Databázove systémy 2010/2011 - FIIT

url: http://is.gd/dbs_11
Prosím správajte sa slušne, podobné výlevy nemám chuť čítať...

Pravidla - prosím akceptujte

Je zakázané používať iné farby ako je čierna a stupne šedej.
Ak do dokumentu nepíšete, používajte read-only verziu (prepíšte argument URL /edit na / preview

Otázky zo skúšok sú podčiarknuté. Miesto na doplnanie slov alebo T/F je vyznačené úvodzovkami, tieto úvodzovky nemažte.

Ku každej odpovedi je možné pridávať otázky, postrehy a komentáre vždy na nový riadok vo farbách šedej. Len otázky a odpovede zo skúšky budú čiernou farbou.

Ak sa diskutujúci dohodnú na správnej odpovedi, odpoveď na otázku sa zmení a diskusia vymaže, odpoveď musí byť jednohlasná, inak sa nechajú všetky návrhy

Pre kopírovanie textu do dokumentu používajte notepad alebo podobné zverstvo ktoré odstráni všetko formátovanie. Okopírovaný text vložíte do notepadu a znovu skopírujete a vložíte sem :)

Ďakujeme.

//mnicky: Ak pri písaní pri niektorej z odpovedí neviete určiť či je správna alebo nie, nevkladajte “F”, ale otáznik: “?”

//mozno by nebolo na skody zmazat tie otazky, ktore sa tu opakuju
//suhlasim

Obsah

RT 2008/2009
RT 2008/2009 2
OT 2008/2009
RT 2007/2008 (pravdepodobne)

-

RT 2008/2009

V príkladovej časti: V rozpracovanej databáze umožňujúcej spravovať údaje o kreditnom štúdiu a ktorá zatiaľ má pripravených niekoľko tabuliek, navrhnite a slovne zdôvodnite potrebné upresnenia, doplnenia a zmeny, tak aby boli realizovateľné príkazy z nasledujúcich príkladov ! Následne ju v štandardnom jazyku SQL [ANSI - 92] zadefinujte (DDL) ako normalizovanú databázu. K vypracovaniu príkladov môžete použiť dvojhárok, ktorý na konci skúšky odovzdáte dozorujúcim [môže byť užitočné pri posudzovaní riešení].

1. Napíšte presný názov atribútu (alebo skupiny atribútov), ktorého hodnota slúži na identifikáciu entity. "**identifikacny kluc** "

//mnicky: kniha, kap. 4.4.2

2. Napíšte jedným slovom názov atribútu, ktorý pozostáva z viacerých atribútov. "**skupinový (=zložený)**"
//mnicky: zlozeny atribut ??
//johnnytaurus: skupinovy?
//mnicky: kniha, kap. 4.4.4.2 hovori ze su to ekvivalentne nazvy. Dal som oba.

3. Entitno-relačné modelovanie
“T“ znamená vytvorenie popisu dát nezávisle od fyzického uloženia databázy
"F" popisuje techniku modelovania dát, uložených v súbore
"T" je spôsob zápisu konceptuálnej schémy
"?" predstavuje tvorbu modelov na konceptuálnej úrovni architektúry databázového systému
"T" pomáha na konceptuálnej úrovni abstrakcie popísať používateľskú aplikáciu
//taj: kniha str. 48
//philips: ako mam tomu rozumiet, preco nie je **T** aj to prve ked je to to iste co je v nazve?
//lebo ho tam nikto neda
// Tak spravna odpoved je len posledna moznost, alebo aj prva ? kto ma knihu nech kukne pls
// Ada: Aj prva moznost, prednaska ER_model strana 4.
//mnicky: podla mna velmi diskutabilne. Ja by som asi nakoniec dal 1. a 5. moznost, ale mozno aj ta 3. a 4. nie su zle.
//liho: 3 je urcite dobre - vychadzajuc z definicie konceptualnej schemy: Konceptuálna schéma je implementačne nezávislá množina dát popisujúca dátový model.
//lanim09 nepopisuje sa nahodou pouzivatelska aplikacia(pouzivatelske pohlady) na extrenej urovni, tym padom by si odpoved odporovala a mala by byt false

-

4. Systém riadenia bázy údajov by mal zabezpečovať
" T " **obnovu databázy**
" T " **ochranu údajov**
" F " zálohovanie databázy
" T " **integritu databázy**
" T " **manipuláciu s dátami**
" T " **paralelný prístup**
" F " normalizáciu databázy
" **T** " **pripojenie k databázam**
" F " generovanie DDL opisu
" T " **definíciu dát**
" F " úložný mechanizmus
" T " **riadenie prístupu**
//materialy uvod str. 20
//mnicky: v knihe (zac. kap. 3) ma aj 'pripojenie k databazam', tak pridavam
//mnicky: a keby nahodou na skuske dal aj 'riadenie katalogov', tak to tam patri tiez
//pd: dosť divné, keď dá do možností veci čo v materiáloch nie sú, ale v knihe áno

5. Ktoré typy údajov sa vzťahujú k databázovému systému?

" F " viazané
" T " **výstupné**
" T " **vstupné**
" T " **perzistentné**
" F " odvodené
" F " volné
//materialy uvod str. 6

6. Relácii v relačnom dátovom modeli zodpovedá (len jedna správna)
"F" stĺpce tabuľky
"**T**" **tabuľka**
"F" riadok tabuľky
"F" stĺpec tabuľky
"F" databáza
//4. prednaska 2.strana

7. Logický žurnál obsahuje informácie, ktoré umožnia obnoviť "**konzistenciu**" databázy v prípade neukončenia transakcie.

8. Databáza je konzistentná, ak vyhovuje množine "**integritných**" obmedzení.
//mnicky: INTEGRITNYCH ??
//taj: tiez si myslim mas to asi z kapitoly 1.3.5 ochrana proti nekonzistencii dat?
//mnicky: tak to oznacujem za spravnu odpoved. Keby mal niekto nieco proti, ozvite sa ;-)

-

//rackom: myslim ze som to niekde videl a bolo to integritnych

9. Keď je transakcia zrušená, potom príkazom "**ROLLBACK**" zabezpečíme obnovenie pôvodného stavu databázy.

10. Kandidátom na primárny kľúč je jeden alebo viac "**atribútov**" .

11. Je primárny kľúč integritným obmedzením relácie?
"**T**" **ÁNO**
"F" NIE
//mnicky: kniha, kap. 11.1

12. Napíšte klasifikáciu integritných obmedzení pre údaje relačnej databázy (5 správnych odpovedí)
"**T**" **integrita stĺpcov**
"**T**" **referenčná integrita**
"**T**" **integrita entít**
"F" integrita vstupu
"**T**" **používateľská integrita**
"F" integrita riadkov
"F" konzistentná integrita
"**T**" **doménová integrita**
"F" prípustná integrita

//mnicky: kniha, kap. 11.1

13. Ktorý z ponúkaných prvkov môže nadobúdať hodnoty z domény?
"F" relácia
"F" tabuľka
"F" databáza
"F" riadok tabuľky
"**T**" **atribút**
//taj: Pre každý atribút sa definuje odbor prípustných hodnôt - doména. materialy 03_02ERmodel , navrhujem ostatne ako false
//mnicky: “domena je mnozina hodnot, ktore moze nadobudat atribut” (kniha, kap. 5.2)

14. V relačnom dátovom modeli je primárny kľúč tabuľky reprezentovaný "**stĺpcom**" alebo skupinou "**stĺpcov**".

//mnicky: ATRIBUTMI alebo skupinou ATRIBUTOV ?
//martingt89: ciatal som aj ze, stlpcom alebo skupinou stlpcov
//taj: suhlas s martingt89 je to na str2 v materialoch 04_3reldbs
//mnicky: suhlasim, ked hovorime o tabulke, skor tam budu stlpce ako atributy

