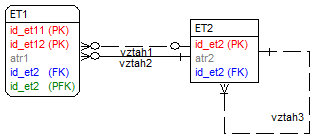
**Otázky na skúšku**

1. Časť

**1. Ako treba opraviť daný entitný diagram?**



1. do ET1 pribudnú 2 atribúty FK, ktoré nebudú súčasťou PK v ET1; do ET2 pribudne 1 atribút FK, ktorý nebude súčasťou PK
2. do ET1 pribudnú 2 atribúty FK, ktoré budú sú časťou PK v ET1; do ET2 pribudne 1 atribút FK, ktorý nebude súčasťou PK
3. do ET1 pribudne 1 atribút FK, ktorý sa stane časťou PK v ET1 a 1 atribút FK, ktorý nebude súčasťou PK v ET1; do ET2 pribudne 1 atribút FK, ktorý nebude súčasťou PK
4. do ET1 pribudne 1 atribút FK, ktorý nebude súčasťou PK v ET1; do ET2 pribudne 1 atribút FK, ktorý bude súčasťou PK v ET2 a 1 atribút FK, ktorý nebude súčasťou PK v ET2

**2. Pri vytvorení tabuľky sa index implicitne vytvára:**

1. nad všetkými stĺpcami tabuľky, ktoré sú definované ako UNIQUE
2. len nad primárnym kľúčom
3. nad cudzími kľúčmi a primárnym kľúčom
4. nad všetkými stĺpcami tabuľky

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



R1(#PK2, akcia)

R2(#PK1, meno, PK2(FK), ako, co)

**4. Príkazy BEGIN WORK, COMMIT WORK, ROLLBACK WORK sú príkazy:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| DDL | Definícia objektov | Create | After | Drop |  |
| DML | Manipulácia s dátami | Insert | Update | Delete | Select |
| DAS | Prístup k dátam | Grant | Revoke |  |  |
| DIS | Správa transakcií | Commit | Rollback | Set transaction |  |

1. jazyka pre manipuláciu s dátami
2. jazyka pre riadenie prístupu k dátam
3. jazyka pre definíciu dát
4. jazyka pre riadenie správy transakcií

**5. Aké druhy nezávislosti poznáte:**

1. konceptuálnu a logickú
2. fyzickú a logickú
3. internú a konceptuálnu
4. nezávislosť od vstupov a výstupov

**6. Ak vzťah ET2 ISA ET1 je vzťah ISA. Čo musí platiť pre primárne(PK) a cudzie(FK) kľuče týchto entitných typov ?**



1. ET2 dedí PK z ET1
2. ET1 dedí PK z ET2
3. PK z ET1 sa stane len FK v ET2, ET2 musí mať vlastný PK
4. PK z ET2 sa stane súčasťou kompozitného PK v ET1

**7. Aká je skutočná kardinalita vzťahu?**



1. 1 1
2. 1:M
3. identifikačná
4. povinná

**8. Kardinalita vzťahu je:**

1. integritné obmedzenie, ktoré vyjadruje možnosť neexistencie entity jedného typu vo vzťahu k existencii entity iného typu
2. integritné obmedzenie, ktoré vyjadruje prípustný počet entít vo vzťahu
3. integritné obmedzenie, ktoré vyjadruje prípustný počet domén vo vzťahu
4. integritné obmedzenie, ktoré vyjadruje nutnosť existencie entity jedného typu vo vzťahu k existencii entity iného typu

**9. What is the purpose of a Unique Identifier?**

1. To identify a specific row within a table, using one or more columns and/or foreign keys.
2. To uniquely determine a table and columns within that table.
3. To identify one unique instance of an entity, by using one or more attributes and/or relationships.
4. Create an entity that is unlike any other entity aside from itself.

**10. Vymenujte vlastnosti relácie:**

Neobsahuje duplicitné n- tice

N- tice sú neusporiadané (zhora dolu)

M- tice atribútov sú neusporiadané (zľava doprava)

Hodnoty atribútov sú atomické

**11. Konceptuálna schéma je:**

1. implementačne nezávislá množina dát popisujúca dátový model
2. implementačne nezávislá množina dát, ktorá presne popisuje dátové štruktúry a prístupné metódy uložených dát v externej pamäti
3. implementačne závisl množina dát popisujúca dátový model
4. implementačne nezávislá množina dát popisujúca užívateľské pohľady aplikácie

**12. Aké je správne poradie pre load dát do týchto tabuliek?**



1. ET1,ET4,ET2,ET3,ET5
2. ET4,ET1,ET3,ET2,ET5
3. ET4,ET2,ET1,ET3,ET5
4. ET5,ET3,ET2,ET1,ET4

**13. Ktoré z príkazov slúžia na vytvorenie nasledovného vzťahu medzi ET1 a ET2?**



1. ALTER TABLE ET2 ADD (FOREIGN KEY (atr1, atr2) REFERENCES ET1);
2. ALTER TABLE ET2 ADD ( PRIMARY KEY (atr1, atr2));
3. ALTER TABLE ET2 ADD ( FOREIGN KEY (atr1) REFERENCES ET1);

ALTER TABLE ET2 ADD ( FOREIGN KEY (atr2) REFERENCES ET1);

1. ALTER TABLE ET1 ADD (FOREIGN KEY (atr1, atr2) REFERENCES ET2);

**14. In a physical data model, an attribute is represented as a/an**

1. Foreign Key
2. Row
3. Column
4. Instance

**15. 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, PK1(FK), kedy)

**16. Ktorý z obrázkov je správna dekompozícia M:N vzťahu?**



**17. Aká je výsledná kardinalita vzťahov medzi ET1 a ET2?**



1. 1:M
2. 1 1
3. M:N

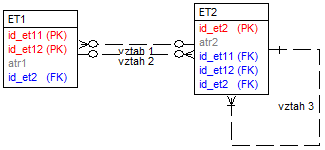
**18. Aký je vzťah medzi reláciou, atribútmi a primárnym kľúčom?**

1. PK a atribúty sú vo vzájomnej relácii
2. PK je podmnožina množiny atribútov v relácii
3. Všetky atribúty relácie musia spolu tvoriť PK
4. PK je práve jeden z atribútov v relácii

**19. Vysvetlite rozdiel medzi vstupnými, výstupnými a perzistentnými dátami**

1. Vstupné sa môžu stať perzistentnými, z perzistentných a vstupných vznikajú výstupné
2. Vstupné vstupujú do systému, výstupné zo systému vystupujú a perzistentné dáta sú parametre aplikácie
3. Všetky vstupné sa stávajú perzistentnými, z perzistentných dát vznikajú výstupné

**20. Ako treba opraviť daný entitný diagram ?** 



1. do ET2 pribudne 5 atribútov FK, ktoré nebudú súčasťou PK v ET2
2. do ET1 pribudne 1 atribút FK, do ET2 pribudnú 3 atribúty FK
3. nie je potrebné nič upraviť
4. do ET1 pribudnú 2 atribúty FK, ktoré nebudú súčasťou PK v ET1, do ET2 pribudne 1 atribút FK, ktorý nebude súčasťou PK v ET1 a jeden atribút FK, ktorý bude súčasťou PK v ET2

**21. Aké je správne poradie pre load dát do týchto tabuliek?**



1. ET1,ET5,ET4,ET2,ET3
2. ET3,ET2,ET4,ET5,ET1
3. ET3,ET2,ET5,ET4,ET1
4. ET2,ET3,ET1,ET4,ET5

**22. When resolving an M:M relationship, the new relationships will always be \_\_\_\_\_\_\_\_\_\_ on the many side.**

1. recursive
2. qredundant
3. mandatory
4. optional

**23. Určite stupeň(s) a kardinalitu(k) nasledovnej relácie**

|  |  |
| --- | --- |
| Stupeň: | Podľa stĺpca teda počtu atribútov |
| Kardinalita: | Podľa riadkov teda počtu entít |



1. s=5,k=3
2. s=5,k=2
3. s=3,k=5
4. s=2,k=5

**24. ISA vzťahy v ERA diagrame:**

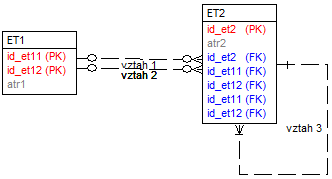
1. tvoria orientovaný cyklus
2. netvoria orientovaný cyklus
3. tvoria neorientovaný cyklus
4. tvoria úplný graf

**25. Ktorá množina príkazov je správna pre vytvorenie vzťahu medzi ET1 a ET2 (Pozor na kardinalitu a povinnosť členstva):**



1. CREATE TABLE ET2( id1 integer NOT NULL PRIMARY KEY, id2 integer NOT NULL PRIMARY KEY, hodnota smallint NULL); ALTER TABLE ET2 ADD( FOREIGN KEY(id2) REFERENCES ET1);
2. CREATE TABLE ET2( id2 integer NOT NULL, id1 integer NOT NULL, hodnota smallint NULL, primary key(id2), FOREIGN KEY(id1) REFERENCES ET1); CREATE UNIQUE INDEX et2\_id1 ON ET2(id1);
3. CREATE TABLE ET2( id2 integer NOT NULL, id1 integer NULL, hodnota smallint NULL, primary key (id2), FOREIGN KEY (id1) REFERENCES ET1 ); CREATE INDEX et2\_id1 ON ET2(id1);
4. CREATE TABLE ET2( id2 integer NOT NULL PRIMARY KEY, id1 integer NOT NULL, hodnota smallint NULL ); ALTER TABLE ET1(FOREIGN KEY(id1) REFERENCES ET2)

**26. Ako treba optaviť daný entitný diagram ?**



1. do ET2 pribudnú: 2 atribúty FK, ktoré budú súčasťou PK v ET2 a 1 atribút FK, ktorý nebude súčasťou PK v ET2
2. do ET2 pribudnú: 1 atribút FK, ktorý bude súčasťou PK v ET2 a 2 atribúty FK, ktoré nebudú súčasťou PK v ET2
3. do ET2 pribudnú: 1 atribút FK, ktorý bude súčasťou PK v ET2 a 4 atribúty FK, ktoré nebudú súčasťou PK v ET2
4. do ET2 pribudne 5 atribútov FK, ktoré nebudú súčasťou PK v ET2

**27. Unigue Identifiers .....**

1. distinguish all entities in a database
2. distinguish one entity from another
3. distinguishes nothing
4. distinguish one instance of an entity from all other instances of that entity

**28. Aký je stupeň (s) a kardinalita (k) relácie R1.**



1. s=4, k=M
2. s=2, k=4
3. s=4, k nie je možné určiť
4. s=2, k=M

**29. Aké je správne poradie pre drop nasledovných tabuliek?**



1. ET2,ET1,ET5,ET4,ET3
2. ET4,ET1,ET2,ET3,ET5
3. ET5,ET3,ET1,ET2,ET4
4. ET1,ET2,ET3,ET4,ET5

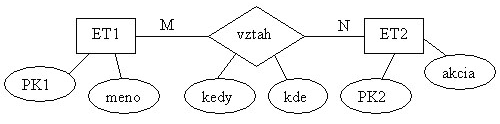
**30. Ktorý z príkazov patrí do DAS:**

1. REVOKE
2. DROP TABLE
3. BEGIN WORK
4. ROLLBACK WORK

**31. Stupeň relácie je:**

Počet atribútov n v relačnej schéme R={A1, A2, ...., An}

**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)

**33. Ktorý z príkazov patrí do DAS:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| DDL | Definícia objektov | Create | After | Drop |  |
| DML | Manipulácia s dátami | Insert | Update | Delete | Select |
| DAS | Prístup k dátam | Grant | Revoke |  |  |
| DIS | Správa transakcií | Commit | Rollback | Set transaction |  |

