**--SELECT**

**Ulohy ---> 2015 ←-**

**-- Uloha 1 - 1.zmazte vsetky info o lete 545**

delete from L\_letenka

where id\_letu = 545;

delete from L\_triedy

where id\_letu = 545;

delete from L\_let

where id\_letu = 545;

**-- Uloha 2 - vypiste mesta kde je pocet letisiek > 2, pozor jedno mesto moze mat viac psc**

SELECT nazov, count(id\_letiska) as "Pocet"

FROM L\_mesto join L\_letisko USING(psc)

group by nazov

having count(id\_letiska) > 2;

**-- Uloha 4 - zobrazte ludi kt. maju priezvisko matiasko alebo co si take**

select \* from L\_osoba

where priezvisko = 'Matiasko';

-- Uloha 4.1

select \* from L\_osoba

where priezvisko like 'M%';

**-- Uloha 5 - zoznam ludi kt. si nekupili letenku v r.k 2014**

select distinct lo.meno, lo.priezvisko

from L\_osoba lo

join L\_letenka ll using(cislo\_dokladu, typ\_dokladu)

where extract(year from ll.datum\_platby) <> 2014;

--alternativne riesenie

select distinct meno, priezvisko

from l\_osoba

where not exists(

select 'x' from l\_letenka

where extract(year from datum\_platby) = 2014

and l\_letenka.cislo\_dokladu = l\_osoba.cislo\_dokladu

and l\_letenka.typ\_dokladu = l\_osoba.typ\_dokladu);

--Pre kontrolu - vypis, kolko leteniek si kupili jednotlive osoby v minulom roku

select meno, priezvisko, count(\*)

from l\_osoba join l\_letenka

using(cislo\_dokladu,typ\_dokladu)

where extract(year from datum\_platby) = 2014

group by(meno,priezvisko);

**-- Uloha 8 - vypiste aktualnych zamestnancov spol. Ryanair**

set linesize 150;

select lo.meno, lo.priezvisko, lz.pozicia, lz.datum\_prijatia, lz.datum\_zrusenia

From L\_osoba lo

join L\_zamestnanec lz using(cislo\_dokladu, typ\_dokladu)

join L\_let\_spolocnost lls using(id\_spolocnosti)

where lls.nazov\_spol = 'Ryanair'

AND lz.datum\_prijatia <= sysdate

AND (nvl(lz.datum\_zrusenia, sysdate) >= sysdate);

**-- Uloha 1 - select koľko minuli osoby s priezviskom Mravec na letenky v roku 2014**

select lo.priezvisko, count(ll.cena)

from L\_osoba lo

join L\_letenka ll using(cislo\_dokladu, typ\_dokladu)

group by (lo.priezvisko)

having lo.priezvisko = 'Mravec';

--alternativne riesenie - count zmeneny na sum, lebo chceme vediet, kolko minuli

--dohromady a nie, kolko letov absolvovali v roku 2014 (doplnena podmienka na

--kontrolu, ci bola letenka zaplatena v roku 2014)

select priezvisko, sum(cena)

from l\_osoba

join l\_letenka using(cislo\_dokladu, typ\_dokladu)

where priezvisko = 'Mravec'

and extract(year from datum\_platby) = 2014

group by priezvisko;

**-- Uloha 2 - select lety buduci mesiac zo slovenska do talianska**

select ll.letisko\_z, lz.id\_krajiny, ll.letisko\_do, ldo.id\_krajiny, ll.datum\_letu

from L\_let ll

join L\_letisko lz ON(ll.letisko\_z = lz.id\_letiska)

join L\_letisko ldo ON(ll.letisko\_do = ldo.id\_letiska)

where lz.id\_krajiny = 'SK' AND ldo.id\_krajiny = 'IT'

AND ll.datum\_letu BETWEEN trunc(add\_months(sysdate, 1), 'MM')

AND last\_day(add\_months(sysdate, 1));

--alternativne riesenie cez vnorene selecty

select id\_letu, letisko\_z, letisko\_do, datum\_letu

from l\_let

where letisko\_z IN (

select id\_letiska from l\_letisko where id\_krajiny = (

select id\_krajiny from l\_krajina where nazov\_krajiny = 'Slovensko'))

and letisko\_do IN (

select id\_letiska from l\_letisko where id\_krajiny = (

select id\_krajiny from l\_krajina where nazov\_krajiny = 'Taliansko'))

and to\_char(datum\_letu,'MM-YYYY') = to\_char(add\_months(sysdate,1),'MM-YYYY');

**ulohy ---> 2014**

**---vypísať meno, priezvisko, a id knihy ak napísal knihu**

**-- a ak nie tak len meno,priezvisko**

select DISTINCT name, surname, title\_id from author

left outer join authors\_of\_book using(author\_id);

(select DISTINCT name,surname,book\_id from author

left join authors\_of\_book using(author\_id)

left join title using(title\_id)

left join book using(title\_id) order by book\_id;)

**-- Vypíšte všetkých čitateľov ktorý sa zaregistrovali po 19:00**

select distinct name, surname from person

join reader using(person\_id)

where to\_char(valid\_from, 'HH24:Mi')>= '19:00';

**---vypísať knihy ktoré boli vydané v roku 2014**

select \* from title where year\_of\_issue = 2014;

**-- všetky žánre u vydavateľa edis**

select genre from title where publisher = ‘edis’;

**!-- mená titulov, a koľko sme zaplati.li za nákup kníh tých titulov**

**--(ak sme nekúpili vypísať 0)**

select distinct title\_name, NVL(sum(price), 0) cena\_za\_nakup

from Title left join Book using(title\_id)

group by title\_name;

**-- suma všetkého toho čo zaplatil čitateľ**

**--(za požičané, stratené, poškodené ...)**

*--a)*

select name, surname, NVL(sum(price),0) poplatky

from Person join Reader using(person\_id)

join Rent\_books using(reader\_id)

group by name, surname

order by name, surname;

*--73 údajov*

*--b)*

select name,surname,sum(price)

from Rent\_books

join reader using (reader\_id)

join person using (person\_id)

group by name,surname;*--,reader\_id; --149 údajov*

*--73 údajov*

**---vypíš všetkých čítateľov a aj ich reader\_id, ak ľudia nie sú čitateľom tak iba ich meno, a priezvisko(left outer join)**

*--a)*

select name, surname, r.reader\_id from person left join reader r using(person\_id);

*--b)*

SELECT reader\_id,name,surname FROM Reader RIGHT JOIN Person using (person\_id) GROUP BY reader\_id,name,surname ORDER BY reader\_id;

**--SELECT(Dátumové)**

**-- vypisať všetky knihy požičané každy druhý pondelok /// alebo niečo podobne**

**--bola vytvorená tabuľka s atribútmi rodné číslo, číslo knihy a bolo do nej potrebné vložiť (zrejme) selectom všetky knihy, ktoré má niekto teraz požičané, alebo niečo také…**

**-- Vypíšte všetkých čitateľov ktorý majú tento rok okrúhle narodeniny  
  
-- Vypíšte všetkých čitateľov a ich posledne výpožičky**

**--Ku čitateľom vypíšte dátum aj čas vypožičky (neviem presne znenie zadania)**

**--Vypísať všetky knihy požičané v nedeľu**

select distinct title\_name from Title join Book using(title\_id) join Rent\_books using(book\_id)

where to\_char(borrow\_date, 'D') = '1'

order by title\_name;

Select title\_name,borrow\_date from title

join book using(title\_id)

join rent\_books using(book\_id)

where borrow\_date is not null and borrow\_date<sysdate and to\_char(borrow\_date,'D') = '1';

**---vypis stratenych knih za posledny rok zo zanru romany.**

select book\_id

from book join title using(title\_id)

where lost\_date between add\_months(sysdate,-12) and sysdate

and genre = 'Novel'

order by 1;

(

select title\_name from title join book using(title\_id)

where genre='Novel'

and months\_between(sysdate,lost\_date) between 0 and 12;

)

**---vypísat isbn, názov, cenu kníh ktoré neboli stratené minulý rok**

**--a zoradiť podľa ceny od najväčšej po najmenšiu cenu**

*--a)*

select isbn, title\_name, sum(price)

from title join book using(title\_id)

where lost\_date is null or months\_between(lost\_date, sysdate) >12

group by isbn, title\_name;

*--85 údajov*

*--b)*

select isbn,title\_name,price from Title,Book

where to\_char(lost\_date,'YYYY')< '2013' or to\_char(lost\_date,'YYYY')='2014'

group by isbn,title\_name,price

order by price;

*--2000 údajov*

*--c)*

SELECT isbn,title\_name,price FROM Title

JOIN Book using (title\_id)

WHERE lost\_date is null

OR to\_char(lost\_date,'YYYY') < to\_char(add\_months(sysdate,-12),'YYYY') --// < 2013

OR to\_char(lost\_date,'YYYY') = to\_char(sysdate,'YYYY') --// = 2014

GROUP BY isbn,title\_name,price

ORDER BY price DESC;

*--441 údajov*

*--d)*

select isbn, title\_name, price from Title join Book using(title\_id)

where lost\_date < add\_months(sysdate, -12) or lost\_date is null

order by price desc;

*--278 údajov*

**--select s časom kde bolo ohraničenie minulý rok**

**(tušim že to bolo že vypisať knihy ktore neboli/boli rok požičané)**

select distinct book\_id

from book join rent\_books using(book\_id)

where return\_date is not null

and months\_between(borrow\_date, return\_date) <= 12;

(

select distinct title\_name from title join book using(title\_id)

join rent\_books using(book\_id)

where months\_between(borrow\_date,return\_date)<=12;

)

**-select s časom kde bolo ohraničenie minulý rok (tušim že to bolo že vypisať knihy ktore neboli/boli rok požičané)**

select distinct title\_name from Title join Book using(title\_id)

join Rent\_books using(book\_id)

where return\_date - borrow\_date >= 365

or (return\_date is null and status = 'P' and sysdate - borrow\_date >= 365)

order by title\_name;