-

//pd: nahodená odpoveď, keď je to v materiáloch tak nie je o čom

15. Základné vlastnosti transakcie (4 správne odpovede)
"F" konečnosť
"F" paralelnosť
"T" **izolovanosť**
"F" úplnosť
"F" sériovosť
"T" **konzistencia**
"T" **atomicita**
"F" determinovanosť
"F" rezultatívnosť
"F" jednoduchosť
"T" **perzistencia**
"F" neobsahuje cyklické operácie

//mnicky: kniha, kap. 14.4
//jm: Z angl. ACID(Atomicity, Consistency, Isolation, Durability)

16. Príkazy, ktorými sa dá ukončiť spracovanie transakcie sú:
"F" END
"F" CHECK
"F" UPDATE
"F" INTERRUPT
"F" CLOSE
"F" WRITE
"**T**" **COMMIT**
"**T**" **ABORT**
"T" ROLLBACK

//liho: yop, agree, ROLLBACK je tiez T, povodna diskusia je na konci

17. Patria príkazy DECLARE, OPEN, CLOSE, FETCH k základným príkazom pre prácu s kurzormi ?

"**T**" **ÁNO**
"F" NIE
//mnicky: spravne a este tam patria aj UPDATE, DELETE (kniha, kap. 10.3.2)

18. Spracovanie transakcií metódou 2PhC znamená rozdelenie transakcie na dve fázy.

-

"T" V prvej fáze sa všetky zmeny zapisujú do logického žurnálu. V druhej fáze sa prepisujú zmeny z logického žurnálu do databázy.
"F" V prvej fáze sa všetky zmeny aktualizujú v databáze. V druhej fáze sa zmeny uchovajú do logického žurnálu.
"F" V prvej fáze sa priradí transakcii časová pečiatka. V druhej fáze sa zapíšu do databázy iba tie údaje, ktoré prislúchajú časovej pečiatke.
//mnicky: kniha, kap. 14.7.2

19. Logické objekty databázy je možné vytvoriť pomocou štandardného dotazovacieho jazyka "**SQL**".
//tiež by som povedal, že SQL je odpoveď
//mnicky: ved ono SQL znamena ~~'standard~~ structured query language' ;-) menim na odpoved...
//a nie nahodou 'structured query language'? :)
//mnicky: tak jo ;-)

20. Štruktúrovaný dotazovací jazyk je najrozšírenejším databázovým jazykom, ktorý sa stal štandardom v oblasti spracovania dát. Oficiálny počiatočný štandard má nasledovné základné časti:

"F" jazyk pre zdieľanie dát z rôznych databáz
"T" jazyk pre manipuláciu s dátami
"T" jazyk pre riadenie prístupu k dátam
"F" jazyk na bezpečné zálohovanie databázy
"T" jazyk pre definíciu dát
"T" jazyk pre riadenie správy transakcií
"F" jazyk pre komunikáciu s procedurálnym rozhraním
"F" jazyk pre spracovanie dokumentov
"F" jazyk na prácu s rámcami
//mnicky: kniha, kap. 6.1.

- Anglicky su to: Data Manipulation Lang., Data Access Statements (niekedy aj Data Control Lang.), Data Definition Lang., Data Integrity Statements (niekedy aj Transaction Control Lang.) - prve pismena tvoria skratky.
- vid. aj. http://is.gd/sql_parts

21. Charakterizujte databázový systém

"T" Pod databázovým systémom (DBS) je treba chápať množinu navzájom súvisiacich dát spoločne s programovým vybavením, ktoré umožňuje prístup a manipuláciu s dátami.
"T" Databázový systém tvoria nasledovné komponenty: Dáta, Hardware,Software, Používatelia
"F" Databázový systém je počítačový systém údajov, spracovávaný na personálnych

-

počítačoch
"**T" Databázový systém (DBS) tvorí databáza (DB) a systém riadenia bázy dát (SRBD)**.

//taj: to prve jednoznacne T je to na strane 4, kapitola 1.1 :)

22. Priraďte typické činnosti jednotlivým triedam používateľov databázových systémov. (na vyber: databázový administrátor, koncový používateľ, aplikačný programátor)

spolupráca s používateľmi "**databázový administrátor**"
monitorovanie výkonnosti systému a ladenie systému "**databázový administrátor**"
umiestenie dát v databáze "**databázový administrátor**"
údržba dát "**databázový administrátor**"
definovanie bezpečnostných pravidiel pre prácu s databázou "**databázový administrátor**"
komunikácia cez rozhranie "**koncový používateľ**"
definovanie archivačných pravidiel a systému obnovy databázy "**databázový administrátor**"
tvorba programov, ktoré využívajú databázu "**aplikačný programátor**"
formulovanie požiadaviek na dáta v systéme "**koncový používateľ**"

// 02_1uvod strana 9
//mnicky: podla mna nejednoznacna, odveci otazka - ved. napr. data moze do db dat aj uzivatel alebo programator, nie iba admin...
//ano ale tam v tej moznosti ide o “umiestnenie dat”, cize skor tym mieria na architekturu podla mna

23. Redundancia dát znamená

"F" opakovanie tabuliek
"**T" výskyt rovnakých atribútov v rôznych tabuľkách**
"F" veľa údajov v jednej tabuľke
"**T" viacnásobný výskyt tých istých údajov v databáze**

//mnicky: myslite ze aj ta druha moznost je spravna? v knihe som nasiel iba tu stvrtu (aj ked je pravda ze stvrta by _mohla_ implikovat druhu...)
//connors: ked som sa nad tym zamyslel, podla mna by ta 2. moznost nemala byt spravna. Ved mozme mat 2 tabulky, ktore obsahuju napriklad atribut id, ale vtedy predsa urcite nejde o redundanciu. Teda ak som spravne pochopil co myslia tym rovnakym atributom... :)
//joffo:suhlasim s connorsom :D
//philips :stvrta je dobre, google prvy odkaz po zadani redundancia dat.
//pd: prvá veta na wikipedii pre data redundancy: “Data redundancy occurs in database systems which have a field that is repeated in two or more tables.” - z toho mi vychádza definitívne aj dvojka ako TRUE.
//mnicky: ide o to, co chapes pod pojmom ‘field’. Ci nazov, alebo vyznam (v prvom pripade to redundancia nebude, v druhom uz ano).
//pd: wikipedia znie skorej k tej druhej možnosti, spomínajú tam adresu kupujúceho v rozličných

-

reláciách a odstránenie tohoto normalizáciou a cudzími kľúčmi; hrozná otázka toto :D
//inak na skuske 0708rt je pod touto otazkou ze moze byt viacero spravnych odpovedi, tak ja sa prikladam k obom
//Kveri: podla mna ta druha odpoved nedava zmysel, co je to rovnaky atribut? a. rovnake meno - moze byt a nie je to redundancia b. rovnaky typ/vyznam/constraint - moze byt a nie je to redundancia c. rovnake data - tiez to nie je redundancia.
//Kveri: druhe nemoze byt spravne, potom by sa to vztahovalo aj na ID

24. Structured Query Language je úplný databázový "**jazyk**". Obsahuje príkazy na:
1.začiatočné "**vytvorenie**" databázy
2.vytvorenie k tomu prislúchajúcich "**fyzických**" objektov
3.vytvorenie logických objektov
4.definovanie autorizácie a riadenia "**prístupov**" pre spravovanie databázy
5."**vyhľadávanie**" a manipuláciu dát v databáze

//taj: zaujimave ale odpovede su na http://www.oernii.sk/sxool/4roc/kubiki/DBS1/DCS2/SQL.TXT kodovanie IBM 852, ak suhlasite dopisem odpovede do dokumentu
//connors: ja osobne som za, odpovede presne sedia z danym dokumentom a btw, preskocili ste 25 :)
//taj: doplnila som:)
//mnicky: nemali by byt slova v moznostiach dva a tri prehodene?
//philips: tiez sa mi to nejak logicky nesedi.
//connors ak myslite to fyzickych a logickych tak moznost 3 je uz napevno dana v skuske, nedala sa menit, podla mna je to spravne ako to je teraz
//philips: aha ok.