1. CREATE UNIQUE INDEX
2. DROP TABLE
3. REVOKE
4. INSERT

**34. Which of the following entities most likely contains invalid attributes?**

1. Entity: Mother. Attributes: Name, Birthdate, Occupation, Salary
2. Entity: Car. Attributes: Owner Occupation, Owner Salary, Speed
3. Entity: Pet. Attributes: Name, Birthdate, Owner
4. Entity: Home. Attributes: Number of Bedrooms, Owner, Address, Date Built

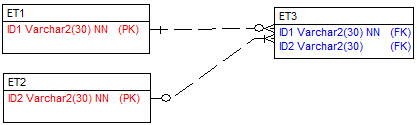
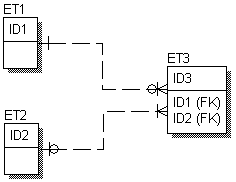
**35. Atribút ADRESA(ŠTÁT, OBEC, ULICA, PSČ) je:**

1. determinant
2. skupinový atribút
3. rekurzívny atribút
4. atomický atribút

**36. What uncommon relationship is described by the statements: “Each LINE must consist of many POINTS and each POINT must be a part of many LINES”**

1. Many to Many Optional
2. One to Many Mandatory
3. One to Many Optional
4. Many to Many Mandatory

**37. Ak máme nasledovný model, potom v tabuľke ET3 budú atribúty ID2, ID1:**



1. ID1 - NOT NULL, ID2 - NOT NULL
2. ID1 - NOT NULL, ID2 - NULL
3. ID1 - NULL, ID2 - NULL
4. ID1 - NULL, ID2 - NOT NULL

**38. Externá schéma je:**

1. implementačne nezávislá množina dát popisujúca užívateľské pohľady aplikácie
2. implementačne závislá množina dát, ktorá presne popisuje dátové štruktúry a prístupové metódy uložených dát v externej pamäti
3. implementačne nezávislá množina dát, ktorá presne popisuje dátové štruktúry a prístupné metódy uložených dát v externej pamäti
4. implementačne závislá množina dát popisujúca dátový model

**39. Pri vytvorení tabuľky sa duplikátny index implicitne vytvára:**

1. pre každý stĺpec PK
2. zvlášť pre každý stĺpec, ktorý tvorí FK zo vzťahu 1:N
3. pre každý stĺpec, alebo množinu stĺpcov, ktoré tvoria PK alebo FK
4. pre množiny stĺpcov, ktoré tvoria FK zo vzťahu 1:N

**40. Ktorý z príkazov nepatri do DDL:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| DDL | Definícia objektov | Create | After | Drop |  |
| DML | Manipulácia s dátami | Insert | Update | Delete | Select |
| DAS | Prístup k dátam | Grant | Revoke |  |  |
| DIS | Správa transakcií | Commit | Rollback | Set transaction |  |

1. ALTER TABLE
2. SELECT
3. CREATE TABLE
4. CREATE INDEX

**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.

**42. Logická nezávislosť dát je:**

1. Nezávislosť dát od prístupových metód
2. Nezávislosť dátového modelu na programe
3. Nezávislosť programu od organizácie dát

**43. V ktorých prípadoch je prípustný viacnásobný výskyt rovnakých dát:**

1. vždy
2. pri riadenej redundancii
3. nikdy

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

1. len interná
2. externá aj interná
3. aplikačná a externá
4. len externá

**45. Perzistentné dáta sú:**

1. dáta, ktoré zanikajú po ukončení programu
2. dáta existujúce aj po ukončení programu
3. odvodené dáta

**46. Popíšte vlastnosti relácie:**

1. CRUDE = stĺpcová, referenčná, užívateľská, domenová, entitná
2. ACID = atomickosť, konzistencia, nezávislosť, trvácnosť
3. ANNU = atomickosť, neusporiadanosť zľava doprava, neusporiadanosť zhora dole, jednoznačnosť

**47. Definujte cudzí klúč:**

1. Množina atribútov, ktorá spĺňa podmienku jednoznačnosti a minimálnosti a je definovaná v inej relácii
2. Množina atribútov, ktorá spĺňa podmienku jednoznačnosti a minimálnosti
3. Množina atribútov, ktorá je definovaná v inej relácii ako kandidát primárneho kľúča

**48. Integrita entít hovorí:**

1. atribút primárneho kľúča môže nadobúdať NULL hodnoty
2. atribút primárneho kľúča nesmie nadobúdať NULL hodnoty
3. 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  • Usporiada teľnosť |
| Integrita stĺpcov | Pre každý stĺpec tabuľky je nutné definovať doménu a prípadne dodatočné integritné obmedzenia. Stĺpcov IO:  • 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. |

**49. Uveďte príklad užívateľskej integrity:**

Dátum skúšky väčší ako dátum.

Kontrola opätovného zapísania predmetu študentovi.

Keď kniha bola vrátená skôr ako bola vypožičaná.

**50. 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:**

1. hodnota FK v relácii R2 sa nemusí rovnať hodnote PK z relácie R1
2. FK je množina atribútov definovaná v relácii R2, ktorá môže byť v inej relácii R1 definovaná ako primárny kľúč PK alebo kandidát PK
3. hodnota FK v relácii R2 sa môže rovnať hodnote PK z relácie R1
4. hodnota FK v relácii R2 sa musí rovnať hodnote PK z relácie R1, alebo NULL

**52. Definujte pojem Doména:**

1. Množina prípustných atribútov
2. Množina prípustných hodnôt atribútov
3. Množina prípustných hodnôt cudzieho kľúča

**53. Ktoré sú hlavné časti SRBD (v každom SRBD musia existovať):**

1. DIS, DAS
2. DDL, DML
3. DDL, DIS
4. DDL, DAS

**54. Fyzická nezávislosť:**

1. nezávislosť definície dát na konceptuálnom modeli
2. nezávislosť programov na organizácii dát a prístupových metódach
3. nezávislosť programov na konceptuálnom modeli

**55. Zdieľateľnosť dát je:**

1. Dáta môžu byť uložené vo viacerých tabuľkách
2. Dáta môžu používať viaceré aplikácie súčasne alebo sekvenčne
3. Dáta môžu byť použité práve raz v aplikácii

**56. Databáza je:**

1. Systém riadenia bázy dát a databáza
2. Programové vybavenie pre definíciu dát, manipuláciu s dátami
3. Množina perzistentných dát

**57. Entita je:**

1. Množina objektov rovnakého typu
2. Popis dát a atribútov objektu
3. Objekt reálneho sveta schopný nezávislej existencie

**58. Relácia je:**

1. je podmnožinou kartézskeho súčinu množiny domén Di na množine atribútov Ai
2. je zjednotením množiny prípustných hodnôt atribútov
3. je množina dvojíc Ai:Di (A-atribút, D-doména)
4. podmnožina kartézskeho súčinu množiny atribútov

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

1. Môže mať viac KPK
2. Môže mať vždy len jeden jednoduchý alebo kompozitný KPK (z jedného alebo množiny atribútov)
3. Môže mať vždy len jeden jednoduchý KPK (z jedného atribútu)

**60. Definujte primárny kľúč:**

1. šetky množiny atribútov, ktoré spĺňajú podmienku jednoznačnosti a minimálnosti
2. Jedna množina atribútov, ktorá spĺňa podmienku jednoznačnosti a minimálnosti
3. 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:**

1. dvojúrovňová (externá a interná)
2. trojúrovňová (externá, konceptuálna, interná)
3. trojúrovňová (externá, interná a aplikačná)

**63. Aké sú úlohy databázového administrátora (DBA):**

Zabezpečiť správny chod a funkčnosť databázového systému

zabezpečiť bezpečnosť prístupu k databáze

**64. Systém riadenia bázy dát je:**

Je množina programov, zabezpečujúca manipuláciu s dátami, ochranou dát, paralelné spracovanie, a pod.

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

1. PK z ET2 bude v ET1 FK, pričom nesmie byt súčasťou PK
2. PK z ET1 bude FK v ET2, pričom môže byt súčasťou PK
3. PK z ET2 bude v ET1 FK a zároveň celým PK
4. PK z ET1 bude v ET2 FK, pričom nesmie byt súčasťou PK

**66. Integrita entít sa týka:**

1. 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):**

Ide o jednu z možností voľby režimu práce.

**70. Relácia:**  
 **Rezervacia\_izby(#objednavatel, #cislo\_izby, #rezervacia\_od, rezervacia\_do, datum\_rezervacie, zaloha)**

Má ?

1. Tri primárne kľúče
2. Jeden kompozitný primárny kľúč
3. Tri cudzie kľúče
4. Jeden kompozitný cudzí kľúč

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

1. Interná
2. Konceptuálna
3. Aplikačnej
4. Externej

**72. Ktorý z príkazov patrí do DAS:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| DDL | Definícia objektov | Create | After | Drop |  |
| DML | Manipulácia s dátami | Insert | Update | Delete | Select |
| DAS | Prístup k dátam | Grant | Revoke |  |  |
| DIS | Správa transakcií | Commit | Rollback | Set transaction |  |

1. CREATE UNIGUE INDEX
2. CREATE VIEW
3. DELETE
4. CREATE TABLE
5. GRANT alebo REVOKE

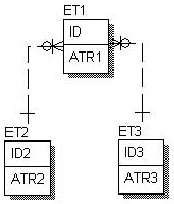
**73. Príkaz GRANT slúži na:**

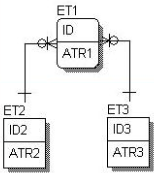
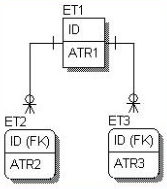
1. Obratie práv užívateľovi
2. Pridelenie práv na prístup k zdrojovým kódom aplikácie
3. Pridelenie prístupových práv na databázový objekt užívateľovi
4. Odobratie práv čítania zdrojových kódov aplikácie

**74. Príkaz COMMIT slúži na:**

1. Potvrdenie transakcie
2. Ukončenie ľubovoľného SQL príkazu bez ohľadu na porušenie konzistencie DB
3. Potvrdenie príkazu SELECT

**75. Určite, ktorý z ERA diagramov modeluje ISA hierarchiu:**



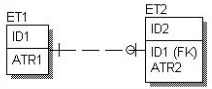


**76. Určite stupeň(s) a kardinalitu(k) nasledovnej relácie:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Id\_druhu** | **Nazov** | **Popis** | **Special\_podmieny** |
| 1 | Jogurt |  | +3 |
| 5 | cukorR | Rafinovaný |  |
| 6 | cukorH | Hnedý |  |
| 8 | Nanuk |  | -15 |
| 12 | Lak | Lak na vlasy | Kozmetika |
| 13 | Aviváž |  | drogéria |