**--vypisat všetky knihy požičané v nedeľu**

select distinct title\_name, title\_id from Title join Book using(title\_id) join Rent\_books using(book\_id)

where to\_char(borrow\_date, 'D') = '1'

order by title\_name;

**--ludia, ktorí is požičali knihy v nedelu...**

*--a)*

select name,surname from Reader join Person using (person\_id)

where reader\_id in (select reader\_id from Rent\_books where to\_char(rent\_date,’DAY’) = ‘SUNDAY’);

*--b)*

select distinct name, surname from Person join Reader using(person\_id)

join Rent\_books using(reader\_id)

where to\_char(borrow\_date, 'D') = 1

order by name, surname;

*--s narodeninami je to ťažšie, z person\_id poskladať dátum, ten previesť do to\_date a to celé dať do where to\_char( ..(to celé).. , ‘DAY’) in (‘Saturday’,’Sunday’);*

-**-select meno, priezvisko, vek ku 31.12 tohto roku**

select name, surname, (*2014* - to\_number('19' || substr(person\_id,1,2))) as age from Person;

*--101 údajov Cailin Pratt age: 1823 ?????*

select name, surname, (to\_char(sysdate, 'YYYY') - to\_char('19' || substr(person\_id,1,2))) as age from Person;

*--101 údajov Cailin Pratt age: 1823 ?????*

*- -toto je tiež zle, napr ten Pratt má substr(person\_id,1,2) =1 , cize ked sa to pozliepa tak to bude 2014-191*

**!--ludia, ktorý sa narodili cez víkend**

*--a)*

*select name, surname*

*from person*

*where mod(to\_number(substr(person\_id, 5,2)),7) = 1 or*

*mod(to\_number(substr(person\_id, 5,2)),7) = 0;*

*--b)*

*select meno, priezvisko, to\_date(decode(substr(rod\_cislo,3,1),5,0,6,1,1,1,0,0)*

*|| substr(rod\_cislo,4,1) || substr(rod\_cislo,5,2) || substr(rod\_cislo,1,2), 'MM.DD.YY') datum\_narodenia from osoba where to\_char(to\_date(decode(substr(rod\_cislo,3,1),5,0,6,1,1,1,0,0) || substr(rod\_cislo,4,1)*

*|| substr(rod\_cislo,5,2) || substr(rod\_cislo,1,2), 'MM.DD.YY'), 'D') in (1,7);*

*--’D’ - tam sa čísluje od nedele ako 1 …*

**-- mena čitateľov, ktorí si požičali za posledný rok menej ako 10 kníh**

--a)

select name, surname, count(\*) pocet\_vypoziciek from Person

join Reader using(person\_id) join Rent\_books using(reader\_id)

where borrow\_date between add\_months(sysdate, -12) and sysdate

group by name, surname

having count(\*) < 10

order by name, surname;

*--tento select vypíše 18 údajov*

*--b)*

select name, surname, count(\*)

from person join reader using(person\_id)

join rent\_books using(reader\_id)

where months\_between(borrow\_date, sysdate) <=12

group by name, surname

having count(\*) < 10;

*--61 údajov*

**--mená titulov kníh, ktoré boli stratené minulý rok**

select title\_name, book\_id

from title join book using(title\_id)

where to\_char(lost\_date, 'YYYY') = to\_char(add\_months(sysdate,-12),'yyyy');

--where to\_char(lost\_date, 'YYYY') =  *(to\_char(sysdate, ‘YYYY’) -1)*;

***--- meno, priezvisko a poplatky čitateľa, ktoré zaplatil***

*select name, surname, sum(price) from person join reader using(person\_id)*

*join rent\_books using(reader\_id) group by name, surname; --73 údajov*

**--- meno, priezvisko a poplatky čitateľa, ktoré zaplatil za minulý rok**

select name, surname, sum(price) poplatky from Person join Reader using(person\_id) join Rent\_books using(reader\_id)

where borrow\_date between add\_months(sysdate, -12) and sysdate

group by name, surname;

*--18 údajov TOTO JE ZA POSLEDNY ROK*

*select name,surname,sum(rent\_books.price)poplatky,borrow\_date from person*

*join reader using(person\_id) join rent\_books using(reader\_id)*

*where to\_char(borrow\_date,'YYYY')=to\_char(sysdate,'YYYY')-1*

*group by name,surname,borrow\_date*

*order by name,surname;*

**--SELECT(Iné)**

**-- Kniha, ktorá nebola ani raz požičaná ( EXISTS, IN)**

*--a) not exists*

select b.book\_id, t.title\_id

from title t join book b on(t.title\_id = b.title\_id)

where not exists (select 'x' from rent\_books rbo where b.book\_id = rbo.book\_id);

*--b) not in*

select b.book\_id, t.title\_id

from title t join book b on(t.title\_id = b.title\_id)

where b.book\_id not in(select rbo.book\_id from rent\_books rbo);

--c)

select distinct title\_id,title\_name from title join book using(title\_id)

where book.book\_id not in(select book\_id from rent\_books) order by title\_id;

**-ožičanie a-select kniha, ktorá je na pspoň jedna**

select title\_name, count(\*) k\_dispozicii from Title join Book using(title\_id)

where book\_id not in (select book\_id from Rent\_books where status = 'P')

group by title\_name

having count(\*) >= 1

order by title\_name;

*--93 údajov*

*SELECT title\_name from Rent\_books*

*JOIN Book USING (book\_id)*

*JOIN Title USING (title\_id)*

*WHERE borrow\_date is null OR return\_date < sysdate;*

*--0 údajov*

**--TRIGGER**

**-- Pridajte do tela Triggra príkaz na to aby pridal do relacie Rent\_Books hodnoty (neviem hodnoty) kde status je R**

**-- triger ak už bola raz kniha predlžena (extension\_date) tak ma vyhodit vynimku Raise\_aplication….**

**--Napiste telo triggra, ktory zabezpeci vymazanie titulu knihy s title\_id.**

delete from authors\_of\_book where title\_id=:old.title\_id;

delete from rent\_books where book\_id in (select book\_id from book where title\_id=:old.title\_id);

delete from book where title\_id=:old.title\_id;

**--telo trigra pri update rent\_books aby vložilo price=100 kde je status L**

if :new.status='L' then :new.price:=:old.price + 100

end if;

**--INSERT**

**--UPDATE uživateľovi s ID 156, status na R, datum výpožičky na aktuálny**

**--vytvor čitateľský preukaz pre osobu s id 12**

insert into reader(reader\_id, person\_id, valid\_from) values(sek.nextVal, ‘12’, sysdate);



**--DELETE**

**-- Zmažte všetky knihy ktoré neboli nikdy požičiané a boli registrované v minulom roku**

**--zmaž všetky knihy, ktoré neboli (nie sú teraz) aktuálne požičané**

DELETE From Rent\_books where borrow\_date is null OR return\_date < sysdate;

**--zmaž knihy, ktoré neboli nikdy požičané**

delete from Book where book\_id not in (select book\_id from Rent\_books);

*--b)*

delete from Book where not exists(select book\_id from Rent\_books where Rent\_books.book\_id = Book.book\_id);

**--zmaž všetky knihy, ktoré nie sú momentálne požičané**

delete from Rent\_books where book\_id in (select book\_id from Rent\_books where status <>'P');

--delete from Book where book\_id not in (select book\_id from Rent\_books);

**---vymazať všetky údaje o tituloch ktoré neboli nikdy požičané**

--a)

delete from book b where not exists

(select 'x' from rent\_books rbo where b.book\_id = rbo.book\_id );

*--123 riadkov vymazaných*

*--b)*

delete from title where title\_id not in (select title\_id from book where book\_id not in (select book\_id from rent\_books));

*--28 riadkov vymazaných*

*--c)*

delete from title t where not exists

(select 'x' from book b, rent\_books rbo where b.title\_id = t.title\_id and b.book\_id = rbo.book\_id);

*--3 riadky vymazané*

*--d)*

delete from Book where book\_id not in (select book\_id from Rent\_books); *--123 riadkov*

delete from AUTHORS\_OF\_BOOK where not exists(select title\_id, author\_id from Title, AUTHOR); *--0 riadkov*

delete from Title where title\_id not in (select title\_id from Book); *--3 riadky vymazané*

**--UPDATE**

**--update zmenu používateľovi borrow\_date na aktualny dátum a stav --knihy na ‘R’,reader\_id na sekv.hodnotu(čislo)**

update rent\_books set borrow\_date = sysdate, status = ‘R’ where reader\_id = ‘hocico’;

**--zmeň status = ‘V’ a read\_id na hodnotu získanú zo sekvenčného čísla..**

update rent\_books set reader\_id = sek.nextVal where status = ‘V’;

**--CREATE, ALTER**

**-- Pridajte do Relácie TITLE novy stlpec TYPE ktorý može nadobudať iba --hodnoty N,B,R,L alebo NULL**

**--vytvorit unikátny index na isbn**

create index title\_isbn ON Title(isbn);

**-- vytvor M:N vzťah medzi tabuľkou GENRE a tabuľkou TITLE**

create table mn( genre\_id integer, title\_id integer, primary key(genre\_id, title\_id));

alter table mn add foreign key(genre\_id) references genre(genre\_id);

alter table mn add foreign key(title\_id) references title(title\_id);

**-- vytvor tabuľku genre, ktorá bude M:N vzťah z titulom**

create table Genre

(genre\_id Integer, genre\_name VARCHAR2(20),primary key(genre\_id));

create table MN\_vztah

( genre\_id Integer,title\_id Integer, primary key(genre\_id, title\_id));

alter table MN\_vztah add foreign key(title\_id) references Title(title\_id);

alter table MN\_vztah add foreign key(genre\_id) references Genre(genre\_id);

