adr_email, adr_wwwuri,
        dat_upisa, prezime, srednje_slovo,
        case locate(prezime, '.')
          when 0
            then prezime
          else
            substr(prezime, locate(prezime, '.') + 1)
      end as prezime,
      case locate('.', prezime)
        when 0
          then 'Nepoznato'
        else
          substr(prezime, 1, locate(prezime, '.') - 1)
```

```
end as srednje_slovo
from  dosije d;
```

51. Opisati ogranicenja domena koje bi primenili u tabeli kurs iz prosirene studentske baze. Napisati SQL naredbu za formiranje tabele kurs_domen koja implementira ta ogranicenja domena koja ste naveli.

Moze se primeniti ogranicenje da vrednost kolone semestar mora biti 1 ili 2.

```
create table kurs_domen
(
id_predmeta integer not null,
godina smallint not null,
semestar smallint not null,
primary key(id_predmeta, godina, semestar),
foreign key fk_predmet (id_predmeta) references predmet,
constraint chk_semestar check(semestar in (1, 2))
);
```

52. Neka je PS prosecna ocena svih studenata na smeru S, i SDS standardna devijacija prosecne ocene svih studenata na smeru S. Formirati MQT Van_intervala koja sadrzi broj indeksa, ime i prezime, naziv smera i prosecnu ocenu studenata cija je prosecna ocena van intervala [PS - SDS, PS + SDS]. Pri tome se PS i SDS odnose na smer koji student studira. Prikazati, iz tabele Van_intervala, cetvrtog i petog studenta sa najvecim prosekom koji studira smer Informatika.

```
create table Van_intervala as (
with smerovi_ps as (
select d.id_smera, avg(ocena*1.0) prosek
from dosije d join ispit i on d.indeks=i.indeks
and i.ocena>5 and i.status_prijave='o'
group by d.id_smera
),
devijacija as (
select id_smera, avg(ocena*1.0) prosek
from dosije d join ispit i on d.indeks=i.indeks
and i.ocena>5 and i.status_prijave='o'
group by id_smera, d.indeks
),
dev2 as (
select id_smera , stddev(prosek) dev
```

```
from devijacija
group by id_smera
),
rez as (
select d.id_smera,prosek-dev levo , prosek+dev desno
from smerovi_ps ps join dev2 d on ps.id_smera=d.id_smera
),
std as (
select d.indeks, s.naziv, s.id_smera, avg(ocena*1.0) prosek
 from dosije d join smer s on s.id_smera=d.id_smera
join ispit i on i.indeks=d.indeks and i.ocena>0 and i.status_prijave='o'
group by d.indeks,s.id_smera,s.naziv )
select indeks, naziv, prosek
from std s join rez r on r.id_smera=s.id_smera
where prosek <levo or prosek>desno
)
data initially deferred
refresh deferred;

with pomocna as
(select dense_rank() over(order by prosek desc) as redni_broj,
       vi.indeks, vi.ime, vi.prezime, vi.naziv
 from   Van_intervala vi
 where  naziv = 'Informatika')
select *
from   pomocna p
where  p.redni_broj in (4,5);
```

53. Definisati pogled duzine(tabela, korisnik, broj atributa, duzina_sloga) koji sadrzi naziv tabele, identifikaciju korisnika koji je formirao tu tabelu, broj atributa u toj tabeli i duzinu sloga tabele (zbir duzina svih atributa u tabeli). Da li je moguce ovim pogledom azurirati osnovnu tabelu nad kojom je pogled napravljen. Obrazloziti odgovor.

```
create view duzine (tabela, korisnik, broj_atributa, duzina_sloga) as
(
select t.name, t.creator, t.colcount, sum(t1.length) -- id kor
from sysibm.systables t join sysibm.syscolumns t1 on t.name = t1.tbname
group by t.name, t.creator
);
```

54. Napisati SQL upit koji prikazuje prvih 10 studenata sa najvecim prosekom ocena koji do sada nisu diplomirali.

```
with prosek as
(
select d.indeks, coalesce(avg(ocena * 1.0), 0) pr
from dosije d left outer join ispit i on d.indeks = i.indeks
join status s on d.indeks = s.indeks and s.status <> 'diplomirao'
group by d.indeks
order by pr desc
)
select p.indeks, p.pr
from prosek p
fetch first 10 rows only;
```

55. Napisati SQL upit koji prikazuje naziv smera za koji vazi da su svi studenti koji ga studiraju polozili sve predmete koji mogu da budu uslovni za neki drugi predmet u toku studija.

```
select distinct s.naziv
from smer s
where not exists(select indeks
                 from dosije d
                 where d.id_smera = s.id_smera
                 and not exists(select up.id_uslovnog
                                from uslovni_predmet up join
                                obavezan_predmet op on up.id_predmeta =
                                op.id_predmeta
                                 where op.id_smera = s.id_smera
                                 and exists(select i.id_predmeta
                                            from ispit i
                                            where i.indeks = d.indeks
                                            and i.id_predmeta = up.id_uslovnog
        a                                 and ocena > 5 and status_prijave = 'o')));
```

56. Na SQL jeziku definisati funkciju ciji je argument tipa character(20), a rezultat ulazna niska u kojoj su sva mala slova zamenjena velikim, i sva velika slova zamenjena malim, dok se karakteri koji nisu slova ne menjaju.

```
create function slova(niska character(20))
returns character(20)
return
translate(niska,
'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ',
```

'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz');

57. Napisati SQL upit kojim se prikazuje danasnji datum uvecan za 2 godine i trenutno vreme na racunaru umanjeno za 10 casova i 30 minuta.

```
values(current_date + 2 years,
        current_time - 10 hours - 30 minutes);
```

58. Formirati pogled preostalo(rbr, ime, prezime, naziv smera, broj_nepolozenih_predmeta) koji sadrzi informacije o broju predmeta svakog studenta koje je preostalo da polozi do zavrsetka studija.

Ovo u sustini nije moguce uraditi, tj. morao bi da postoji jos neki dodatni uslov, jer ne postoje podaci o izbornim predmetima i moglo bi samo da se izracuna koliko je studentu ostalo nepolozenih obaveznih predmeta na tom smeru do kraja studija, ali posto ne postoje podaci o izbornim predmetima na pojedinim smerovima, ne bi mogao da se izracuna ukupan broj predmeta...

59. Napisati upit kojim se primarni kljuc tabele dosije definise nad atributom jmbg umesto postojeceg kljuca nad atributom indeks.

```
alter table dosije drop primary key;
alter table dosije add constraint kljuc primary key(jmbg);
```

60. Napisite upit kojim se prikazuju podaci o svim studentima koji su rodjeni u mestu ciji naziv sadrzi najvise jedan blanko karakter i sadrzi 'eog' na 16 poziciji.

```
select *
from dosije
where length(trim(mesto_rodjenja)) - length((replace(trim(mesto_rodjenja), ' ', ''))) <= 1
and substr(mesto_rodjenja, 16, 3) = 'eog';
```

61. Napisati SQL upit koji formira tabelu Dosije2013 koja ima iste atribute kao i tabela Dosije. Prosiriti tabelu atributom prosek koji treba da sadrzi trenutni prosek ocena studenta zaokruzen na dve decimale, i atributom redosled koji sadrzi trenutni redosled tog studenta po proseku u okviru smera koji studira. Upisati u tabelu Dosije2013 podatke koji se odnose na studente upisane 2013. godine i popuniti vrednosti atributa prosek i redosled.

```
create table Dosije2013 like dosije;
```

```
alter table Dosije2013 add constraint kljuc primary key (indeks)

alter table Dosije2013 add prosek dec(4, 2);

alter table Dosije2013 add redosled int;

insert into Dosije2013
with pomocna as
(
select d.indeks, dec(coalesce(avg(ocena * 1.0), 0), 4, 2) prosek
from dosije d left outer join ispit i on d.indeks = i.indeks
and ocena > 5 and status_prijave = 'o'
where year(datum_upisa) = 2013
group by d.indeks
)
select d.*, p.prosek, rank() over(order by p.prosek desc) rbr
from pomocna p join dosije d on p.indeks = d.indeks;
```

62. Formirati pogled polaganje(rbr, ime, prezime, naziv_smera, najvise_puta_polagan_predmet, broj_polaganja, poslednja_ocena, prethodna_ocena) koji sadrzi informacije o predmetu koga je student najvise puta polagao, koliko puta je polagao taj predmet, ocenu na poslednjem polaganju i ocenu na pretposlednjem polaganju. Ukoliko je neki predmet polagan najvise jedan put, tada za ocenu na poslednjem polaganju treba navesti '---'. Da li je preko ovog pogleda moguce azurirati tabelu dosije? Ako jeste, uneti (preko pogleda) u tabelu dosije slog sa imenom i prezimenom studenta Petar Petrovic i brojem indeksa 313/2013. Ukoliko nije, navesti zasto nije i uslove pod kojim u RSUBP DB2 moze da se azurira osnovna tabela preko pogleda koji je definisan nad njim.

```
create view polaganje(rbr, ime, prezime, naziv_smera, najvise_puta_polagan_predmeta,
broj_polaganja, poslednja_ocena, prethodna_ocena) as
(
with student as
(
select i.indeks, i.id_predmeta, count br
from ispit i
group by i.indeks, i.id_predmeta
),
pol as
(
select indeks, id_predmeta, br
from student s
```

```
where br >= all(select br from student s1 where s.indeks = s1.indeks)
)
select ime, prezime, naziv, id_predmeta, br,
(select ocena from ispit i where i.indeks = d.indeks and i.id_predmeta = p.id_predmeta
and broj_polaganja = p.br),
case when br > 1 then (select char(ocena) from ispit i where i.indeks = d.indeks and
i.id_predmeta = p.id_predmeta and broj_polaganja = p.br-1)
else '---'
end
from pol p join dosije d on p.indeks = d.indeks
join smer s on d.id_smera = s.id_smera
);
```

63. Napisati upit kojim se prikazuju podaci o svim studentima koji su rodjeni u mestu ciji naziv sadrzi najvise dva blanko karaktera i sadrzi 'nei' na 10. poziciji, i ne sadrzi % niti \. Pri ispitivanju da li je na 10. poziciji niska 'nei' ne koristiti funkcije SUBSTR, SUBSTRING, SUBSTR2, SUBSTR4, SUBSTRB, kao ni klauzulu LIKE. Naziv mesta rodjenja je pisan bez blanko karaktera na pocetku ili kraju.

```
select *
from dosije d
where instr(mesto_rodjenja, 'nei', 10) = 10
and length(mesto_rodjenja) - length(replace(mesto_rodjenja, ' ', '')) <= 2
and length(mesto_rodjenja) - length(replace(mesto_rodjenja, '%', '')) = 0
and length(mesto_rodjenja) - length(replace(mesto_rodjenja, '\', '')) = 0;
```

64. Napisati SQL upit koji prikazuje komentare koji se odnose na tabele koje je napravio korisnik koji izvrsava upit.

```
select comments from user_tab_comments;
```

65. Napisati SQL upit kojim se formira sinonim Studenti za tabelu Dosije.

```
create synonym Studenti for Dosije;
```

66. Sta je efekat izvrsavanja SQL upita:

```
select ime, prezime, count(distinct id_smera)
from dosije
group by cube (ime, prezime);
```

Napisati jedan SQL upit, bez koriscenja GROUP BY CUBE ili drugih OLAP naredbi, koji prozivodi isti rezultat.

Efekat bi bio da se izdvoje sve moguce kombinacije, sto bi u ovom slucaju bilo da se izracuna broj razlicitih id_smera i grupise po imenu i prezimenu(ne posebno, vec da se izracuna za svaki par imena i prezimena), zatim da se grupise po imenu tj. da se izracuna broj razlicitih id_smera za svako ime pri cemu ce u koloni prezime stajati null, zatim isto tako i za prezime, i na kraju ce biti uzeta u obzir i kombinacija null null. Ovo bi u ovom slucaju kao i za druge OLAP naredbe grupisanja(grouping sets, rollup) moglo da se odradi i sa union all, sto je prikazano ispod.

```
select ime, null, count(distinct id_smera)
from dosije
group by ime

union all

select null, prezime, count(distinct id_smera)
from dosije
group by prezime

union all

select null, null, count(distinct id_smera)
from dosije

union all

select ime, prezime, count(distinct id_smera)
from dosije
group by ime, prezime;
```

67. Napisati deklaraciju tabele izborni_predmet koja sadrzi informacije o studentima koji su se prijavili za pohadjanje pojedinih izbornih predmeta. Tabela treba da sadrzi informacije koje jednoznacno odredjuju prijavljenog studenta, izborni predmet, kao i redosled, vreme i datum prijavljivanja. Redosled prijavljivanja je odredjen redosledom upisa u tabelu, i kao i vreme i datum treba da se upisuju automatski, bez unosenja podataka od strane studenta. U tabeli definisati odgovarajuce primarne i spoljasnje kljuceve. Da li je, na nivou deklaracije tabele, moguce ograniciti broj prijavljenih studenata za pojedini predmet?