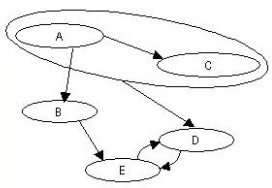
1. s = 4, k = 6
2. s = 2, k = 6
3. s = 6, k = 4

**77. Určite aké je členstvo ET1 a ET2 vo vzťahu:**

****

1. ET1 nemôže existovať bez ET2, ET2 nemôže existovať bez ET1
2. ET1 môže existovať bez ET2, ET2 nemôže existovať bez ET1
3. ET1 môže existovať ET2, ET2 môže existovať ET1
4. 1 : N

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

****

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

**79. 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, ID\_PACIENTA)

PACIENT(#ID\_PACIENTA, ID\_LEKARA)

LEKAR – PACIETN M:N

LEKAR nemusí mať pacienta(nepovinné členstvo)

PACIENT musí mať lekara(povinné členstvo)

**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

**81. Vločková schéma sa používa na modelovanie:**

1. Tabuľky faktov
2. Dátové kocky
3. Dimenzie

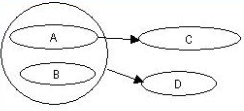
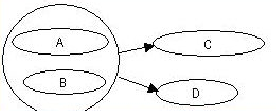
**82. V indexovom súbore s hustým (úplným) indexom:**

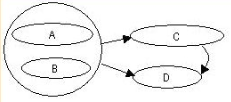
1. Pre skupinu záznamov v dátovom súbore existuje aspoň jeden záznam v indexov súbore
2. Pre každý záznam v dátovom súbore existuje práve jeden záznam v indexov súbore
3. Pre skupinu záznamov dátovom súbore existuje jeden záznam v indexov súbore

**83. Aký je rozdiel medzi B-stromom a B+stromom:**

1. V počte úrovní
2. V zreťazení listových vrcholov
3. Žiadny
4. V počte smerníkov medziľahlého vrcholu

**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**

****

****

**85. Dátová kocka \_\_\_\_\_\_\_\_\_\_ obsahuje merateľnú dimenziu. Doplňte**

1. nemusí
2. musí
3. nesmie

**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 ?**

1. Áno
2. Nie

**87. Redundancia:**

1. Je viacnásobný výskyt rovnakých dát. objektov
2. Je súčasný prístup dvoch užívateľov k dát. objektu
3. Ukladanie dát v B stromoch

**88. Nezávislosť dát:**

1. Znamená vzájomnú nezávislosť uloženia dát
2. Znamená nezávislosť programu a prístupových metód od zmien dátových štruktúr
3. 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:**

1. Predstavuje pohľad užívateľov na dáta
2. Predstavuje prístupové metódy k dátam
3. Predstavuje organizáciu uloženia dát a prístupové metódy k dátam

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

1. Medzi externou a konceptuálnou úrovňou
2. Medzi konceptuálnou a internou úrovňou
3. Nedá sa zabezpečiť

**93. Primárny kľúč:**

1. je množina atribútov, pre ktorú platí pravidlo jednoznačnosti
2. je množina atribútov, ktorá nejednoznačne určuje n-ticu relácie
3. je množina atribútov, pre ktorú platí pravidlo jednoznačnosti a minimálnosti

**94. Každý RDBS musí mať:**

1. DML alebo DDL
2. DML a DDL
3. DDL a DIS
4. DDL a DAS

**95. DIS (Data Integrity statements) = Príkazy pre ochranu integrity sú:**

1. Poskytuje príkazy pre prácu s transakciami a riadenie paralelného prístupu k dátam
2. Pre určenie prístupových práv
3. Pre prácu s Indexami

**96. Úplný RDBS obsahuje:**

1. Len DDL
2. Len DDL a DML
3. DDL, DML, DAS a DIS

**97. Transakcia je postupnosť operácií OI I= 1, ..., n ktorá sa:**

1. Vykoná ako jedna operácia tak, že sa udrží konzistencia DB.
2. Vykoná ako jedna operácia tak, že sa nemusí udržať konzistencia DB.
3. Vykoná v ľubovoľnom poradí tak, že sa nemusí udržať konzistencia DB.

**98. Vymenujte vlastnosti transakcie:**

1. CRUDE = stĺpcová, referenčná, užívateľská, doménová, entitná
2. ANNU = atomickosť, neusporiadanosť zľava doprava, zhora dole, jednoznačnosť
3. ACID = atomickosť, konzistencia, nezávislosť, trvácnosť

Atomickosť, konzistencia, izolovanosť(nezávislosť), trvanlivosť

**99. Logický žurnál obsahuje:**

1. Kópiu DB
2. Informácie o zmenách DB( syst. Inf., pôvodné záznamy, zmenené záznamy resp. stránky)
3. Systémové informácie

**100. Pri potvrdzovacom protokole s priamym zápisom (LOG AHEAD) sa:**

1. DB modifikuje v 1. Fáze
2. DB modifikuje v 2. Fáze
3. DB modifikuje súčasné s log. Žurnálom

**101. Metóda kontrolného bodu sa požíva pri:**

1. Chybách systému
2. Poškodení média s BD
3. Pri chybách aplikácie

**102. Dve transakcie môžu uviaznuť len pri (Ak váhate, tak si to nakreslíte)**

1. Práci s jedným objektom
2. Práci s viacerými objektmi
3. Práci s jedným objektom alebo viacerými objektmi

**103. Pri použití zdieľaného zámku na objekt X:**

1. Je možné sprístupniť objekt inej transakcii len pre operáciu READ
2. Iná transakcia nesmie používať objekt
3. Iná transakcia sme používať len na operáciu WRITE

**104. Pri zamykaní môže nastať uviaznutie, ktoré:**

1. Môžeme prechádzať
2. Môžeme detekovať
3. Nemôže detekovať

**105. Pri predchádzaní uviaznutiu metódu čas. Pečiatok WAIT – DIE pri práci s tým istým objektom DB zruší:**

1. Staršia transakcia mladšiu
2. Mladšia transakcia staršiu
3. Mladšia sama seba

**106. Sérializovateľný rozvrh je:**

Rozvrh S nad množinou transakcií T je serializovateľný, ak dáva rovnaké výsledky ako sériový rozvrh nad množinou transakcií T.

**107. Uviaznutie nastane ak v čakacom grafe:**

1. Existuje cyklus
2. Neexistuje cyklus
3. Existuje orientovaná cesta medzi všetkými bežiacimi transakciami

**108. Homogénny DDBS je systém:**

1. s rovnakým SRBD
2. s rôznymi SRBD
3. s rôznymi DB
4. s rovnakými DB

**109. Uviaznutie v DDBS:**

1. nemôže vzniknúť
2. nevieme detekovať
3. vieme detekovať pomocou dodatočného prenosu čakacieho grafu

**110. Metódu časových pečiatok v DDBS:**

1. je možné použiť
2. nie je možné použiť

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

1. medzi externou a konceptuálnou úrovňou
2. medzi konceptuálnou a internou úrovňou
3. nedá sa zabezpečiť

**112. Typ entity:**

1. Je objekt reálneho sveta schopný nezávislej existencie
2. Vyjadruje väzbu medzi entitami
3. Je množina vlastností objetov rovnakého typu

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

1. Nutnosť existencie entity pri priradení do vzťahu
2. Vyjadruje vzťah medzi dvomi entitami
3. Reprezentuje množinu prípustných hodnôt

**114. Konzistencia databázy:**

1. Znamená, že DB je v každom okamihu správna
2. Znamená opakované spracovanie
3. Typovú kontrolu vstupných dát

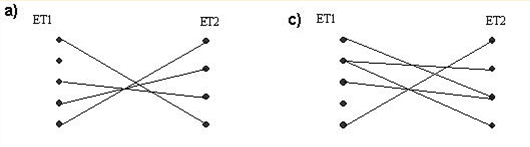
**115. Koncetuálna schéma obsahuje množinu dát:**

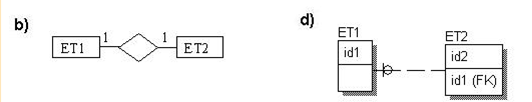
1. Implementačne závislú popisujúcu dátové užívateľské požiadavky aplikácie
2. Implementačné závislú popisujúcu dátový model aplikácie
3. Implementačné nezávislú popisujúca dátový model aplikácie

**116. Entitiy**

1. Je objekt reálneho sveta schopný nezávislej existencie
2. Vyjadruje väzbu medzi entitami
3. Je množina vlastností objektov rovnakého typu

**117. Ktorý z nasledovných obrázkov je výskytový diagram vzťahu 1:1:**





1. Len obrázky a, b, c
2. Len obrázky a, b
3. Len obrázky b, d
4. Len obrázok a

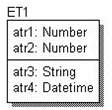
**118. Definujte povinné členstvo entity vo vzťahu:**

1. Povinnosť výskytu aspoň jedného atribútu relácie
2. Hodnoty atribútov relácie musia byť zadefinované
3. Povinnosť existencie entity vo vzťahu k inej entite
4. Možnosť existencie entity vo vzťahu k inej entite

**119. Príkaz ROLLBACK slúži na:**

1. Potvrdenie transakcie
2. Vrátanie DB do pôvodného stavu ako bolo pred začatím transakcie
3. Vrátenie DB do pôvodného stavu ako bola pred posledným vykonávaným príkazom

**120. Ktorý z príkazov zodpovedá definícii danej entity ?**

****

1. Create table ET1 (atr1 int not null, atr2 int not null, atr3 varchar2(10), atr4 date not null, primary key (atr1, atr2));
2. Create table ET1 (atr1 int not null, atr2 int not null, atr3 varchar2(10), atr4 date not null);
3. Create table ET1 (atr1 int not null primary key, atr2 int not null primary key, atr3 varchar2(10), atr4 date not null);
4. Create table ET1 (atr1 int, atr2 int, atr3 varchar2(10), atr4 date not null, unique (atr1, atr2))

**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?

1. Nie
2. Áno

**122. Hyperkocka(Multidimenzionálna databáza) je \_\_\_\_\_\_\_\_\_\_\_\_\_\_ priestor bázových dimenzií, z ktorých jedna musí byť merateľná dimenzia, a nad ktorým môže existovať m- rozmerný priestor agregovaných dimenzií (m >> n):**

1. 3- rozmerný
2. 4- rozmerný
3. n- rozmerný

**123. Navrhnite štruktúru logického žurnálu:**

ID transakcie

ID činnosti

* CHECK poin
* Begin Work
* Abort
* Commit
* Operácia

ID operácie

* Insert
* Delete
* Update
* Select

Čas udalostí

Záznam PRED operáciou

Záznam PO operácii

Záznam z CHECK pointu

**124. Pri dvojfázovom potvrdzovacom protokole:**

1. sa v prvej fáze prideľujú zámky a v druhej fáze uvoľňujú zámky
2. sa v prvej fáze zapisuje do log. Žurnálu a v druhej fáze do databázy
3. sa v prvej fáze zapisuje do databázy a v druhej fáze do log. žurnálu

**125. Určite správnu maticu zamykania, ak platí:**

**X – exclusive**

**S – share**

**-- bez zámku**

****

**126. Uviaznutiu môžeme predchádzať pri metóde:**

1. metóde kontrolného bodu
2. časových pečiatok
3. pri žiadnej
4. zamykania

**127. Pri ktorom protokole sa DB modifikuje pred zápisom do log. žurnálu:**

1. žiadnom
2. časových pečiatok
3. metóde kontrolného bodu
4. metóde dvojfázového potvrdzovania protokolu