**3b. Pomocou lineárneho zápisu entít a vzťahov navrhnite ERA diagram pre vzťah LEKÁR – PACIENT,**

**definujte atribúty patriace vzťahu a uveďte možnú kardinalitu a povinnosť členstva s uvedením slovného popisu integritného obmedzenia vyplývajúceho zo vzťahu:**

**LEKAR (#ID\_LEKARA)**

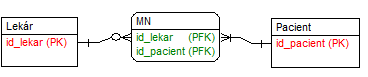
**PACIENT(#ID\_PACIENTA)**

**MN(#ID\_LEKARA,#ID\_PACIENTA)**

**LEKAR – PACIETN M:N**

**LEKAR nemusí mať pacienta(nepovinné členstvo)**

**PACIENT musí mať lekara(povinné členstvo)**



***Create table Lek\_r (id\_lekar Integer NOT NULL ,primary key (id\_lekar) ) ;***

***Create table Pacient (id\_pacient Integer NOT NULL , primary key (id\_pacient) );***

***Create table MN (id\_lekar Integer NOT NULL , id\_pacient Varchar2 (30) NOT NULL , primary key (id\_lekar,id\_pacient) ) ;***

***Alter table MN add foreign key (id\_lekar) references Lekar (id\_lekar) ;***

***Alter table MN add foreign key (id\_pacient) references Pacient (id\_pacient)***

**3b. Pomocou lineárneho zápisu entít a vzťahov navrhnite ERA diagram pre vzťah POISŤOVNA – POISTENEC (Myslite na všetky druhy poistenia). Definujte atribúty patriace vzťahu a uveďte možnú kardinalitu a povinnosť členstva s uvedením slovného popisu integritného obmedzenia vyplývajúceho zo vzťahu:**

**POISTOVNA(#id\_poistovne)**

**POISTENEC(#id\_poistenca, druh\_poistenia)**

**MN(#id\_poistovne,#id\_poistenca)**

**KARDINALITA M:N**

**POISTOVNA nemusí mať poistencov(nepovinné členstvo)**

**POISTENEC musí mať POISTOVNU (povinné členstvo)**

**3bodova otazka). Pomocou lineárneho zápisu entít a vzťahov navrhnite ERA diagram pre vzťah OSOBA - ŠKOLA (Myslite na všetky druhy škôl, vrátane jazykových). Definujte atribúty patriace vzťahu a navrhnite kardinalitu vzťahu s uvedením stručného popisu integritného obmedzenia vyplývajúceho zo vzťahu.**

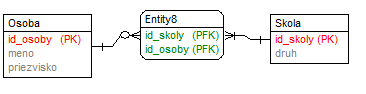
**OSOBA(#id\_osoby, meno, priezvisko)**

**SKOLA(#id\_skoly, druh)**

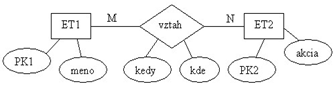
**Entity8(#id\_osoby, #id\_skoly)**

**škola musí byt priradená osobe - povinne členstvo**

**osoba nemusí byt priradená skole - nepovinne členstvo**



**32. Pomocou lineárneho zápisu transformujte daný entitno-relačný diagram do dátového.(Atribúty, ktoré sú súčasťou PK označte symbolom #. Cudzie kľúče označte pomocou (FK)).**



R1(#PK1,meno)

R2(#PK2,akcia)

R3(PK1(PFK),PK2(PFK),kedy,kde)

**38. Externá schéma je:**

a.) implementačne nezávislá množina dát popisujúca užívateľské pohľady aplikácie

b.) implementačne závislá množina dát, ktorá presne popisuje dátové štruktúry a prístupové metódy uložených dát v externej pamäti

c.) implementačne nezávislá množina dát, ktorá presne popisuje dátové štruktúry a prístupné metódy uložených dát v externej pamäti

d.) implementačne závislá množina dát popisujúca dátový model

**41. Pri operácii UPDATE vysvetlite pojem REŠTRIKCIA (RESTRICTED) pre zmenu hodnôt primárneho kľúča:**

Ak sa užívateľ pokúsi zmeniť hodnotu PK ktorá sa v inej relácií nachádza ako FK tak to nedovolí.

**44. Ktoré úrovne architektúry DBS sú implementačne závislé:**

a.) len interná

b.) externá aj interná

c.) aplikačná a externá

d.) len externá

**45. Perzistentné dáta sú:**

a.) dáta, ktoré zanikajú po ukončení programu

b.) dáta existujúce aj po ukončení programu

c.) odvodené dáta

**46. Popíšte vlastnosti relácie:**

a.) CRUDE = stĺpcová, referenčná, užívateľská, domenová, entitná

b.) ACID = atomickosť, konzistencia, nezávislosť, trvácnosť

c.) ANNU = atomickosť, neusporiadanosť zľava doprava, neusporiadanosť zhora dole, jednoznačnosť

**47. Definujte cudzí klúč:**

a.) Množina atribútov, ktorá spĺňa podmienku jednoznačnosti a minimálnosti a je definovaná v inej relácii

b.) Množina atribútov, ktorá spĺňa podmienku jednoznačnosti a minimálnosti

c.) Množina atribútov, ktorá je definovaná v inej relácii ako kandidát primárneho kľúča

**48. Integrita entít hovorí:**

a.) atribút primárneho kľúča môže nadobúdať NULL hodnoty

b.) atribút primárneho kľúča nesmie nadobúdať NULL hodnoty

c.) atribút primárneho kľúča musí nadobúdať NULL hodnotu

|  |  |
| --- | --- |
| Doménová integrita | Doménová integrita reprezentuje množinu integritných obmedzení, ktoré zdieľajú všetky hodnoty atribútov priradených k tejto doméne. Druhy doménovej integrity:  • Typ dát  • Množina prípustných hodnôt  • Usporiadateľnosť |
| Integrita stĺpcov | Pre každý stĺpec tabuľky je nutné definovať doménu, prípadne dodatočné integritné obmedzeniastĺpcov  DruhyIO:  • Dodatočné IO pre rozsah hodnôt , ktoré sú podmnožinou príslušnej domény  • NULL alebo NOT NULL  • DISTINCT alebo DUPLICATE |
| Integrita entít | Atribút ktorý je súčasťou PK nesmie nadobúdať nedefinované hodnoty (nesmie byt NULL) |
| Referenčná integrita | Atribút, ktorého sa referenčná integrita týka sa nazýva cudzí kľúč. Hodnota tohto atribútu je buď prázdna alebo obsahuje hodnotu primárneho kľúča s inej relácie. PK sa týka hlavnej, FK sa týka závislej relácie. |
| Užívateľská integrita | Spojená s dátumami. |

**3b. Určite miesta, kde je porušená stĺpcová integrita:**



Citatel.os\_cislo, Kniha.nazov, duplicita # id\_citatel(C5)

**3b. Určite miesta, kde je porušená užívateľská integrita:**

**Čitateľ nemôže vrátiť knihu skôr ako si ju požičal. Kniha nemôže byť požičaná keď je stratená. Uzivatelska integrita. Každý čitateľ musí mať meno.**

**49. Uveďte príklad užívateľskej integrity:**

Dátum skúšky väčší ako dátum.

Kontrola opätovného zapísania predmetu študentovi.

Keď kniha bola vrátená skôr ako bola vypožičaná.

Tá istá kniha bola požičaná dvom ľuďom súčasne.

**50. Popíšte stlpcovuv integritu:**

**50.b Popíšte doménovú integritu:**

Doménová integrita reprezentuje množinu integritných obmedzení, ktoré zdieľajú všetky hodnoty atribútov priradených k tejto doméne. Druhy doménovej integrity:

• Typ dát

• Množina prípustných hodnôt

• Usporiada teľnosť

**51. Referenčná integrita hovorí:**

a.) hodnota FK v relácii R2 sa nemusí rovnať hodnote PK z relácie R1

b.) FK je množina atribútov definovaná v relácii R2, ktorá môže byť v inej relácii R1 definovaná ako primárny kľúč PK alebo kandidát PK

c.) hodnota FK v relácii R2 sa môže rovnať hodnote PK z relácie R1

d.) hodnota FK v relácii R2 sa musí rovnať hodnote PK z relácie R1, alebo NULL

**52. Definujte pojem Doména:**

a.) Množina prípustných atribútov

b.) Množina prípustných hodnôt atribútov

c.) Množina prípustných hodnôt cudzieho kľúča

**54. Fyzická nezávislosť:**

a.) nezávislosť definície dát na konceptuálnom modeli

b.) nezávislosť programov na organizácii dát a prístupových metódach

c.) nezávislosť programov na konceptuálnom modeli

**55. Zdieľateľnosť dát je:**

a.) Dáta môžu byť uložené vo viacerých tabuľkách

b.) Dáta môžu používať viaceré aplikácie súčasne alebo sekvenčne

c.) Dáta môžu byť použité práve raz v aplikácii

**56. Databáza je:**

a.) Systém riadenia bázy dát a databáza

b.) Programové vybavenie pre definíciu dát, manipuláciu s dátami

c.) Množina perzistentných dát

**57. Entita je:**

a.) Množina objektov rovnakého typu

b.) Popis dát a atribútov objektu

c.) Objekt reálneho sveta schopný nezávislej existencie

**58. Relácia je:**

a.) je podmnožinou kartézskeho súčinu množiny domén Di na množine atribútov Ai

b.) je zjednotením množiny prípustných hodnôt atribútov

c.) je množina dvojíc Ai:Di (A-atribút, D-doména)

d.) podmnožina kartézskeho súčinu množiny atribútov

**59. Koľko kandidátov primárneho kľúča môže mať relácia:**

a.) Môže mať viac KPK

b.) Môže mať vždy len jeden jednoduchý alebo kompozitný KPK (z jedného alebo množiny atribútov)

c.) Môže mať vždy len jeden jednoduchý KPK (z jedného atribútu)

**60. Definujte primárny kľúč:**

a.) šetky množiny atribútov, ktoré spĺňajú podmienku jednoznačnosti a minimálnosti

b.) Jedna množina atribútov, ktorá spĺňa podmienku jednoznačnosti a minimálnosti

c.) Množina atribútov, ktorá spĺňa podmienku jednoznačnosti a minimálnosti a je definovaná v inej relácii ako Kandidát primárneho kľúča

**61. Aký je rozdiel medzi typom entity a entitou:**

Entita je objekt reálneho sveta schopný nezávislej existencie.