26. Čo znamená integrovanie údajov?
"F" Integrovanie dát znamená, že uložené data v súboroch navzájom nesúvisia
"F" Integrovanie dát znamená, že dáta udržiavané v databáze môžu byť umiestnené v jednom súbore tak, aby bola minimalizovaná duplicita výskytov dát a zároveň, aby program mohol súčasne sprístupniť dáta z viacerých súborov
"T" Integrovanie dát znamená, že dáta udržiavané v databáze môžu byť umiestnené vo viacerých súboroch tak, aby bola minimalizovaná duplicita výskytov dát a zároveň, aby program mohol súčasne sprístupniť dáta z viacerých súborov

//02_1uvod strana 6

27. Databázový systém môžeme chápať ako počítačový systém správy uložených záznamov v súbore. V takom systéme sú najdôležitejšie nasledovné funkcie: (na vyber: výber, oprava, zrušenie, vloženie, vytvorenie)
"**vyber**" dát z existujúceho súboru
"**oprava**" dát v existujúcom súbore
"**zrusenie**" dát z existujúceho súboru
"**vlozenie**" nových dát do existujúceho súboru

-

"**vytvorenie**" nového prázdneho súboru, do ktorého sa budú ukladať dáta
//mnicky: starsie poznamky ku skuske (dbs_zhrnutie_uciva_na_skuskuII.doc)

//taj: na strane 4v knihe je to iste a je tam “pridanie” miesto “vytvorenie”, len ak by to bolo automaticky porovnavane...dajte si pozor
// nestras, nedopisovalo sa, ale vyberalo z danych moznosti

28. Príkaz INSERT INTO KURZY umožní aktualizovať hodnoty údajov v tabuľke s menom KURZY.

" F " ÁNO
" T " **NIE**
// INSERT INTO vkladá nové údaje, na aktualizovanie slúži UPDATE a nazvy tabuliek by nemali byt velkymi pismenami

29. Pri spracovaní transakcií sa používa technika zamykania objektov, ktorá umožní "**paralelný**" prístup k zdieľaným objektom.
// Ada: ako moze byt paralelny pristup pri zamykani objektov?
//mnicky: tiez mi to znie divne, ale v knihe sa pise: “zamykanie je technika, kt. umozni paralelny pristup k zdielanym objektom DB.” (kap. 15.4.1)
//pd: vďaka lockom je možný paralelný prístup bez toho aby vznikla nekonzistentná databáza, asi myslia to
//Kveri: su rozne urovne lockov, mysql ma asi 5, postgres asi 8, v strucnosti: read lock - chces citat, nikto iny nemoze pisat ale ostatni mozu citat, write lock - chces pisat, nikto iny nemoze citat ani pisat.

30. Databáza je množina x dát, ktorá je používaná v aplikačnom systéme daného používateľa. Na mieste pismena x vyberte správnu možnosť.

"F" dokumentačných
"F" vstupujúcich
"F" textových a číselných
"F" konzistentných
"F" závislých
"F" neredundatných
"F" integritných
"**T**" **perzistentných**

31. Základný postup pri tvorbe entitno-relačného modelu je nasledovný:
1. identifikujú sa "**typy entit**" ako množiny objektov rovnakého typu.
2. identifikujú sa "**typy vztahov**", do ktorých môžu vstupovať entity identifikovaných typov
3. na základe primeranej úrovne abstrakcie sa priradia jednotlivým typom entít a vzťahov "**atributy**", ktoré bližšie popisujú vlastnosti entít a vzťahov.
4. formulujú sa rôzne "**IO**", vyjadrujúce s väčšou či menšou presnosťou súlad schémy s

-

modelovanou realitou.

//03_2ERmodel strana 5
//IO = integritne obmedzenia
//pd: pozn.: v tejto úlohe nie sú option boxy, ale treba dopisovať ručne ;), aby ste sa nespoliehali náhodou...

32. Návrh popisu údajov na konceptuálnej úrovni sa zvyčajne vyjadruje

"**T**" **konceptuálnym modelom**
"F" diagramom toku údajov
"F" funkcionálnym modelom
"?" koncepčnou schémou
"F" diagramom aktivít
"**T**" **entitno-relačným modelom**
//mnicky: aspon podla knihy, kap. 4.3 a 4.4 sa da tak usudzovat. *Koncepcna schema* sa v knihe nespomina, iba *konceptualna schema*. Je otazne ci ide o to iste...
//pd: no idea :/

33. Označte správne komponenty architektúry rozhrania ODBC.

"F" ESQL Source Component
"F" ODBC API
"F" Database Request Module
"F" Precompiler
"F" ODBC CLI
"**T**" **Application**
"F“ ODBC SQL
"?" Drive Manager // Ak sa mysli Driver Manager tak ANO
"**T**" **Data Source**
"F" Linker
"**T**" **Driver**

34. Ak chceme jednu či viac operácií nad databázovými objektami spracovať s transakčnou podporou, je potrebné tieto operácie zapuzdriť príkazmi, ktoré ohraničujú začiatok a koniec príslušnej "**transakcie**"

//transakcie?
//mnicky: ano, transakcie - kniha, kap. 14.3.1, priklad 14.4

35. Vyberte typy kurzorov, ktoré sú implementované v MS SQL Server 2000 (Počet správných odpovědí je 4. Vyberte je.)

-

"F" cyklické kurzory
"F" sekvenčné kurzory
"T" **rýchle kurzory typu "iba dopredu"**
"F" kurzory typu " iba dozadu" //tiez by som bol za ROLLBACK
"T" **statické kurzory**
"T" **kurzory riadené sadou kľúčov**
"T" **dynamické kurzory**
//mnicky: osobne by som dal skor *sekvencne kurzory*, ako *staticke...*
//prednaska kurzory.doc *staticke*
//mnicky: v knihe su zasa niekde spominane staticke, ale nie presne v tomto kontexte, takze moze byt ze je to OK
//mnicky: snad tam taketo 'implementacne zavisle' otazky tento rok neda

36. Jedna z architektúr SRBD využíva lokálne počítačové siete. Na jednom počítači sú v súboroch uložené dáta. Operačný systém na tomto počítači zabezpečí používateľom z ostatných počítačov prístup k dátam.
Aplikácie aj dáta sú spracovávané na používateľských počítačoch. O akú architektúru sa jedná?

"F" klientská
"F" globálna
"**T**" **súborová**
"F" lokálna
"F" sieťová
"F" distribuovaná
"F" centralizovaná

//02_1uvod strana 26

37. Napíšte veľkými písmenami skrátený názov jazyka, ktorý je použitý v nižšie uvedenom kóde "**ESQL**".

```
main() {
EXEC SQL BEGIN DECLARE SECTION;
int OrderID;
int CustID;
char SalesPerson[10];
char Status[6];
EXEC SQL END DECLARE SECTION;
/* Zadanie čísla objednávky */
printf („Zadaj cislo: ");
scanf ("%d", &OrderID);
/* Vykonanie SQL dotazu */
EXEC SQL SELECT CustID, SalesPerson, Status FROM Orders
WHERE OrderID = :OrderID
INTO :CustID, :SalesPerson, :Status;
```

-

```
/* Zobrazenie výsledku */
printf („Cislo zakaznika: %d\n", CustID);
printf („Meno predajcu: %s\n",sc SalesPerson);
printf ("Status: %s\n", Status);
exit();
}
```

//mnicky: podla googlu vyzera ze ide o DYNAMIC SQL (DSQL). Kto ma rovnaky nazor??