```
create table izborni_predmet
(
indeks integer not null,
id_predmeta integer not null,
```

```
rbr_prijave integer not null generated always as identity(start with 1),
vreme time with default current_time,
datum date with default current_date,
primary key(indeks, id_predmeta, rbr_prijave),
foreign key(indeks) references dosije,
foreign key(id_predmeta) references predmet
);
```

Ne znam kako bi moglo da se to ogranici na nivou deklaracije tabele, preko trigera bi lako moglo, na nivou deklaracije tabele moze da se doda check constraint, ali ne znam da li bi na taj nacin mogao da se ogranici broj za pojedini predmet, mogao bi da se ogranici ukupni broj ili tako nesto slicno...

68. Napisati korinsnicki definisanu funkciju isecak(poluprecnik, ugao) ciji su argumenti velicina poluprecnika kruga i broj stepni kruznog isecka, a rezultat povrsina tog isecka zapisana kao realan broj u pokretnom zarezu pomocu dekadne osnove. Vrednosti π ne navoditi u obliku konstante vec kao vrednost neke od skalarnih funkcija.

```
create function isecak(poluprecnik float, ugao float)
returns float
return poluprecnik * poluprecnik * radians(180) * ugao / 360;
```

69. Napisati upit kojim se prikazuje, po korisnicima, broj tabela koje je taj korisnik napravio, broj atributa u tim tabelama, kao i prosecna vrednost i standardna devijacija duzine atributa u tabelama.

```
select char(creator, 40) as korisnik, count(distinct t.name) as br_tabela, sum(t.colcount)
as br_atributa, dec(round(avg(k.length * 1.0),3),12,3) as prosek_duzine_atributa,
dec(round(stddev(k.length * 1.0),3),12,3) as standardna_devijacija_duzine_atributa
from  sysibm.systables as t join sysibm.syscolumns k
on      t.name = k.tbname
group by creator;
```

70. Formirati pogled ocene(indeks, ime, prezime, naziv_smera, najcesce_dobijana_ocena, broj_ocena, poslednji_datum_dobijanja_ocene) koji sadrzi informacije o studentu, ocenu koju je on najcesce dobijao na ispitu, koliko puta je dobio tu ocenu i kada je dobio tu ocenu poslednji put. U obzir uzeti sva polaganja (bez obzira da li je ocena ponistena ili ne).

```
create view ocene(indeks, ime, prezime, naziv_smera, najcesca_dobijana_ocena,
broj_ocena,
                poslednji_datum_dobijanja_ocene) as
```

```
with broj_ocena as
(
select indeks, ocena, count(*) br
from ispit
where ocena is not null
group by indeks, ocena
),
najcesca_ocena as
(
select distinct indeks, br, (select max(ocena) from broj_ocena br_o1 where br_o.indeks
= br_o1.indeks) najcesca
from broj_ocena br_o
)
select distinct d.indeks, ime, prezime, naziv, najcesca, n_o.br, (select
max(coalesce(datum_usmenog, datum_pismenog))
                                                    from ispit i1
                                                        where n_o.indeks = i1.indeks
and najcesca = ocena)
from najcesca_ocena n_o
join dosije d on n_o.indeks = d.indeks
join smer s on d.id_smera = s.id_smera;
```

71. Formirati pogled preostalo(broj_indeksa, ime, prezime, naziv_smera_koji_student_studira, broj_bodova) koji sadrzi informacije o broju bodova koji je preostao svakom od studenata do zavrsetka studija. Napisati i SQL upit kojim se daje dozvola korisnicima STUDENT i NASTAVNIK za brisanje pogleda preostalo uz mogucnost prenosenja dozvole na druge korisnike.

```
create view preostalo (broj_indeksa, ime, prezime, naziv_smera_koji_student_studira,
broj_bodova) as
(
select d.indeks, ime, prezime, s.naziv,
case when sum(p.bodovi) is null then s.bodovi else s.bodovi-sum(p.bodovi) end
from dosije d join smer s on d.id_smera = s.id_smera
left outer join ispit i on d.indeks = i.indeks and ocena > 5 and status_prijave = 'o'
left outer join predmet p on i.id_predmeta = p.id_predmeta
group by d.indeks, ime, prezime, s.naziv, s.bodovi
);

grant delete on table preostalo to STUDENT, NASTAVNIK with grant option;
```

72. Napisati upit kojim se formira pogled mesto_rodjenja(broj_indeksa, ime, prezime,

mesto_rodjenja) koji sadrzi broj indeksa, ime i prezime studenta, JMBG i mesto rodjenja svih studenata koji su upisani na studije 2012. godine ili kasnije i koji su rodjeni u petak 13. Pri tome, u pogled treba izdvojiti samo one studente cije je ime i prezime zapisano velikim pocetnim slovom. Pogled oformiti tako da bude onemogucen unos slogova u tabelu dosije koji ne zadovoljavaju definiciju pogleda.

```
create view mesto_rodjenja(broj_indeksa, ime, prezime, mesto_rodjenja) as
(
select d.indeks, ime, prezime, jmbg, mesto_rodjenja
from dosije d
where year(datum_upisa) >= 2012 and day(datum_rodjenja) = 13 and
dayname(datum_rodjenja) = 'Friday'
and substr(ime,1,1)=upper(substr(ime,1,1))
and substr(prezime,1,1)=upper(substr(prezime,1,1))
)
with check option;
```

73. Jovanca micic je krenuo na put oko sveta relacijom Jagodina-Beograd-Moskva-Peking-Vasington-London-Beograd-Jagodina. Sa sobom je poneo racunar sa instaliranim RSUBP DB2. Na racunaru nije menjao nikakve postavke koje se odnose na vreme i datum. U svakom trenutku tokom puta moze da dodje do podatka o trenutnom lokalnom vremenu i casovnoj zoni, ali ne i podatak koje je trenutno vreme u rodnom gradu. Napisati korisnicki definsanu funkciju koja kao argumente ima casovnu zonu i (lokalno) vreme i koja kao rezultat vraca trenutno vreme u Jagodini.

```
create function vreme (zona integer,vreme time)
returns time
return
values vreme - (zona/10000) HOURS - (mod(zona/100,100)) MINUTES -
mod(zona,100) SECONDS + current timezone;
```

74. Napisati SQL upite pomocu kojih se daje dozvola za formiranje i brisanje spoljasnjih kljuceva nad tabelom ispit svim korisnicima sem korisniku STUDENT.

```
grant references on table ispit to public;
revoke references on table ispit from STUDENT;
```

75. Formirati pogled najlaksi_i_najtezi(naziv_predmeta, prosek_ocena, standardna_devijacija_ocena, broj_studenata_koji_su_polozili_ispit) koji sadrzi informacije o 10 predmeta sa najvecom i 10 predmeta sa najmanjom prosecnom ocenom studenata koji su polozili taj predmet, kao i standardnu devijaciju ocena na tom predmetu ukljucujuci sve dobijene ocene od 5 do 10, kao i broj studenata koja su

polozili taj predmet. Ukoliko neki predmet nije polozlio nijedan student, on treba da se nadje u pogledu pri cemu za prosek ocena i standardnu devijaciju treba da stoji vrednost -1, a za broj studenata tekst 'ispit nije polozio nijedan student'.

```
create view najlaksi_i_najtezi(naziv_predmeta, prosek_ocena,
standardna_devijacija_ocena, broj_studenata_koji_su_polozili_ispit) as
(
with broj as
(
select p.id_predmeta, count(indeks) br
from predmet p left outer join ispit i
on ocena > 5 and status_prijave = 'o' and p.id_predmeta = i.id_predmeta
group by p.id_predmeta
),
predmeti as
(
(
select b.id_predmeta,
case when br = 0 then -1
else (select avg(ocena *  1.0) from ispit i1 where i1.id_predmeta = b.id_predmeta
        and ocena > 5 and status_prijave = 'o')
end prosek,
case when br = 0 then -1
else (select stddev(ocena) from ispit i1 where i1.id_predmeta = b.id_predmeta)
end devijacija,
br brp
from broj b join ispit i on b.id_predmeta = i.id_predmeta
group by b.id_predmeta, br
order by prosek
fetch first 10 rows only
)

union
(
select b.id_predmeta,
case when br = 0 then -1
else (select avg(ocena *  1.0) from ispit i1 where i1.id_predmeta = b.id_predmeta
        and ocena > 5 and status_prijave = 'o')
end prosek,
case when br = 0 then -1
else (select stddev(ocena) from ispit i1 where i1.id_predmeta = b.id_predmeta)
end devijacija,
```

```
br brp
from broj b join ispit i1 on b.id_predmeta = i1.id_predmeta
group by b.id_predmeta, br
order by prosek desc
fetch first 10 rows only
)
)

select naziv, prosek, devijacija,
case when brp =  0 then 'Ispit nije polozio nijedan student'
else char(brp)
end
from predmeti join predmet p on predmeti.id_predmeta = p.id_predmeta
order by prosek
)
```

76. Formirati MQT duplikat koji sadrzi iste podatke kao i pogled definisan u prethodnom pitanju uz uslov da je prosecna ocena povecana za 1 na predmetima koje je polozio bar jedan student. Napuniti tako definisanu tabelu.

```
create table duplikat (naziv, prosek, devijacija, broj) as
(
select naziv_predmeta,
case when prosek_ocena > -1 then prosek_ocena + 1 else prosek_ocena end,
standardna_devijacija_ocena, broj_studenata_koji_su_polozili_ispit
from najlaksi_i_najtezi
)

data initially deferred
refresh deferred;
```

77. Napisati korisnicki definisanu funkciju ciji su argumenti niska maksimalne duzine 30000 i niska maksimalne duzine 10, a rezultat broj pojavljivanja druge niske u prvoj.

```
create function niske(niska1 varchar(30000), niska2 varchar(10))
returns int
return (length(trim(niska1)) - length(replace(trim(niska1), niska2, ''))) /
length(trim(niska2))
```

78. Formirati tabelu ispit_januar koja ima iste atribute kao i tabela ispit. Upisati u formiranu tabelu podatke o polaganim ispitima u januarskom ispitnom roku, bez obzira na godinu polaganja, a zatim definisati primarne i spoljasnje kljuceve tabele ispit_januar.

```
create table ispit_januar like ispit;

insert into ispit_januar
select *
from ispit
where oznaka_roka = 'jan';

alter table ispit_januar
add constraint primarni primary key(indeks, id_predmeta, godina_roka, oznaka_roka)
add    constraint spoljasnji foreign key(indeks) references dosije
add    constraint spoljasnji_1 foreign key(id_predmeta) references predmet
add    constraint spoljasnji_2 foreign key(godina_roka, oznaka_roka) references
ispitni_rok
add    constraint spoljasnji_3 foreign key(indeks, id_predmeta, godina, semestar)
references upisan_kurs;
```

79. Napisati SQL upit kojim se formira sinonim aktivni_studenti za tabelu dosije.datum

```
--pretpostavimo da se misli na dosije, ne znam sta je ovo dosije.datum
create synonym aktivni_studenti for dosije;
```

80. Napisati SQL naredbu kojom se korisniku STUDENT daje dozvola za promenu vrednosti atributa ime i prezime iz prethodnog pitanja. Da li se ta dozvola odnosi i na tabelu dosije? Obrazloziti odgovor.

```
grant update(ime, prezime) on table aktivni_studenti to STUDENT;
```

Odnosi se, jer je to samo drugi naziv za dosije.

81. Napisati SQL upit koji, za sve tabele koje je napravio korisnik STUDENT prikazuje naziv tabele i ukupnu duzinu svih atributa (u bajtovima) u tabeli.

```
select t.name, sum(t1.length)
from sysibm.systables t join sysibm.syscolumns t1 on t.name = t1.tbname
where t.creator = 'STUDENT'
group by t.name;
```

82. Formirati pogled ispisani(ime, prezime, naziv_smera, datum_upisa, naziv_predmeta, prosek_ocena, broj_polozenih_predmeta) koji sadrzi informacije o ispisanim studentima. Ukoliki student nije polozio nijedan predmet, kao broj polozenih ispita treba da stoji tekst 'nije polozio nijedan predmet'.

```
create view ispisani(ime, prezime, naziv_smera, datum_upisa, prosek_ocena,
broj_polozenih_predmeta) as
(
select ime, prezime, s.naziv, datum_upisa, coalesce(avg(ocena * 1.0), 0) prosek,
case
        when count(id_predmeta) = 0 then 'nije polozio nijedan predmet'
        else char(count(id_predmeta))
end
from dosije d join smer s on d.id_smera = s.id_smera
join status st on d.indeks = st.indeks and st.status = 'ispisan'
left outer join ispit i on d.indeks = i.indeks
and ocena > 5 and status_prijave = 'o'
group by d.indeks, ime, prezime, s.naziv, datum_upisa
);
```

83. Formirati MQT ispisani_2015 koja sadrzi podatke kao i pogled definisan u prethodnom pitanju i dodatni atribut rbr koji predstavlja redni broj ispisanog studenta u 2015. godini. Vrednost atributa rbr treba da se automatski poveca za svakog ispisanog studenta.

```
create table ispisani_2015(rbr,ime, prezime, naziv_smera, datum_upisa, prosek_ocena,
broj_polozenih_predmeta) as
(
select rank() over (order by datum desc) rbr,
                i.*
from ispisani i join status s on i.indeks = s.indeks
where year(datum) = 2015
)
data initially deferred
refresh deferred;
```

84. Definisati korisnicki defnisan tip podataka UGAO koji predstavlja mogucu velicinu ugla u trouglu. Defnisiati korisnicki definisanu funkciju ciji su argumenti ostar ugao u pravouglom trouglu i velicina katete naspram tog ugla, a rezultat velicina hipotenuze tog pravouglog trougla. Argument koji predstavlja ugao je tipa UGAO.

