ODGOVORI:

1 **(2 boda)** Što je to model podataka i od se kojih komponenti sastoji. Definirajte bazu podataka. Koje vrste informacija u osnovi sadržava svaka baza podataka? Koji modeli podataka se koriste u DBMS-ovima za oblikovanje baze podataka? Nabrojite nekoliko relacijskih DBMS-ova.

Odgovor

MODEL PODATAKA *je* ***formalni sustav*** *koji mora imati barem sljedeće tri komponente:*

a) ***Skup objekata*** *koji su osnovni elementi baze podataka;*

b) ***Skup operacija*** *koje možemo izvoditi nad objektima definiranim pod* ***a)*** *i kojima se mogu pretraživati, dobivati i modificirati podaci o tim objektima;*

c) ***Skup općih pravila integriteta*** *podataka koja implicitno ili eksplicitno definiraju skup konzistentnih stanja podataka ili promjena stanja, ili oboje i koja su općenita u smislu da su primjenjiva na bilo koju bazu podataka koja koristi taj model.*

Baza podataka je centralno mjesto **informacijskog sustava**. Pohranjeni podaci u bazi podataka opisuju **trenutno** stanje dijela realnog svijeta za koji je i razvijen informacijski sustav, naravno na način pogodan za računalnu obradu.

BAZA PODATAKA *je* ***skup*** *međusobno* ***povezanih podataka*** *pohranjenih* ***bez nepotrebne zalihosti*** *s ciljem da na optimalni način posluže u raznim primjenama. Podaci se spremaju neovisno o programima koji ih koriste, zajedničkim pristupom dodaju se novi podaci te mijenjaju i premještaju postojeći.*

Svaka **baza podataka** sadrži u osnovi **dvije vrste informacija**:

- **opis *ENTITETA***,
- **prikaz odnosa** između ***ENTITETA***.

Način na koji prikazujemo **opis entiteta** i njihove međusobne **odnose** ovisi o odabranom **modelu podataka**, odnosno **DBMS**-u koji podržava taj model podataka.

Ovisno o modelu podataka koji podržavaju, **DBMS**-ove dijelimo na

- hijerarhijske **DBMS**-ove,
- mrežne **DBMS**-ove,
- relacijske **DBMS**-ove (**RDBMS**-ove),
- prošireno - relacijske **DBMS**-ove (**RDBMS**-ove),
- objektno-relacijske **DBMS**-ove (**ORDBMS**-ove),
- objektno-orijentirane **DBMS**-ove (**OODBMS**-ove).

Relacijski DBMS-ovi: DB2, INFORMIX, INGRES, MICROSOFT, SQL/SERVE, ORACLE, SYBASE, SUPRA.

2 **(2 boda)** Koja je uloga Sustava za potporu poslovanja (ERP) u okviru Integriranog informacijskog sustava tvrtke? Koje osnovne poslovne funkcije mora omogućiti?

Odgovor

Sustav za potporu poslovanja (ERP - *Enterprice Resource Planning,* u daljnjem tekstu ERP) je sustav koji se sastoji od odgovarajućih aplikacija i baze podataka. U doslovnom prijevodu ERP znači planiranje resursa poduzeća, međutim naziv sustav za potporu poslovanja puno bolje odgovara njegovoj funkciji nego izravan prijevod. U praksi se udomaćio izraz ERP sustav.

ERP mora omogućiti obavljanje sljedećih osnovnih poslovnih funkcija:

- Računovodstvo i financije,
- Upravljanje ljudskim potencijalom,
- Upravljanje osnovnim sredstvima,
- Upravljanje nabavom i prodajom.

Za obavljanje odgovarajućih funkcija nužna je i pridružena baza podataka. Ona predstavlja operativnu transakcijsku bazu i zato se sustavi gdje je dominantna transakcija nazivaju OLTP (*On Line Transaction Processing*). Logika transakcijskog sustava izgrađena je tako da obrađuje poslovne promjene i da prati odvijanje pojedinih poslovnih funkcija. U bazi podataka pohranjeni su podaci koji opisuju trenutno stanje poslovnih funkcija tvrtke. Takva se baza naziva i produkcijska baza podataka tvrtke.