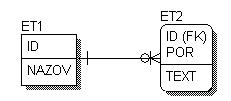
**128. Pri metóde čas. pečiatok WOUND – WAIT pri práci s tým istým objektom DB zruší:**

1. mladšia transakcia staršiu
2. staršia transakcia mladšiu
3. mladšia sama seba
4. staršia sama seba

**129. Typ entity:**

1. popis objektov s rovnakými charakteristikami
2. objekt reálneho sveta schopný samostatnej existencie, ktorý je jednoznačne identifikovateľný
3. objekt reálneho svet schopný samostatnej existencie, ktorý musí byť jednoznačne identifikovateľný

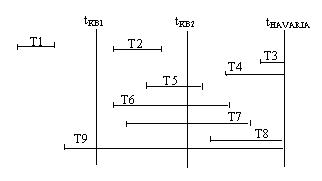
**130. Určite príkaz, ktorý vytvorí PK tabuľky ET2:**

****

1. ALTER TABLE ET2 ADD (PRIMARY KEY(ID));
2. ALTER TABLE ET2 ADD (PRIMARY KEY(ID, POR));
3. ALTER TABLE ET2 ADD (PRIMARY KEY (ID, POR, TEXT));
4. ALTER TABLE ET2 ADD (PRIMARY KEY(ID));

ALTER TABLE ET2 ADD (PRIMARY KEY(POR));

**131. Metóda kontrolného bodu . Operáciu UDNO aplikujem pri obnove databázy po havárií na transakcie:**

****

1. T3, T4, T8, T9
2. T3, T4, T8
3. T5, T6, T7
4. T5, T6, T7, T9

**132. 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, id\_poistenca)

POISTENEC(#id\_poistenca, id\_poistovne)

KARDINALITA M:N

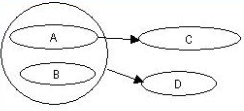
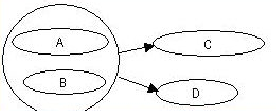
POISTOVNA nemusí mať poistencov(nepovinné členstvo)

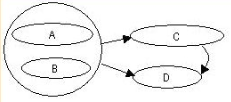
POISTENEC musí mať POISTOVNU (povinné členstvo)

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

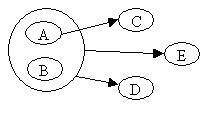
1. AK je relácia v BCNF musí byť aj v 3NF
2. Ak je relácia v 3NF nemôže byť v BCNF
3. Ak je relácia v BCNF nemusí byť v 3NF

**134. 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:**

****

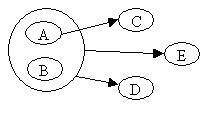
****

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

****

A -> C, AB -> E, AB -> D

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

****

1. Áno
2. Nie

**137. Definícia DDBS:**

Distribuovaný databázový systém je množina uzlov počítačovej siete, navzájom prepojených v komunikačnej sieti, každý z uzlov je samostatný databázový systém, samotná databáza je množina navzájom prepojených databáz.

**138. Vysvetlite pojem fragmentačná transparencia:**

1. Užívateľ musí definovať fragmentáciu všetkých relácií v DDBS
2. Užívateľ nesmie zbadať rozdiel medzi prácou s fragmentovanou reláciou a originálnou
3. Užívateľ musí vedieť kde sa fragmenty pôvodnej relácie nachádzajú

**139. Ak je FK súčasťou PK potom:**

1. Nemôžeme nastaviť ON UPDATE (NULLIFIED)

**140. Definujte Unikátnosť:**

Neexistuje v relácií dva a viac n-tíc, ktoré majú rovnaké hodnoty atribútov tvoriacich KPK.

**141. Definujte Minimálnosť:**

Neredukovateľnosť, žiadna podmnožina atribútov KPK nespĺňa podmienku unikátnosti.

**142. Definujte Kandidáta primárneho kľúča:**

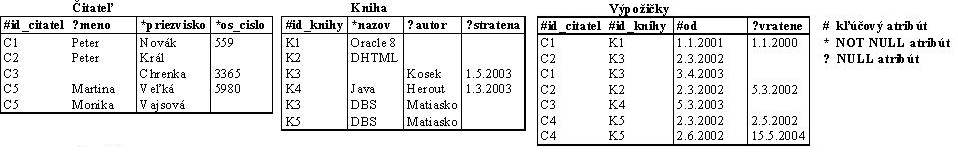
Množina atribútov, ktorá spĺňa podmienku jednoznačnosti a minimálnosti.

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

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

os\_cislo

**144. Určite miesta, kde je porušená stĺpcová integrita:**

****

čitatel.os\_cislo, Kniha.nazov

**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 ani v 3NF

**146. Hviezdicová schéma sa používa na modelovanie:**

Tabuľky faktorov

**147. Čo je to kurzor:**

Je mechanizmus, ktorý umožní sprístupniť riadky tabuľky.

Kurzor je objekt, ktorý sprístupňuje záznamy v množine získanej príkazom SELECT

**148. Sekvenčný kurzor je:**

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

**149. Scroll kurzor je:**

1. kurzor, ktorý umožní priamy prístup k vybranému záznamu
2. kurzor, ktorý umožní spracovávanie záznamov len v jednom smere od začiatku do konca.
3. kurzor, ktorý zobrazí naraz viacej riadkov výslednej množiny

**150. Koľko tabuliek faktov môže byť v jednej dátovej kocke :**

Viac ako jenda

**151. B strom rozpoznáva:**

Dva druhy vrcholov

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

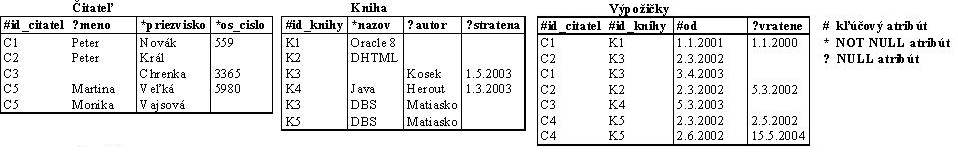
**A -> B, A -> C, A -> D, B -> E je táto relácia v 2NF ?**

1. Áno
2. Nie

**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.

**154. Určite miesta, kde je porušená stĺpcová integrita:**

****

Čitateľ nemôže vrátiť knihu skôr ako si ju požičal. Kniha nemôže byť požičaná keď je stratená.

**155. Najmenšia pamäťová jednota na disku je:**

Sektor

**156. Čo je stopa:**

Množina bodov (čo bod to bit) na povrchu disku, ktorá je sústredená do kruhu na diskovej platni.

**157. Write failure je:**

Neschopnosť správne zaznamenať dáta na disk

**158. Model čítania z disku môžeme reprezentovať funkciou, ktorá vráti dvojicu (w, s), kde:**

w sú dáta sektora a s je stavový bit

**159. Pri nepárnej parite platí:**

Ak je v sektore nepárny počet bitov s hodnotou 1, paritný bit nadobúda hodnotu 1

**160. Kontrolná suma sa vypočíta pre :**

Len pre každý sektor

**161. Stabilná pamäť:**

Je tvorená dvojicami sektorov XL a XR, pričom každý sektor obsahuje kontrolnú sumu

**162. Ak máme sekvenčný súbor, potom nový záznam môže pridať:**

Len na koniec súboru

**163. Pre každú operáciu UPDATE tabuľky sa Indexový súbor nad primárnym kľúčom:**

Nemusí zmeniť

**164. Riedky index je:**

Pre skupinu záznamov v dátovom súbore existuje jeden záznam

**165. Aký je rozdiel medzi B a B+ stromom:**

B strom tvorí jeden súbor, kde sú aj kľúče aj dáta. B plus strom sa skladá z 2 súborov..

**166. Vrcholy B stromu typu LISTU:**

Musia byť na rovnakej úrovni.

**167. Ak už existuje na objekt A zámok typu S-LOCK, potom:**

Iná transakcia môže získať len zámok typu S-LOCK

**168. Ak udalosť spôsobí vyslanie správy z uzla S1 na uzol S2, tak:**

Byť väčšia ako čas odoslania správy z uzla S1

**169. Výpadok typu jedna znamená:**

Že pod transakciu nebolo možné spustiť a globálna transakcia nedostane správu.

**170. O akú množinu sa jedna:**

**SELECT id, meno, priezvisko**

**FROM osoba;**

SET

**171. Fyzický plán vykonania tvorí:**

Postupnosť operácií relačnej algebry doplnené o operáciu pre prechádzanie tabuľkou

**172. Jednému SQL dotazu môže zodpovedať:**

Viac rôznych plánov vykonania.

**173. Pre blokový Nested- Loop Join je odporúčané:**

Menšiu z relácií použiť vo vonkajšom cykle.

**174. BAG označuje typ, ktorého:**

Hodnoty tvoria konečnú množinu s duplicitami.

**175. V ODL je používanie kľúčov:**

Nepovinné

**176. Kurzor v DBS Oracle umožňuje získať:**

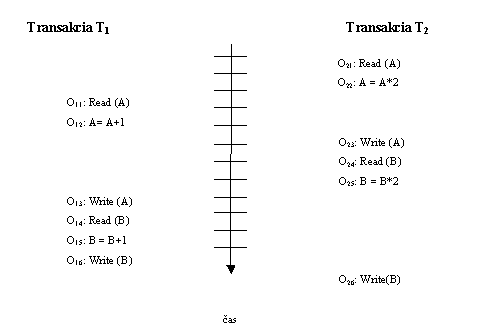
Len hodnoty nasledujúcej položky v kurzore

**177. Doplňte do hlavičky funkcie určenej na triedenie objektov:**

ORDER member FUNCITON zorad\_meno(b t\_osoba) RETURN integer AS ....

**178. Podmienka serializovateľnosti je:**

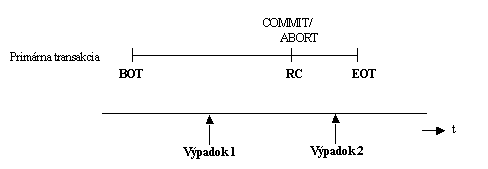
Všetky operácie transakcie sa vykonávajú postupne za sebou – sériovo.