//taj: mne to znie presne ako toto: http://www.postgresql.org/docs/9.1/static/ecpg-sql-connect.html pozrite ten dlhy example

//mnicky: hej, ale tam ide iba o ukazku pouzitia prikazu SELECT, nie o to, aky jazyk je toto. A ja si myslim, ze ide o SQL, konkretne o Dynamic SQL …

//taj: no skusala som hladat dalej vyslo mi ze je to presne odtialto: http://msdn.microsoft.com/en-us/library/ms714570(v=vs.85).aspx embedded sql, ale neviem ako sa presne cleni sql ci tamten nazov je odpoved... pozri to prosim ta, skor mi to pride ako metoda kombinacia c a sql, nie jazyk

//mnicky: je mozne ze ide o *Embedded SQL*. Najlepsia je vsak poznamka z wikipedie: “Embedded SQL for C has been deprecated as of Microsoft SQL Server 2008 although earlier versions of the product support it.” (http://en.wikipedia.org/wiki/Embedded_SQL)

//taj: tak ja neviem mohol by sa aj niekto dalsi vyjadrit :D, dobra poznamka btw
//podla mna urcite Embedded SQL
//pd: podľa toho čo čítam ako by malo ESQL vyzerať, mohlo by to byť naozaj ono
//mnicky: davam ESQL ako spravnu odpoved

38. Konceptuálna úroveň architektúry databázového systému vyjadruje (len 1 správna)

"T" abstraktnú reprezentáciu dátového modelu
"F" opis reality esenciálnym modelom
"F" funkcionálny model
"F" koncept uloženia údajov
"F" konceptuálny pohľad používateľa

//mnicky: ja by som dal tu prvu moznost, ale nie som si isty...
//taji: suhlas - strana 24 v knihe na zaciatku kapitoly 2.2 konceptualny pohlad je abstraktnou reprezentaciou celeho modelu (aj ked nepisu ze datoveho :)
//bal: suhlasim s prvou moznostou, ties vychadzam z knihy...

//pd: z časti Konceptuálne modelovanie v materiáloch: “Konceptuálne modely sú pokusom

-

umožniť vytvorenie popisu dát v databáze, t.j. konceptuálnej schémy, nezávisle na fyzickom uložení databázy. Tento popis by mal čo najvernejšie vystihovať konceptuálny pohľad človeka na danú časť reálneho sveta.” - z toho mi vychádza posledná možnosť

//mnicky: Nie. Ved o pohlad pouzivatela sa stara ‘externa uroven’. Konceptualna sice ma ‘co najvernejsie vystihovat’ pohlad cloveka, ale je pokusom o ‘vytvorenie popisu dat v db’, cize to ma byt PRVA moznost. Z knihy: “Externa uroven predstavuje uroven jednotlivych pouzivatelov. (...) Konceptualny pohlad sa moze dost vyrazne lisit od sposobu pouzivatelskych definicii v externych pohladoch. (...) Konceptualny pohlad reprezentuje cely informacny pohlad na databazu, odvodeny z ext. pohladov.” (kap. 2.2.1 a 2.2.2). Cize koncept. pohlad je _odvodeny_ z pouzivatelskych, ale _nie je totozny_.
...takze navrhujem oznacit PRVU MOZNOST.

//pd: okey dajme prvú

39 Máme databázu, ktorá umožňuje spravovať údaje o študentoch, predmetoch a o výberoch predmetov študentami.
Databáza obsahuje nasledovných 5 tabuliek:
1/ CREATE TABLE Student
(Id_student Int, Meno Char, Priezvisko Char, Rocnik_studia Int, Odbor Char)
2/ CREATE TABLE Predmet
(Id_predmet Int , Kredity Int, Nazov Char, Odbor Char, Rocnik Int, Semester Char, Skratka Char, Ucitel Int, Hodin_prednasok Int, Hodin_cviceni Int, Povinny Char, Volitelny Char)
3/ CREATE TABLE Vyber
(Id_predmet Int, Id_student Char, Kredity Int, Rok Char, Ucit_cvici Int, Body_za_cvic Int, Datum_skusky Date, Ucit_skusa Char, Body_za_skusku Int, Znamka Char , Roc_studia Char)
4/ CREATE TABLE Ucitel
(Id_ucitela Int, Meno Char, Priezvisko Char, Tituly Int, Ustav Char)
5/ CREATE TABLE Uci
(Id_predmet Int, Id_ucitel Char, Akad_rok Char, Prednasa Bool, Cvici Bool)

Navrhnite a zdôvodnite potrebné zmeny, ktoré odstránia anomálie a chyby v definíciách tabuliek. Ku každej tabuľke
definujte aj primárny kľúč a všetky potrebné cudzie kľúče a následne napíšte opravené výsledné definície tabuliek.

Tabulka Student - Id_student PK, Odbor int FK

Tabulka Predmet - Id_predmet PK, Semester int, Ucitel FK, Povinny bool, Volitelny bool, Odbor vytvorit prepojovaciu tabulku Predmet - Odbor (jeden predmet moze byt vo viacerych odboroch)

Tabulka Vyber - Id_vyber PK, Id_predmet FK, Id_student int FK, rok int, Ucit_cvici odstranit, Ucit_skusa odstranit, Znamka int, Roc_studia int, Kredity odstrnit vytvorit

-

prepojovaciu tabulku predmet_odbor

Tabulka Ucitel - Tituly char, Ustav int FK, Id_ucitela PK

Tabulka Uci - Id_uci PK, Id_predmet FK, Id_ucitel int FK, Akad_rok int

//este by trebalo zapisat to v sql, ale netusim ci je toto spravne a co myslel tou normalizovanou formou
//taktiez netusim ako ten navrh ma byt podrobny a az s akymi extremnostami mam pocitat
//tie sql prikazy by niekto nevedel vypracovat, zdaju sa mi p
nekud zlozite :((

//mnicky: moje priklady na MyISAM engine su v skuske nizsie

40. K zadefinovaným tabuľkám uveďte SQL príkaz, ktorým vypíšete zoznam učiteľov (priezvisko, meno) a ku každému z nich v jeho predmete počet študentov, ktorých v tom predmete cvičí - priezviská a mená usporiadané abecedne.
//Matho:

```
SELECT priezvisko, meno, COUNT( v.student_id ) AS pocet_studentov_ktorych_cvici
FROM `mnicky_v2_vyber` v
JOIN `mnicky_v2_uci` u ON v.predmet_id = u.predmet_id
JOIN `mnicky_v2_ucitel` ul ON u.ucitel_id = ul.id_ucitel
WHERE u.cvici = '1'
GROUP BY priezvisko, meno
ORDER BY priezvisko ASC , meno ASC
```

//Matho: dobrym zvykom byva vytvorit si skratky z tabuliek - a pri joinovani sa odvolavat na stlpce v tvare skratka.stlpec aby sme vedeli z ktorej tabulky nam taha data ked mame vo viacerych tabulkach rovnake nazvy slpcov

41. K zadefinovaným tabuľkám uveďte SQL príkaz, ktorým vypíšete zoznam odborov, ročníkov, priezvisk a mien všetkých študentov usporiadaný po ročníkoch, odboroch a abecedne podľa priezviska a mena./u
//podujeme sa niekto aj na ostatne SQL
//Matho: sql su vypracovane v skuske nizsie

42. K zadefinovaným tabuľkám uveďte SQL príkaz, ktorým vypíšete po ročníkoch zoznamy názvov predmetov končiacich skúškou a ku každému predmetu maximálny, minimálny a priemerný počet získaných bodov zvlášť za cvičenia a zvlášť za skúšku - usporiadaný najprv podľa ročníkov a v nich abecedne podľa názvu predmetu.