3 (**3 boda**) Što je to apstrakcija? Objasnite postupke generalizacije, klasifikacije i agregacije. Postupkom klasifikacije i generalizacije nacrtajte generičku strukturu za sljedeće pojave objekata: Automobili: **CITROEN C4 ZG 2345-A, RENAULT CLIO ZG 7786-C, FORD FOCUS SP 3333-B;** Autobusi: **SETRA ZG 4343-AB, IVECO RI 4444-B**

Odgovor

APSTRAKCIJA *je, općenito,* ***zanemarivanje*** *određenih aspekata ili svojstava nekog objekta promatranja, koji nisu važni za svrhu objekta u određenom kontekstu. Ona je nužna u procesu spoznavanja svijeta, a predstavlja* ***uočavanje glavnog***, ***općeg***, ***nužnog*** *ili* ***bitnog*** *te hotimično ispuštanje sporednog, posebnog, slučajnog ili nebitnog – ona je proces teorijskog uopćavanja.*

KLASIFIKACIJA *je vrsta* ***apstrakcije*** *kod koje se stvarni ili apstraktni objekti (entiteti) identificiraju, opisuju i grupiraju u* ***klase*** *(tipove) prema* ***zajedničkim svojstvima*** *(obilježjima). Ona se opisuje vezom "****jest pojava****". Obrnuti postupak od klase (tipa) objekta (entiteta) prema njegovim pojavama nazivamo* **INSTANTACIJOM**.

GENERALIZACIJA *je vrsta* ***apstrakcije*** *kod koje se uspostavlja veza između* ***više klasa objekata*** *(tipova entiteta)* ***niže razine apstrakcije*** *i* ***klase objekta*** *(tipa entiteta)* ***više razine apstrakcije****. To je apstrahiranje detalja od najniže do najviše razine zajedničkih svojstava, a označava se vezom "****jest****". Klase više razine su* ***superklase*** *(nadtipovi) klasa niže razine (****podklasa****, podtipova). Obrnuti postupak od generalizacije naziva se* **SPECIJALIZACIJOM**.

AGREGACIJA *je postupak* ***apstrakcije*** *gdje se* ***formira*** *novi pojam višeg stupnja na temelju* ***odnosa postojećih pojmova****. Ona se koristi kao:*

- ***Agregacija jednostavnih atributa,*** *čime se opisuje entitet (objekt),*
- ***Agregacija entiteta*** *(objekata) u novi entitet (objekt).*

Motorna vozila ***(generalizacija)***

Automobili **Autobusi** ***(klasifikacija)***

CITROEN C4 ZG 2345-A RENAULT CLIO ZG 7786-C FORD FOCUS SP 3333-B **SETRA ZG 4343-AB IVECO RI 4444-B**

4 (**1 bod**) Navedite osnovna ***pravila integriteta*** u relacijskom modelu podataka.

Odgovor

Osnovna ***pravila integriteta*** u relacijskom modelu podataka dijelimo na:

- ***OGRANIČENJA STRUKTURE*** (engleski: *structural integrity constraints*),
  - Ograničenje domene
  - Ograničenje "NUL" vrijednosti
  - Ograničenje jedinstvenosti ključa
  - Referencijsko ograničenje
-
- ***OGRANIČENJA PONAŠANJA*** (engleski: *behavioral integrity constraints*)
  - FUNKCIJSKA OVISNOST,
  - VIŠEZNAČNA OVISNOST,
  - SPOJNA OVISNOST.