Typ entity je množina objektov rovnakého typu, charakteristík a vlastností

**62. Architektúra DBS je:**

a.) dvojúrovňová (externá a interná)

b.) trojúrovňová (externá, konceptuálna, interná)

c.) trojúrovňová (externá, interná a aplikačná)

**65. Čo musí platiť pre primárne(PK) a cudzie(FK) kľúče dvoch entitných typov ET1 a ET2, aby kardinalita vzťahu bola 1:N?**

a.) PK z ET2 bude v ET1 FK, pričom nesmie byt súčasťou PK

b.) PK z ET1 bude FK v ET2, pričom môže byt súčasťou PK

c.) PK z ET2 bude v ET1 FK a zároveň celým PK

d.) PK z ET1 bude v ET2 FK, pričom nesmie byt súčasťou PK

**66. Integrita entít sa týka:**

a.) len atribútov primárneho kľúča

b.) len atribútov cudzieho kľúča

c.) len atribútov primárneho, alebo cudzieho kľúča

d.) všetkých NOT NULL atribútov

**67. Popíšte stĺpcovú integritu:**

Pre každý stĺpec tabuľky je nutné definovať doménu a prípadne dodatočné IO.

**68. Pri operácií DELETE vysvetlite pojem REŠTRIKCIA (RESTICTED):**

Ak sa v referencovanej tabuľke pokúsime vymazať hodnotu atribútu ktorú nadobúda FK v inej tabuľke tak nebude táto operácia

vykonaná.

**69. Pri operácií DELETE vysvetlite pojem NULOVANIA (NULLFIED):**

**Bicycles (id, type, owner\_id...) = (17, "Fast One", 31...) Owners (id, name...) = (31, "Joe Biker"...)**

**nullify pri delete znamená, že ak je vymazaný záznam v rodičovskej tabuľke, zodpovedajúce záznamy v tabuľkách potomkov sa NEVYMAŽÚ, ale nastavia na NULL (http://www.techonthenet.com/oracle/foreign\_keys/foreign\_null.php).**

**napr.: Ak owner\_id nastavime ze moze byt NULL a Joe Biker vymazeme, tak Bicycle s ID 17 owner\_id sa zmeni na NULL**

**139. Ak je FK súčasťou PK potom:**

**a.) Nemôžeme nastaviť ON UPDATE (NULLIFIED)**

**153. Pri operácii DELETE vysvetlite pojem kaskáda (CASCADE):**

**Keď sa pokúsime vymazať hodnotu atribútu v ref. tabuľke, ktorú nadobúda FK v inej tabuľke zmažú sa rodičovské riadky a zároveň aj “child“ riadky.**

**71. Do ktorej úrovne patrí stanovenie aká metóda indexácie bude použitá:**

**a.) Interná**

**b.) Konceptuálna**

**c.) Aplikačnej**

**d.) Externej**

**73. Príkaz GRANT slúži na:**

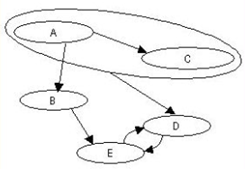
**a.) Obratie práv užívateľovi**

**b.) Pridelenie práv na prístup k zdrojovým kódom aplikácie**

**c.) Pridelenie prístupových práv na databázový objekt užívateľovi**

**d.) Odobratie práv čítania zdrojových kódov aplikácie**

**78. Vypíšte všetky funkčné závislosti grafu:**



**A -> C A -> B B -> E E <-> D AC -> D**

**fUNKCNE ZAVISLOSTI VYJADRENE CEZ LINEARNY ZAPIS**

**R1(#A,C)**

**R2(#A,#C,D)**

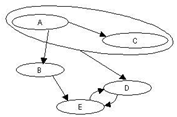
**R3(#A,B)**

**R4(#B,E)**

**R5(#E,D)**

**R6(#D,E)**

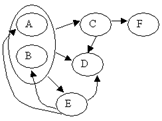
**194. Dekomponujte pomocou lineárneho zápisu nasledovnú reláciu podľa atribútu E:**



**R1(#A,C,B,E)**

**R2(#E,D)**

**195. Dekomponujte pomocou lineárneho zápisu nasledovnú reláciu:**



**R1(A#, B#, C, E) , R2(C#, D, F), ? R3(#E, A, B, D)**

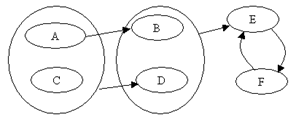
**80. Majme reláciu R(#A, B, C, D, E), pre ktorú sú dané nasledovné funkčné závislosti:**

**A -> B, BC -> E a ED -> A**

**Určite zoznam všetkých determinatov relácie R?**

**A, BC, ED**

**197. Funkčná závislosť ktorého atribútu na (A, C) je úplná:**

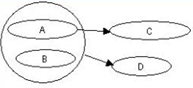


**a.) len atribútov B**

**b.) atribútov B a D**

**c.) len atribútov D**

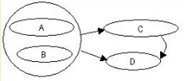
**84. Majme reláciu R(#A, #B, C, D). Aké sú v relácii funkčné závislosti, ak relácia je v 1.NF a nie je v 2.NF**



**A,B - > D uplna**

**A->C elementarna**

**84. Majme reláciu R(#A, #B, C, D). Aké sú v relácii funkčné závislosti, ak relácia je v 2.NF a nie je v 3.NF**

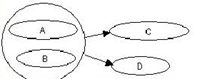


**A,B -> C**

**A,B -> D**

**C -> D**

**84. Majme reláciu R(#A, #B, C, D). Aké sú v relácii funkčné závislosti, ak relácia je v 3.NF**



**A,B -> C**

**A,B -> D**

**86. Majme reláciu R(#A, B, C, D, E), pre ktorú sú dané nasledovné funkčné závislosti:**

**A -> B, A -> E, ED -> A je táto relácia v 3NF ?**

**a.) Áno**

**b.) Nie(JE TRANZITIVNA)**

**121. Majme reláciu R (#A, B, C, D, E), pre ktorú sú dané nasledovné funkčné závislosti:**

**A -> B, BC -> E a ED -> A**

**Je relácia v BCNF?**

**a.) Nie**

**b.) Áno**

**145. Majme reláciu R(#A, B, C, D, E) pre ktorú sú dané nasledovné funkčné závislosti (B je kandidát PK)**

**A -> B, A -> C, A -> D, A -> E, B -> A, B -> C, B -> D, D -> E. Potom:**

**Relácia nie je ani v BCNF(nie každý determinat je kandidátom PK) ani v 3NF (je tranzitívna A <-> B...)**

**152. Majme reláciu R(#A, B, C, D, E), pre ktorú sú dané nasledovné funkčné závislosti:**

**A -> B, A -> C, A -> D, B -> E je táto relácia v 2NF ?**

**a.) Áno (každý nekľúčový atribút je úplne funkcne závislý od primárneho kľúča)**

**b.) Nie**

**každý nekľúčový atribút je úplne funkčne závislý od primárneho kľúča.**

**133. Aký je vzťah medzi BCNF a 3NF:**

**a.) AK je relácia v BCNF musí byť aj v 3NF**

**b.) Ak je relácia v 3NF nemôže byť v BCNF**

**c.) Ak je relácia v BCNF nemusí byť v 3NF**

**6. Kedy je relácia v BCNF:**

**a.) keď každý determinant je kandidátom primárneho kľúča**

**b.) keď je v 2NF a žiaden nekľúčový atribút nie je tranzitívne funkčne závislý na primárnom kľúči**

**c.) keď každý kandidát primárneho kľúča je determinantom**

**d.) keď je v 3NF a každý nekľúčový atribút je determinantom**

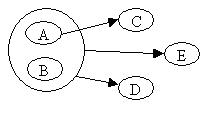
**142. Definujte Kandidáta primárneho kľúča:**

**Množina atribútov, ktorá spĺňa podmienku jednoznačnosti a minimálnosti.**

**143. Ak relácia R ma nasledovný zoznam kandidátov PK, potom určite PK:**

**KPK1: os\_cislo, KPK2: rodne\_cislo, datum\_zapisu, KPK3: login... os\_cislo**

**135. Dekomponujte pomocou lineárneho zápisu nasledovnú reláciu:**

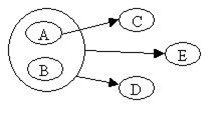


**R1(#A,C)**

**R2(#A,#B,E)**

**R3(#A,#B,D)**

**136. Je zadaná relácia R(#A,#B,C,D,E) s nasledovnými funkčnými závislosťami v 2 normálnej forme:**

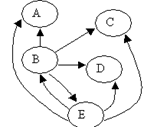


**a.) Áno**

**b.) Nie (pretože atribút C nie je úplne funkčne závislý od celej množiny atributov PK, ale iba od jeho podmnožiny)**

**(KAžDý NEKLúčOVý ATRIBúT JE úPLNE FUNKčNE ZáVISLý OD PRIMáRNEHO KľúčA)**

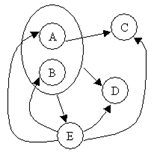
**190. Je relácia s nasledovnými funkčnými závislosťami v BCNF:**