43. K zadefinovaným tabuľkám uveďte SQL príkaz, ktorým vypíšete usporiadane pre každý akademický rok, ročník štúdia v každom odbore počet získateľných kreditov z povinných

-

predmetov.

44. K zadefinovaným tabuľkám uveďte SQL príkaz, ktorým vypíšete zoznam názvov a skratiek predmetov a ku každému predmetu aj jeho prednášajúceho ako aj zoznam učiteľov cvičiacich daný predmet - názvy predmetov, ako aj priezviská a mená učiteľov budú usporiadané abecedne.

RT 2008/2009 /2

1. Požiadavky pre dobre definovanú koncepuálnu schému sa dajú zhrnúť do 8 bodov. Doplňte prázdne miesta
1. Žiadny typ entity nemá v konceptuálnej schéme viac ako " **jeden**" zdroj ISA hierarchie.
2. ISA vzťahy netvoria v E-R diagrame orientovaný "**cyklus** ".
3. Identifikačné typy vzťahov netvoria v E-R diagrame "**orientovany**" cyklus.
4. Typ entity v ISA hierarchii, ktorý nie je zdrojom, nie je identifikačne "**zavisly**" na žiadnom type entity.
5. Mena typov entít a vzťahov sú jednoznačné "**globalne**" mená, zatiaľ čo mená atribútov vrátane zdedených sú jednoznačne lokálne mená v rámci daného typu objektu
6. Ak vystupuje typ entity v type vzťahu viackrát, je charakterizovaný rôznymi úlohami, t.j. ďalšími identifikátormi.
7. Ak je typ entity zdroj ISA hierarchie, potom ma "**identifikacny**" kľúč. Ostatné typy entít nemajú indentifikačný kľúč.
8. Ak je identifikovaný slabý entitný typ E pre dva rôzne identifikačné vzťahy ten istý, potom buď:
(a) pre E existujú dva "**rozne**" identifikačné zdroje, alebo
(b) ten istý identifikačný zdroj vystupuje v dvoch úlohách a tie je nutne odlíšiť označením úloh

//03_2ERmodel strana 23

4. Pri výskyte dvoch entít poznáme kardinalitu typu:
"T" **1:N**
"T" **1:1**
"F" MN:MB
"T" **M:N**
"F" 0:0
"F" N:1

//v prednaske vztah N:1 nema
//ja som tam nasiel aj taky vztah, sice bol v zatvorke ale bol tam...
//nie je náhodu do isté 1:N a N:1 len to je opačne zapísané? preto to možno bolo aj v zátvorke
//ja som tiez za to, ze by to N:1 tam malo byt, vsak to je podla mna uplne logicke
//mnicky: vsimnite si, ze v otazke je rec o TYPE kardinality, nie o kardinalite ako takej.
Cize podla mna tam patria iba tie tri zakladne. Navyse v knihe (kap. 4.4.5.1) sa ine typy

-

nespominaju...
//Matho - pripajam sa k mnickemu. //dury:nesuhlasil som, ale uz suhlasim s mnickym, Mathom

5. Nech predmety sú najskôr identifikované svojou skratkou napr. AL1, SI3, a potom svojim názvom, napr. ALGEBRA, SIETE. Napíšte príklad jednej inštancie z uvedených parametrov "**AL1 ALGEBRA**".
//to co je toto za otazku? chape tomu niekto?
//jedna instancia by mala byt jeden riadok cize "AL1 ALGEBRA", aspon ja som to tak pochopil...
//pd: inštanciu by som chápal ako riadok, čiže AL1 ALGEBRA znie fajn podľa mňa
//mnicky: jj, suhlasim. A v knihe je tiez daco podobne (kap. 4.4.1)

6. Ideálne spracovanie množiny transakcii {T1, T2, …. , Tn} je paralelne
"F" ANO
"T" NIE
//a nie je to nahodou "seriove"?
//mas pravdu, odpoved bola zmenena

//jm: Z hľadiska konzistencie databázy by bolo ideálne, aby množina transakcií, ktoré potrebujeme vykonať, bola spracovaná sériovým spôsobom. Toto sekvenčné vykonanie transakcií je také, ktoré vylúči súčasné spracovanie operácií viacerých transakcií, zodpovedá špeciálnemu rozvrhu, pri ktorého spracovaní nedochádza k strate výsledkov operácií a ani k nekonzistencii. Takým rozvrhom je **sériový rozvrh**. - 10_9paralelizmy.doc

7. Stav databázy sa mení počas operácií X. Po ich vykonaní je dôležité zabezpečiť, aby databáza bola v konzistentnom stave. Vyberte za X správnu množinu SQL príkazov.
"T" insert, update, delete
"F" grant, select, insert
"F" create, selextm update
"F" insert, create, drop

8. Spracovanie transakcií metódou priameho zápisu je charakteristická tým, že pri každej operácii "**aktualizujeme**" logický žurnál a po modifikácii logického žurnálu aj samostatnú databázu (na vyber: aktualizujeme, resetujeme, uchováme, presunieme, prečítame, nezmeníme)

//zeby modifikujeme? ;)
//mnicky: "updatujeme" - je to veta z knihy, kap. 14.7.1
//a nie "aktualizujeme"?
//update == aktualizovat
// ale je napisane vybrat z moznosti takze aktualizujeme
// pd: vidíš, tu sú aj možnosti, opravené na "aktualizujeme", keby niečo tak aj updatujeme je dobre

-

9. Pri paralelnom spracovaní transakcii je nutné riešiť poradie spracovania jednotlivých operacií pomocou rozvrhov.
"T" **ANO**
"F" NIE

10. Spracovanie transakcii metódou CHECK POINT spočíva v dvoch fázach. Prvá fáza predstavuje periodické zaznamenávanie informácií o stave databázového systému v kontrolných bodoch. Druhá fáza sa používa len pre "**obnovu databazy**" po systémovej chybe (na vyber obnovenie databázy, prepísanie logického žurnálu, obnovenie pôvodného stavu logického žurnálu, čítanie databázy)

11. Príkaz INSERT patri do skupiny DDL jazyka SQL
"F" ANO
"T" **NIE**

12. V roku 1992 vznikol nový štandard štruktúrovaného dotazovacieho jazyka. Jeho skrátene označenie je "**SQL2**".

//neviem isto
//nie SQL2? 12.12.
//mnicky: hej, asi SQL2 - vid. http://en.wikipedia.org/wiki/SQL#Standardization
//pd: no na prednáške stále spomínal čosi s SQL-92 štandardom, neviem či to nebude ono
//peto : sql2 = sql-92 (http://en.wikipedia.org/wiki/SQL -tabulka)
//pd: ja len tak, ak to vyhodnocuje automaticky, že asi tam bude skorej to SQL-92 ;)
//fillky: vzniklo to v roku 1992 cize SQL-92 je logicke. Zdroj http://en.wikipedia.org/wiki/SQL-92
//mnicky: otazne je, co sa povazuje za 'skratene oznacenie' - ci SQL-92 alebo SQL2...??
//tempus: na wikipedii je spominane SQL-92, ja som skor za to... I ked revizia z 1999 sa uz // oznacovala ako SQL-3
//http://en.wikipedia.org/wiki/SQL-92