```
--ne radi iz nekog razloga kad se pozove funkcija

create type UGAO as double;

create function ostar(ugao UGAO, a double)
```

```
returns double
return a / sin(radians(double(ugao)));
```

85. Napisati upit koji prikazuje nazive svih tabela koje je korisnik koji izvrsava upit napravio u poslednje dve godine i 17 dana.

```
select table_name
from user_tables ut join sysibm.systables t on ut.table_name = t.name
where (current_date - date(substr(ctime, 1, 10))) <= '20017'
```

86. Napisati SQL naredbu kojom se atributu id_smera u bazi menja tip sa celobrojne vrednosti na realan broj sa dekadnom osnovom.

Nejasno je kako bi to moglo u celoj bazi da se uradi i verovatno je to greska ovde i trebalo bi u nekoj tabeli npr. dosije. Ne znam kako bi to moglo da se uradi, a da se ne izgube podaci. Ispod je prikazano neko resenje koje nisam siguran da je bas tacno, ali pretpostavljam da bi nesto ovako slicno trebalo da izgleda.

```
create table dosije_pom like dosije;

insert into dosije_pom
select * from dosije;

alter table dosije_pom
alter column id_smera
set data type decfloat;

drop table dosije;

create table dosije
(
indeks integer not null,
id_smera decfloat not null,
ime varchar(25) not null,
prezime varchar(25) not null,
pol char(1) not null,
jmbg char(13) not null,
datum_rodjenja date,
mesto_rodjenja varchar(100),
drzava_rodjenja varchar(100),
ime_oca varchar(50),
ime_majke varchar(50),
```

```
ulica_stanovanja varchar(100),
kucni_broj varchar(20),
mesto_stanovanja varchar(100),
postanski_broj varchar(20),
drzava_stanovanja varchar(100),
telefon varchar(50),
mobilni_telefon varchar(25),
email varchar(50),
"www uri" varchar(100),
datum_upisa date not null,
primary key (indeks),
foreign key(id_smera) references smer
);

insert into dosije
select * from dosije_pom;

drop table dosije_pom;
```

87. Neka korisnik STUDENT ima dozvolu za citanje kompletne tabele ispit. Napisati naredbe kojima mu se oduzima dozvola za citanje vrednosti atributa semestar i brojpol.

```
revoke select on table ispit from STUDENT;

create view pogled_ispit(indeks, id_predmeta, godina, godina_roka, oznaka_roka,
datum_prijave, nacin_prijave, status_prijave,
datum_pismenog, bodovi_pismenog, datum_usmenog, bodovi_usmenog, bodovi, ocena,
nastavnik, napomena) as
(
select indeks, id_predmeta, godina, godina_roka, oznaka_roka, datum_prijave,
nacin_prijave, status_prijave,
datum_pismenog, bodovi_pismenog, datum_usmenog, bodovi_usmenog, bodovi, ocena,
nastavnik, napomena
from ispit
);

grant select on table pogled_ispit to STUDENT;
```

88. Neka tabela I_godina sadrzi podatke o studentima koji su upisani u I godinu studija. Struktura tabele je ista kao i struktura tabele dosije. Napisati upit koji koriscenjem naredbe MERGE unosi u tabelu dosije podatke o novim studentima. U slucaju da je neki od vec aktivnih studenata ponovo polagao prijemni radi upisa na studije, u vec

postojecim podacima u tabeli dosijea zameniti samo broj indeksa i datum upisa, ali ne i ostale podatke.

```
merge into dosije d using
(
select indeks, id_smera, ime, prezime, pol, jmbg, datum_rodjenja, mesto_rodjenja,
drzava_rodjenja,
      ime_oca, ime_majke, ulica_stanovanja, kucni_broj, mesto_stanovanja,
postanski_broj,
      telefon, mobilni_telefon, email, "www uri", datum_upisa
from I_godina
) as t
on d.indeks = t.indeks
when matched then
        update
        set (d.indeks, d.datum_upisa) = (t.indeks, t.datum_upisa)
when not matched then
        insert
        values(t.indeks, t.id_smera, t.ime, t.prezime, t.pol, t.jmbg, t.datum_rodjenja,
t.mesto_rodjenja,
                t.drzava_rodjenja, t.ime_oca, t.ime_majke, t.ulica_stanovanja,
t.kucni_broj,t.mesto_stanovanja,
                t.postanski_broj, t.telefon, t.mobilni_telefon, t.email, t."www uri",
t.datum_upisa);
```

89. Formirati MQT ime_deo_prezimena koja sadrzi ime, prezime, broj indeksa, smer studija, broj polozenih ispita i poziciju imena u prezimenu, pri cemu vazi da se ime nalazi u prezimenu pocev od navedene pozicije.

```
create table ime_deo_prezimena1 (ime, prezime, indeks, id_smera, broj, poz) as
(
select ime, prezime, indeks, id_smera,
(
select count(id_predmeta)
from ispit i
where ocena > 5 and status_prijave = 'o'
and i.indeks = d.indeks
) broj,
locate(prezime, ime) poz
from dosije d
)
```

data initially deferred
refresh deferred;

90. Napisati upit koji prikazuje tip podatka i ukupnu duzinu svih atributa tog tipa u tabelama koje je napravio korisnik koji izvrsava upit.

```
select coltype, sum(length)
from sysibm.syscolumns
where tbcreator = current_user
group by coltype;
```

91. Napisite naredbu kojom se korisniku STUDENT daje dozvola da azurira atribute ime i prezime u tabeli dosijea.

```
grant update(ime, prezime) on table dosije to STUDENT;
```

92. Formirati pogled upisani(rbr, ime, prezime, naziv, smera, datum_upisa, broj polozenog predmeta) koji sadrzi informacije o studentima koji su upisani 2009. ili 2010. godine, i imaju bar 3 polozena ispita. Atribut rbr automatski dobija vrednost prema datumu upisa studenta (najranije upisani student dobija rbr=1). Kakav je efekat navodjenja WITH CHECK OPTION klauzule u definiciji pogleda? Zasto?

```
create view upisani(rbr, ime, prezime, naziv_smera, datum_upisa,
broj_polozenih_predmeta) as
with pomocna(ime, prezime, naziv_smera, datum_upisa, broj_polozenih_predmeta) as
(select ime, prezime, s.naziv, dat_upisa, count(id_predmeta)
 from     dosije d join smer s
 on       d.id_smera = s.id_smera join ispit i
 on       d.indeks = i.indeks
 where      status_prijave = 'o' and ocena > 5 and year(dat_upisa) in (2009, 2010)
 group by   d.indeks, d.ime, d.prezime, s.naziv, d.dat_upisa
 having     count(id_predmeta) >= 3
)
select rank() over (order by datum_upisa), ime, prezime, naziv_smera, datum_upisa,
broj_polozenih_predmeta
from pomocna p;
```

Navodjenje with check option klauzule u definiciji pogleda ime efekat da slogovi koji se unose u tabelu na kojoj je zasnovan pogled moraju da zadovoljavaju definiciju pogleda.

93. Formirati pogled preostalo_60 koji sadrzi sve informacije o studentima iz tabele dosije kojima je do zavrsetka studija preostalo najvise 60 bodova.

```
create view preostalo_60 as
with preostalo as
(
select d.indeks, case when sum(p.bodovi) is null then s.bodovi else s.bodovi -
sum(p.bodovi) end bodova
from dosije d join smer s on d.id_smera = s.id_smera
left outer join ispit i on d.indeks = i.indeks
and ocena > 5 and status_prijave = 'o'
left outer join predmet p on p.id_predmeta = i.id_predmeta
group by d.indeks, s.bodovi
)
select d.*
from preostalo p join dosije d on p.indeks = d.indeks
where bodova <= 60;
```

94. Napisati upit kojim se, na osnovu pogleda preostalo_60 iz prethodnog pitanja, formira pogled uspeh_po_smerovima koji sadrzi broj indeksa, ime i prezime studenta, identifikaciju smera koji student studira i godinu rodjenja svih studenata koji su upisani na studije juna 2010. godine ili kasnije i kojima je do zavrsetka studija ostalo 60 ili manje bodova. Da li je moguc unos podataka u tabelu dosije preko pogleda uspeh_po_smerovima Obrazloziti odgovor.

```
create view uspeh_po_smerovima(indeks, ime, prezime, id_smera, godina_rodjenja) as
select indeks, ime, prezime, id_smera, year(datum_rodjenja)
from preostalo_60
where datum_upisa >= '01.06.2010';

--Nije moguc unos u tabelu dosije, jer se u from klauzuli referise
--na pogled preostalo_60 koji ne moze da se azurira, jer sadrzi join
```

95. Napisati SQL upit kojim se prikazuju ime, prezime, naziv smera, naziv predmeta, datum polaganja ispita i ocena svih studenata koji su polozili ispit koji je polagan 13og dana u mesecu ciji je naziv "petak".

```
select ime, prezime, s.naziv, p.naziv, coalesce(datum_usmenog, datum_pismenog),
ocena
from ispit i join dosije d on i.indeks = d.indeks
join predmet p on i.id_predmeta = p.id_predmeta
join smer s on d.id_smera = s.id_smera
where ocena > 5 and status_prijave = 'o'
and day(coalesce(datum_usmenog, datum_pismenog)) = 13
```

```
and dayname(coalesce(datum_usmenog, datum_pismenog)) = 'петак';
```

96. Napisati SQL upit kojim se prikazuju, bez koriscenja WITH naredbe, za svaki smer standardna devijacija srednjih ocena studenata koji studiraju na tom smeru.

```
select stddev((select coalesce(avg(ocena * 1.0), 0)
                    from dosije d left outer join ispit i on d.indeks = i.indeks
                    and ocena > 5 and status_prijave = 'o'
                    where d1.indeks = d.indeks and d1.id_smera = d.id_smera
                    group by d.indeks, d.id_smera))
from dosije d1
group by d1.id_smera;
```

97. Napisati sql naredbu kojom se korisniku STUDENT oduzima dozvola za promenu tipa nekog atributa tabele dosije.

```
revoke alter on table dosije form STUDENT;
```

98. Napisati sql naredbu kojom se korisniku STUDENT daje dozvola za brisanje tabele dosije uz mogucnost prenosenja dozvole na druge korisnike.

```
grant control on table dosije to STUDENT with grant option;
```

99. Napisati sql upit kojim se prikazuje kardinalnost i stepen relacije dosije.

```
select count(*) kardinalnost, colcount stepen
from dosije, sysibm.systables
where name = 'DOSIJE'
group by colcount;
```

100. Napisati SQL upit pomocu kojih se korisniku KOR1 daje dozvola da cita atribute broj indeksa i datum rodjenja studenata u tabeli dosije.

```
create view dozvola(indeks, datum_rodjenja) as
select indeks, datum_rodjenja
from dosije;

revoke select on table dosije from KOR1;
grant select on table dozvola to KOR1;
```

101. Napisati SQL upit pomocu kojih se korisniku KOR1 omogucava da brise sadrzaj tabele semestar uz mogucnost prenosenja dozvole na druge korisnike.

grant delete on table semestar to KOR1 with grant option;

102. Formirati pogled kandidati koji sadrzi sve podatke iz tabele dosijea o studentima koji su upisani u prosloj kalendarskoj godini, imaju mobilne telefone i cija glavna strana na sajtu nema adresu koja sadrzi vise od 2 znaka '%'.

```
create view kandidati as
select d.*
from dosije d
where year(datum_upisa) = 2014 and mobilni_telefon is not null
and mobilni_telefon <> ''
and length("www uri") - length(replace(trim("www uri"), '%', '')) <= 2
with cascaded check option;
```

103. Napisati upit kojim se na osnovu podataka u pogledu kandidati iz prethodnog pitanja, formira pogled lokalni_sajt, koji sadrzi broj indeksa, ime i prezime studenta, datum upisa, broj mobilnog telefona, elektronsku adresu sajta svih studenata koji imaju elektronsku adresu na racunaru alas, pri cemu je moguce da su slova u reci alas pisana malim ili velikim slovima. Pogled oformiti tako da bude onemogucen unos slogova u tabelu dosije koji ne zadovoljavaju definiciju oba pogleda.

```
create view lokalni_sajt(indeks, ime, prezime, datum_upisa, mobilni_telefon, email) as
select indeks, ime, prezime, datum_upisa, mobilni_telefon, email
from kandidati
where locate(lower(email), 'alas') <> 0
with cascaded check option;
```

104. Napisati SQL upit kojim se prikazuju ime i prezime studenta, i mesto stanovanja u kome su, ako postoje drugo i 14-to pojavljivanje karaktera '%' zamenjeni niskom '\%'.

```
select ime, prezime,
case
        when locate_in_string(mesto_stanovanja, '%', 1, 2) <> 0
        then insert(mesto_stanovanja, locate_in_string(mesto_stanovanja, '%', 1, 2), 1, '\
%')
        when locate_in_string(mesto_stanovanja, '%', 1, 14) <> 0
        then insert(mesto_stanovanja, locate_in_string(mesto_stanovanja, '%', 1, 14), 1, '\
%')
end
from dosije;
```

105. Napisati SQL upit kojim se prikazuju, bez koriscenja WITH naredbe, imena, prezimena i prosek prvih 5 studenata sa najvecim prosekom koji imaju to ime i prezime. Pri tome su podaci koji se odnose na jednog studenta razdvojeni uspravnom crtom ('|') od podataka za drugog studenta.