**a.) Áno**

**b.) Nie**

**191. Je daná relácia v BCNF:**

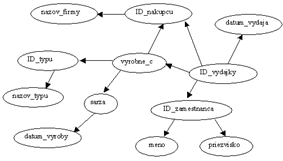


**a.) Áno (ak by nebola fun. závislosť A -> C, ale AB -> C)**

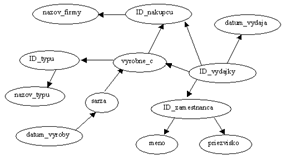
**b.) Nie**

**188. Ktorý model funkčných závislostí zodpovedá nasledovnému popisu?**

**V sklade máme súčiastky, ktoré majú svoje výrobné číslo, šaržu, typové označenie a názov typu. Súčiastky jednej šarže sú vyrábané v jeden deň. Tieto súčiastky sú postupne vydávané zo skladu do predajne, pričom potrebujeme vedieť ktorý zamestnanec vydal jednotlivé súčiastky, kedy a pre ktorého zákazníka (na jednu výdajku sa môže vydať viac súčiastok,, ale len pre jedného zákazníka).**



**196. Ak máme nasledovné funkčné závislosti, potom:**



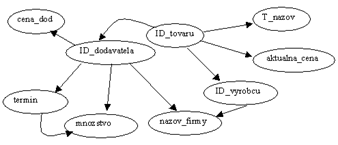
**a.) v jeden deň sa môže vyrobiť len jeden výrobok z viacerých šarží**

**b.) v jeden deň sa môžu vyrobiť viaceré výrobky z viacerých šarží**

**c.) v jeden deň sa môžu vyrobiť viaceré výrobky rovnakej šarže**

**d.) v jeden deň sa môže vyrobiť len jeden jediný výrobok**

**200. Funkčná závislosť medzi termínom a množstvom znamená, že v jednom termíne:**



**a.) musia byť všetky dodávky všetkých tovarov rovnakého množstva**

**b.) môžu byť dodané rôzne množstvá rôznych tovarov aj od rovnakého dodávateľa**

**c.) musia byť všetky dodávky len rovnakých tovarov rovnakého množstva**

**88. Nezávislosť dát:**

**a.) Znamená vzájomnú nezávislosť uloženia dát**

**b.) Znamená nezávislosť programu a prístupových metód od zmien dátových štruktúr**

**c.) Znovu použiteľnosť dát**

**89. Databázový systém je:**

**DB + SRBD, Štvorica: Data, SW, HW, Users**

**90. Vymenujte úrovne architektúry DBS:**

**Externá, konceptuálna, interná**

**91. Interná úroveň:**

**a.) Predstavuje pohľad užívateľov na dáta**

**b.) Predstavuje prístupové metódy k dátam**

**c.) Predstavuje organizáciu uloženia dát a prístupové metódy k dátam**

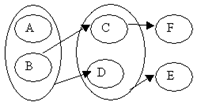
**92. Logická nezávislosť sa v architektúre zabezpečuje:**

**a.) Medzi externou a konceptuálnou úrovňou**

**b.) Medzi konceptuálnou a internou úrovňou**

**c.) Nedá sa zabezpečiť**

**208. Navrhnite primárny kľúč danej relácie:**



**Kompozitný PK AB**

**93. Primárny kľúč:**

**a.) je množina atribútov, pre ktorú platí pravidlo jednoznačnosti**

**b.) je množina atribútov, ktorá nejednoznačne určuje n-ticu relácie**

**c.) je množina atribútov, pre ktorú platí pravidlo jednoznačnosti a minimálnosti**

**98. Vymenujte vlastnosti transakcie:**

**a.) CRUDE = stĺpcová, referenčná, užívateľská, doménová, entitná**

**b.) ANNU = atomickosť, neusporiadanosť zľava doprava, zhora dole, jednoznačnosť**

**c.) ACID = atomickosť, konzistencia, nezávislosť, trvácnosť**

**98. Vymenujte vlastnosti RELACIE:**

**a.) CRUDE = stĺpcová, referenčná, užívateľská, doménová, entitná**

**b.) ANNU = atomickosť, neusporiadanosť zľava doprava, zhora dole, jednoznačnosť**

**c.) ACID = atomickosť, konzistencia, nezávislosť, trvácnosť**

**98. Vymenujte INTEGRITNE OBMEDZENIA:**

**a.) CRUDE = stĺpcová, referenčná, užívateľská, doménová, entitná**

**b.) ANNU = atomickosť, neusporiadanosť zľava doprava, zhora dole, jednoznačnosť**

**c.) ACID = atomickosť, konzistencia, nezávislosť, trvácnosť**

**111. Fyzická nezávislosť sa pri architektúre zabezpečuje:**

**a.) medzi externou a konceptuálnou úrovňou**

**b.) medzi konceptuálnou a internou úrovňou**

**c.) nedá sa zabezpečiť**

**112. Typ entity:**

**a.) Je objekt reálneho sveta schopný nezávislej existencie**

**b.) Vyjadruje väzbu medzi entitami**

**c.) Je množina vlastností objetov rovnakého typu**

**113. Povinné členstvo entity vo vzťahu vyjadruje IO:**

**a.) Nutnosť existencie entity pri priradení do vzťahu**

**b.) Vyjadruje vzťah medzi dvomi entitami**

**c.) Reprezentuje množinu prípustných hodnôt**

**114. Konzistencia databázy:**

**a.) Znamená, že DB je v každom okamihu správna**

**b.) Znamená opakované spracovanie**

**c.) Typovú kontrolu vstupných dát**

**115. Koncetuálna schéma obsahuje množinu dát:**

**a.) Implementačne závislú popisujúcu dátové užívateľské požiadavky aplikácie**

**b.) Implementačné závislú popisujúcu dátový model aplikácie**

**c.) Implementačné nezávislú popisujúcu dátový model aplikácie**

**116. Entita**

**a.) Je objekt reálneho sveta schopný nezávislej existencie**

**b.) Vyjadruje väzbu medzi entitami**

**c.) Je množina vlastností objektov rovnakého typu**

**118. Definujte povinné členstvo entity vo vzťahu:**

**a.) Povinnosť výskytu aspoň jedného atribútu relácie**

**b.) Hodnoty atribútov relácie musia byť zadefinované**

**c.) Povinnosť existencie entity vo vzťahu k inej entite**

**d.) Možnosť existencie entity vo vzťahu k inej entite**

**119. Príkaz ROLLBACK slúži na:**

**a.) Potvrdenie transakcie**

**b.) Vrátanie DB do pôvodného stavu ako bolo pred začatím transakcie**

**c.) Vrátenie DB do pôvodného stavu ako bola pred posledným vykonávaným príkazom**

**129. Typ entity:**

**a.) popis objektov s rovnakými charakteristikami**

**b.) objekt reálneho sveta schopný samostatnej existencie, ktorý je jednoznačne identifikovateľný**

**c.) objekt reálneho svet schopný samostatnej existencie, ktorý musí byť jednoznačne identifikovateľný**

**201. Počas vykonávania transkacie**

**a.) DB môže byť dočasne aj v nekonzistentnom tvare**

**b.) DB musí byť stále v konzistentnom tvare\***

**c.) DB nesmie byť v nekonzistentnom tvare**

**204. Perzistencia transakcie znamená:**

**a.) že potvrdené výsledky transakcií môžu byť vrátené pomocou príkazu rollback**

**b.) že potvrdené výsledky transakcií sú uložené v databáze**

**c.) že všetky zadané operácie budú vykonané**

**206. Ktorý z príkazov slúži na pridelenie práva select z tabuľky tab1 užívatelovi vajsová**

**a.) USER CAN select ON tab1 TO vajsova;**

**b.) GRANT select ON tab1 TO vajsova;**

**c.) DROP RIGHT select ON tab1 TO vajsova;**

**d.) REVOKE select ON tab1 FROM vajsova;**

**KURZORY**

**147. Čo je to kurzor:**

**a.) Kurzor je objekt, ktorý sprístupňuje záznamy v množine získanej príkazom SELECT na dalsie spracovanie**

**148. Sekvenčný kurzor je:**

**Kurzor, ktorý umožní spracovávanie záznamov len v jednom smere od začiatku do konca.**

**149. Scroll kurzor je:**

**a.) kurzor, ktorý umožní priamy prístup k vybranému záznamu**

**b.) kurzor, ktorý umožní spracovávanie záznamov len v jednom smere od začiatku do konca.**

**c.) kurzor, ktorý zobrazí naraz viacej riadkov výslednej množiny**

**185. Kde má zmysel použiť kurzor ?**

**Kurzor sa používa na sprístupnenie riadkov tabuľky**

**44. Zoraďte tieto príkazy do správneho poradia, ak cur1 je správne deklarovaný kurzor pre SELECT priezvisko FROM os\_udaje; a priezv je premenná typu os\_udaje. priezvisko%TYPE.:**