12. Napíšte typ údajovej štruktúry, do ktorej je uložená relácia v pamäti počítača " "
//connors: tata otazka bola vynechana, tak som ju doplnil, skoda ze neviem aj odpoved :)
// fragment ? Mozno aj tabulka na wiki: http://en.wikipedia.org/wiki/Data_model#Three_perspectives posledny odsek;
//taj: navrh: tabulka? :P 5.2. str91 kniha

13. Relačný dátový model je založený na
"F" teórii množín
"T" relačnej algebre
"T" teórii relácii
//04_3reldbs strana 3
//ja by som doplnil aj teoriu mnozin "V 70-tych rokoch vznikajú relačné databázové systémy, ... Hlavným prínosom týchto systémov je, že sú postavené na matematickej teórii množín s využitím relačnej algebry a kalkulu." Prva prednaska strana 13

-

//connors: ta poucka co si napisal plati pre databazovy system, nie je pre datovy model, podla mna je to dobre ako to je teraz
//pd: v jednej zo starých skúšok bola podobná otázka, len jedna možnosť správna a tie dve čo tu máme vyznačené boli práve v tej jednej možnosti, takže by to malo byť dobre

14. Najpoužívanejší databázový systém je “**relacny**” databázový systém. Ma tieto charakteristiky:

- je založený na teórii “**relacii**”, množinovom “**pojme**” relácií a relačnej “**algebre**”
- všetky vzťahy sú reprezentované vo forme “**tabuliek**”, pretože dvojrozmerné “**tabulky**” stačia na modelovanie reálnych vzťahov
- kľúčovým pojmom používaným pri relačnom modelovaní je “**relacia**”

//04_3reldbs strana 3

(na vyber: relácii, tabuľky, pojme, relácia, tabuliek, algebre, relačný)

17. Vyberte názov všetkých skupín príkazov štandardného dotazovacieho jazyka pre relačné databázy:
"F" DAU
"F" DAL
"**T**" **DML**
"**T**" **DDL**
"F" DTL
"**T**" **DAS**
"F" DCB
"F" DIB
"**T**" **DIS**
"F" DCT
"F" DDD
"F" DTS
"F" DMA
"F" DMD

19. Systém na riadenie bazy údajov je množina “**programov**”, ktorá zabezpečuje definíciu a manipuláciu s dátovými štruktúrami
//taj: asi “programov” kapitola 3, prva veta, nejaky suhlas?
//kudlohlavec: jj suhlasim- “programov”

20. Označte zakladné typy databazového systému podľa historického vývoja
"F" transakčný systém
"**T**" **expertný systém**
"**T**" **objektový systém**
"**T**" **textový systém**
"F" globálny systém

-

"F" entitno-relačný systém
"T" súborový systém
"F" homogenný systém
"F" korelačný systém
"F" heterogénny systém
"T" objektovo-relačný systém
"F" lokálny systém
"T" relačný systém
"F" komunikačný systém
"T" znalostný systém

//02_1uvod
//mnicky: ja tam vidim aj relacny system :-). Taktiez textovy system je spominany na rovnakej urovni ako znalostny a expertny (kniha, kap. 1.5), tak bud su vsetky tri T alebo F
//aj entitno-relacny?
//pd: entito-relačný tam spomenutý nevidím nikde, dávam ho ako F

21. Môže byť redundancia údajov výhodná?
"T" ANO
"F" NIE
//asi ANO - vid NoSQL databazy typu CouchDB, ALE podla 02_1uvod strana 11 NIE - redundancia by mala byt odstranena, alebo minimalizovana
//mnicky: jasne ze moze. napr. pri optimalizacii zameranej na rychlost
//taj: v knihe pisu ze ano konkretne str. 17, kap. 1.3.4 ale tam si to asi aj nasiel, takze ano:)

22. Perzistentné údaje sú
"F" údaje, ktoré sa spracovávajú ale neukladajú do databázy
"F" všetky vstupné údaje
"T" údaje, ktoré existujú aj po skončení programu
"F" všetky výstupné údaje

23. Označte základné funkcie systému riadenia bázy dát
"F" komunikácia so serverom
"F" prihlásenie používatela
"T" manipulácia s údajmi
"F" sériový prístup k údajom
"T" definícia údajov
"F" definícia rozhrania
"F" mapovanie jednej schémy do druhej

//02_1uvod strana 21
//mnicky: kniha, zac. kap. 3

-

25. Externá úroven architektúry databázového systému sa týka (len 1 správna)
"F" opisu exteriéru
"F" externých metód
"F" externých pracovníkov
"T" individuálych používateľov
"F" opisu prístupových spôsobov k externým údajom
"F" aplikačného rozhrania

//02_1uvod strana 17

26. Na hlavnom pocítači sa spracovávajú požiadavky od používateľov a spúšťajú aplikácie, ktoré používajú dátové služby. Aplikácia a SRBD komunikujú cez spoločnú pamäť, ktorá je prideľovaná operačným systémom. Používatelia komunikujú s databázovym systémom prostredníctvom terminálu. O akú architektúru sa jedná? (1 spravna)
"F" serverová
"F" dátova
"T" centralizovaná
"F" prezentačná
"F" súborová
"F" terminálová

//02_1uvod strana 24

27. Vyberte z uvedených možností tie správne, ktoré definujú kurzor
"?" Kurzor je mechanizmus, ktorý spracováva množinu riadkov získaných príkazom SELECT
"T" Kurzor je mechanizmus, ktorý umožní sprístupniť riadky tabuľky
"F" Kurzor je pozičný ukazovateľ spojený so stĺpcami tabuľky databázy
"F" Kurzor je ukazovateľ ľubovolného riadku na obrazovke
"T" Kurzor je mechanizmus, ktorý spracováva dáta množinového charakteru
"T" Kurzor je objekt jazyka SQL, ktorý čísluje záznamy v množine získanej príkazom SELECT a dovoľuje aktualizovať alebo odstraňovať bežný záznam adresovaný kurzorom
"?" Kurzor je špeciálna konštrukcia, ktorá umožni pracovať s množinou dát získaných príkazom SELECT

//nie som si isty, prednaska kurzory.doc strana 1
//mnicky: cela otazka divna:

- v knihe (kap. 10.3) su spominane 2. a 6., moznost, tak o ostatnych neviem naisto...
- prva moznost by mohla byt aj dobre....aspon logicky
- ta tretia sa mi nepozdava kvoli tomu, ze kurzor by mal byt spojeny s riadkami a nie stlpcami, tak som ju dal otaznikom
- ta piata je velmi vseobecna, aj ked v tych poznamkach sa daco podobne vyskytuje
- ta posledna by skoro sedela na dalsiu definiciu z knihy, pokial by na zaciatku nebolo “specialna konstrukcia” ale “objekt jazyka SQL”....takze kto vie...

-

- *vase nazory?*

//pd: kurzor je spojený s riadkami, nie so stĺpcami (neviem o operácií nad stĺpcom ktorú by robil), prvá znie logicky správne a posledné.. no neviem tu fakt. otázka za bod, nemyslím že to má cenu trápiť sa s tým nejak moc.

28. Názvy typov kurzorov môžu byť odlišné v jednotlivých rozhraniach API a implantáciách. Podľa deklarácie kurzora rozpoznávame dva základné typy kurzorov (len 2 správne)
"F" posuvný
"**T" dynamický**
"F" rotačný
"**T" statický**
"F" rýchly
"F" spätný
"F" sekvenčný
//priklanam sa k dynamicky a staticky ale nikde som to nenasiel
//mnicky: mas pravdu - kniha, kap. 10.3.1.
Potom je este delenie podla spracovania: sekvencny, posuvny, kurzor pre zmeny.
//fillky: v materialoch(kurzory str. 5) je uvedene delenie na 4 typy: staticke, dynamicke a rychle, kurzory riadene sadou klucou
//mnicky: to sice hej, ale nie je to 'delenie podla deklaracie'. Delenie podla deklaracie je v knihe a su tam iba tie dva typy.