**179. Vykonanie daného rozvrhu je:**

1. Akceptovateľne len v distribuovaných systémoch
2. Akceptovateľne
3. Neakceptovateľne

**180. Ak na tomto istom uzle udalosť i predchádza udalosť j, tak pre časové pečiatky týchto udalosti platí:**

1. TS(i) > TS(j)
2. TS(i) < TS(j)
3. TS(i) >= TS(j)

**181. V prípade výpadku typu 2, a v prípade, že jedna podtransakcia vyslala lokálny ABORT táto podtransakcia:**

1. Môže ukončiť svoju činnosť, ale ostatné podtransakcie musia čakať na obnovenie činnosti globálnej transakcie.
2. Nesmie ukončiť svoju činnosť, ale bude čakať na správu o ukončení ostatných podtransakcií nesmie ukončiť svoju činnosť, ale vyšle ostatným podstransakcieám správu o svojom ukončení.
3. Nesmie ukončiť svoju činnosť, ale vyšle ostatným podtransakciám správu o svojom ukončení

**182. Vertikálne fragmentujte reláciu DRUH TOVARU na aspoň dva fragmenty. Fragmentáciu vykonajte pomocou operácií relačnej algerbry:**

**183. Horizontálne fragmentujte reláciu NAKUP podľa pokladne aspoň na 3 fragmenty (ak sa jedná o hypermarket). Fragmentáciu vykonajte pomocou príkazov SQL:**

**184. Horizontálne fragmentujte reláciu POLOZKA podľa zamestnanca, ktorý účtoval nákup, aspoň na 3 fragmenty. Fragmentácia vykonajte príkazov SQL:**

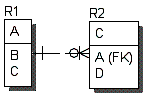
**185. Kde má zmysel použiť kurzor ?**

Kurzor sa používa na sprístupnenie riadkov tabuľky

**186. Dátová kocka môže obsahovať \_\_\_\_\_\_\_\_\_\_\_\_ časovú dimenziu. Doplňť:**

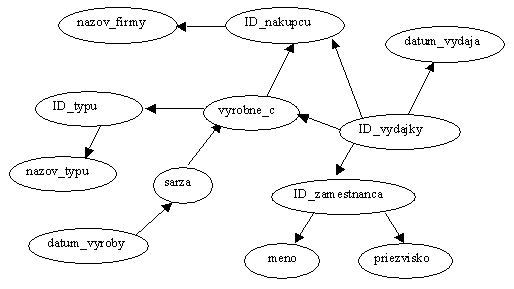
1. Viac ako jednu
2. Ani jednu
3. Maximálne jednu

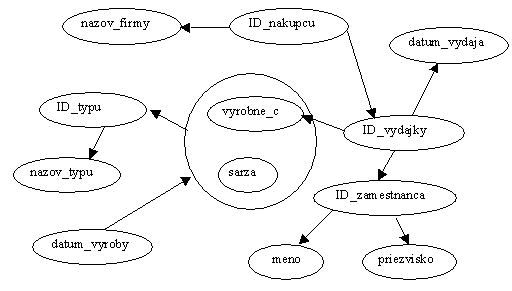
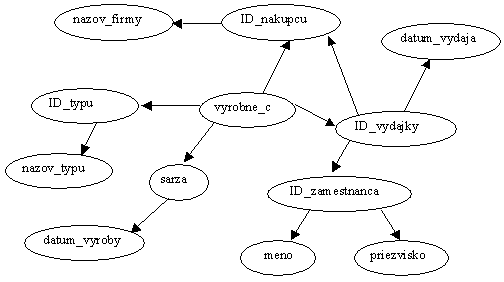
**187. Čo platí pre kardinalitu – card(R1) = K1, card(R2) = K2 a card(R1 JOIN R2) = K3, ak máme nasledovný dátový diagram:**

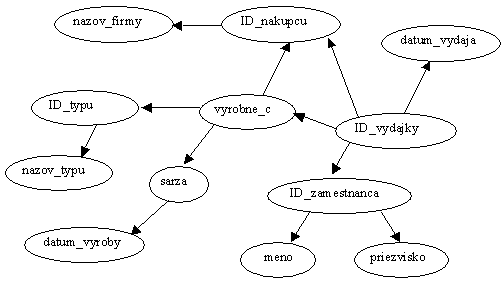
****

1. K3 = K1
2. K3 = max(K1, K2)
3. K3 = K2
4. K3 >= K2

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

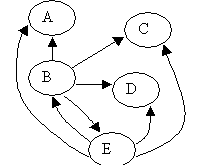
** V sklade máme súčiastky, ktoré majú svoje výrobné číslo, šaržu, typové označenie a názov typu. Súčiastky jednej šarže sú vyrábané v jeden deň. Tieto súčiastky sú postupne vydávané zo skladu do predajne, pričom potrebujeme vedieť ktorý zamestnanec vydal jednotlivé súčiastky, kedy a pre ktorého zákazníka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

****

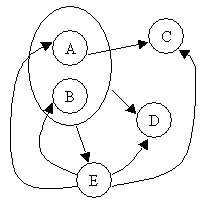
****

**189. V dátových skladoch sa výpočitateľne dáta:**

1. Môžu ukladať, len ak sa neukladajú dáta, z ktorých boli vypočítané
2. Nesmú ukladať
3. Môžu ukladať

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

1. Áno
2. Nie

**191. Je daná relácia v BCNF:**

1. Áno
2. Nie

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

1. môžeme použiť len v prípade riadkového triggra
2. môžeme vždy použiť
3. nikdy nemôžeme použiť

**193. 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.**

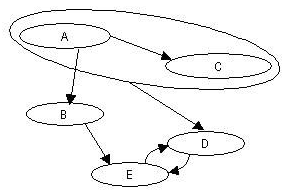
OSOBA(#id\_osoby, meno, priezvisko)

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

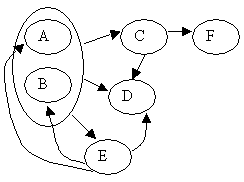
škola musí byt priradená osobe - povinne členstvo

osoba nemusí byt priradená skole - nepovinne členstvo

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

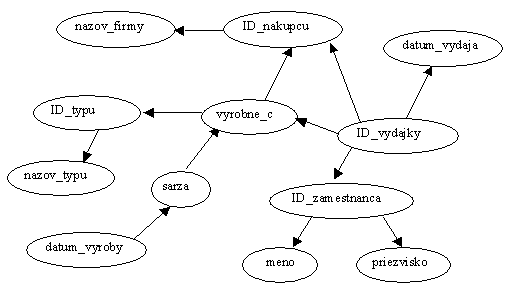
****

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

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

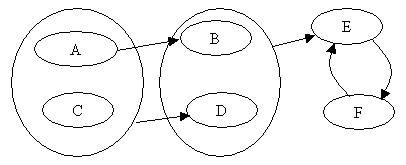
AB -> C, AB ->D, AB -> E, E -> B, C -> D, E -> D, C -> F, E -> A

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



1. v jeden deň sa môže vyrobiť len jeden výrobok z viacerých šarží
2. v jeden deň sa môžu vyrobiť viaceré výrobky z viacerých šarží
3. v jeden deň sa môžu vyrobiť viaceré výrobky rovnakej šarže
4. v jeden deň sa môže vyrobiť len jeden jediný výrobok

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

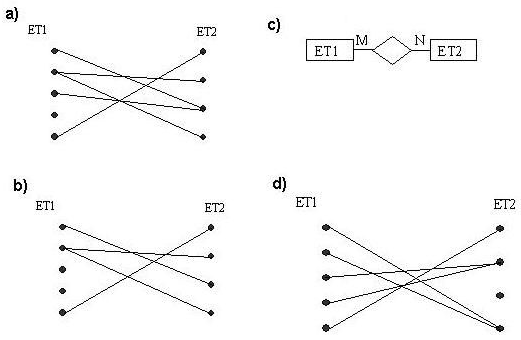
****

1. len atribútov B
2. atribútov B a D
3. len atribútov D

**198. Schéma je:**

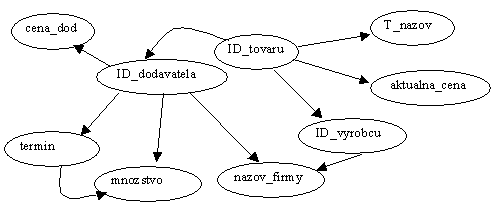
1. Špecifická množina dát popisujúca manipuláciu s dátami
2. Špecifická množina dát popisujúca dátový model aplikácie
3. Množina perzistentných dát

**199. Ktorý z nasledovných obrázkov je príklad výskytového diagramu vzťahu M:N:**

****

1. Len obrázok a, c
2. Len obrázok b
3. Len obrázok a, c, d
4. Len obrázok a, b, d

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

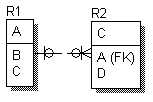
****

1. musia byť všetky dodávky všetkých tovarov rovnakého množstva
2. môžu byť dodané rôzne množstvá rôznych tovarov aj od rovnakého dodávateľa
3. musia byť všetky dodávky len rovnakých tovarov rovnakého množstva

**201. Počas vykonávania transkacie**

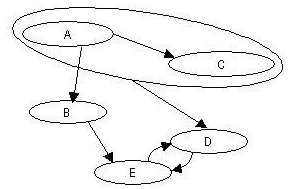
1. DB môže byť dočasne aj v nekonzistentnom tvare
2. DB musí byť stále v konzistentnom tvare
3. DB nesmie byť v nekonzistentnom tvare

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

****

1. K3 = K1 \* K2
2. K3 >= K1
3. K3 = K1 + K2
4. K3 <= K1

**203. Funkčná závislosť medzi atribútmi A a E je:**

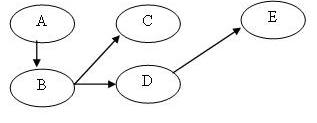
****

1. elementárna
2. vzájomná
3. tranzitívna
4. triviálna

**204. Perzistencia transakcie znamená:**

1. že potvrdené výsledky transakcií môžu byť vrátené pomocou príkazu rollback
2. že potvrdené výsledky transakcií sú uložené v databáze
3. že všetky zadané operácie budú vykonané

**205. Funkčná závislosť ktorých atribútov je tranzitívne závislá na atribúte B:**



1. len atribútu A
2. len atribútu D
3. atribútov C, D, E
4. len atribútu E

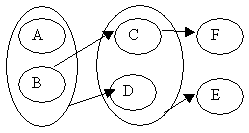
**206.** **Ktorý z príkazov slúži na pridelenie práva select z tabuľky tab1 užívatelovi vajsová:**

1. USER CAN select ON tab1 TO vajsova;
2. GRANT select ON tab1 TO vajsova;
3. DROP RIGHT select ON tab1 TO vajsova;
4. REVOKE select ON tab1 FROM vajsova;

**207.Pri metóde priameho zápisu:**

1. až po všetkých zápisoch do log. žurnálu modifikuje DB
2. sa po každom zápise do log. žurnálu modifikuje DB
3. až po všetkých modifikáciách DB sa operácie zapíšu do log. žurnálu
4. sa po každej modifikácii DB zapíše operácia do log. žurnálu

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



Kompozitný PK AB

1. Časť

**1. Čo znamená ak je nastavené:**

**ALTER TABLE os\_udaje**

**ADD (FORGEIGN KEY (obec\_id) REFERENCES obec(id) ON UPDATE RESTRICTED);**

1. operácia UPDATE atribútu obec\_id v relácii os\_udaje bude zamietnutá, pokiaľ existuje aspoň jeden riadok v relácii obec s hodnotou id zhodnou s hodnotou obec\_id.
2. operácia UPDATE atribút id v relácii obec bude povolená, len pokiaľ existuje aspoň jeden riadok v relácii os\_udaje s hodnotou obec\_id zhodnou s hodnotou id.
3. operácia UPDARE atribútu id v relácii obec bude zamietnutá, pokiaľ existuje aspoň jeden riadok v relácii os\_udaje s hodnotou obec\_ zhodnou s hodnotou id.id
4. operácia UPDATE atribútu obec\_id v relácii os\_udaje bude povolená, len pokiaľ existuje aspoň jeden riadok v relácii obec s hodnotou id zhodnou s hodnotou obec.id

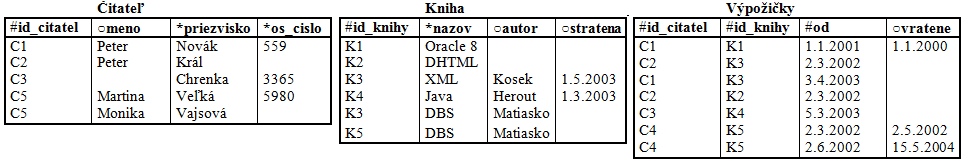
**2. Čo znamená ak je nastavené:**

**ALTER TABLE os\_udaje**

**ADD (FOREIGN KEY (obec\_id) REFERENCES obec (id) ON DELETE NULLIFIED);**

1. operácia DELETE bude povolená len na riadky v relácii obec, ktorých hodnota id bude NULL.
2. ak bude zmazaný riadok z relácie obec, potom v relácii os\_udaje bude v referencovaných riadkoch nastavená hodnota obec\_id na NULL
3. ak bude zmazaný riadok z relácie os\_udaje, potom v relácii obec bude v referencovaných riadkoch nastavená hodnota id na NULL
4. ak bude zmenená hodnota id z relácie obec, potom v relácii os\_udaje bude v referencovaných riadkoch nastavená hodnota obec\_id na NULL

**3.**

****

a.) Navrhnite aspoň 2 prípady kontroly užívateľskej integrity:

Kontrola hodnôt dátumu vrátené nesmie byť väčšie ako sysdate .