```
select listagg(concat(ime_prezime, char(prosek)), '|')
from (select trim(ime) || ' ' || trim(prezime) ime_prezime,
        dec(avg(ocena * 1.0), 4, 2) prosek
        from dosije d join ispit i on d.indeks = i.indeks
        and ocena > 5 and status_prijave = 'o'
        group by ime, prezime
        order by 2 desc
        fetch first 5 rows only);
```

III Funkcionalne zavisnosti i normalizacija

Napomena 1: Pri odredjivanju zatvorenja skupa atributa potrebno je pisati kako se doslo do svakog koraka, npr. A+=A jer A → A ili ako A izvodi B onda je potrebno napisati kako se do toga doslo, na osnovu koje FZ, nije dovoljno napisati samo A+=AB. Ja to nisam pisao ovde, mrzelo me, ali je potrebno to pisati, jer su to ta detaljna objasnjenja koja trazi Mitic.
Napomena 2: Sto se tice nereducibilnog skupa, potrebno je uvek pisati kako se doslo do cega, sto sam manje vise pisao u svakom zadatku.
Napomena 3: Pri odredjivanju kandidata za kljuc, ako se neka promenljiva ne nalazi s desne strane automatski mora da se nadje u kandidatu za kljuc, takodje, ako se ne nalazi s leve strane onda ne moze da se nadje u kandidatu za kljuc. Kada se nadju kandidati za kljuc nadskupove tih kandidata ne treba ispitivati, oni nisu kandidati za kljuc, jer ne ispunjavaju uslov minimalnosti, ali jesu superkljucevi.

106. Neka relacija R={A,B,C,D,E,F,G} sadrzi sledece FZ:
1) AD → BF
2) CD → EGC
3) BD → F
4) E → D
5) F → C
6) D → F.
Odrediti nereducibilni skup funkcionalnih zavisnosti relacije R.

I korak - dekompozicija:
1) AD → B
2) AD → F
3) CD → E
4) CD → G
5) CD → C
6) BD → F
7) E → D
8) F → C
9) D → F
II korak - uklanjanje FZ koje mogu da se izvedu iz drugih FZ:
- uklanjamo FZ 2) AD → F: prosirujemo FZ 9) sa A i dobijamo AD → AF i kada izvrsimo dekompoziciju imamo AD → F
- uklanjamo FZ 5) CD → C: prosirujemo FZ 9) sa C i dobijamo CD → CF i kada izvrsimo dekompoziciju imamo CD → C
- uklanjamo FZ 6) BD → F: prosirujemo FZ 9) sa B i dobijamo BD → BF i kada izvrsimo dekompoziciju imamo BD → F

Sada imamo:
1) $AD \rightarrow B$
2) $CD \rightarrow E$
3) $CD \rightarrow G$
4) $E \rightarrow D$
5) $F \rightarrow C$
6) $D \rightarrow F$
- zatim iz FZ 5) i 6) imamo iz tranzitivnosti $D \rightarrow C$ tako da je C redundantno s leve strane funkcionalnih zavisnosti 2) i 3).
Dakle, konacno resenje je:
1) $AD \rightarrow B$
2) $D \rightarrow E$
3) $D \rightarrow G$
4) $E \rightarrow D$
5) $F \rightarrow C$
6) $D \rightarrow F$

107. Neka relacija R={A,B,C,D,E,H} sadrzi sldece FZ:
1) $CD \rightarrow AB$
2) $C \rightarrow D$
3) $D \rightarrow EH$
4) $AE \rightarrow C$
5) $A \rightarrow C$
6) $B \rightarrow D$
Odrediti nereducibilan skup funkcionalnih zavisnosti relacije R. Obrazloziti korake u radu.

I korak - dekompozicija:
1) $CD \rightarrow A$
2) $CD \rightarrow B$
3) $C \rightarrow D$
4) $D \rightarrow E$
5) $D \rightarrow H$
6) $AE \rightarrow C$
7) $A \rightarrow C$
8) $B \rightarrow D$
II korak - uklanjanje FZ koje mogu da se izvedu iz drugih FZ:
- uklanjamo FZ 6) $AE \rightarrow C$: prosirujemo FZ 7) sa E i dobijamo $AE \rightarrow CE$ i kada izvrsimo dekompoziciju imamo $AE \rightarrow C$

Sada imamo:
1) $CD \rightarrow A$

2) $CD \rightarrow B$
3) $C \rightarrow D$
4) $D \rightarrow E$
5) $D \rightarrow H$
6) $A \rightarrow C$
7) $B \rightarrow D$
- zatim imamo da je D sa leve strane redundantno u FZ 1) i 2) , jer imamo FZ 3)
- iz toga onda imamo da $C \rightarrow B$, pa iz tranzitivnosti sa 7) dobijamo FZ 3) i zato je uklanjamo

Konacno resenje:
1) $C \rightarrow A$
2) $C \rightarrow B$
3) $D \rightarrow E$
4) $D \rightarrow H$
5) $A \rightarrow C$
6) $B \rightarrow D$

108. Dokazati da su sledeca dva skupa funkcionalnih zavisnosti ekvivalentna:
- $A \rightarrow B$ $AB \rightarrow C$ $D \rightarrow AC$ $D \rightarrow E$
- $A \rightarrow BC$ $D \rightarrow AE$

Potrebno je pokazati da se iz prvog skupa FZ mogu izvesti sve FZ iz drugog skupa i obratno.
=>
1) Ako prosirimo $A \rightarrow B$ sa A dobijamo $A \rightarrow AB$, a kako je $AB \rightarrow C$ iz tranzitivnosti dobijamo da je $A \rightarrow C$, a posto i $A \rightarrow B$ to znaci da $A \rightarrow BC$
2) Kada izvrsimo dekompoziciju na $D \rightarrow AC$ dobijamo $D \rightarrow A$, a kako je $D \rightarrow E$ znaci da je $D \rightarrow AE$

<=
1) Kada izvrsimo dekompoziciju na $A \rightarrow BC$ dobijamo $A \rightarrow B$
2) Kada prosirimmo $A \rightarrow BC$ sa B dobijamo $AB \rightarrow BC$ i kada izvrsimo dekompoziciju dobijamo $AB \rightarrow C$
3) Kada prosirmo $A \rightarrow BC$ sa A dobijamo $A \rightarrow ABC$ i kada izvrsimo dekompoziciju dobijamo $A \rightarrow AC$ i iz kada izvrsimo dekompoziciju na $D \rightarrow AE$ dobijamo $D \rightarrow A$ i sada iz tranzitivnosti imamo $D \rightarrow AC$
4) Kada izvrsimo dekompoziciju na $D \rightarrow AE$ dobijamo $D \rightarrow E$

109. Odrediti zatvorenje skupa FZ relacije R={A,B,C,D,E}:
1) $A \rightarrow BC$
2) $CD \rightarrow E$

3) $B \rightarrow D$
4) $E \rightarrow A$
Obrazloziti korake u radu.

5) $A \rightarrow B$ (dekompozicija 1)
6) $A \rightarrow D$ (tranzitivnost 5+3)
7) $A \rightarrow C$ (dekompozicija 1)
8) $A \rightarrow CD$ (unija 6,7)
9) $A \rightarrow E$ (tranzitivnost 8+2)
10) $A \rightarrow ABCDE$ (unija 5,6,7,9 i prosirenje A)
11) $E \rightarrow ABCDE$ (tranzitivnost 4+10)
12) $CD \rightarrow ABCDE$ (tranzitivnost 2+11)
13) $BC \rightarrow CD$ (prosirenje 3 C)
14) $BC \rightarrow ABCDE$ (tranzitivnost 12+13)

Ukljucujuci i relacije koje su projekcije izvedenih relacija.

110. Neka je data relacija R={A,B,C,D,E,F} sa skupom FZ:
1) $A \rightarrow BCD$
2) $BC \rightarrow DE$
3) $B \rightarrow D$
4) $D \rightarrow A$
Primenom Armstrongovih aksioma odrediti zatvorenje skupa FZ relacije R i zatvorenje X+ skupa atributa projekcije X={B,C} relacije R.
Detaljno obrazloziti svaki korak u radu.

Zatvorenje skupa FZ:
5) $A \rightarrow ABCDE$
Dekoompozicijom FZ 1) dobijamo $A \rightarrow BC$ i iz tranzitivnosti sa 3) dobijamo $A \rightarrow DE$ i kada uradimo uniju dobijamo $A \rightarrow BCDE$ i kada prosirimo sa A dobijamo FZ 5).
6) $BC \rightarrow ABCDE$
Dekompozicijom FZ 2) dobijamo $BC \rightarrow D$ i iz tranzitivnosti sa 4) dobijamo $BC \rightarrow A$, zatim napravimo uniju FZ 2) sa tima i imamo $BC \rightarrow ADE$ i kada prosirimo sa BC dobijamo FZ 6).
7) $B \rightarrow ABCDE$
Iz tranzitivnosti 3) i 4) imamo $B \rightarrow A$ i kada prosirimo to sa B i uradimo uniju sa 3) imamo $B \rightarrow ABD$, zatim dekompozicijom 1) dobijamo $A \rightarrow BC$ i iz tranzitivnosti sa 2) imamo $A \rightarrow DE$ i kada izvrsimo dekompoziciju imamo $A \rightarrow E$ i iz tranzitivnosti toga i 4) imamo $D \rightarrow E$ a iz tranzitivnosti 3) i ovoga imamo $B \rightarrow E$ i sada kada uradimo uniju sa $B \rightarrow ABD$ imamo $B \rightarrow ABDE$, takodje iz 1) imamo $A \rightarrow C$ i iz tranzitivnosti toga i 4) imamo $D \rightarrow C$ a iz tranzitivnosti toga i 3) imamo $B \rightarrow C$ i uniju toga i $B \rightarrow ABDE$ nam daje FZ 7).

8) $D \rightarrow ABCDE$
Iz tranzitivnosti 4) i 1) imamo $D \rightarrow BCD$ i kada uradimo uniju toga i 4) imamo $D \rightarrow ABCD$, zatim kada uradimo dekompoziciju FZ 1) i dobijemo $A \rightarrow BC$, iz tranzitivnosti toga i FZ 2) imamo $A \rightarrow DE$ i kada uradimo dekompoziciju imamo $A \rightarrow E$ i iz tranzitivnosti toga i 4) imamo $D \rightarrow E$ i kada uradimo uniju sa $D \rightarrow ABCD$ i toga dobijamo $D \rightarrow ABCDE$

Naravno ovo nisu sve FZ, treba ukljuciti u ovaj skup i sve relacije koje su projekcije izvedenih relacija.

zatvorenje X+ skupa atributa projekcije X={B,C}:
Ovde se trivjalno vidi da je B+ = BC, C+ = C i BC+ = BC

111. Neka je dat relvar R={A,B,C,D,E,G} i skup F FZ:
1) $AB \rightarrow C$
2) $C \rightarrow A$
3) $BC \rightarrow D$
4) $ACD \rightarrow B$
5) $D \rightarrow EG$
6) $BE \rightarrow C$
7) $CG \rightarrow BD$
8) $CE \rightarrow AG$
Odrediti kandidate za kljuc relacije R. Obavezno obrazloziti korake u radu.

I korak	II korak		
A+=A	AB+=ABCDEG	BD+=ABCDEG	DE+=DEG
B+=B	AC+=AC	BE+=ABCDEG	DG+=DEG
C+=AC	AD+=ADEG	BG+=BG	EG+=EG
D+=DEG	AE+=AE	CD+=ABCDEG	
E+=E	AG+=AG	CE+=ABCDEG	
G+=G	BC+=ABCDEG	CG+=ABCDEG	

Kandidati za kljuc su AB, BC, BD, BE, CD, CE, CG.
III korak
Ne ispitujemo trojke koje sadrze kombinacije AB, BC, BD, BE, CD, CE, CG, jer su one superkljucevi.

ADE+=ADEG
ADG+=ADEG
AEG+=AEG
DEG+=DEG

IV korak

Nemamo vise sta da ispitujemo, tako da su kandidati za kljuc AB, BC, BD, BE, CD, CE, CG.

112. R={A,B,C,D,E,F} i skup FZ:
1) $AB \rightarrow D$
2) $B \rightarrow C$
3) $AE \rightarrow B$
4) $A \rightarrow D$
5) $D \rightarrow EF$
Odrediti minimalni pokrivac skupa funkcionalnih zavisnosti i dekomponovati relaciju R tako da novodobijeni skup relacija bude u BCNF. Obavezno obrazloziti korake u radu.

I korak dekompozicija:
1) $AB \rightarrow D$
2) $B \rightarrow C$
3) $AE \rightarrow B$
4) $A \rightarrow D$
5) $D \rightarrow E$
6) $D \rightarrow F$

II korak - uklanjamo FZ koje mogu da se izvedu iz drugih FZ:
- uklanjamo FZ 1) $AB \rightarrow D$: prosirujemo FZ 4) sa B i kada izvrsimo dekompoziciju dobijamo $AB \rightarrow B$
Sada imamo:
1) $B \rightarrow C$
2) $AE \rightarrow B$
3) $A \rightarrow D$
4) $D \rightarrow E$
5) $D \rightarrow F$
- iz FZ 3) i 4) imamo $A \rightarrow E$ tako da je E sa leve strane FZ 2) redundantno
Konacno resenje:
1) $B \rightarrow C$
2) $A \rightarrow B$
3) $A \rightarrow D$
4) $D \rightarrow E$
5) $D \rightarrow F$

Za drugi deo zadatka pogledati miticev slajd o normalizaciji primer 2, tamo je resenje identicnog zadatka.

113. Neka je dat relvar R={A,B,C,D,E,F,G,H} i skup F FZ:
1) $CD \rightarrow A$

2) $EC \rightarrow H$
3) $GHB \rightarrow AB$
4) $C \rightarrow D$
5) $EG \rightarrow A$
6) $H \rightarrow B$
7) $BE \rightarrow CD$
8) $EC \rightarrow B$
Odrediti nereducibilni pokrivac skupa funkcionalnih zavisnosti F i sve kandidate za kljuc relacije R. Obavezno obrazloziti sve korake u radu.

Nereducibilni pokrivac:
I korak - dekompozicija:
1) $CD \rightarrow A$
2) $EC \rightarrow H$
3) $GHB \rightarrow A$
4) $GHB \rightarrow B$
5) $C \rightarrow D$
6) $EG \rightarrow A$
7) $H \rightarrow B$
8) $BE \rightarrow C$
9) $BE \rightarrow D$