**(1) FETCH cur1 INTO priezv;**

**(2) LOOP**

**(3) IF cur1%FOUND THEN**

**(4) OPEN cur1;**

**(5) CLOSE cur1;**

**(6) ELSE**

**(7) END IF;**

**(8) EXIT;**

**(9) dbms\_output.put\_line(priezv);**

**(10) END LOOP;**

**4, 2, 1, 3, 9, 6, 8, 7, 10, 5**

**(4) OPEN cur1;**

**(2) LOOP**

**(1) FETCH cur1 INTO priezv;**

**(3) IF cur1%FOUND THEN**

**(9) dbms\_output.put\_line(priezv);**

**(6) ELSE**

**(8) EXIT;**

**(7) END IF;**

**(10) END LOOP;**

**(5) CLOSE cur1;**

**51. Majme tabuľku R s nasledovnou schémou:**

**CREATE TABLE R (id integer);**

**Aký výsledok bude po spustení nasledovných príkazov, ak tabuľka R je na začiatku prázdna?**

**INSERT INTO R values(10);**

**COMMIT;**

**DELETE FROM R;**

**ROLLBACK;**

**SELECT count(\*) FROM R;**

**a.) 10**

**b.) 0**

**c.) 1**

**51. Majme tabuľku R s nasledovnou schémou:**

**CREATE TABLE R (id integer);**

**Aký výsledok bude po spustení nasledovných príkazov, ak tabuľka R je na začiatku prázdna?**

**INSERT INTO R values(10);**

**COMMIT;**

**DELETE FROM R;**

**ROLLBACK;**

**SELECT max(\*) FROM R;**

**a.) 10**

**b.) 0**

**c.) 1**

**58. Majme pohlad:**

**CREATE VIEW v\_pohl as**

**SELECT os.\***

**FROM osoba os**

**WHERE os.id\_osoby NOT IN (SELECT zakaznik**

**FROM objednavky );**

**A nasledovný trigger:**

**CREATE TRIGGER trig**

**INSTEAD OF INSERT on v\_pohl**

**BEGIN**

**INSERT INTO log\_table VALUES (user, sysdate);**

**END;**

**Koľko riadkov bude vložených do tabuľky OSOBA, ak nasledovný príkaz prebehol bez chyby?**

**INSERT INTO v\_pohl (id\_osoby, meno, priezvisko, psc, rod\_cislo)**

**VALUES (321, 'Jana','Kratka','01001','875512/2341');**

**a.) 2**

**b.) 0**

**c.) 1**

**59. Majme tabulku R s nasledovnou schémou:**

**CREATE TABLE R (id integer);**

**Aký výsledok bude po spustení nasledovných príkazov, ak tabuľka R je na začiatku prázdna?**

**INSERT INTO R VALUES(10);**

**ROLLBACK;**

**SELECT count(\*) FROM R;**

**a.) 1**

**b.) 0**

**c.) 10**

**60. Aký výsledok bude vypísaný?**

**CREATE TABLE tab1(pom integer);**

**INSERT INTO tab1 values (0);**

**COMMIT;**

**INSERT INTO tab1 values (1);**

**SAVEPOINT sp1;**

**INSERT INTO tab1 values (2);**

**SAVEPOINT sp2;**

**INSERT INTO tab1 values (3);**

**ROLLBACK TO sp2;**

**SELECT MAX(pom) FROM tab1;**

**a.) 1**

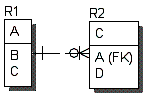
**b.) 3**

**c.) 0**

**d.) 2**

**RELAČNÁ ALGEBRA**

**187. Čo platí pre kardinalitu – card(R1) = K1, card(R2) = K2 a card(R1 JOIN R2) = K3, ak máme nasledovný dátový diagram:**



**a.) K3 = K1**

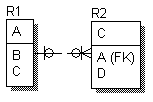
**b.) K3 = max(K1, K2)**

**c.) K3 = K2 ?**

**d.) K3 >= K2**

**e.) K3 <= K2 ak je členstvo pri R1 nepovinné (alebo môžno aj toto)**

**202. Čo platí pre kardinalitu card(R1) = K1, card(R2) = K2 a card(R1 x R2) = K3, ak máme nasledovný dátový diagram:**



**a.) K3 = K1 \* K2**

**b.) K3 >= K1**

**c.) K3 = K1 + K2**

**d.) K3 <= K1**

**202. Čo platí pre kardinalitu card(R1) = K1, card(R2) = K2 a card(R1 + R2) = K3, ak máme nasledovný dátový diagram:**

**202. Čo platí pre kardinalitu card(R1) = K1, card(R2) = K2 a card(R1 - R2) = K3, ak máme nasledovný dátový diagram:**

**31. Určite, ktorý z výrazov relačnej algebry zodpovedá nasledovnému príkazu:**

**SELECT z.os\_cislo, n.kedy, n.id\_nakupu FROM zamestnanec z, nakup n;**

**a.) πzos\_cislo (zamestnanec) x π kedy, id\_nakupu (nakup)**

**b.) zamestnanec x nakup**

**c.) πzos\_cislo(zamestnanec) x nakup**

**32. Určite výraz relačnej algebry zodpovedajúci nasledovnému príkazu:**

**SELECT id**

**FROM osoba**

**WHERE id not in (SELECT id FROM zamestnanec);**

**a.) πid(osoba – zamestnanec)**

**b.) πid(osoba) – πid(zamestnanec)**

**c.) πid(δosoba.id <> zamestnanec.id(((osoba x zamestnanec))**

**d.) θosoba.id <> zamestnanec.id(osoba, zamestnanec)**

**33. Ktoré z nasledovných výrazov sú korektné výrazy relačnej algebry:**

**πid(osoba) – πos\_cislo(zamestnanec)**

**δpopis jogurt(druh\_tovaru)**

**osoba ᴗ zamestnanec**

**πid(osoba) ᴗ πid(zamestnanec)**

**osoba x zamestnanec**

**a.) všetky výrazy**

**b.) výrazy b, d, e**

**c.) výrazy a, c, e**

**d.) ani jeden výraz**

**34. Určite výraz relačnej algebry zodpovedajúci nasledovnému príkazu:**

**SELECT id, meno FROM osoba;**

**a.) θ id, meno(osoba)**

**b.) δ id, meno(osoba)**

**c.) π id, meno(osoba)**

**035. Ak máme reláciu:**

**A (rodne\_cislo, meno, priezvisko, telefon, #osobne\_cislo, rocnik, st\_skupina, #cislo\_predmetu, nazov, pocet\_kreditov);**

**Určite závislosť osobne\_cislo – názov v relácií A je:**

**a.) neexistuje**

**b.) tranzitívna**

**c.) vzájomná**

**38. Ktoré z nasledujúcich výrazov sú korektné výrazy relačnej algebry:**

**a.) xos\_cislo(zamestnanec) ᴗ xkedy(nakup)**

**b.) xos\_cislo(zamestnanec) – xos\_cislo(nakup)**

**c.) zamestnanec ᴖ nakup**

**d.) δnazov=jogurt(druh\_tovaru)**

**e.) zamestnanec ᴗ nakup**

**a.) Všetky výrazy**

**b.) Výraz b, d (a,e nie je dobre, lebo relacie nie su union kompatibilne (nemaju totoznu mnozinu atributov), c tiez nie je dobre, lebo prienik sa musi robit podla rovnakeho atributu v oboch relaciach)**