Kontrola hodnôt dátumu od nesmie byť väčšia ako vrátene.

b.) Označte miesta s porušenou stĺpcovou integritou:

os\_cislo má byť not null a sú tam voľné miesta

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*3

***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;*

**4. Uveďte príklad, kde by ste využili kartézsky súčin:**

Vypísať všetky kombinácie mena, priezviska a čísla predmetu, ktoré si môže študent zapísať.

**5. Nech r(A,B,C) a s(B,C,A) sú relácie a nech a∈ dom(A) a b∈dom(B). Ktoré z nasledujúcich výrazov sú korektné výrazy relačnej algebry ?**

1. πB(r) - πA(s) (1)
2. σA=a,B=b(s) (1)
3. r ∪ s (1)
4. r ∩ s (1)

πx(y) znamená projekciu relácie y na množinu atribútov X

σpodm(y) znamená výber tých riadkov relácie y, ktoré spĺňajú podmienku *podm*

y ∪ z znamená zjednotenie relácií y a z

y ∩ z znamená prienik relácií y a z

1. Operáciu prienik relácií R1 a R2 vieme vyjadriť pomocou operácie ................... uveďte príklad ? rozdiel
2. Operáciu spojenie môžeme vyjadriť pomocou operácií ? karteziánsky súčin, selekcia
3. Vymenujte binárne operácie relačnej algebry ? zjdednotenie, rozdiel, prienik, karteziansky súčin, spojenie, delenie
4. Vymenujte unárne operácie relačnej algebry ? výber, projekcia
5. Vymenujte množinové operácie ? zjednotenie, rozdiel, prienik, karteziánsky súčin
6. Vymenujte relačné operácie ? výber, projekcia, spojenie, delenie
7. Operáciu delenie relácií D a d vyjadrite pomocou operácií ? D / d = R1 – R2 R1 = ∏ A1, A2, ... Ap(D) R2 = ∏ A1, A2, ...Ap (R1 x d) - D)

**6. Kedy je relácia v BCNF:**

1. keď každý determinant je kandidátom primárneho kľúča
2. keď je v 2NF a žiaden nekľúčový atribút nie je tranzitívne funkčne závislý na primárnom kľúči
3. keď každý kandidát primárneho kľúča je determinantom
4. keď je v 3NF a každý nekľúčový atribút je determinantom

**7. Je relácia:**

**student(cislo\_studenta, meno, adresa, st\_skupina)**

**predmet(cislo\_predmetu, nazov, pocet\_kreditov)**

**zap\_predmety(cislo\_studenta, cislo\_predmetu, skrok, znamka)**

1. Determinant v relácií ZapPredmety je - cislo\_studenta, cislo\_predmetu, skrok
2. Je daná relácia Zap\_Predmety v BCNF - áno
3. Je relácia ZapPremety v 3NF – áno
4. V relácii A(OC, MENO, CisPredemtu, NazovPredmetu, OCUcitela, MenoUcitela) sú daterminanty (zdôvodnite) - OC, OCUcitela, CisPredmetu
5. Zavislosť OC – NazovPredmetu v relácii A je - ani funkčná, ani vzájomná, ani tranzitívna
6. Uveďte a zdôvodnite najvyššiu NF relaćie Student - je v 3NF aj BCNF
7. Zdôvodnite najvyššiu NF relácie Student - OC, CP
8. Je relácia ZapPremety v 3NF - áno
9. Závislosť CisPredmet – NazovPredmetu v relácii A je - vzájomná

**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 ?

1. BEFORE alebo AFTER INSERT
2. INSTEAD OF INSERT
3. AFTER INSERT
4. 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 ?

1. BEFORE insert
2. nie je možná operácia INSERT
3. nie je potrebné definovat žiadny trigger
4. 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 ?

1. DELETE bude vždy fungovať aj bez triggra
2. AFTER delete
3. 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 ?

1. riadok bude vložený iba do pohľadu
2. riadok bude vložený iba do tabuľky os\_udaje
3. riadok bude vložený do pohľadu aj do tabuľky os\_udaje
4. 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 ?

1. riadok bude vložený iba do pohľadu
2. riadok bude vložený iba do tabuľky os\_udaje
3. riadok bude vložený do pohľadu aj do tabuľky os\_udaje
4. 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)

1. nie je potrebné definovať žiadny trigger
2. nie je možná operácia INSERT
3. BEFORE insert
4. INSTEAD OF insert
5. 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)

1. nie je možná operácia INSERT
2. BEFORE insert
3. INSTEAD OF insert
4. 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 ?

1. BEFORE, alebo AFTER
2. len INSTEAD OF
3. len AFTER
4. 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 **BEGIN** pre každý riadok zvlášť

**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 ?

1. 1 riadok
2. ani jeden riadok
3. 10 riadkov
4. 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 ?

1. 1 riadok
2. ani jeden riadok
3. 10 riadkov
4. 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 ?

1. AFTER
2. INSTEAD OF
3. BEFORE
4. 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 ?

1. BEFORE, AFTER, alebo INSTEAD OF
2. BEFORE
3. BEFORE alebo AFTER
4. AFTER

**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 ?

1. AFTER
2. INSTEAD OF ZLE
3. BEFORE
4. 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 ?

1. Before
2. After
3. 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);**

1. Tento pohľad nie je ani možné vytvoriť.
2. V selecte je syntaktická chyba.
3. 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);**

1. Tento pohľad nie je ani možné vytvoriť.
2. V selecte je syntaktická chyba.
3. 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 ?

1. BEFORE, alebo AFTER
2. len INSTEAD OF
3. len AFTER
4. 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 ?

1. riadok bude vložený iba do pohľadu
2. riadok bude vložený iba do tabuľky osoba
3. riadok bude vložený do pohľadu aj do tabuľky osoba
4. 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 ?

1. BEFORE, AFTER, alebo INSTEAD OF
2. BEFORE
3. 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 ?

1. Before
2. After
3. BEFORE alebo AFTER

**29. Určte ktorý z nasledovných príkazov vytvorí všetky vzťahy medzi entitou Rodič a Dieťa:**

****

1. ALTER TABLE DIETA ADD (FOREIGN KEY(MATKA, OTEC) REFERENCES (RC));
2. ALTER TABLE RODIC ADD(FOREIGN KEY(RC) REFERENCES (MATKA));
3. ALTER TABLE DIETA ADD(FOREIGN KEY(MATKA) REFERENCES (RC));

ALTER TABLE DIETA ADD(FOREIGN KEY(OTEC) REFERENCES (RC));

1. ALTER TABLE RODIC ADD(PRIMARY KEY (RC));

**30. Príkazom ALTER TABLE TAB MODIFY (PLAT NUMBER UNIGUE):**

1. Vytvorí tabuľkUru s jedným stĺpcom PLAT
2. Nastaví existujúcemu stĺpcu PLAT doménu NUMBER a definuje IO na daný stĺpec. V prípade, že stĺpec PLAT neexistuje v tabuľke TAB, týmto príkazom ho vytvoríme
3. Pridá nový stĺpec PLAT do tabuľky TAB
4. Nastaví existujúcemu stĺpcu PLAT doménu NUMBER a definuje IO na daný stĺpec

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

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

1. πzos\_cislo (zamestnanec) x π kedy, id\_nakupu (nakup)
2. zamestnanec x nakup
3. 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);**

1. πid(osoba – zamestnanec)
2. πid(osoba) – πid(zamestnanec)
3. πid(δosoba.id <> zamestnanec.id(((osoba x zamestnanec))
4. 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**

1. všetky výrazy
2. výrazy b, d, e
3. výrazy a, c, e
4. ani jeden výraz

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

**SELECT id, meno FROM osoba;**

1. θ id, meno(osoba)
2. δ id, meno(osoba)
3. π id, meno(osoba)

**35. 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:

1. neexistuje
2. tranzitívna
3. vzájomná

**36. Majme trigger, ktorý zabezpečí nastavenie hodnoty autoincrement stĺpca:**

**CREATE OR REPLACE TRIGGER tab\_seq\_ins**

**.... INSERT ON tab\_seq**

**REFERENCING NEW AS novy**

**FOR EACH ROW**

**BEGIN**

**SELECT sekv\_id NEXTVAL INTO :novy,id FROM dual;**

**END;**

Doplňte do triggru:

1. INSTEAD OF
2. Len BEFORE
3. BEFORE alebo AFTER
4. Len AFTER

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

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

1. Áno
2. Nie

**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**

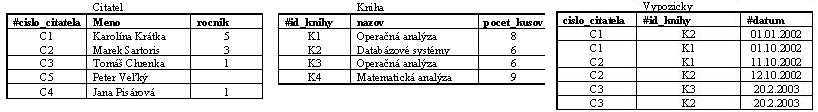
**d.) δnazov=jogurt(druh\_tovaru)**

**e.) zamestnanec ᴗ nakup**

1. Všetky výrazy
2. Výraz b, d
3. Výraz a, b, d
4. Výraz a, c, e

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

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



1, 2, 1

**40. Ktorý z príkazov slúži na odobratie práva selectu z tabuľky tab1 užívateľovi vajsová:**

1. GRANT select ON tab1 TO vajsova
2. DROP RIGHT select ON tab1 TO vajsova;
3. USER CAN select ON tab1 TO vajsova;
4. REVOKE select ON tab1 FROM vajsova;

**41. Môže trigger:**

**AFTER INSERT ON zap\_predmety**

Obsahovať príkaz INSERT INTO log\_zp?