10) $EC \rightarrow B$
II korak - uklanjanje FZ koje mogu da se izvedu iz drugih FZ:
- uklanjamo FZ 4) $GHB \rightarrow B$: prosirujemo FZ 7) sa BG i dobijamo $GHB \rightarrow BG$ i kada izvrsimo dekompoziciju imamo $GHB \rightarrow B$
- uklanjamo FZ 9) $BE \rightarrow D$: na osnovu tranzitivnosti iz 8) i 5) imamo $BE \rightarrow D$
- uklanjamo FZ 10) $EC \rightarrow B$: na osnovu tranzitivnosti iz 2) i 7) dobijamo $EC \rightarrow B$
- uklanjamo D sa leve strane FZ 1) $CD \rightarrow A$: prosirujemo $C \rightarrow D$ sa C i dobijamo $C \rightarrow CD$ i onda imamo iz tranzitivnosti toga i 1) $C \rightarrow A$ sto znaci da je D sa leve strane FZ 1) redundantno.
- uklanjamo i B s leve strane FZ 3), jer je i ono redundantno zbog FZ 6)
Konacno resenje:
1) $C \rightarrow A$
2) $EC \rightarrow H$
3) $GH \rightarrow A$
4) $C \rightarrow D$
5) $EG \rightarrow A$
6) $H \rightarrow B$
7) $BE \rightarrow C$

Kandidati za kljuc:

S obizrom da sa desne strane relacije nemamo EFG to znaci da nasi kandidati za kljuc moraju sadrzati EFG, tako da pocinjemo odatle, medjutim, znamo i da u kandidatima za kljuc ne mogu da se nalaze A i D, jer se ne nalaze s leve strane, tako da imamo:
EFG+ = EFGA
EFGB+ = EFGBCAHD
EFGC+ = EFGCBAHD
EFGH+ = EFGHABCD

Vidimo da su kandidati za kljuc EFGB, EFGC, EFGH i dalje nemamo sta da ispitujemo, svi nadskupovi ovih kombinacija ce biti superkljucevi, ali ne i kandidati za kljuc.

114. Neka je dat relvar R={B,C,D,F,G,H} i skup FZ:
1) BG → CD
2) G → F
3) CD → GH
4) C → FG
5) F → D
Odrediti minimalni pokrivac skupa funkcionalnih zavisnosti. Neka je R1={C,D,G} projekcija relacije R. Odrediti skup funkcionalnih zavisnosti koje su vazece u R1. Dekomponovati relaciju R tako da novodobijeni skup relacija bude u BCNF. Obavezno obrazloziti sve korake u radu.

Minimalni pokrivac:
I korak - dekompozicija:
1) BG → C
2) BG → D
3) G → F
4) CD → G
5) CD → H
6) C → F
7) C → G
8) F → D
II korak - uklanjamo FZ koje mogu da se izvedu iz drugih FZ:
- uklanjamo FZ 2) BG → D: iz tranzitivnosti 3) i 8) imamo G → D i kada prosirimo to sa B i izvrsimo dekompoziciju imamo BG → D
- uklanjamo FZ 4) CD → G: kada prosirimo FZ 7) sa D i izvrsimo dekompoziciju imamo CD → G
- uklanjamo FZ 6) C → F: iz tranzitivnosti 7) i 3) imamo C → F

Sada imamo:
1) BG → C

2) $G \rightarrow F$
3) $CD \rightarrow H$
4) $C \rightarrow G$
5) $F \rightarrow D$
- iz tranzitivnosti 4) i 2) imamo $C \rightarrow F$, a zatim iz tranzitivnosti toga i 5) imamo $C \rightarrow D$, tako da je D s leve strane FZ 3) redundantno i uklanjamo ga.
Konacno resenje:
1) $BG \rightarrow C$
2) $G \rightarrow F$
3) $C \rightarrow H$
4) $C \rightarrow G$
5) $F \rightarrow D$

Za projekciju R1 funkcionalne zavisnosti koje su i dalje vazece su trivijalno $C \rightarrow G$, zatim iz tranzitivnosti toga i 2) dobijamo $C \rightarrow F$, a iz tranzitivnosti toga i 5) imamo $C \rightarrow D$. Takodje je i $G \rightarrow D$ vazeca relacija u projekciji, jer nju dobijamo iz tranzitivnosti 2) i 5). Sada kombinovanjem ove 3 relacije mozemo da dobijemo jos neke, ali to nece ovde biti radjeno, jer ih moze biti dosta.

Da bi novodobijeni skup relacije bio u BCNF prvo moramo da nadjemo kandidate za kljuc:
Posto sa desne strane nemamo B krecemo od toga, B se sigurno nalazi u kandidatu za kljuc, ispitujemo kombinacije samo sa F, G i C, jer se nalaze s leve strane:
BG+ = BGCFDH
BC+ = BCGDFH
BF+ = BFD
Ne ispitujemo i ostale dvojke, jer ocigledno da nece biti kandidati za kljuc.
Samo FZ 1) ne narusava skup, medjutim, mozemo da prosirimo relacije 2), 3) i 4) sa B, tako da ni one nece narusavati skup, jer ce se u tom slucaju s leve strane FZ nalaziti neki od kandidata za kljuc, tako da samo FZ 5) narusava skup tako da on nije u BCNF.

Sada radimo razbijanje:
R1 = {F, D}, R2 = {B, C, F, G, H}.

115. Neka je dat relvar R={A,B,C,D} i skup FZ:
1) $A \rightarrow BC$
2) $B \rightarrow C$
3) $A \rightarrow B$
4) $AB \rightarrow C$
5) $AC \rightarrow D$
Odrediti minimalni pokrivac skupa funkcionalnih zavisnosti. Dekomponovati relaciju R

tako da novodobijeni skup relacija bude u BCNF. Obrazloziti korake u radu.

Za minimalni pokrivac pogledati miticev slajd funkcionalne zavisnosti, prvi primer koji je radio.
Da bi skup relacija bio u BCNF potrebno je da nadjemo prvo kandidate za kljuc nakon nereducibilnog skupa FZ.
1) $A \rightarrow B$
2) $B \rightarrow C$
3) $A \rightarrow D$

Ocigledno je A+ = ABCD tako da je A kandidat za kljuc i relacija koja narusava da skup relacija bude u BCNF je FZ 2) i sada vrsimo razbijanje tako da je R1 = {B, C}, a R2 = {A, B, D}.

116. Neka je dat relvar R={A,B,C,D,E,F,G,H,I,J,K,L,M,N} i skup F FZ:
1) $L \rightarrow M$
2) $CK \rightarrow CD$
3) $CKN \rightarrow JABHN$
4) $CK \rightarrow EFGK$
Odrediti kandidate za kljuc relacije R. Dokazati prethodni rezultat koriscenjem funkcionalnih zavisnosti i Armstrongovih aksioma. Detaljno obrazloziti korake u radu. Obavezno navesti koje pravilo je koriscenu u izvodjenju.

Kandidati za kljuc:
Sa desne strane nemamo ILC i odatle pocinjemo
ILC+ = ILCM
ILCK+ = ILCKMDEFG
ILCKN+ = ILCKMDEFGNABHJ

Kandidat za kljuc je ILCKN, kombinacije koje su nadskupovi ove su superkljucevi i vidimo da nemamo vise sta da ispitujemo.

Da bismo dokazali ovo krecemo od ILCKN i treba da pokazemo da se iz njega moze izvesti ILCKMDEFGNABHJ

Posto vazi samo-odredjenje imamo $ILCKN \rightarrow ILCKN$, kada uradimo dekompoziciju imamo $ILCKN \rightarrow CK$ i iz tranzitivnosti sa 2) imamo $ILCKN \rightarrow CD$ i kada uradimo dekomompoziciju imamo $ILCKN \rightarrow D$ i unija toga i pocetne relacije nam daje $ILCKN \rightarrow ILCKND$. Takodje iz tranzitivnosti $ILCKN \rightarrow CK$ i 4) imamo $ILCKN \rightarrow EFGK$ i kada izvrsimo dekompoziciju imamo $ILCKN \rightarrow EFG$ i unija toga i $ILCKN \rightarrow ILCKND$ nam daje $ILCKN \rightarrow ILCKNDEFG$. Kada uradimo dekompoziciju pocetne relacije imamo

i ILCKN → CKN i iz tranzitivnosti toga i 3) imamo ILCKN → JABHN i kada uradimo dekomompoziciju imamo ILCKN → JABH i unija toga i ILCKN → ILCKNDEFG nam daje ILCKN → ILCKNDEFGABHJ. Dekompozicija pocetne relacije nam daje i ILCKN → L i iz tranzitivnosti toga i 1) imamo ILCKN → M i unija toga i ILCKN → ILCKNDEFGABHJ nam daje konacno ILCKN → ILCKNDEFGABHJM.

117. Neka je dat relvar R={A,B,C,D,E,G} i skup F FZ:
1) AB → C
2) C → A
3) BC → D
4) ACD → B
5) D → EG
6) BE → C
7) CG → BD
8) CE → AG
Odrediti minimalan pokrivac skupa FZ relacije R. Detaljno obrazloziti korake u radu.

Videti Miticev prvi primer sa 10. slajda gde su primeri, identican je samo sto je tamo slovo F umesto G.

118. Neka je dat relvar R={A,B,C,D,E} i skup F FZ:
1) A → B
2) AB → C
3) D → AC
4) D → E
Odrediti zatvorenje skupa funkcionalnih zavisnosti F+ primenom Armstrongovih aksioma. Obavezno obrazloziti sve korake u radu.

Isti postupak kao i u zadatku 95, gomila novih relacija ce se dobiti...

119. Ispitati koji od sledecih skupova funkcionalnih zavisnosti je ekvivalentan skupu F iz prethodnog pitanja:
1) A → BC, D → E, A → D
2) A → BC, D → AE
3) AB → C, D → E
4) Nijedan od prethodnih skupova
Obavezno obrazloziti sve korake u radu.

1) skup nije zato sto se ne moze izvesti iz prvog skupa A → D
2) skup jeste ekvivalentan
dokaz:
=>

1) Prosirimo FZ 1) sa A i dobijamo $A \rightarrow AB$ i iz tranzitivnosti sa 2) dobijamo $A \rightarrow C$ i posto imamo $A \rightarrow B$ kada izvrsimo kompoziciju dobijamo $A \rightarrow BC$
2) Izvrsimo dekompoziciju FZ 3) i imamo $D \rightarrow A$ i kako je $D \rightarrow E$ kada izvrsimo kompoziciju dobijamo $D \rightarrow AE$
<=
1) Posto imamo $A \rightarrow BC$ kada izvrsimo dekompoziciju dobijamo $A \rightarrow B$
2) Posto imamo $A \rightarrow BC$ kada prosirimo sa B i izvrsimo dekompoziciju dobijamo $AB \rightarrow C$
3) Posto je $D \rightarrow AE$ kada izvrsimo dekompoziciju imamo $D \rightarrow A$ i $D \rightarrow E$ i iz tranzitivnosti $D \rightarrow A$ i $A \rightarrow BC$ dobijamo $D \rightarrow BC$ i kada izvrsimo dekompoziciju imamo $D \rightarrow C$ i kada izvrsimo kompoziciju sa $D \rightarrow A$ dobijamo $D \rightarrow AC$
4) Kada izvrsimo dekompoziciju $D \rightarrow AE$ dobijamo $D \rightarrow E$

120. Neka je dat relvar R={A,B,C,D,F} i skup FZ:
1) $A \rightarrow BC$
2) $C \rightarrow AD$
3) $E \rightarrow ABC$
4) $F \rightarrow CD$
5) $CD \rightarrow BEF$
6) $AB \rightarrow D$
Odrediti minimalni pokrivac skupa funkcionalnih zavisnosti. Dekomponovati relaciju R tako da novodobijeni skup relacija bude u BCNF. Obavezno obrazloziti sve korake u radu.

I korak - dekompozicija:
1) $A \rightarrow B$
2) $A \rightarrow C$
3) $C \rightarrow A$
4) $C \rightarrow D$
5) $E \rightarrow A$
6) $E \rightarrow B$
7) $E \rightarrow C$
8) $F \rightarrow C$
9) $F \rightarrow D$
10) $CD \rightarrow B$
11) $CD \rightarrow E$
12) $CD \rightarrow F$
13) $AB \rightarrow D$
II korak - uklanjamo FZ koje mogu da se izvedu iz drugih FZ
- uklanjamo FZ 6) $E \rightarrow B$: iz tranzitivnosti FZ 5) i FZ 1) imamo $E \rightarrow B$
- uklanjamo FZ 7) $E \rightarrow C$: iz tranzitivnosti FZ 5) i FZ 2) imamo $E \rightarrow C$
- uklanjamo FZ 10) $CD \rightarrow B$: iz tranzitivnosti FZ 3) i FZ 1) imamo $C \rightarrow B$ i kada prosirimo to sa D i izvrsimo dekompoziciju imamo $CD \rightarrow B$

- uklanjamo FZ 13) AB → D: iz tranzitivnosti FZ 2) i FZ 4) imamo A → D i kada prosirimo to sa B dobijamo AB → BD i posle dekompozicije imamo AB → D
- uklanjamo FZ 9) F → D: iz tranzitivnosti 8) i 4) imamo F → D

Sada imamo:
1) A → B
2) A → C
3) C → A
4) C → D
5) E → A
6) F → C
7) CD → E
8) CD → F

S obzirom da C → D to znaci da je D redundantno s leve strane FZ 7) i 8) tako da sada imamo:
1) A → B
2) A → C
3) C → A
4) C → D
5) E → A
6) F → C
7) C → E
8) C → F
-sada mozemo da uklonimo i FZ 3), jer nju dobijamo iz tranzitivnosti FZ 7) i 5)
i konacno resenje je:
1) A → B
2) A → C
3) C → D
4) E → A
5) F → C
6) C → E
7) C → F

Trazimo kandidate za kljuc ovog skupa FZ
Ocigledno je A+ = ABCDEF, C+ = ABCDEF, E+ = ABCDEF, F+ = ABCDEF, dok je B+ = B i D+ = D, BD+ = BD, kandidati za kljuc s A, C, E ili F, sve vece kombinacije koje sadrze neko od ovih slova su superkljucevi. S obzirom da nema atributa s leve strane koji nisu kandidati za kljuc ovaj skup je vec u BCNF.