5 (**2 boda**) Zadana je relacijska shema R(A,B,C,D,E,F,G), gdje je AB primarni i jedini ključ. Osim funkcijskih ovisnosti neključnih atributa o primarnom ključu, uočene su i sljedeće funkcijske ovisnosti:

B→C,
E→A,
A→D
G→F,
F ↛G

Vaš je zadatak da izvršite vertikalnu normalizaciju dekompozicijom isključivo do 3NF i prikažete dobivene relacijske sheme (konačnim relacijskim shemama treba podcrtati primarni ključ).

Postupak rješavanja:

Analizom uočenih funkcijskih ovisnosti zaključujemo:

B→C parcijalna funkcijska ovisnost neključnog atributa o ključu, parcijalna funkcijska ovisnost je specijalni slučaj tranzitivne funkcijske ovisnosti, naime vrijedi AB →B (trivijalna funkcijska ovisnost) B –/→AB, B →C

E→A tranzitivna funkcijska ovisnost ključnog atributa o ključu, uklanjanje je bitno za ***BOYCE-CODDOVU*** normalnu formu

A→D parcijalna funkcijska ovisnost neključnog atributa o ključu, parcijalna funkcijska ovisnost je specijalni slučaj tranzitivne funkcijske ovisnosti, naime vrijedi AB →A (trivijalna funkcijska ovisnost) A –/→AB, A →D

G→F tranzitivna ovisnost neključnog atributa o ključu

Postupak normalizacije:

Druga normalna forma

R1(B, C)

R2(A, D)

R3(A,B,E,F,G)

Treća normalna forma:

R1(B, D)

R2(A, C)

R31(G, F)

R32(A,B,E,G)

6 (**2 boda**) Što znate o Coddovim kriterijima za potpune RDBMS-ove (zbog čega su doneseni, od kojih se grupa pravila sastoje i koliko pravila ima u svakoj grupi; ne treba nabrajati pojedinačna pravila za svaku grupu) ?

Odgovor

U početku razvoja relacijskih sustava nad hijerarhijske i mrežne modele se dodao relacijski sloj (programska podrška koja je oponašala relacijski model). Ti sustavi su prema korisnicima imali sve prednosti relacijskog modela u smislu postavljanja upita, ali su bili vrlo spori zbog starog pristupa pohranjenim podacima koji su koristili mrežni i hijerarhijski model. Govorilo se da je relacijski model dobar, ali spor i da se može koristiti samo za male baze podataka. Da bi osigurao daljnji razvoj relacijskog modela i promijenio prethodni stav, ***Codd*** je 1985. godine definirao kriterije (pravila) koje bi morao zadovoljavati **DBMS** da bi spadao u kategoriju **potpunog relacijskog sustava**:

- **dvanaest osnovnih pravila**,
- **osamnaest manipulacijskih pravila** (svojstava),
- **devet strukturnih pravila** (svojstava)
- **tri pravila integriteta**.

RDBMS mora zadovoljiti tzv. "***nulto***" pravilo da bi bilo uopće spadao u kategoriju relacijskog modela podataka:

P0 ***OSNOVNO PRAVILO*** *(Foundation Rule)* - Svaki sustav koji se proglašava relacijskim mora moći u potpunosti upravljati bazom podataka kroz svoje relacijske sposobnosti. Operacije nad podacima provode se kao operacije nad skupom zapisa.

Ostalo dodatak koji se ne traži u odgovoru

DVANAEST OSNOVNIH PRAVILA su:

P1 ***PRAVILO O INFORMACIJAMA*** *(Information Rule)* - Sve informacije u relacijskoj bazi eksplicitno su prikazane na logičkoj razini na točno određeni način i to kao vrijednosti u tablicama. Također i imena tablica, imena atributa/kolona te imena domena prikazani su u katalogu. Katalog je aktivan, dinamički te prikazuje metapodatke (podatke koji opisuju podatke u bazi). Katalog odražava pravo stanje u bazi te pomaže pri održavanju baze podataka.

P2 ***PRAVILO O PRISTUPU*** *(Guaranteed Access Rule)* - Svaki podatak koji se nalazi u relacijskoj bazi može se logički dohvatiti uz pomoć imena tablice, imena kolone te vrijednosti primarnog ključa.