1. Áno
2. Nie

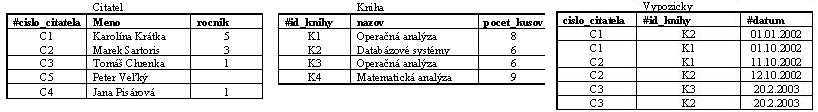
**42. Aký bude výsledok nasledovného selectu:**

**SELECT count(\*)**

**FROM citatel , vypozicky**

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

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



2, 2, 2

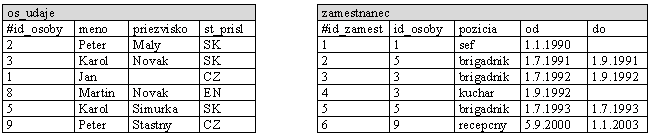
**43. Aký bude výsledok nasledovného selectu:**

**SELECT count(\*)**

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

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

**GROUP BY o.id\_osoby**



1, 2, 2, 1

**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

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

**SELECT id\_jedla, kedy**

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

**WHERE nazov = ‘FRI‘;**

1. Πid\_jedla, kedy(δnazov =´FRI´(platnost\_menu x vydajne))
2. Πid\_jedla, kedy(δnazov =´FRI´(platnost\_menu JOIN vydajne))
3. Πid\_jedla, kedy(platnost\_menu JOIN (δnazov =´FRI´(vydajne)))

**46. Ak chceme definovať trigger na kaskádu pre operácie DELETE z nasledovného pohľadu**

**CREATE VIEW v\_jedlo AS**

**SELECT por, nazov, kedy**

**FROM menu, jedlo**

**WHERE menu.id\_jedla = jedlo.id\_jedala;**

Potom v tele triggra budú operácie DELETE FROM nasledovných tabuliek(pozor aj na poradie):

1. objednavky, platnost\_menu
2. objednavky, platnost\_menu, vydajne, menu, jedlo
3. menu, jedlo, platnost\_menu
4. objednavky, platnost\_menu, menu, jedlo

**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);**

1. πid\_jedla(jedlo) JOIN (πid\_jedla(δid\_vydajne = 3(platnost\_menu)))
2. πid\_jedla(jedlo) - πid\_jedla(δid\_vydajne = 3(platnost\_menu))
3. 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)**

1. všetky výrazy
2. výrazy 2, 4
3. výrazy 2, 3, 5
4. výrazy 1, 2, 4

**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;

**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%´;**

1. πs.meno,s.priezvisko,o.kedy(δnazov like ´Kur%´(jedlo JOIN objednavky JOIN stravnik)
2. δnazov like ´Kur%´(jedlo JOIN πo.kedy,id\_jedla(objednavky)) JOIN πs.meno,s.priezvisko (stravnik)
3. δnazov like ´Kur%´(jedlo JOIN πo.kedy(objednavky)) JOIN πs.meno,s.priezvisko (stravnik)

**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;

1. 10
2. 0
3. 1

**52. Nasledovný príkaz SELECT**

**SELECT meno, priezvisko INTO p\_meno, p\_priezv**

**FROM os\_udaje**

**WHERE podmienka;**

1. Môžeme použiť len v tele triggra
2. Nikdy nemožeme použit v tele procedúry
3. Môžeme použiť len ak daný select vracia práve jeden riadok
4. Môžeme vždy použiť v tele procedúry

**53. Ak máme dátový model “menza“ a všetky vzťahy nastavené na “ON DELETE RESTRICTED“, potom nasledovný príkaz**

**DELETE**

**FROM jedlo**

**WHERE id\_jedla = 321**

1. Môže vyvolať výnimku
2. Musí vyvolať výnimku
3. Nikdy nevyvolá výnimku

**54. Pomocou selectu zistite kardinalitu relácie stravník.**

SELECT count(\*) FROM stravník;

**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

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

**SELECT id**

**FROM osoba**

**WHERE id NOT IN (SELECT id**

**FROM zamestnanec);**

1. Θosoba.id<>zamestnanec.id(osoba, zamestnanec)
2. Πid(osoba – zamestnanec)
3. Πid(δosoba.id<>zamestnanec.id(osoba x zamestnanec))
4. 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);**

1. Πid\_osoby(os\_udaje) – πid\_osoby(zamestnanec)
2. Os\_udaje – πid\_osoby(zamestnanec)
3. Πid\_osoby(os\_udaje) ᴗ πid\_osoby(zamestnanec)

Majme pohľad:

**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');

1. 2
2. 0
3. 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;**

1. 1
2. 0
3. 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;**

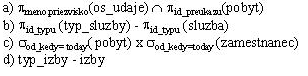
**SELECT MAX(pom) FROM tab1;**

1. 1
2. 3
3. 0
4. 2

**62. Akým príkazom by ste vymazali tabuľku author:**

DROP TABLE author;

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

****

1. výrazy a,c,d
2. výrazy b,c
3. výrazy a,d
4. výrazy b,d

**64. Zoraďte tieto príkazy do správneho poradia, ak cur1 je správne deklarovaný kurzor pre príkaz**

**SELECT priezvisko**

**FROM os\_udaje;**

**a priezv je premenná typu os\_udaje.priezvisko%TYPE.**

**(1) LOOP**

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

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

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

**(5) ELSE**

**(6) OPEN cur1;**

**(7) END IF;**

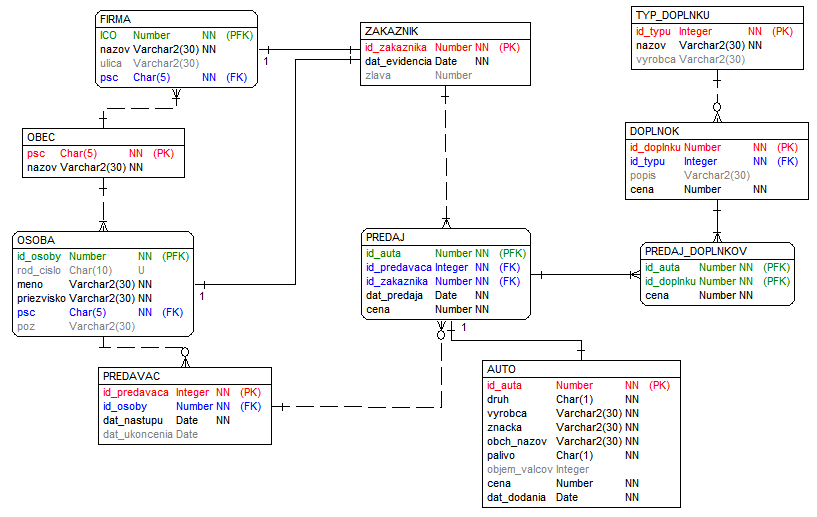
**(8) EXIT;**

**(9) CLOSE cur1;**

**(10) END LOOP;**

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

1. Časť



**---------------------------------------------------------------------------------------UPDATE---------------------------------------------------------------------------------------**

**1. Zmeňte dátum ukončenia predavača Jána Mrkvičku na aktuálny dátum.**

***UPDATE*** *predavac*

***SET*** *dat\_ukoncenia =* ***SYSDATE***

***WHERE*** *id\_osoby* ***IN****(****SELECT*** *id\_osoby*

***FROM*** *osoba*

***WHERE*** *meno = ‘Jan’*

***AND*** *priezvisko = ‘Mrkvicka‘);*

**2. Zmeňte zákazníkovi zľavu (na percenta) podľa počtu kúpených aut a doplnkov za minulý rok.**

***UPDATE*** *zakaznik z*

***SET*** *zlava = (****SELECT COUNT****(\*)*

***FROM*** *predaj p* ***JOIN*** *predaj\_doplnkov pd* ***ON****(predaj.id\_auta = predaj\_doplnkov.id\_auta)*

***WHERE*** *p.id\_zakaznika = z.id\_zakaznika*

***AND TO\_CHAR****(p.dat\_predaj, ‘YYYY’) =* ***TO\_CHAR****(****SYSDATE****, ‘YYYY’) -1*

***GROUP BY*** *p.id\_zakaznika)*

*+ (****SELECT*** *count(\*)*

***FROM*** *predaj*

***WHERE*** *id\_zakaznika = z.id\_zakaznika*

***AND******TO\_CHAR****(p.dat\_predaj, ‘YYYY’) =* ***TO\_CHAR****(****SYSDATE****, ‘YYYY’) -1*

***GROUP BY*** *p.id\_zakaznika);*

alebo

***UPDATE*** *zakaznik z*

***SET*** *zlava = (****SELECT******COUNT****(id\_doplnku) +* ***COUNT****(****DISTINCT*** *id\_auta)*

***FROM*** *predaj p* ***JOIN*** *predaj\_doplnkov pd* ***ON****(predaj.id\_auta = predaj\_dopl.id\_auta*

***WHERE*** *p.id\_zakaznika = z.id\_zakaznika*

***AND******TO\_CHAR****(p.dat\_predaja, ‘YYYY’) =* ***TO\_CHAR****(****SYSDATE****, ‘YYYY’) -1*

***GROUP BY*** *p.id\_zakaznika);*

**3. Zmeňte cenu autám o 10% u ktorých prešiel mesiac od dátumu dodania.**

***UPDATE*** *auto*

***SET*** *cena = cena \* 0,9*

***WHERE*** *dat\_dodania <=* ***ADD\_MONTHS****(****SYSDATE****, -1);*

**----------------------------------------------------------------------------------------DELETE---------------------------------------------------------------------------------------**

**4. Vymaž z evidencie predavačov ktorý už nepracujú a nepredali žiadne auto.**

***DELETE***

***FROM*** *predavac*

***WHERE*** *dat\_ukoncenia <* ***SYSDATE***

***AND*** *id\_predavaca* ***NOT IN****(****SELECT*** *id\_predavaca*

***FROM*** *predaj);*

**5. Vymaž všetky údaje o predaji automobilu.**

***DELETE***

***FROM*** *predaj\_doplnkov*

***WHERE*** *id\_auta = ....;*

***DELETE***

***FROM*** *predaj*

***WHERE*** *id\_auta = ....;*

**6. Vymažte všetky údaje o predaji automobilu s id\_auta = 5.**

***DELETE***

***FROM*** *predaj\_doplnkov*

***WHERE*** *id\_auta =’5’;*

***DELETE***

***FROM*** *predaj*

***WHERE*** *id\_auta = ‘5’;*

**----------------------------------------------------------------------------------------INSERT--------------------------------------------------------------------------------------**

**7. Pridať zákazníka s rodným číslo 754545/8575, menom Karol, priezviskom Ulicny z mesta Prievidza. O id\_zakazníka sa postará trigger.**

***INSERT INTO*** *osoba(rod\_cislo, meno, priezvisko, psc)*

***VALUES*** *(‘754545/8575’,’Karol’,’Ulicny’,’97101’);*

**8. Pridať nového zákazníka s rodným čislo 750112/1212, menom Karol, priezviskom Urman z mesta Prievidza. Novú hodnotu id\_zakaznika sa postará trigger a sequence.**

***INSERT INTO*** *osoba(rod\_cislo, meno, priezvisko, psc)*

***VALUES*** *(‘750112/1212’,’Karol’,’Urman’,’97101’);*

**----------------------------------------------------------------------------------------SELECT---------------------------------------------------------------------------------------**