121. Neka je dat relvar R={A,B,C,D,E,F,G,H} i skup F FZ:
1) BE → GH

2) G → FA
3) D → C
4) F → B
Odrediti minimalan kljuc relacije R. Ispitati da li je relacija R u BCNF. Ako nije, dekomponovati relaciju R tako da novodobijeni skup relacija bude u BCNF. Obavezno obrazloziti sve korake u radu.

Minimalan kljuc:
S obzirom da sa leve strane nemamo DE krecemo odatle:
DE+ = DEC
DEB+ = DEBCGHFA
DEF+ = DEFBGHCA
DEG+ = DEFBGHCA

Minimalni kandidati za kljuc su DEB, DEF i DEG.

Relacija ocigledno nije u BCNF i sve zavisnosti narusavaju da relacija bude u BCNF. Medjutim, ako bismo prosirili FZ 1) sa D, FZ 2) sa DE, FZ 3) sa EF, FZ 4) sa DE dobicemo skup FZ koji je vec u BCNF. (Mitic kaze da ovo moze ovako!)

122. Neka je dat relvar R={A,B,C,D,E,G,H} i skup FZ:
1) A → CD
2) B → AB
3) AC → E
4) DE → B
5) CG → H
6) C → G
Odrediti minimalni pokrivac skupa funkcionalnih zavisnosti i sve kandidate za kljuc relacije R. Obavezno obrazloziti sve korake u radu.

Kandidati za kljuc:
I korak
A+=ABCDEGH
B+=ABCDEGH
C+=CGH
D+=D
E+=E
G+=G
H+=H
Kandidati za kljuc su A i B i u narednim koracima necemo razmatrati nikakve kombinacije koje sadrze A i B, jer su one superkljucevi.
II korak

CD+=CDGH
CE+=CEGH
CG+=CGH
CH+=CGH
DE+=ABCDEGH
DG+=DG
DH+=DH
EG+=EG
EH+=EH
GH+=GH
Kandidat za kljuc je DE i u narednim koracima necemo razmatrati nikakve kombinacije koje sadrze DE, jer su one superkljucevi.
III korak
CDG+=CDGH
CDH+=CDGH
CEG+=CEGH
CEH+=CEGH
DGH+=DGH
EGH+=EGH
IV korak
CDGH+=CDGH
CEGH+=CEGH
Kandidati za kljuc su A, B, DE.

Minimalni pokrivac:
I korak: dekomompozicija
1) $A \rightarrow C$
2) $A \rightarrow D$
3) $B \rightarrow A$
4) $B \rightarrow B$
5) $AC \rightarrow E$
6) $DE \rightarrow B$
7) $CG \rightarrow H$
8) $C \rightarrow G$
II korak: uklanjamo FZ koje mogu da se izvedu iz drugih FZ:
- uklanjamo FZ 4) $B \rightarrow B$ jer ona trivijalno vazi
- uklanjamo C sa leve strane FZ 5) $AC \rightarrow E$: kada prosirimo 1) sa A dobijamo $A \rightarrow AC$ i iz tranzitivnosti sa 5) imamo $A \rightarrow E$, pa C sa leve strane FZ 5) postaje redundantno.
- uklanjamo G sa leve strane FZ 7): kada prosirimo 8) sa C imamo $C \rightarrow CG$ i iz tranzitivnosti sa 7) imamo $C \rightarrow H$, pa G sa leve strane FZ 7) postaje redundantno.
Konacno resenje:
1) $A \rightarrow C$

2) A → D
3) B → A
4) A → E
5) DE → B
6) C → H
7) C → G

123. Neka je data relaciona promenljiva R={A,B,C,D,E,F,G} i skup F FZ:
1) ABC → DE
2) AB → D
3) DE → ABCF
4) E → C
Odrediti nereducibilni pokrivac skupa funkcionalnih zavisnosti F i sve kandidate za kljuc relacije R. Obavezno obrazloziti sve korake u radu. Navodjenje samo rezultata pojedinih koraka nece biti priznato kao delimicno uradjen zadatak.

Nereducibilni pokrivac:
I korak - dekompozicija:
1) ABC → D
2) ABC → E
3) AB → D
4) DE → A
5) DE → B
6) DE → C
7) DE → F
8) E → C
II korak - uklanjanje FZ koje mogu da se izvedu iz drugih FZ:
- uklanjamo FZ 1) ABC → D: prosirujemo 3) sa C i dobijamo ABC → DC i kada izvrsimo dekompoziciju imamo ABC → D
- uklanjamo FZ 6) DE → C: prosirujemo 8) sa D i dobijamo DE → CD i kada izvrsimo dekompoziciju imamo DE → C
Konacno resenje:
1) ABC → E
2) AB → D
3) DE → A
4) DE → B
5) DE → F
6) E → C

Kandidati za kljuc:
S desne strane nemamo nigde G, tako da kandidat za kljuc mora sadrzati G. S leve strane nemamo nigde F, tako da kandidat za kljuc nece sadrzati F, nema svrhe ispitivati nijednu

dvoclanu kombinaciju osim DE, jer jedino ona odredjuje B i dobijamo
DEG+=ABCDEFG
Dalje, ispitivanjem drugih kombinacija dobili bismo ABEG+=ABCDEFG i ABCG+=ABCDEFG i kandidati za kljuc su DEG, ABEG, ABCG.

124. Neka je data relaciona promenljiva R={A,B,C,D,E} i skup FZ:
1) AB → C
2) DE → C
3) B → D
Odrediti sve funkcionalne zavisnosti koje onemogucavaju da relacije R bude u BCNF. Dekomponovati relaciju R tako da novodobijeni skup relacija bude u BCNF. Navesti sve funkcionalne zavisnosti koje nisu ocuvane u prethodnoj dekompoziciji. Obavezno obrazloziti sve korake u radu. Navodjenje samo rezultata pojedinih koraka nece biti priznato kao delimicno uradjen zadatak.

Prvo odredjujemo kandidate za kljuc:
Posto sa desne strane nemamo A, B i E krecemo od ABE i vidimo da je ABE+ = ABCDE i ocigledno je da je to jedini kandidat za kljuc i sada dekomponujemo, posto sve relacije narusavaju da skup bude u BCNF:
R1 = {A, B, C}, Rx = {A, B, D, E}
R2 = {B, D}, R3 = {A, B, E}
Funkcionalna zavisnost koja nije ocuvana u ovoj dekompoziciji je DE → C.

125. Neka je data relaciona promenljiva R={A,B,C,D,E,F,G,H,I,J} i skup F FZ:
1) HD → A
2) D → EFG
3) HDB → CJ
4) H → IJ
5) A → BC
Odrediti nereducibilni pokrivac skupa funkcionalnih zavisnosti F. Dekomponovati relaciju R tako da novodobijeni skup relacija bude u BCNF. Navesti sve funkcionalne zavisnosti koje nisu ocuvane u prethodnoj dekompoziciji. Obavezno obrazloziti sve korake u radu. Navodjenje samo rezultata pojedinih koraka nece biti priznato kao delimicno uradjen zadatak.

Nereducibilni pokrivac:
I korak - dekompozicija:
1) HD → A
2) D → E
3) D → F
4) D → G
5) HDB → C

6) HDB → J
7) H → I
8) H → J
9) A → B
10) A → C
II korak - uklanjamo FZ koje mogu da se izvedu iz drugih FZ:
- uklanjamo FZ 5) HDB → C: iz tranzitivnosti 1) i 10) imamo HD → C i kada to prosirimo sa B i uradimo dekompoziciju imamo HDB → C
- uklanjamo FZ 6) HDB → J: prosirimo 8) sa DB i uradimo dekompoziciju i imamo HDB → J
Konacno resenje:
1) HD → A
2) D → E
3) D → F
4) D → G
5) H → I
6) H → J
7) A → B
8) A → C

Odredjujemo kandidate za kljuc: S obzirom da sa desne strane nemamo D i H vidimo da je DH+ = ABCDEFGHIJ i to je kandidat za kljuc i sve relacije osim prve narusavaju skup relacija tako da on nije u BCNF. Medjutim, prosirivanjem FZ 2), 3), 4) sa H i FZ 5), 6) sa D dobijamo da samo FZ 7) i 8) onemogucavajuda skup FZ bude u BCNF. Sada radimo razbijanje:
R1 = {A, B}
R2 = {A, C}
R3 = {A, D, E, F, G, H, I, J}

126. Neka je data relaciona promenljiva R={A,B,C,D,E,F,G,H} i skup S FZ:
1) AB → C
2) AC → B
3) AD → E
4) B → D
5) BC → A
6) E → G
Odrediti nereducibilni pokrivac funkcionalnih zavisnosti S i sve kandidate za kljuc relacije R. Obavezno obrazloziti sve korake u radu. Navodjenje samo rezultata pojedinih koraka nece biti priznato kao delimicno uradjen zadatak.

Ovde je skup FZ vec nereducibilan.

Posto sa desne strane nemamo F i H krecemo od FH. Medjutim, sa leve strane nemamo kombinaciju tih slova i vidi se da nijedna trojka ne moze biti kandidat za kljuc, a ocigledno je i da A i jos neko slovo mora da se nadje u kandidatu za kljuc zbog toga sto tako mozemo doci do C, B, E ili da se BC nadje u kandidatu, jer vodi do A.
FHAB+ = FHABCDEG
FHAC+ = FHACBDEG
FHBC+ = FHBCADEG
FHAE+ = FHAEG
Nema svrhe da ispitujemo jos i neku petorku, ocigledno je da su kandidati za kljuc FHAB, FHAC i FHBC, a nista dalje ne ispitujemo, jer sve vece kombinacije koje sadrze ove su superkljucevi.

127. Prikazati funkcionalne zavisnosti koje trenutno postoje u studentskoj bazi podataka. Ispitati da li je skup tih funkcionalnih zavisnosti nereducibilan, i ako nije odrediti nereducibilan skup. Ispitati da li su relacije u studentskoj bazi u BCNF. Obavezno obrazloziti sve korake u radu. Navodjenje samo rezultata pojedinih koraka nece biti priznato kao delimicno uradjen zadatak.

???

128. Neka je data relaciona promenljiva R = {A, B, C, D, E, F, G, H} i skup F FZ:
1) $CD \rightarrow A$
2) $EC \rightarrow H$
3) $GHB \rightarrow AB$
4) $C \rightarrow D$
5) $EG \rightarrow A$
6) $H \rightarrow B$
7) $BE \rightarrow CD$
8) $EC \rightarrow B$
Odrediti nereducibilni pokrivac skupa funkcionalnih zavisnosti F i sve kandidate za kljuc relacije R. Obavezno obrazloziti sve korake u radu. Navodjenje samo pojedinih koraka (npr. zatvorenja skupa atributa bez objasnjenja kako se do njega doslo) nece biti priznato kao delimicno uradjen zadatak.

I korak - dekompozicija:
1) $CD \rightarrow A$
2) $EC \rightarrow H$
3) $GHB \rightarrow A$
4) $GHB \rightarrow B$
5) $C \rightarrow D$
6) $EG \rightarrow A$
7) $H \rightarrow B$

8) BE → C
9) BE → D
10) EC → B
II korak - uklanjamo FZ koje mogu da se izvedu iz drugih FZ:
- uklanjamo FZ 9) EC → B: iz 2) i 7) zbog tranzitivnosti sledi EC → B.
- uklanjamo FZ 4) GHB → B: prosirimo FZ 7) sa GB i dobijamo BGH → BG i kada izvrsimo dekompoziciju imamo GHB → B
- uklanjamo FZ 9) BE → D: iz tranzitivnosti FZ 8) i 5) imamo BE → D.
Sada imamo:
1) CD → A
2) EC → H
3) GHB → A
4) C → D
5) EG → A
6) H → B
7) BE → C
S obzirom da C → D to znaci da je D redundantno s leve strane FZ 1). Slicno, zbog FZ 6) H → B, to znaci da je B redundantno s leve strane FZ 3).
Konacno resenje:
1) C → A
2) EC → H
3) GH → A
4) C → D
5) EG → A
6) H → B
7) BE → C

Sada trazimo kandidate za kljuc:
Posto s desne strane nemamo E, F, G znaci da odatle krecemo da trazimo kljuc. Takodje, u kandidatu za kljuc se ne mogu nalaziti A i D, jer ih nemamo s leve strane.

EFG+ = EFGA
EFGB+ = EFGBCHAD
EFGC+ = EFGCDHBA
EFGH+ = EFGHABCD

Nema svrhe da ispitujemo nijednu vecu kombinaciju, jer ce ona sigurno sadrzati neki od kandidata za kljuc. Dakle, kandidati za kljuc su EFGB, EFGC, EFGH.

129. Neka je data relaciona promenljiva R = {A, B, C, D, E, F, G} i skup F FZ:
1) AB → CF
2) BG → C

3) AEF → C
4) ABG → ED
5) CF → AE
6) A → CG
7) AD → FE
8) AC → B
Odrediti sve kandidate za kljuc relacije R i nereducibilni pokrivac skupa funkcionalnih zavisnosti F. Obavezno obrazloziti SVE korake u radu. Navodjenje samo rezultata pojedinih koraka (npr. zatvorenja skupa atributa bez objasnjenja kako se do njega doslo) nece biti priznato kao delimicno uradjen zadatak.
Prvo cemo traziti nereducibilni pokrivac:
I korak - dekomompozicija:
1) AB → C
2) AB → F
3) BG → C
4) AEF → C
5) ABG → E
6) ABG → D
7) CF → A
8) CF → E
9) A → C
10) A → G
11) AD → F
12) AD → E
13) AC → B
II korak - uklanjamo FZ koje mogu da se izvedu iz drugih FZ:
- uklanjamo FZ 1) AB → C: kada prosirimo FZ 9) sa B imamo AB → CB i posle dekompozicije imamo AB → C.
- uklanjamo FZ 4) AEF → C: kada prosirimo FZ 9) sa EF imamo AEF → CEF i posle dekompozicije imamo AEF → C.
- vidi se i da je C s leve strane FZ 13) redundantno, jer vazi FZ 9) A → C.
Sada imamo:
1) AB → F
2) BG → C
3) ABG → E
4) ABG → D
5) CF → A