30. Architektúra ODBC je určená viacerými komponentami. Počet komponent je štandardne "**4**" (2,3,4,5)

39 Máme databázu, ktorá umožňuje spravovať údaje o študentoch, predmetoch a o výberoch predmetov študentami.
Databáza obsahuje nasledovných 5 tabuliek:
1/ CREATE TABLE Student
(Id_student Int, Meno Char, Priezvisko Char, Rocnik_studia Int, Odbor Char)
2/ CREATE TABLE Predmet
(Id_predmet Int , Kredity Int, Nazov Char, Odbor Char, Rocnik Int, Semester Char, Skratka Char, Ucitel Int, Hodin_prednasok Int, Hodin_cviceni Int, Povinny Char, Volitelny Char)
3/ CREATE TABLE Vyber
(Id_predmet Int, Id_student Char, Kredity Int, Rok Char, Ucit_cvici Int, Body_za_cvic Int, Datum_skusky Date, Ucit_skusa Char, Body_za_skusku Int, Znamka Char , Roc_studia Char)
4/ CREATE TABLE Ucitel
(Id_ucitela Int, Meno Char, Priezvisko Char, Tituly Int, Ustav Char)
5/ CREATE TABLE Uci
(Id_predmet Int, Id_ucitel Char, Akad_rok Char, Prednasa Bool, Cvici Bool)

-

Navrhnite a zdôvodnite potrebné zmeny, ktoré odstránia anomálie a chyby v definíciách tabuliek. Ku každej tabuľke
definujte aj primárny kľúč a všetky potrebné cudzie kľúče a následne napíšte opravené výsledné definície tabuliek.

Tabulka Student - Id_student PK, Odbor int FK

Tabulka Predmet - Id_predmet PK, Semester int, Ucitel FK, Povinny bool, Volitelny bool, Odbor vytvorit prepojovaciu tabulku Predmet - Odbor (jeden predmet moze byt vo viacerych odboroch)

Tabulka Vyber - Id_vyber PK, Id_predmet FK, Id_student int FK, rok int, Ucit_cvici odstranit, Ucit_skusa odstranit, Znamka int, Roc_studia int, Kredity odstrnit vytvorit prepojovaciu tabulku predmet_odbor

Tabulka Ucitel - Tituly char, Ustav int FK, Id_ucitela PK

Tabulka Uci - Id_uci PK, Id_predmet FK, Id_ucitel int FK, Akad_rok int

//este by trebalo zapisat to v sql, ale netusim ci je toto spravne a co myslel tou normalizovanou formou
//taktiez netusim ako ten navrh ma byt podrobny a az s akymi extremnostami mam pocitat

//mnicky: mathove priklady na InnoDB engine su v skuske vyssie

//mnicky: tu je moj pokus: http://ideone.com/TGvY4 - urcite nie je dokonaly a nechcelo sa mi vsade pisat NOT NULL, ani dalsie CHECK a UNIQUE constraints. Tiez by sa dalo este vyclenit dalsie samostatne tabulky ako napr. semester a rocnik, ale to uz je zbytocny masaker podla mna ;-). A este sa da uvazovat nad tabulkou OSOBA, ktora by referencovala foreign key TYP (ucitel, student, tech. pracovnik...).

//mnicky: inak vytvoril som tie tabulky v tej databaze co bola spominana tu: .fiitkar.sk/forum/viewtopic.https://wwwphp?p=106850#p106850 a pre toho kto ma db radsej graficky: http://ideone.com/cg5Cq

//mnicky: databaza **je uz naplnena** (dufam ze je tam vsetko ok ;-) takze mozete skusat.
Ak by niekto chcel skript na jej vytvorenie a naplnenie, tak tuna: http://www.uloz.to/9273717/nh566304db-sql //edit2: zrandomizovane pocty bodov.

//matus: pozeral som na query v http://ideone.com/TGvY4 a stale mi tam chyba informacia ze ktoreho studenta cvici aky ucitel. V tabulke Vyber z prikladu taka informacia je, ale v tych create table prikazoch vidim len vztah ci ucitel skusa alebo prednasa predmet, nevidim vztah ze akeho studenta cvici aky ucitel. Je mozne ze som si len nieco nevsimol...

-

Dalo by sa to riesit cez foreign_key z MNICKY_VYBER na MNICKY_UCI a potom by ani nebolo treba stlpec predmet_id v MNICKY_VYBER... Alebo spravit novu spojovaciu tabulku medzi MNICKY_UCI a MNICKY_STUDENT, ale to asi nie je najlepsie riesenie ak ma student viac zaznamov v MNICKY_VYBER s rovnakym predmetom.

//Matho: odstranit anomalie znamena urobit normalizaciu. Kniha strana 100 - “Tato anomalia odstranena pomocou procesu normalizacie.”

40. K zadefinovaným tabuľkám uveďte SQL príkaz, ktorým vypíšete zoznam učiteľov (priezvisko, meno) a ku každému z nich v jeho predmete počet študentov, ktorých v tom predmete cvičí - priezviská a mená usporiadané abecedne.

//mnicky: asi dajako takto (dal som to pre kazdy rok aj rocnik zvlast):

```
SELECT meno, priezvisko, nazov, rok, rocnik_studia, COUNT(student_id)
FROM MNICKY_UCITEL
        INNER JOIN MNICKY_UCI ON id_ucitel = ucitel_id
        INNER JOIN MNICKY_PREDMET ON predmet_id = id_predmet
        INNER JOIN MNICKY_VYBER ON MNICKY_VYBER.predmet_id = id_predmet
WHERE cvici = 1
GROUP BY nazov, priezvisko, rok, rocnik_studia
ORDER BY priezvisko, meno;
```

//mnicky: tak ako to chcu oni je to takto:

```
SELECT meno, priezvisko, nazov, COUNT(student_id)
FROM MNICKY_UCITEL
        INNER JOIN MNICKY_UCI ON id_ucitel = ucitel_id
        INNER JOIN MNICKY_PREDMET ON predmet_id = id_predmet
        INNER JOIN MNICKY_VYBER ON MNICKY_VYBER.predmet_id = id_predmet
WHERE cvici = 1
GROUP BY id_ucitel
ORDER BY priezvisko, meno;
```

//pd: SELECToval by som len to co vyzaduju, teda priezvisko, meno a pocet
//Matho: ma to byt zoradene podla priezvisk aj mien, tj ORDER BY priezvisko ASC, meno ASC, takto ti to sortne iba podla priezviska - efekt uvidite ked si premenujete navrata napr na bozenu bielikovu. Myslim ze INNER nie je treba pisat, staci JOIN a INNER sa berie ako defaultny

//mnicky: @pd aj ja by som na skuske selectoval iba co vyzaduju, tu som to vsak dal schvalne aby bolo vidno ako je to v databaze realne. Nie je problem odstranit tie dve slova potom :-). @matho oki, doplnil som tam aj ‘meno’ a viem, ze INNER je implicitne, pouzivam ho vsak rad explicitne, je to take zrejmejsie ;-).

-

//preco taky zlozity GROUP BY? nestaci len podla Uci.ucitel_id?

//mnicky: jasne ze stacilo, ja som to vsak chcel urobit pre kazdy rok aj rocnik zvlast (vid. co som pisal vyssie) a vtedy to nestaci. Ale dal som tam variantu presne ako chcu oni, aby ste boli spokojni ;-)

//pd: @mnicky: riadok “INNER JOIN MNICKY_PREDMET ON predmet_id = id_predmet” je zbytočný, nie? :)

41. K zadefinovaným tabuľkám uveďte SQL príkaz, ktorým vypíšete zoznam odborov, ročníkov, priezvisk a mien všetkých študentov usporiadaný po ročníkoch, odboroch a abecedne podľa priezviska a mena.