**9. Vypíšte značky aut predaných predavačom s ID = 26 za posledný mesiac.**

***SELECT DISTINCT*** *znacka*

***FROM*** *auto* ***JOIN*** *predaj* ***USING****(id\_auta)*

***WHERE*** *id\_predavaca = 26*

***AND*** *dat\_predaja >* ***ADD\_MONTHS****(****SYSDATE****, -1);*

**10. Vypíšte zoznam predavačov ktorý za posledný rok/ mesiac nepredali žiadne auto.**

***SELECT*** *meno, priezvisko, id\_predavaca*

***FROM*** *osoba* ***JOIN*** *predavac* ***USING****(id\_osoby)*

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

***FROM*** *predaj*

***WHERE*** *dat\_predaja >* ***ADD\_MONTHS****(****SYSDATE****, -12));*

**11. Vypíšte obec v ktorej sa nachádza najviac zákazníkov.**

***SELECT*** *obec.nazov,* ***CHAR****(\*)* ***AS*** *pocet*

***FROM*** *obec* ***JOIN*** *firma* ***USING****(psc)*

***WHERE******ROWNUM*** *= 1*

***GROUP BY*** *psc, obec.nazov*

***ORDER BY*** *pocet;*

alebo

***SELECT*** *nazov*

***FROM*** *(****SELECT*** *nazov,* ***RANK****()* ***OVER*** *(****ORDER BY COUNT****(\*)* ***DESC****)* ***AS*** *rn*

***FROM*** *obec* ***JOIN*** *firma* ***USING****(psc)* ***JOIN*** *zakanik* ***ON****(firma.ICO****=****zakaznik.id\_zakaznika)*

***GROUP BY*** *psc, nazov)*

***WHERE*** *rn = 1;*

**12. Vypíšte obec s ktorej pochádza najviac predavačov.**

***SELECT*** *nazov*

***FROM*** *(****SELECT*** *nazov,* ***RANK****()* ***OVER*** *(****ORDER BY COUNT****(\*)* ***DESC****)* ***AS*** *rn*

***FROM*** *obec* ***JOIN*** *osoba* ***USING****(psc)* ***JOIN*** *predavac* ***USING****(id\_osoby)*

***GROUP BY*** *psc, nazov)*

***WHERE*** *rn = 1;*

**13. Vypíš takých predavačov ktorý predali viac ako 5 doplnkov k autu.**

***SELECT*** *meno, priezvisko, id\_predavaca*

***FROM*** *osoba* ***JOIN*** *predavac* ***USING****(id\_osoby)*

***JOIN*** *predaj* ***USING****(id\_predavaca)*

***JOIN*** *predaj\_doplnkov* ***USING****(id\_auta)*

***HAVING******COUNT****(id\_doplnku) > 5*

***GROUP BY*** *id\_predavaca, meno, priezvisko;*

alebo

***SELECT*** *id\_predavaca, meno, priezvisko,* ***COUNT****(\*)* ***AS*** *pocet*

***FROM*** *osoba* ***JOIN*** *predavac* ***USING****(id\_osoby)*

***JOIN*** *predaj* ***USING****(id\_predavaca)*

***JOIN*** *predaj\_doplnkov* ***USING****(id\_auta)*

***GROUP BY*** *id\_predavaca, meno, priezvisko*

***HAVING COUNT****(\*) > 5*

***ORDER BY*** *pocet* ***DESC****;*

**14. Vypíšte najlepšieho predajcu, pre každý kalendárny mesiac, teda takého čo predal najviac aut.**

***SELECT*** *pr.id\_predavaca* ***AS*** *predavac,* ***TO\_CHAR****(pre.dat\_predaja,‘MM’)* ***AS*** *mesiac,* ***COUNT****(\*)* ***AS*** *pocet\_predanyc*

***FROM*** *predavac pr* ***JOIN*** *predaj pre* ***ON****(pre.id\_predavaca = pr.id\_predavaca)*

***GROUP BY*** *pr.id\_predavaca,* ***TO\_CHAR****(pre.dat\_predaja,‘MM‘)*

***HAVING COUNT****(\*) = (****SELECT******MAX****(pocet)*

***FROM*** *(****SELECT******TO\_CHAR****(dat\_predaja,’MM’)* ***AS*** *mes,* ***COUNT****(\*)* ***AS*** *pocet*

***FROM*** *predavac* ***JOIN*** *predaj* ***USING****(id\_predavaca)*

***GROUP BY*** *id\_predavaca,* ***TO\_CHAR*** *(dat\_predaja,’MM’))*

***WHERE*** *mes=* ***TO\_CHAR****(pre.dat\_predaja,’MM’));*

**15. Vypíšte najlepšieho predajcu, pre každý kalendárny mesiac, teda podľa celkovej sumy za predané autá v danom mesiaci.**

***SELECT*** *pr.id\_predavaca* ***AS*** *predavac,* ***TO\_CHAR****(pre.dat\_predaja,‘MM’)* ***AS*** *mesiac,* ***SUM****(pre.cena)* ***AS*** *cena*

***FROM*** *predavac pr* ***JOIN*** *predaj pre* ***ON****(pre.id\_predavaca = pr.id\_predavaca)*

***GROUP BY*** *pr.id\_predavaca,* ***TO\_CHAR****(pre.dat\_predaja,‘MM‘)*

***HAVING SUM****(pre.cena) = (****SELECT******MAX****(cen)*

***FROM*** *(****SELECT******TO\_CHAR****(dat\_predaja,’MM’)* ***AS*** *mes,* ***SUM****(cena)* ***AS*** *cen*

***FROM*** *predavac* ***JOIN*** *predaj* ***USING****(id\_predavaca)*

***GROUP BY*** *id\_predavaca,* ***TO\_CHAR*** *(dat\_predaja,’MM’))*

***WHERE*** *mes=* ***TO\_CHAR****(pre.dat\_predaja,’MM’));*

**16. Vypísať psc obcí s ktorých nepochádza žiaden zákazník.**

***SELECT*** *psc*

***FROM*** *obec*

***WHERE*** *psc* ***NOT IN****(****SELECT*** *psc* ***FROM*** *osoba* ***JOIN*** *zakaznik* ***ON****(osoba.id\_osoby=zakaznik.id\_zakaznika));*

**17. Vypísať priemernú cenu predaných doplnkov pre každé auto a k tomu vypísať aj značku auta.**

***SELECT*** *id\_auta, znacka,* ***AVG****(predaj\_doplnkov.cena)*

***FROM*** *auto* ***JOIN*** *predaj* ***USING****(id\_auta)* ***JOIN*** *predaj\_doplnkov* ***USING****(id\_auta)*

***GROUP BY*** *id\_auta, znacka;*

**18. Vypísať predavačov ktorý už bol vyhodený a znova prijatý.**

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

***FROM*** *osoba* ***JOIN*** *predavac* ***USING*** *(id\_osoby)*

***GROUP BY*** *meno, priezvisko, id\_osoby*

***HAVING*** *count > 1;*

**19. Vypísať predavačov ktorý predali auto skôr ako bolo dodané.**

***SELECT DISTINCT*** *meno, priezvisko*

***FROM*** *osoba* ***JOIN*** *predavac* ***USING****(id\_osoby)* ***JOIN*** *predaj* ***USING****(id\_predavaca)* ***JOIN*** *auto* ***USING****(id\_auta)*

***WHERE*** *dat\_predania < dat\_dodania;*

**20. Vypísať predavačov a počet automobilov ktoré dodali za menej ako 6 mesiacov.**

***SELECT*** *id\_predavaca, meno, priezvisko,* ***COUNT****(id\_auta)* ***AS*** *pocet*

***FROM*** *osoba* ***JOIN*** *predavac* ***USING****(id\_osoby)* ***JOIN*** *predaj* ***USING****(id\_predavaca)* ***JOIN*** *auto* ***USING****(id\_auta)*

***WHERE*** *dat\_dodania >* ***ADD\_MONTHS****(****SYSDATE****, -6)*

***GROUP BY*** *meno, priezvisko, id\_predavaca;*

**21. Vypísať zamestnancov a priemerný počet aut ktoré predali za posledný mesiac.**

**22. Vypíšte informácie o autách ktoré boli predané za posledný mesiac.**

***SELECT*** *druh, vyrobva, znacka, obch\_nazov, palivo, objem\_valcov, auto.cena*

***FROM*** *auto* ***JOIN*** *predaj* ***USING*** *(id\_auta)*

***WHERE*** *dat\_predaja >* ***ADD\_MONTHS****(****SYSDATE****,-1);*

**23. Vypíšte údaje o autách, ku ktorým bolo predaných viac ako 5 doplnkov.**

***SELECT*** *id\_auta, druh, vyrobca, znacka, obch\_nazov, palivo, objem\_valcov, cena*

***FROM*** *auto* ***JOIN*** *predaj* ***USING****(id\_auta)* ***JOIN*** *predaj\_doplnkov* ***USING****(id\_auta)*

***HAVING******COUNT****(id\_doplnku) > 5;*

**24. Vypíšte id\_aut a ich značky, ktoré boli predané za menej ako 100.000 Eur.**

***SELECT DISTINCT*** *predaj.id\_auta, auto.znacka*

***FROM*** *predaj* ***JOIN*** *auto****USING****(id\_auta)*

***WHERE*** *predaj.cena < 1000000*

***GROUP BY*** *predaj.id\_auta, auto.znacka;*

**25. Vypíšte zoznam predavačov ktorý skončili za posledný mesiac.**

***SELECT*** *meno, priezvisko*

***FROM*** *osoba* ***JOIN*** *predavac* ***USING****(id\_osoby)*

***WHERE*** *dat\_ukoncenia >* ***ADD\_MONTHS****(****SYSDATE****, -1);*

**26. Vypíšte zoznam predavačov ktorý za posledný rok nepredali dieslove auto.**

***SELECT*** *id\_predavaca*

***FROM*** *predavac*

***WHERE******NOT EXIST****(****SELECT*** *‘x’*

***FROM*** *predaj* ***JOIN*** *auto a* ***USING****(id\_auta)*

***WHERE*** *predavac.id\_predavaca = p.id\_predavaca*

***AND*** *p.dat\_predaja >* ***ADD\_MONTHS****(****SYSDATE****, -12)*

***AND*** *a.palivo = ‘D’);*

*Alebo*

*SELECT meno, priezvisko, id\_predavaca*

*FROM osoba JOIN predvac USING(id\_osoby)*

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

*FROM predaj JOIN auto USING(id\_auta)*

*WHERE dat\_predaja > ADD\_MONTHS(SYSDATE,-12)*

*AND palivo = ´D´);*

**27. Vypíšte počet zákazníkov ku každej obci, ktorí si kúpili nejaké auto za posledných 30 dní.**

***SELECT*** *obec.nazov,* ***COUNT****(id\_zakaznika)* ***AS*** *pocet\_zakaznikov,*

***FROM*** *predaj* ***JOIN*** *zakaznik* ***USING****(id\_zakaznika)*

***JOIN*** *osoba* ***ON****(zakaznik.id\_zakaznika=osoba.id\_osoby)*

***JOIN*** *obec* ***USING****(psc)*

***WHERE*** *dat\_predaja >=* ***ADD\_MONTHS****(****SYSDATE****, -1)*

***GROUP******BY*** *obec.nazov;*

**28. Vypíšte počet automobilov pre každého výrobcu, ktoré neboli predané dlhšie ako 6 mesiacov od dátumu dodania.**