6) CF → E
7) A → C
8) A → G
9) AD → F
10) AD → E

11) $A \rightarrow B$
- sada vidimo da je B redundantno s leve strane FZ 1), 3) i 4), jer vazi FZ 11) $A \rightarrow B$.
- takodje je i G redundantno s leve strane FZ 3) i 4), jer vazi FZ 8) $A \rightarrow G$.
- uklanjamo FZ 7) $A \rightarrow C$: vidimo da iz unije 11) i 8) vazi $A \rightarrow BG$ i iz tranzitivnosti sa FZ 2) imamo $A \rightarrow C$.
Sada imamo:
1) $A \rightarrow F$
2) $BG \rightarrow C$
3) $A \rightarrow E$
4) $A \rightarrow D$
5) $CF \rightarrow A$
6) $CF \rightarrow E$
7) $A \rightarrow G$
8) $AD \rightarrow F$
9) $AD \rightarrow E$
10) $A \rightarrow B$
- sada zbog $A \rightarrow D$, D s leve strane FZ 8) i 9) postaje redundantno, a posto to znaci da dobijamo $A \rightarrow E$ i $A \rightarrow F$, ali njih mozemo odmah ukloniti, jer se vec nalaze u skupu.
- uklanjamo FZ 6) $CF \rightarrow E$: iz tranzitivnosti FZ 5) i 3) imamo $CF \rightarrow E$.
Konacno resenje:
1) $A \rightarrow F$
2) $BG \rightarrow C$
3) $A \rightarrow E$
4) $A \rightarrow D$
5) $CF \rightarrow A$
6) $A \rightarrow G$
7) $A \rightarrow B$

Trazimo kandidate za kljuc:
Sa leve strane nemamo nigde D, E tako da kombinacije koje ukljucuju njih necemo ispitivati.
A+=ABCDEFG
Necemo ispitivati dvojke koje sadrze A, jer su one superkljucevi.
BC+=BC
BF+=BF
BG+=BGC
CF+=ABCDEFG
CG+=CG
FG+=FG
Ne ispitujemo trojke koje sadrze A ili CF, jer su one superkljucevi.
BCG+=BCG
BFG+=BFGCADE

Kandidati za kljuc su A, CF, BFG.

130. Neka ja dat relvar R = {A, B, C, D, E, F, G} i skup F FZ:
1) $AD \rightarrow BF$
2) $CD \rightarrow ECG$
3) $BD \rightarrow F$
4) $E \rightarrow D$
5) $F \rightarrow C$
6) $D \rightarrow F$
Dekomponovati relaciju R tako da novodobijeni skup relacije bude u BCNF. Da li postoje zavisnosti (i ako postoje koje su) koje nisu ocuvane u procesu dekomompozicije u BCNF?
Obavezno obrazloziti SVE korake u radu. Navodjenje samo rezultata pojedinih koraka nece biti priznato kao delimicno uradjen zadatak.

Prvo cemo naci nereducibilni pokrivac:
I korak - dekompozicija:
1) $AD \rightarrow B$
2) $AD \rightarrow F$
3) $CD \rightarrow E$
4) $CD \rightarrow C$
5) $CD \rightarrow G$
6) $BD \rightarrow F$
7) $E \rightarrow D$
8) $F \rightarrow C$
9) $D \rightarrow F$
II korak - uklanjamo FZ koje mogu da se izvedu iz drugih FZ:
- uklanjamo FZ 2) $AD \rightarrow F$: kada prosirimo FZ 9) sa A i izvrsimo dekompoziciju imamo $AD \rightarrow F$.
- uklanjamo FZ 6) $BD \rightarrow F$: kada prosirimo FZ 9) sa B i izvrsimo dekompoziciju imamo $BD \rightarrow F$.
- uklanjamo FZ 4) $CD \rightarrow C$: iz tranzitivnosti FZ 9) i 8) imamo $D \rightarrow C$ i kada to prosirimo sa C imamo $CD \rightarrow C$.
- s obziram da iz tranzitivnosti FZ 9) i 8) imamo $D \rightarrow C$, odatle sledi da je C sa leve strane FZ 3) i 5) redundantno.
Sada imamo:
1) $AD \rightarrow B$
2) $D \rightarrow E$
3) $D \rightarrow G$
4) $E \rightarrow D$
5) $F \rightarrow C$

6) $D \rightarrow F$

Sada trazimo kandidate za kljuc:
Posto sa desne strane nemamo A, kandidat za kljuc mora sadrzati A, a sa leve strane nemamo nigde A jednoclano tako da jednoclane kombinacije ne proveravamo. Takodje sa leve strane nemamo ni B, C, G, tako da necemo proveravati kombinacije koje ih sadrze.
AD+=ADBEGCF
AE+=AEDBFGC
AF+=AFC
Kandidati za kljuc su AD i AE.
Vidimo da sve FZ osim 1) onemogucavaju da skup bude u BCNF. Medjutim, mzoemo da prosirimo FZ 2), 3), 4) i 6) sa A i onda nece narusavati skup i ostaje samo FZ 5) koja onemogucava da skup bude u BCNF. Sada radimo dekomomponovanje:
R1 = {F, C}
R2 = {A, B, D, E, F, G}

IV Teorija - dodatak

131. Sta je sistem baza podataka?

To je u osnovi sistem za racunarsko zapisivanje i cuvanje slogova, tj. sistem cija je svrha da cuva informacije i dozvoli korisniku da te informacije dobije i azurira po zelji.

132. Sta su entiteti i odnosi?

Entiteti su objekti koji su na neki nacin prikazani u bazi podataka i mozemo ih definisati kao objekte o kojima zelimo da sakupljamo informacije. Odnosi predstavljaju vezu izmedju dva ili vise entiteta i oni takodje moraju na neki nacin biti predstavljeni u bazi. Dijagram koji predstavlja entitete i njihove odnose se zove E/R dijagram.

133. Sta je model podataka?

To je apstraktna, samostalna logicka definicija objekata, operatora, itd. koji cine apstraktnu masinu sa kojom korisnici komuniciraju. Implementacija datog modela podataka je fizicka realizacija na realnoj masini sa svim ciniocima apstraktne masine koji prave taj model.

134. Na kom modelu podataka je zasnovan sql?

Na relacionom modelu gde su podaci prikazani kao redovi u tabelama i operatori vrse operacije na tim redovima i kreiraju nove tabele.

135. Opisati korake u pristupu bazi.

1) Korisnik ispostavlja zahtev (npr. SQL upit)
2) SUBP prihvata zahtev i analizira ga
3) Da bi odredio potrebne operacije SUBP proverava spoljasnju shemu korisnika, odgovarajuce spoljasnje/konceptualno preslikavanje, konceptualnu shemu, konceptualno/unutrasnje preslikavanje i definicije memorijskih struktura.
4) SUBP izvrsava potrebne operacije (tj. zahtev korisnika) nad bazom.

136. Kako se upravlja prenosom podataka u bazi?

Zahtevi korinsnika se prenose od mesta nastanka do SUBP-a preko komunikacionih poruka; rezultati se takodje vracaju istim putem. Prenosom poruka upravlja deo softvera koji se naziva komunikacioni upravljac podataka (DC manager). DC manager nije deo SUBP-a; oni rade zajedno formirajuci DB/DC sistem.

137. Opisati klijent-server arhitekturu i distribuiranu obradu.
SBP moze da se posmatra kao da ima dve komponente: server (u sustini SUBP) i klijent(ispostavlja zahteve serveru). Klijent-server arhitektura je omogucena i u distribuiranoj obradi. Kod distribuirane obrade isti posao se (delom) izvrsava na razlicitim racunarima koji su spojeni u mrezu i skoro da se upotrebljava kao sinonim za klijent/server. Podaci iz jedne baze mogu da budu smesteni na dva ili vise servera i klijent moze da pristupi do vise servera.

138. Sta je relaciona algebra i koja je njena svrha?

Algebra je formalni matematicki sistem koji se sastoji od skupa objekata i operacija nad tim objektima. Za definiciju formalnog sistema je potrebno prikazati sintaksu, dati semantiku i pravila izvodjenja dokaza. Relaciona algebra je familija algebri sa dobro zasnovanom semantikom koja se koristi za modeliranje relacija (objekata) smestenih u relacionoj bazi podataka i za definisanje upita na njima. U sustini predstavlja skup operatora ciji su operandi i rezultati relacije. Prvu verziju je dao Codd 1972. i kasnije je prosirivana od strane raznih autora. Njena svrha je pisanje relacionih izraza koji se koriste za:
- definisanje prostora za dohvatanje podataka
- definisanje prostora za azuriranje podataka
- definisanje pravila integriteta
- definisanje izvedenih relacija
- definisanje pravila zastite
- ...

139. Sta je relacioni racun? Opisati ukratko razliku izmedju relacionog racuna torki i domena?

Relacioni racun je opisan, neproceduralan jezik. Ako se posmatra kao deo relacionog modela za obradu podataka, mozemo reci da je relacioni racun logicki ekvivalent relacione algebre. Zasnovan je na predikatskom racunu, postoje kvantifikatori FORALL i EXISTS i kvantifikacija i rad sa slobodnim i vezanim promenljivm su u skladu sa pravilima predikatskog racuna. Postoje dve varijante racuna:
- racun orijentisan ka torkama
- racun orijentisan ka domenima koji je osnova za QBE
Kod relacionog racuna torki, promenljiva torki ima opseg iz skupa navedenih relacija i dopustene vrednosti koje pripradaju torkama iz tih relacija. Kod relacionog racuna domena, promenljiva domena ima opseg iz skupa navedenih domena i dopustene vrednosti koje pripadaju tim domenima. Koriscenjem promenljivih torki traze se torke za koji je predikat tacan. Opseg vazenja promenljivih kod relacionog racuna domena su domeni, a ne relacije i moguce 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 tacan akko postoji torka u relaciji R takva da je za svaki konkretan par poredjenje A = x tacno.

140. Detaljno opisati karakteristike relacione baze.

Relacioni sistemi zahtevaju da se baza prikaze korisniku u obliku tabele (nacin smestanja i cuvanja na medijumu nije specificiran). Informacioni princip: Celokupan informacioni kontekst baze se prikazuje na tacno jedan nacin kao eksplicitne vrednosti u pozicijama vrsta i kolona tabele. Posledica informacionog principa je da nema pokazivaca koji medjusobno povezuju tabele i oni mogu da postoje na fizickom nivou, a razlika u odnosu na nerelacione modele je u tome sto su tamo pokazivaci deo struktura koje se prikazuju korisnicima i koje korisnik koristi u obradi.
Rezultat primene svakog od operatora je tabela - osobina zatvorenja relacionih sistema, a rezultat primene operatora je istog tipa kao i njegov argument => mogu da se pisu ugnjezdeni relacioni izrazi. Sve operacije se primenjuju na ceo skup istovremeno, a ne samo na pojedinacni red i rezultat operacije nije nikada pojedinacni red vec je uvek kompletna tabela koja sadrzi skup redova.

141. Opisati formalni pogled na relacioni model.

On se sastoji od:
- Otvorenog skupa skalarnih tipova (koji ukljucuje i tip logickih vrednosti boolean)
- Generatora relacionih tipova i njihove odgovarajuce interpretacije
- Mogucnosti definisanja relacionih promenljivih za generisane relacione tipove
- Operacija relacione dodele kojom se dodeljuju relacione vrednosti definisanim relacionim promenljivama
- Otvorenog skupa opstih relacionih operatora ("relaciona algebra") za izvodjenje relacionih vrednosti iz drugih relacionih vrednosti.

142. Ukratko opisati postupak optimizacije.

Relacioni jezici su neproceduralni i za relacioni sistem se cesto kaze da vrsi automatsku "navigaciju" (nad sacuvanim podacima). U nerelacionim sistemima korisnik je odgovoran za "navigaciju". Odluka kako vrsiti automatsku navigaciju spada u domen komponente SUBP nazvane optimizator. U opstem slucaju optimizator odredjuje strategiju na osnovu razmatranja:
- koji relvar-i su referencirani u zahtevu i koliko su oni veliki
- koji indeksi postoje
- kako su podaci fizicki smesteni na disku
- ...

143. Sta je katalog?