P3 ***PRAVILO O SISTEMATIČNOM PRIKAZU INFORMACIJA KOJE NEDOSTAJU*** *(Systematic treatment of null values) (Missing Information Rule)* - Za prikaz informacija koje ne postoje, koristi se "nul" vrijednost. "Nul" vrijednost je nezavisna od tipa podataka, različita od nule, različita od praznog niza i različita od niza praznina.

P4 ***PRAVILO O DINAMIČKOM INTERAKTIVNOM KATALOGU BAZIRANOM NA RELACIJSKOM MODELU*** *(Dyinamic On-line Catalog Based on the Relational Model)* - Opisi podataka su na logičkoj razini prikazani na isti način kao i "obični"

podaci. Korisnici s pravom pristupa mogu nad katalogom primjenjivati isti relacijski upitni jezik kao i nad običnim podacima. U sustavu postoji samo jedan jezik i samo jedan model za pretragu i pohranu podataka.

P5 *PRAVILO O SVEOBUHVATNOM JEZIKU* *(Comprehensive Data Sublanguage Rule)* - Relacijski sustav može podržavati različite jezike i različite načine korištenja podataka, ali barem jedan od njih mora biti sveobuhvatan, odnosno mora omogućavati: **definiciju podataka, definiciju pogleda, rukovanje s podacima interaktivno i uz pomoć programa, definiranje posebnih korisniku prilagođenih pravila integriteta, kontrolu pristupa - autorizaciju, određivanje granica transakcija.**

P6 *PRAVILO O IZMJENAMA KROZ POGLEDE* *(View Updating Rule)* - Sustav omogućava izmjene kroz poglede (virtualne tablice - views) u svim situacijama u kojima su izmjene teorijski moguće. Teorijski je moguća izmjena kroz pogled ukoliko postoji vremensko nezavisan algoritam za nedvosmisleno određivanje jednog niza odgovarajućih izmjena na osnovnim tablicama.

P7 *PRAVILO O OPERACIJAMA NAD PODACIMA* *(High-level Insert, Update and Delete)* - Prilikom dodavanja, izmjena i brisanja s osnovnim ili virtualnim tablicama rukuje se kao s jednim operandom. Ovo pravilo omogućava optimizaciju vremena izvođenja, omogućava sustavu da sam izabere pristupne puteve.

P8 *PRAVILO O FIZIČKOJ NEZAVISNOSTI PODATAKA* *(Physical Data Independence)* - Promjene fizičkog prikaza podataka i promjene pristupnih puteva ne smiju uzrokovati promjene u aplikacijskim programima.

P9 *PRAVILO O LOGIČKOJ NEZAVISNOSTI PODATAKA* *(Logical Data Independence)* - Promjenom strukture osnovnih tablica bez gubitka informacija ne mijenjaju se aplikacijski programi i terminalske aktivnosti.

P10 *PRAVILO O NEZAVISNOSTI INTEGRITETA* *(Integrity Independence)* - Pravila integriteta definiraju se uz pomoć relacijskog jezika i pohranjuju se u katalog, a ne u aplikacijske programe.

P11 *PRAVILO O NEZAVISNOSTI DISTRIBUCIJE* *(Distribution Independence)* - Relacijski sustav omogućava da aplikacijski program i terminalske aktivnosti ostanu nepromijenjene prilikom prve distribucije podataka i svih kasnijih redistribucija. Ukoliko su podaci distribuirani, tada jedna transakcija može zahtijevati podatke s različitih udaljenih stanica. Sustav mora omogućiti obnovu na različitim stanicama, fleksibilnost dekompozicije, mogućnost rekompozicije, optimizaciju prijenosa i mogućnost analize zahtjeva.