**c.) Výraz a, b, d**

**d.) Výraz a, c, e**

**45. Určite ktorý z výrazov relačnej algebry zodpovedá nasledovnému príkazu.**

**SELECT id\_jedla, kedy**

**FROM platnost\_menu p, vydajne v**

**WHERE nazov = ‘FRI‘;**

**a.) Πid\_jedla, kedy(δnazov =´FRI´(platnost\_menu x vydajne))**

**b.) Πid\_jedla, kedy(δnazov =´FRI´(platnost\_menu JOIN vydajne))**

**c.) Πid\_jedla, kedy(platnost\_menu JOIN (δnazov =´FRI´(vydajne)))**

**47. Určite ktorý z výrazov relačnej algebry zodpovedá nasledovnému príkazu:**

**SELECT id\_jedla**

**FROM jedlo**

**WHERE id\_jedla NOT IN (SELECT id\_jedla**

**FROM platnost\_menu**

**WHERE id\_vydajne = 3);**

**a.) πid\_jedla(jedlo) JOIN (πid\_jedla(δid\_vydajne = 3(platnost\_menu)))**

**b.) πid\_jedla(jedlo) - πid\_jedla(δid\_vydajne = 3(platnost\_menu))**

**c.) πid\_jedla(jedlo) - δid\_vydajne = 3(platnost\_menu)))**

**48. Ktoré z nasledovných výrazov sú korektné výrazy relačnej algebry:**

**1.) πid\_stravnika(stravnik) – objednavky**

**2.) πid\_jedla(jedlo) - πid\_jedla(objednavky)**

**3.) δnazov like ´Buchty%´(jedlo JOIN objednavky)**

**4.) objednavky ᴖ platnost\_menu**

**5.) πid\_stravnika(objednavky) ᴗ πid\_stravnika(stravnik)**

**a.) všetky výrazy**

**b.) výrazy 2, 4**

**c.) výrazy 2, 3, 5**

**d.) výrazy 1, 2, 4**

**50. Určite ktorý z výrazov relačnej algebry zodpovedá nasledovné príkazu:**

**SELECT s.meno, s.priezvisko, o.kedy**

**FROM jedlo j, objednavky o, stravnik s**

**WHERE j.id\_jedla = o.ide\_jedla**

**AND o.id\_stravnika = s.id\_stravnika**

**AND j.nazov like ´Kur%´;**

**a.) πs.meno,s.priezvisko,o.kedy(δnazov like ´Kur%´(jedlo JOIN objednavky JOIN stravnik)**

**b.) δnazov like ´Kur%´(jedlo JOIN πo.kedy,id\_jedla(objednavky)) JOIN πs.meno,s.priezvisko (stravnik)**

**c.) δnazov like ´Kur%´(jedlo JOIN πo.kedy(objednavky)) JOIN πs.meno,s.priezvisko (stravnik)**

**56. Určite výraz relačnej algebry zodpovedajúci nasledovnému prikazu:**

**SELECT id**

**FROM osoba**

**WHERE id NOT IN (SELECT id**

**FROM zamestnanec);**

**a.) Θosoba.id<>zamestnanec.id(osoba, zamestnanec)**

**b.) Πid(osoba – zamestnanec)**

**c.) Πid(δosoba.id<>zamestnanec.id(osoba x zamestnanec))**

**d.) Πid(osoba) – πid(zamestnanec)**

**57. Určite výraz relačnej algebry zodpovedajúci nasledovnému prikazu:**

**SELECT id\_osoby**

**FROM os\_udaje**

**WHERE id\_osoby NOT IN (SELECT id\_osoby**

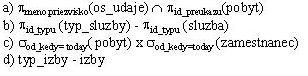
**FROM zamestnanec);**

**a.) Πid\_osoby(os\_udaje) – πid\_osoby(zamestnanec)**

**b.) Os\_udaje – πid\_osoby(zamestnanec)**

**c.) Πid\_osoby(os\_udaje) ᴗ πid\_osoby(zamestnanec)**

**63. Ktoré z nasledujúcich výrazov sú korektné výrazy relačnej algebry:**



**a.) výrazy a,c,d b.) výrazy b,c c.) výrazy a,d d.) výrazy b,d**

**TRIGGRE A POHĽADY**

**192. Ak máme trigger ON objednavka, potom v tele triggra príkaz INSERT INTO objednava**

**a.) môžeme použiť len v prípade riadkového triggra**

**b.) môžeme vždy použiť**

**c.) nikdy nemôžeme použiť**

**8. Ak máme nasledovný pohľad:**

**CREATE OR REPLACE VIEW zamest**

**AS**

**SELECT os\_cislo, meno, priezvisko, pozicia**

**FROM osoba, zamestnanec**

**WHERE osoba.id=zamestnanec.id;**

**Aký trigger budete potrebné definovať, aby operácia INSERT INTO zamest fungovala správne ?**

**a.) BEFORE alebo AFTER INSERT**

**b.) INSTEAD OF INSERT**

**c.) AFTER INSERT**

**d.) BEFORE INSERT**

**9. Majme nasledovný pohľad:**

**CREATE OR REPLACE VIEW pohl4 (meno, priezvisko, rocnik, os\_cislo)**

**AS**

**SELECT ou\_meno, ou\_priezvisko, st\_rocnik, st\_os\_cislo**

**FROM os\_udaje, student**

**WHERE ou\_rod\_cislo = st\_ou\_rod\_cislo;**

**Aký typ triggra je potrebné definovať, aby operácia INSERT z pohľadu fungovala korektne ?**

**a.) BEFORE insert**

**b.) nie je možná operácia INSERT**

**c.) nie je potrebné definovat žiadny trigger**

**d.) AFTER insert**

**10. Majme nasledovný pohľad:**

**CREATE OR REPLACE VIEW pohl4 (meno, priezvisko, rocnik, rod\_cislo, os\_cislo)**

**AS**

**SELECT ou\_meno, ou\_priezvisko, st\_rocnik, ou\_rod\_cislo, st\_os\_cislo**

**FROM os\_udaje, student**

**WHERE ou\_rod\_cislo = st\_ou\_rod\_cislo;**

**WITH READ ONLY;**

**Aký typ triggra je potrebné definovať, aby operácia DELETE z pohľadu fungovala korektne ?**

**a.) DELETE bude vždy fungovať aj bez triggra**

**b.) AFTER delete**

**c.) BEFORE delete**

**d.) nie je možná operácia DELETE**

**11. Majme nasledovný pohľad a príkaz Insert:**

**CREATE OR REPLACE VIEW pohl6 (meno, priezvisko, rod\_cislo)**

**AS**

**SELECT ou\_meno, ou\_priezvisko, ou\_rod\_cislo**

**FROM os\_udaje**

**WHERE ou\_meno LIKE 'S%';**

**INSERT INTO pohl6**

**VALUES ( 'Karol', 'Novy', '790502/1212');**

**Čo sa stane ?**

**a.) riadok bude vložený iba do pohľadu**

**b.) riadok bude vložený iba do tabuľky os\_udaje**

**c.) riadok bude vložený do pohľadu aj do tabuľky os\_udaje**

**d.) riadok nebude nikam vložený – nastane chyba**

**12. Majme nasledovný pohľad a príkaz Insert:**

**CREATE OR REPLACE VIEW pohl6 (meno, priezvisko, rod\_cislo)**

**AS**

**SELECT ou\_meno, ou\_priezvisko, ou\_rod\_cislo**

**FROM os\_udaje**

**WHERE ou\_meno LIKE 'S%';**

**WITH CHECK OPTION;**

**INSERT INTO pohl6**

**VALUES ( 'Karol', 'Novy', '790502/1212');**

**Čo sa stane ?**

**a.) riadok bude vložený iba do pohľadu**

**b.) riadok bude vložený iba do tabuľky os\_udaje**

**c.) riadok bude vložený do pohľadu aj do tabuľky os\_udaje**

**d.) riadok nebude nikam vložený – nastane chyba**

**13. Majme nasledovný pohľad a príkaz Insert.**

**CREATE OR REPLACE VIEW pohl6 (meno, priezvisko, rod\_cislo)**

**AS**

**SELECT ou\_meno, ou\_priezvisko, ou\_rod\_cislo**

**FROM os\_udaje**

**where ou\_meno like 'S%'**

**WITH CHECK OPTION;**

**INSERT INTO pohl6**

**VALUES ( 'Stano', 'Novy', '790502/1212');**

**Aký typ triggra je potrebné definovať, aby predchádzajúci INSERT do pohľadu fungoval korektne ?**

**(Ak v tabuľke os\_udaje sú práve 3 NOT NULL stĺpce – ou\_meno, ou\_priezvisko, ou\_rod\_cislo)**

**a.) nie je potrebné definovať žiadny trigger**

**b.) nie je možná operácia INSERT**

**c.) BEFORE insert**

**d.) INSTEAD OF insert**

**e.) AFTER insert**

**14. Majme nasledovný pohľad a príkaz Insert.**

**CREATE OR REPLACE VIEW pohl6 (meno, priezvisko,rod\_cislo, os\_cislo)**

**AS**

**SELECT ou\_meno, ou\_priezvisko, ou\_rod\_cislo, st\_os\_cislo**

**FROM os\_udaje, student**

**where ou\_rod\_cislo = st\_ou\_rod\_cislo;**

**INSERT INTO pohl6**

**VALUES ( 'Stano', 'Novy', '790502/1212',5269);**

**Aký typ triggra je potrebné definovať, aby predchádzjúci INSERT do pohľadu fungoval korektne? (Ak v tabuľke os\_udaje sú práve 3 NOT NULL stĺpce – ou\_meno, ou\_priezvisko, ou\_rod\_cislo a v tabuľke student len st\_ou\_rod\_cislo a st\_os\_cislo)**

**a.) nie je možná operácia INSERT**

**b.) BEFORE insert**

**c.) INSTEAD OF insert**

**d.) AFTER insert**

**15. Majme nasledovný trigger:**

**CREATE OR REPLACE TRIGGER t\_log\_zp**

**........ UPDATE ON zap\_predmety**

**FOR EACH ROW**

**BEGIN**

**INSERT INTO log\_table\_zp**

**VALUES (user, sysdate);**

**END;**

**Doplňte do Triggra ?**

**a.) BEFORE, alebo AFTER**

**b.) len INSTEAD OF**

**c.) len AFTER**

**d.) len BEFORE**

**16. Majme nasledovný trigger:**

**CREATE OR REPLACE TRIGGER t\_log\_zp**

**AFTER UPDATE ON zap\_predmety**

**FOR EACH ROW**

**- ak operácia pracuje s viacerými riadkami relácie tak trigger sa vykoná pre každý riadok zvlášť**

**BEGIN**

**INSERT INTO log\_table\_zp**

**VALUES (USER, SYSDATE);**

**END;**

**Koľko riadkov bude vložených do tabuľky log\_table\_zp, ak operácia UPDATE zap\_predmety modifikovala 10 riadkov ?**

**a.) 1 riadok**

**b.) ani jeden riadok**

**c.) 10 riadkov**

**d.) 2 riadky**

**17. Majme nasledovný trigger:**

**CREATE OR REPLACE TRIGGER t\_log\_zp**

**AFTER UPDATE ON zap\_predmety**

**BEGIN**

**INSERT INTO log\_table\_zp**

**VALUES (USER, SYSDATE);**

**END;**

**Koľko riadkov bude vložených do tabuľky log\_table\_zp, ak operácia UPDATE zap\_predmety modifikovala 10 riadkov ?**

**a.) 1 riadok**

**b.) ani jeden riadok**

**c.) 10 riadkov**

**d.) 2 riadky**

**18. Majme nasledovný trigger, ktorý zabezpečí kaskádu pre operáciu DELETE študenta:**

**CREATE OR REPLACE TRIGGER ST\_DEL\_CASCADE**

**.......... DELETE ON STUDENT**

**FOR EACH ROW**

**BEGIN**

**DELETE FROM zap\_predmety**

**WHERE zp\_st\_os\_cislo =:old.st\_os\_cislo**

**END;**

**Doplňte do tela triggra ?**

**a.) AFTER**

**b.) INSTEAD OF**

**c.) BEFORE (predtym, nez vymazeme studenta z tabulky student musime najprv vymazat cudzi kluc v tabulke zap\_predmety)**

**d.) BEFORE alebo AFTER**

**19. Majme nasledovný trigger, ktorý má automaticky nastaviť atribúty uživ a dátum hodnotami kto a kedy naposledy menil daný riadok:**

**CREATE OR REPLACE TRIGGER zap\_predmety\_log**

**......... INSERT OR UPDATE ON zap\_predmety**

**REFERENCING new as novy**

**FOR EACH ROW**

**BEGIN**

**SELECT user, sysdate into :novy.uziv, :novy.datum\_zm**

**FROM dual;**

**END;**

**Doplňte do Triggra ?**

**a.) BEFORE, AFTER, alebo INSTEAD OF**

**b.) BEFORE**

**c.) BEFORE alebo AFTER**

**d.) AFTER (pozri otazku 17. (dve otazky vyssie); keby sme dali BEFORE a insert by bol zly tj. nic by nevlozil do tabulky zap\_predmety, do logovacej tabulky by sa napriek tomu ulozil zaznam o vlozeni; vid tiez https://docs.oracle.com/database/121/TDDDG/tdddg\_triggers.htm#TDDDG52100)**

**20. Majme nasledovný trigger, ktorý má automaticky nastaviť atribúty uziv a datum hodnotami kto a kedy naposledy menil daný riadok.**