Katalog je mesto gde se cuvaju sve sheme i odgovarajuca preslikavanja izmedju njih. Te detaljne informacije koje se nalaze u katalogu se zovu metadata koja je neophodna da bi SBP radio kako treba, npr. optimizator koristi informacije iz kataloga o indeksima i drugim strukturama, kao i mnoge druge informacije koje mu mogu pomoci da implementira korisnicke zahteve. Sistem za autorizaciju koristi informacije iz kataloga o korisnicima i bezbednosnim ogranicenjima da bi dozvolio ili zabranio takve zahteve.

144. Sta su osnovni relvar-i i pogledi?

Relvar predstavlja relacionu promenljivu, a relacija specificnu relacionu vrednost. Relacioni sistemi moraju da obezbede nacin koriscenja osnovnih relvara, a pored toga osnovni relvari moraju biti imenovani (npr. TABLE, koji je osnovni relvar u SQL-u). Pored ovog, relacioni sistem podrzava i drugaciju vrstu imenovanih relvara koja se naziva pogled cija je vrednost u svakom trenutku neka izvedena relacija. Vrednost pogleda je (trenutni) rezultat izvrsavanja odredjenog relacionog izraza koji se navodi pri formiranju pogleda. Pogledi predstavljaju razlicit nacin gledanja na podatke, dok osnovni relvari predstavljaju podatke koji su fizicki smesteni u bazi podataka. Sistem pretvara upit pri formiranju pogleda u ekvivalentan upit nad osnovnim relvar-ima. Pretvaranje se vrsi supstitucijom koja je moguca na osnovu osobine relacionog zatvorenja.

145. Navesti relacione domene koje podrzava SQL.

- Brojevi (numbers)
- Niske karaktera (character strings)
- Niske bitova (bit strings)
- Datumi (dates)
- Vremena (times)
- Kombinacija datuma i vremena (timestamps)
- Intervali godina/mesec (year/month intervals)
- Intervali dan/vreme (day/time intervals)

146. Sta je relacija i koje su njene osobine?

Neka je dat skup od n tipova ili domena Ti (i = 1,2,...,n), pri cemu ne moraju svi tipovi da budu medjusobno razliciti. R je relacija nad tim tipovima ako se sastoji iz dva dela, zaglavlja i tela gde vazi:
1) Zaglavlje je skup od n atributa oblika Ai: Ti gde su Ai (koji svi moraju da budu razliciti) imena atributa relacije R, a Ti odgovarajuca imena tipova (i = 1,2,...,n)
2) Telo je skup od m torki t gde je t skup komponenti oblika Ai : vi u kojima je vi

vrednost tipa Ti.
m se naziva kardinalnost, a n stepen (arnost) relacije R. Osobine relacija su:
- Nema ponovljenih (duplih torki)
- Torke su neuredjene, od vrha ka dnu
- Atributi su neuredjeni, s leva u desno
- Svaka torka sadrzi tacno jednu vrednost za svaki atribut. Za relaciju koja zadovoljava ovu osobinu se kaze da je normalizovana, odnosno da je u prvoj normalnoj formi.
Prve tri osobine su upravo one po kojima se relacija razlikuje od tabele, jer kod tabela ima ponavljanja redova i redovi i kolone su uredjeni.
Za relaciju koja ima prazan skup torki vazi da ima neprazno zaglavlje, a prazno telo, a relacija koja ima praznu torku ima prazno zaglavlje i telo s jednom torkom bez komponenti.
Svaki atribut ima neki tip koje moze biti ugradjen ili korinsnicki definisan, atomski (skalarni) ili ucauren (nije skalaran). Atributi relacije mogu da budu proizvoljnog tipa.

147. Ukratko opisati nedostajuce vrednosti i njihovu svrhu.

U svakodnevnoj praksi se cesto javlja problem nedostataka podataka i postoji potreba da se indikator o nedostatku vrednosti cuva u bazi i da se na odgovarajuci nacin vrsi obrada takvih podataka. Najcesci pristup prihvacen i u praksi je koriscenje 'nedostajuce vrednosti' (NULL) odnosno trovalentne (3VL) logike koja se sastoji od 3 vrednosti: tacno, netacno i nepoznato iako je Codd predlozio koriscenje 4-valentne logike, jer postoje dve vrste nedostajucih vrednosti: vrednost je nepoznata ili vrednost nije promenljiva, vrednost ne postoji, ...

148. Sta je integritet baze podataka? Kako glasi zlatno pravilo?

Pojam integritet se u kontekstu baza podataka odnosi na preciznost, punovaznost i korektnost podataka u bazi. Odrzavanje integriteta podataka je od najvece vaznosti za RSUBP, zbog toga se u sistemu definisu pravila (tzv. ogranicenja integriteta) koja se primenjuju na podatke. Intuitivno, ogranicenje integriteta je logicki izraz pridruzen bazi za koga se zahteva da njegovo izracunavanje uvek daje vrednost tacno, ogranicenja se proveravaju pri formiranju objekata u bazi ili menjanju njihovog sadrzaja. Zlatno pravilo glasi: Ni jednoj operaciji azuriranja nije dozvoljeno da ostavi bilo koji relvar u stanju koje narusava bilo koje od ogranicenja tog relvar-a.

149. Kako se biraju kandidati za kljuceve? Koje vrste kljuceva postoje? Ukratko ih opisati.

Kandidat za kljuc relacije predstavlja podskup atributa X te relacije, ako vazi:
- Pravilo jedinstvenosti: ne postoje dve torke u relaciji R koje imaju iste vrednosti za X, i
- Pravilo minimalnosti: ne postoje pravi podksup skupa X koji zadovoljava pravilo

jedinstvenosti.
Vrste kljuceva su:
- Primarni kljuc: jedan od kandidata za kljuc
- Alternativni kljucevi: ostali kandidati
- Spoljasnji (strani) kljuc: skup atributa jednog relvar-a R2 cije vrednosti treba da odgovaraju vrednostima nekog kandidata za kljuc nekog relvar-a R1
- Superkljuc: nadskup kandidata za kljuc; poseduje jedinstvenost, ali ne i minimalnost.

150. Opisati referencijalni integritet.

Osnovna ideja ocuvanja integriteta u ovom slucaju je da sve vrednosti u tabelama treba da budu usaglasene. Spoljasnji kljuc predstavlja referencu na torku koji sadrzi odgovarajuci primarni kljuc. Odatle je problem osiguravanja da baza podataka ne sadrzi pogresne spoljasnje kljuceve poznat kao problem referencijalnog integriteta, a ogranicenja koja to omogucavaju se nazivaju referencijalna ogranicenja. Relacija koja sadrzi primarne kljuceve se naziva roditelj relacija, a relacija koja sadrzi spoljasnje kljuceve koji se referisu na roditelj relaciju se naziva dete relacija. Referencijalni integritet glasi: Baza ne sme da sadrzi neuparene vrednosti spoljasnjih kljuceva. Relvar-i koji nemaju kandidate za kljuc (tj. sadrze duple slogove) se ponasaju nepredvidivo u pojedinim situacijama. Sistem koji ne poseduje znanje o kandidatima za kljuc ponekad pokazuje karakteristike koje nisu "cisto relacione".

151. Sta su tvrdnje, a sta okidaci?

Tvrdnja je logicka vrednost koja mora uvek da bude ispunjena, a okidac je niz akcija koje su pridruzene odredjenim dogadjajima i koji se izvrsavaju svaki put kada se takav dogadjaj dogodi. Implementacije RSUBP ne podrzavaju tvrdnje, ali podrzavaju okidace.

152. Navesti osobine i funkcije pogleda. Sta je logicka nezavisnost podataka?

Osobine pogleda su da definicija pogleda kombinuje spoljasnju semu, preslikavanje izmedju spoljasnjeg i konceptualnog nivoa i spoljasnje/spoljasnje preslikavanje. Funkcije su da obezbedjuje automatsku zastitu za skrivene podatke, da razliciti korisnici (istovremeno) vide iste podatke na razlicite nacine, da uproscavaju slozene operacije (slicno makroima u programskim jezicima) i omogucuju logicku nezavisnost podataka. Logicka nezavisnost podataka omogucuje da prosirenjem relacija baze ne sme da ima efekat na izvrsavanje aplikativnih programa i da restruktuiranje baze ne sme da ime efekat na postojece aplikativne programe. Nova i stara baza treba da budu informaciono ekvivalentne.

153. Kako se vrsi azuriranje pogleda?

Azuriranje pogleda ne sme da narusi ogranicenja integriteta nad pogledima, ogranicenja integriteta nad pogledima su izvedena iz ogranicenja integriteta osnovnih relvar-a. Ako je D baza, V pogled nad D i X funkcija nad D kojom se definise pogled V, tada za dati pogled V = X(D) i operaciju azuriranja U nad V, potrebno je odrediti operaciju azuriranja U1 nad D tako da vazi U(X(D)) = X(U1(D)). Postoje dva pristupa pri azuriranju pogleda:
- Codd-ov pristup: definisanje pogleda koji mogu da se azuriraju.
- Date-in pristup: svi pogledi mogu da se azuriraju. Operacije azuriranja se izvode izbegavanjem ogranicenja integriteta u medjukoracima azuriranja, azuriranje se izvodi kao brisanje postojecih i unosenje novih podataka.
SQL/92 podrska za azuriranje pogleda je vrlo ogranicena i jedini pogledi koji se smatraju mogucim za azuriranje su pogledi koji su izvedeni iz jedne osnovne tabele preko kombinacija restrikcije i projekcije. U SQL/92 pogled moze da se azurira ako vazi:
- izraz kojim se definise pogled je select izraz koji ne sadrzi JOIN, UNION, INTERSECT ili EXCEPT;
- select klauzula ne sadrzi kljucnu rec DISTINCT;
- svaka select stavka sadrzi (kvalifikovano) ime koje predstavlja referencu na kolonu osnovne tabele;
- from klauzula sadrzi referencu na tacno jednu tabelu koja je ili osnovna tabela ili pogled koji moze da se azurira;
- where klauzula select izraza ne sadrzi podupit u kome se from klauzula referise na istu tabelu kao i from klauzula u select izrazu na najvisem nivou;
- select izraz ne sadrzi group by niti having klauzulu.

154. Sta je normalizacija?
Normalizacija je proces zamene relacija skupom relacija koje su u pogodnijem obliku. Svrha normalizacije je izbegavanje redundantnosti i pojedinih anomalija azuriranja. U procesu normalizacije operator projekcije se vise puta primenjuje na datu relaciju na takav nacin da spajanjem projekcija moze da se dodje do pocetne relacije. Na taj nacin, proces normalizacije je reverzibilan i cuva informacije, tj. uvek je moguce da se uzme izlaz iz procesa i preslika unatrag do ulaza.

155. Sta su nedostaci trece normalne forme?
1) ima vise od jednog kandidata za kljuc
2) kandidat za kljuc je kompozitan
3) kompozitni kandidati za kljuceve se preklapaju
Ovi slucajevi su obuhvaceni Bojs-Kodovom normalnom formom.

156. Opisati 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 proojekcije relvara u 3NF radi eliminisanja preostalih FZ u kojima na levoj strani nije kandidat za kljuc. 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 elminisati ZS koje ne slede iz kandidata za kljuc(eve). Dobijeni skup relvara je u 5NF.
U praksi se cesto ne sprovodi puna normalizacija zbog dobrih performansi, jer puna normalizacija dovodi do velikog broja logicki razdvojenih relvar-a. Takodje, veliki broj razdvojenih relvar-a znaci veliki broj razdvojenih datoteka u kojima se cuvaju, a veliki broj datoteka znaci veliki broj U/I operacija. U praksi se normalizacija najcesce sprovodi do 3NF.