P12 *PRAVILO O NENARUŠAVANJU*

(Non Subversion Rule) - Ako relacijski sustav posjeduje jezik niže razine koji omogućava pojedinačni dohvat zapisa, taj jezik ne može narušiti ili zaobići pravila integriteta i ograničenja definirana uz pomoć relacijskog jezika više razine.

DEVET STRUKTURNIH SVOJSTAVA:

S1. *RELACIJE*, odnosno ekvivalentno ***TABLICE*** s imenovanim stupcima gdje poredak redova i stupaca nije bitan i gdje ne postoje ponavljajuće grupe.

S2. ***BAZNE TABLICE*** - prikazuju spremljene podatke.

S3. ***TABLICE UPITA*** - rezultat bilo kojeg upita je druga tablica, koja može biti sačuvana i kasnije ponovo korištena.

S4. ***TABLICE POGLEDA*** - virtualne tablice koje se interno prikazuju s jednom ili više relacijskih komandi, ali ne i sa spremljenim podacima. Definirane komande nad baznim i izvedenim relacijama izvedu se tek nakon poziva pogleda.

S5. ***TABLICE SNIMAKA*** *(Snapshoot tables)* - tablice koje su procijenjene vrijednima i spremljene u bazi podataka zajedno s ulazom u katalog, specificiranim datumom, vremenom kreiranja i opisom.

S6. ***ATRIBUTI*** - svaki stupac bilo koje relacijske tablice je atribut.

S7. ***DOMENA*** - skup vrijednost iz koje jedan ili više stupaca ostvaruje svoje vrijednosti.

S8. ***PRIMARNI KLJUČ*** - svaka bazna tablica ima jedan ili više stupaca čije vrijednosti jednoznačno identificiraju jedan red u tablici. Primarni ključ osigurava svojstvo jedinstvenog adresiranja u relacijskom modelu koje je implementacijski, programski i sklopovski nezavisno.

S9. ***STRANI KLJUČ*** - bilo koji stupac u bazi podataka koji ima istu domenu kao i primarni ključ neke bazne relacije. Strani ključ omogućava realizaciju referencijskog integriteta bez uvođenja vidljivih linkova za korisnike ili programere.

TRI PRAVILA INTEGRITETA:

I1. ***ENTITETSKI INTEGRITET***,

I2. ***REFERENCIJSKI INTEGRITET***,

I3. ***KORISNIČKI DEFINIRAN INTEGRITET*** *(User-defined integrity)*.

OSAMNAEST MANIPULACIJSKIH SVOJSTAVA:

M1. ***THETA RESTRIKCIJA*** *(Theta select)*,

M2. ***PROJEKCIJA***,

M3. ***THETA SPAJANJE***,

M4. ***VANJSKO THETA-SPAJANJE*** *(Outer theta join)*,

M5. ***DIJELJENJE***.

M6. ***UNIJA***,

M7. ***PRESJEK*****,**

M9. ***RAZLIKA SKUPOVA***,

M9. ***VANJSKA UNIJA***,

M10. ***RELACIJSKO DODJELJIVANJE*** *(Relational assignment)*.

M11. ***MOŽDA THETA RESTRIKCIJA*** *(Theta select maybe)*,

M12. ***MOŽDA THETA SPAJANJE***,

M13. ***MOŽDA VANJSKO THETA SPAJANJE***,

M14. ***MOŽDA DIJELJENJE***.

M15. ***THETA RESTRIKCIJA BEZ UVAŽAVANJA SEMANTIKE*** (**SN**) *(Theta select semantic override* (s/o)),

M16. ***THETA SPAJANJE SN***,

M17. ***VANJSKO THETA SPAJANJE SN***,

M18. ***DIJELJENJE SN.***

"***theta***" označava jedan od operatora usporedbe:

- jednako,
- nejednako,
- veće nego,
- manje nego,
- veće nego ili jednako,
- manje nego ili jednako.

7 **(2 boda)** Od kojih se faza sastoji postupak oblikovanja baze podataka? U čemu se razlikuje oblikovanje distribuirane baze podataka od centralizirane ?