**CREATE OR REPLACE TRIGGER zap\_predmety\_log**

**......... INSERT OR UPDATE ON zap\_predmety**

**REFERENCING new as novy**

**FOR EACH ROW**

**BEGIN**

**SELECT user, sysdate into :novy.uziv, :novy.datum\_zm FROM dual;**

**END;**

**Doplňte do tela triggra ?**

**a.) AFTER**

**b.) INSTEAD OF**

**c.) BEFORE**

**d.) BEFORE alebo AFTER**

**21. Majme nasledovný trigger, ktorý má automaticky nastaviť atribúty uziv a datum hodnotami kto a kedy naposledy menil daný riadok:**

**CREATE OR REPLACE TRIGGER t\_log\_zp**

**........ update on zap\_predmety**

**BEGIN**

**insert into log\_table\_zp**

**values (user, sysdate);**

**End;**

**Doplňte do tela triggra ?**

**a.) Before**

**b.) After**

**c.) Before alebo After**

**22. Je možné v procedúre použiť nasledovný SELECT – príkaz:**

**SELECT \* FROM zamestnanec;**

**Vysvetlite prečo a kedy áno alebo nie ?**

**Nie, pretože ani v trigroch ani v procedúrach nemôžeme požívať SELECT. Taktiež nemôžeme používať COMMIT ROOLBACK.**

**Použiť môžeme z DML – INSERT, UPDATE a DELETE. Ďalej SELECT INTO.**

**23. Aký je problém s daným pohľadom?**

**CREATE VIEW v\_semdni**

**AS**

**SELECT \***

**FROM semina sem JOIN den\_seminara den ON (sem.id\_seminara = den.id\_seminara);**

**a.) Tento pohľad nie je ani možné vytvoriť.**

**b.) V selecte je syntaktická chyba.**

**c.) Nie je nijaký problém.**

**24. Ak máme definovaný pohľad**

**CREATE VIEW v\_predava**

**AS**

**SELECT \***

**FROM osoba o JOIN predavac p ON(o.id\_osoby = p.id\_osoby);**

**a.) Tento pohľad nie je ani možné vytvoriť.**

**b.) V selecte je syntaktická chyba.**

**c.) Nie je nijaký problém.**

**25. Majme nasledovný trigger:**

**CREATE OR REPLACE TRIGGER t\_log\_zp**

**............... UPDATE ON zap\_predmety**

**FOR EACH ROW**

**BEGIN**

**INSERT INTO log\_table\_zp**

**VALUES (user, sysdate);**

**END;**

**Doplňte do Triggra ?**

**a.) BEFORE, alebo AFTER**

**b.) len INSTEAD OF**

**c.) len AFTER\***

**d.) len BEFORE**

**26. Majme nasledovný pohľad a príkaz Insert:**

**CREATE OR REPLACE VIEW pohlou (meno, priezvisko)**

**AS**

**SELECT meno, priezvisko**

**FROM osoba**

**WHERE meno LIKE 'P%';**

**INSERT INTO pohlou**

**VALUES ( 'Peter', 'Novy');**

**Čo sa stane ?**

**a.) riadok bude vložený iba do pohľadu**

**b.) riadok bude vložený iba do tabuľky osoba**

**c.) riadok bude vložený do pohľadu aj do tabuľky osoba**

**d.) riadok nebude nikam vložený – nastane chyba**

**27. Majme nasledovný trigger, ktorý má automaticky nastaviť atribúty uživ a dátum hodnotami kto a kedy naposledy menil daný riadok:**

**CREATE OR REPLACE TRIGGER zap\_predmety\_log**

**......... INSERT OR UPDATE ON zap\_predmety**

**REFERENCING new AS novy**

**FOR EACH ROW**

**BEGIN**

**SELECT user, sysdate into :novy.uziv, :novy.datum\_zm**

**FROM dual;**

**END;**

**Doplňte do Triggra ?**

**a.) BEFORE, AFTER, alebo INSTEAD OF**

**b.) BEFORE**

**c.) BEFORE alebo AFTER**

**28. Majme nasledovný trigger, ktorý má automaticky nastaviť atribúty uziv a datum hodnotami kto a kedy naposledy menil daný riadok:**

**CREATE OR REPLACE TRIGGER t\_log\_zp**

**........ update on zap\_predmety**

**BEGIN**

**insert into log\_table\_zp**

**values (user,sysdate)**

**End;**

**Doplňte do tela triggra ?**

**a.) Before**

**b.) After**

**c.) BEFORE alebo AFTER**

**58. Majme pohlad:**

**CREATE VIEW v\_pohl as**

**SELECT os.\***

**FROM osoba os**

**WHERE os.id\_osoby NOT IN (SELECT zakaznik**

**FROM objednavky );**

**A nasledovný trigger:**

**CREATE TRIGGER trig**

**INSTEAD OF INSERT on v\_pohl**

**BEGIN**

**INSERT INTO log\_table VALUES (user, sysdate);**

**END;**

**Koľko riadkov bude vložených do tabuľky OSOBA, ak nasledovný príkaz prebehol bez chyby?**

**INSERT INTO v\_pohl (id\_osoby, meno, priezvisko, psc, rod\_cislo)**

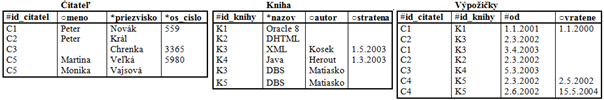
**VALUES (321, 'Jana','Kratka','01001','875512/2341');**

**a.) 2**

**b.) 0**

**c.) 1**

**SELECTY**



**Napíšte aký bude výsledok nasledovného SQL príkazu:**

**a.) *SELECT count(\*)* 5**

***FROM citatel***

**b.) *SELECT count(os\_cislo)* 3**

***FROM citatel***

**c.) *SELECT count(\*)* 3**

***FROM citatel***

***WHERE os\_cislo IS NOT NULL***

**d.) *SELECT count(DISTINCT id\_citatel)* 4**

***FROM citatel***

**e.) *SELECT count(DISTINCT os\_cislo)* 1**

***FROM citatel* 1**

***GROUP BY meno* 1**

**0**

**f.) *SELECT COUNT(vratene)* 3**

***FROM vypozicky;***

**g.) *SELECT count(\*)* 1**

***FROM vypozicky* 2**

***GROUP BY id\_knihy;* 1**

**1**

**2**

**h.) *SELECT count(id\_knihy)* 2**

***FROM vypozicky* 2**

***GROUP BY id\_citatel;* 1**

**2**

**i.) *SELECT count(DISTINCT id\_knihy)* 5**

***FROM vypozicky***

**j.) *SELECT count(\*)* 1**

***FROM Kniha* 1**

***GROUP BY pocet\_kusov*; 1**

**1**

**0**

**k.) *SELECT count(\*)* 2**

***FROM citatel, vypozicky* 2**

***WHERE citatel.cislo\_citatela = vypozicky.cislo\_citatela* 1**

***GROUP BY vypozicky.cislo\_citatela;* 0**

**0**

**l.) *SELECT count(\*)* C1 2**

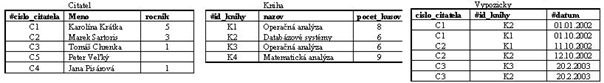
***FROM kniha, vypozicky* C2 2**

***WHERE kniha.id\_knihy = vypozicky.id\_knihy* C3 3**

***GROUP BY cislo\_citatela;***

**39. Aký bude výsledok nasledovného selectu:**

**SELECT count(\*) FROM kniha GROUP BY pocet\_kusov;**



**1, 2, 1**

**42. Aký bude výsledok nasledovného selectu:**

**SELECT count(\*)**

**FROM citatel , vypozicky**

**WHERE citatel.cislo\_citatela = vypozicky.cislo\_citatela**

**GROUP BY vypozicky\_cislo\_citatela;**

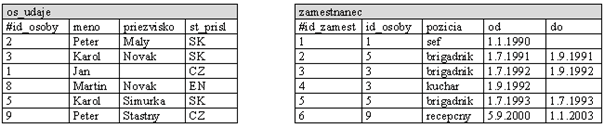
**43. Aký bude výsledok nasledovného selectu:**

**SELECT count(\*)**

**FROM os\_udaje o, zamestnanec z**

**WHERE o.id\_osoby = z.id\_osobv**

**GROUP BY o.id\_osoby**



**49. Opravte nasledovný SQL príkaz, tak aby ho bolo možné vykonať:**

**SELECT id\_stravnika, meno, priezvisko**

**FROM stravnik**

**WHERE subst(rod\_cislo, 3, 1) in (´5´,´6´)**

**UNION**

**SELECT id\_stravnika**

**FROM objednavky;**

**SELECT id\_stravnika, meno, priezvisko**

**FROM stravnik**

**WHERE substr(rod\_cislo,3,1) in ('5','6')**

**UNION**

**SELECT id\_stravnika, meno, priezvisko**

**FROM objednavky;**

**55. Opravte nasledovný SQL príkaz, tak aby ho bolo možné vykonať:**

**SELECT meno, priezvisko, 0**

**FROM os\_udaje ou, zamestanenc z**

**WHERE ou, id\_osoby = z.id:\_osoby**

**UNION**

**SELECT meno, priezvisko, count(\*)**

**FROM os\_udaje ou, pobyt p**

**WHERE ou.id\_osoby = p.id\_osoby**

**SELECT meno, priezvisko, count(\*)**

**FROM os\_udaje ou, zamestanenc z**

**WHERE ou, id\_osoby = z.id:\_osoby**

**UNION**

**SELECT meno, priezvisko, count(\*)**

**FROM os\_udaje ou, pobyt p**

**WHERE ou.id\_osoby = p.id\_osoby**

**37. Je možne v procedúre použiť nasledovný selec-príkaz:**

**SELECT MAX(os\_cislo) INTO p\_oc FROM zamestnanec;**

**a.) Áno**

**b.) Nie**