Odgovor

FAZE U POSTUPKU OBLIKOVANJA BAZE PODATAKA:

- Formulacija i analiza zahtjeva - analiza objektnog sustava
- Konceptualno oblikovanje
- Implementacijsko oblikovanje
- Fizičko oblikovanje

DISTRIBUIRANA BAZA PODATAKA predstavlja takvu organizaciju gdje su podaci ***smješteni*** u **različitim čvorovima mreže**, ali međusobno povezani u jedan **integrirani sustav**.

Ukoliko usporedimo faze oblikovanja centralizirane baze podataka s fazama oblikovanja distribuirane baze podataka možemo ustanoviti da se postupak oblikovanja distribuirane baze podataka ***razlikuje*** u osnovi za **faze:**

- **određivanja particija** baze podataka,
- **oblikovanja lokalnih logičkih modela baze podataka**.

PARTICIJA predstavlja **disjunktni podskup globalne logičke baze podataka**. Ona se često naziva i **logičkim fragmentom baze podataka**.

Svaka se particija opiše s veličinom particije, statističkim podacima o učestalosti korištenja dotične particije u pojedinim obradama te strukturom podataka koja mora točno odražavati informacije sadržane u globalnoj logičkoj bazi podataka.

8. (**2 boda**) Što je logičko optimiranje upita (u odnosu na fizičko)?

Odgovor:

Logično optimiranje upita je odabir najefikasnijeg izraza relacijske algebre (tj. onog koji se najbrže izvršava) u skupu međusobno ekvivalentnih izraza (dva izraza relacijske algrebre su ekvivalentna ako za svaku ispravnu instancu baze rezultiraju istim skupom n-torki). Primjerice, selekciju je bolje izvršiti prije spajanja, nego nakon njega.
Fizičko optimiranje odnosi se na odabir odgovarajućeg načina fizičkog dohvata podataka iz datoteka (indeksi ili linearno pretraživanje) i na odabir odgovarajućeg algoritma izvođenja operacija relacijske algebre (algoritam spajanje, algoritam sortiranja).

9. **(2 boda**) Koje algoritme sustavi za upravljanje bazama podataka koriste za spajanje dviju relacija(tablica)? Koji od njih će najefikasnije izvršiti sljedeće θ-spajanje tablica A i B:

```
SELECT * FROM A, B WHERE A.x > B.y;
```

Odgovor:

Algoritmi spajanja su:

- spajanje ugniježđenom petljom
- spajanje sortiranjem i udruživanjem
- spajanje raspršivanjem

Budući da potonja dva algoritma funkcioniraju samo za operator jednakosti, gornji izraz može se izvršiti isključivo ugniježđenom petljom.

10. **(2 boda**) Koje su faze razvoja strukture zapisa podataka (prema sve kvalitetnijim metapodacima) od tekstualnih dokumenata do ontologija. Poredajte ih kronološki uz kratki opis svake faze.

Odgovor:

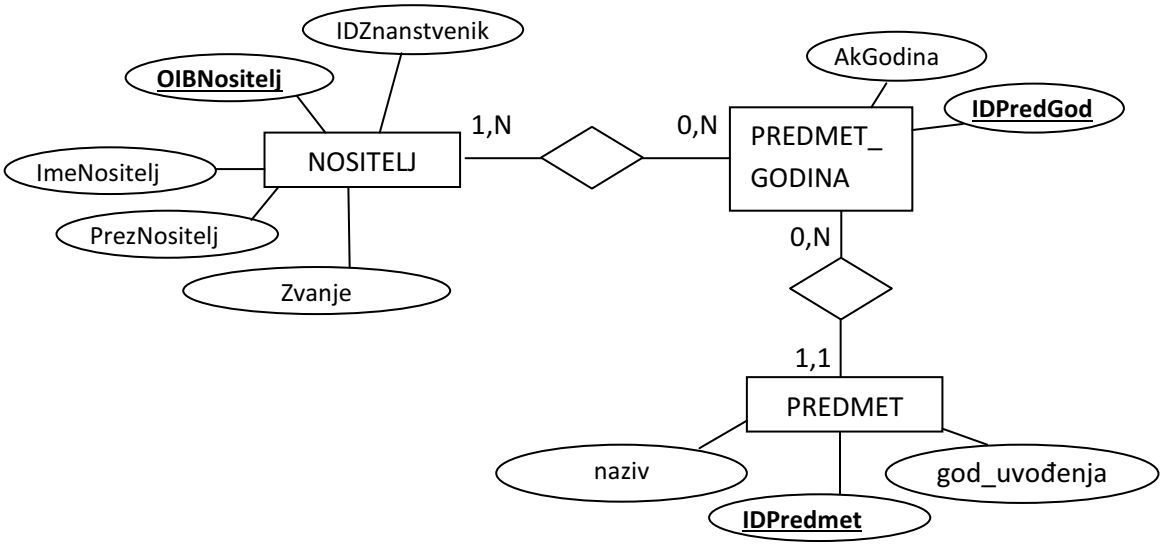
1. Tekst i baze podataka (logika se nalazi u aplikacijama, a ne u podacima)
2. XML dokumenti (metapodaci su odvojeni od podataka; neovisnost podataka od aplikacije za određenu domenu; podatke može koristiti više aplikacija)
3. Taksonomije i rječnici (podaci se odnose na više različitih domena; postoji precizna klasifikacija; postoje jednostavne relacije između kategorija)
4. Ontologije i pravila (postoji mogućnost dobivanja novih podataka iz onih već postojećih; podaci su "pametno" organizirani i opisane su konkretne relacije te sofisticirani formalizmi)

11. **(1 bod**) U modelu entiteti-veze objasnite gornju i donju granicu pridruživanja.

Odgovor:

Gornja i donja granica pridruživanja definiraju najveći odnosno najmanji broj instanci danog entiteta koje mogu biti pridružene preostalom entitetu (skupu entiteta, u slučaju da je stupanje veze veći od 2). Donja granica zapravo se tiče opcionalnosti sudjelovanja entiteta u vezi i može biti 0 ili 1. Gornja granica može biti 1, neki drugi pozitivni broj ili M (nedefinirano mnogo; u teoriji beskonačno).

12. **(4 boda**) Na osnovu donjeg ER dijagrama potrebno je napisati SQL naredbe koje će poslužiti za konačno kreiranje tablica u bazi podataka. Radi se o predmetima na fakultetu za koje se bilježe podaci o njihovim nositeljima u različitim godinama izvođenja. Svaki nositelj predmeta je znanstvenik uveden u bazu znanstvenika u Republici Hrvatskoj te mu se pridružuje i jedinstveni broj IDZnanstvenik.




```
CREATE TABLE predmet (
        IDPredmet INTEGER PRIMARY KEY,
        naziv VARCHAR2(128),
        god_uvodjenja SMALLINT
)

CREATE TABLE predmet_godina (
        IDPredGod INTEGER PRIMARY KEY,
        gk_godina SMALLINT,
        IDPredmet INTEGER REFERENCES predmet (IDPredmet)
)

CREATE TABLE nositelj (
        OIBNositelj CHAR(11) PRIMARY KEY,
        IDZnanstvenik INTEGER NOT NULL UNIQUE,
        imeNositelj VARCHAR2(128),
        prezNositelj VARCHAR2(128),
        zvanje VARCHAR2(128)
)

CREATE TABLE nositelj_predmet_godina (
        OIBNositelj CHAR(11) REFERENCES nositelj(OIBNositelj),
        IDPredGod INTEGER REFERENCES predmet_godina(IDPredGod),
        PRIMARY KEY (OIBNositelj, IDPredGod)
)
```

13. **(3 boda)** Tablica NEKRETNINA stvorena je naredbom

```
CREATE TABLE nekretnina
 (mbr INTEGER PRIMARY KEY,
  ulica VARCHAR2(128),
  kbr VARCHAR2(5),
  pbr INTEGER,
  povrsina DECIMAL(5,2)).
```

NEKRETNINA

MBR	ULICA	KBR	PBR	POVRSINA
1001	Tratinska	32	10000	80.37
1002	Korzo	18	51000	104.54
1003	Ilica	20B	10000	104.54

Tablica TAB stvorena je naredbom `CREATE TABLE tab (y VARCHAR2(256))` i na početku je bez zapisa. Sadržaj tablice NEKRETNINA dan je desno. Prikažite sadržaj tablice `tab` (skicirajte tablicu) nakon izvršavanja PL/SQL bloka na slici dolje.

GRUPA A

```
DECLARE
  CURSOR c1 IS
    SELECT pbr, COUNT(*)
    FROM nekretnina
    GROUP BY pbr ORDER BY pbr;
  CURSOR c2 (x nekretnina.pbr%TYPE) IS
    SELECT mbr FROM nekretnina
    WHERE pbr=x ORDER BY mbr;
  line tab.y%TYPE;
BEGIN
  FOR rec1 IN c1 LOOP
     line:=rec1.pbr||':';
     FOR rec2 IN c2(rec1.pbr) LOOP
        line:=line||rec2.mbr||';';
     END LOOP;
     INSERT INTO tab VALUES (line);
  END LOOP;
  COMMIT;
END;
```

GRUPA B

```
DECLARE
  CURSOR c1 IS
    SELECT pbr, COUNT(*)
    FROM nekretnina
    GROUP BY pbr ORDER BY pbr;
  CURSOR c2 (x nekretnina.pbr%TYPE) IS
    SELECT mbr FROM nekretnina
    WHERE pbr=x ORDER BY mbr DESC;
  line tab.y%TYPE;
BEGIN
  FOR rec1 IN c1 LOOP
     line:=rec1.pbr||':';
     FOR rec2 IN c2(rec1.pbr) LOOP
        line:=line||rec2.mbr||';';
     END LOOP;
     INSERT INTO tab VALUES (line);
  END LOOP;
  COMMIT;
END;
```

GRUPA A

Y
10000:1001;1003;
51000:1002;

GRUPA B

Y
10000:1003;1001;
51000:1002;

14. **(2 boda)** Tablica TABLICA stvorena je naredbom `CREATE TABLE tablica (x INTEGER).` Prikažite njen sadržaj (skicirajte tablicu) nakon izvršavanja PL/SQL bloka na slici dolje.

GRUPA A

```
BEGIN
  DELETE FROM tablica;
  COMMIT;
  SAVEPOINT a;
  INSERT INTO tablica VALUES(1);
  SAVEPOINT b;
  INSERT INTO tablica VALUES(2);
  ROLLBACK to SAVEPOINT a;
  INSERT INTO tablica VALUES(3);
  SAVEPOINT c;
  INSERT INTO tablica VALUES(4);
  COMMIT;
  INSERT INTO tablica VALUES(5);
  SAVEPOINT d;
  INSERT INTO tablica VALUES(6);
  ROLLBACK to SAVEPOINT d;
  COMMIT;
END;
```

GRUPA B

```
BEGIN
  DELETE FROM tablica;
  COMMIT;
  SAVEPOINT a;
  INSERT INTO tablica VALUES(1);
  SAVEPOINT b;
  INSERT INTO tablica VALUES(2);
  SAVEPOINT c;
  INSERT INTO tablica VALUES(3);
  ROLLBACK to SAVEPOINT b;
  INSERT INTO tablica VALUES(4);
  COMMIT;
  INSERT INTO tablica VALUES(5);
  SAVEPOINT d;
  INSERT INTO tablica VALUES(6);
  ROLLBACK to SAVEPOINT d;
  COMMIT;
END;
```

GRUPA A

X
3
4
5

GRUPA B

X
1
4
5