//mnicky: easy-peasy :). Akurat ze ja som v tej tabulke zrusil atribut ‘rocnik’ pri studentoch, cize to je bez neho:

```
SELECT nazov AS odbor, meno, priezvisko
FROM MNICKY_STUDENT
        INNER JOIN MNICKY_ODBOR ON id_odbor = odbor_id
ORDER BY odbor, priezvisko, meno;
```

//mnicky: ak by sme to ale silou mocou chceli aj s rocnikmi, mozeme pouzit atribut ‘rocnik’ z tabulky predmetov (btw, ja som tie tabulky naplnal tak, ze kazdy student je vo viacerych rokoch vo viacerych rocnikoch naraz, pretoze sa mi s tym nechcelo prilis hrat ;-) a potom je to cca takto:

```
SELECT DISTINCT rocnik_studia AS rocnik, nazov AS odbor, meno, priezvisko
FROM MNICKY_STUDENT
        INNER JOIN MNICKY_ODBOR ON id_odbor = odbor_id
        INNER JOIN MNICKY_VYBER on student_id = id_student
ORDER BY rocnik, odbor, priezvisko, meno;
```

//Matho: Takze ja len zopakujem - DISTINCT je tam napisany len preto ze nasa databaza umoznila studentov viacnasobnu registraciu, takze na teste tam DISTINCT nepisat.
.//mnicky: hej, no, nechcelo sa mi hrat s tolkym poctom studentov, tak som tam kazdeho regol viackrat ;-)

42. K zadefinovaným tabuľkám uveďte SQL príkaz, ktorým vypíšete po ročníkoch zoznamy názvov predmetov končiacich skúškou a ku každému predmetu maximálny, minimálny a priemerný počet získaných bodov zvlášť za cvičenia a zvlášť za skúšku - usporiadaný najprv podľa ročníkov a v nich abecedne podľa názvu predmetu.

-

//mnicky: takto nejako (este som potom trochu zrandomizoval hodnoty bodov v tej tabulke, aby to nevyzeralo rovnako - mal som ich odzaciatku davat nahodne, ale co uz ;-)

```
SELECT rocnik_studia AS rocnik, skratka AS predmet, MAX(body_za_cvic) AS cvicenia_max,
        MIN(body_za_cvic) AS cvicenia_min, AVG(body_za_cvic) AS cvicenia_avg,
        MAX(body_za_skusku) AS skuska_max, MIN(body_za_skusku) AS skuska_min,
        AVG(body_za_skusku) as skuska_avg
FROM MNICKY_PREDMET
        INNER JOIN MNICKY_VYBER ON predmet_id = id_predmet
GROUP BY rocnik, predmet
ORDER BY rocnik, predmet;
```

// preco nestaci GROUP BY id_predmet ?

43. K zadefinovaným tabuľkám uveďte SQL príkaz, ktorým vypíšete usporiadane pre každý akademický rok, ročník štúdia v každom odbore počet získateľných kreditov z povinných predmetov.

//mnicky: iba takto so subselectom mi to slo (pouzil som atribut PREDMET_ODBOR.rocnik a nie ten MNICKY_VYBER.rocnik_studia, lebo ten hovori iba o zapisanych predmetoch):

```
SELECT rok, rocnik, odbor, SUM(kredity) AS kreditov
FROM (
        SELECT DISTINCT rok, rocnik, MNICKY_ODBOR.nazov AS odbor, skratka, kredity
        FROM MNICKY_VYBER
                FULL JOIN MNICKY_PREDMET
                INNER JOIN MNICKY_PREDMET_ODBOR ON
        MNICKY_PREDMET_ODBOR.predmet_id = id_predmet
                INNER JOIN MNICKY_ODBOR ON odbor_id = id_odbor
                INNER JOIN MNICKY_TYP_PREDMETU ON typ_predmetu_id =
        id_typ_predmetu AND MNICKY_TYP_PREDMETU.nazov = 'povinny'
        ) AS temp
GROUP BY rok, rocnik, odbor
ORDER BY rok, rocnik, odbor;
```

44. K zadefinovaným tabuľkám uveďte SQL príkaz, ktorým vypíšete zoznam názvov a skratiek predmetov a ku každému predmetu aj jeho prednášajúceho ako aj zoznam učiteľov cvičiacich daný predmet - názvy predmetov, ako aj priezviská a mená učiteľov budú usporiadané abecedne.

//mnicky: takto nejako:

```
SELECT nazov AS predmet, skratka, priezvisko, meno, prednasa, cvici
```

-

```
FROM MNICKY_PREDMET
        INNER JOIN MNICKY_UCI on predmet_id = id_predme
        INNER JOIN MNICKY_UCITEL on ucitel_id = id_ucitel
ORDER BY predmet, priezvisko, meno;
//toto
```

RT 2007/2008 (pravdepodobne)

http://www.trenchtown.sk/skuska1.pdf
//connors: je mozne ze su tam nejake chyby, pastol som to dost na rychlo, kludne opravte :)

1. Všetky vstupné údaje sa stávajú perzistentnými
"F" YES
"T" NO

7. Je správne zakreslený typový entitno-relačný diagram? //tu asi chýba obrázok
"" YES
"" NO

//pd: ale tak tipnúť si snáď môžeme, nie?! :)
//mnicky: 5eur na YES :-D
//pd: ja by som povedal NIE, keď sa na to pozrem, chýbajú mi tam nejaké väzby :D

9. Inštancia je
"T" výskyt objektu entitného typu
"T" jeden záznam v tabuľke
"F" entita
"F" inicializácia primárneho kľúča
"F" tabuľka
"T" existencia objektu, ktorý je určený definovanou entitou

//mnicky: zeby prve alebo posledne?
//kasi: ten jeden zaznam (chapem ako riadok) v tabulke?(fillky: podla mna ano)
//pd:asi áno, v tom prípade asi prvé, druhé a posledné; označil som tri možnosti, ak niečo, píšte

10. Nech učiteľ a študent majú niektoré atribúty spoločné a je nimi jednoznačne určená entita OSOBA. Doplňte správne vzťah v uvedenej schéme.
OSOBA(OSOBNÉ ČÍSLO, MENO, ROD_CISLO, ADRESA)
UČITEĽ(AKADEMICKÁ_HODNOSŤ) "**ISA**" OSOBA
ŠTUDENT(ODBOR, SPECIALIZACIA, ROČNÍK) "**ISA**" OSOBA

//kasi: mam chapat ako kardinalitu?
//pd: skorej ma to ťahá k “IS-A” (alebo ISA či ako sa to píše)
//George: som za ISA
//kasi: po prebehnuti prednasky ERmodel je to urcite ISA je tam presne ten isty priklad

-

//pd: nahadzujem odpoveď :)

11. Napíšte malými písmenami názvy piatich rôznych typických operácií s reláciami
"výber"
"projekcia"
"karteziánsky súčin"
"prienik"
"rozdiel"
"spojenie"
"zjednotenie"
"delenie"
//toto su vsetky co boli v materialoch...
//pd: pozrel by som si radšej aj anglické názvy (union, division etc.)

12. Atribúty, ktoré sú súčasťou nejakého primárneho kľúča sa nazývajú kľúčové, ostatné atribúty nazývame **"nekľúčové"**

13. Relačná databáza je množina "**relacii**"

//podla zhrnutie_na_skuskII.doc

14. Relačný databázový systém je založený na
"F" teórii grúp s využitím štatistických metód
"F" teórii množín v spojení s predikátovou logikou
"F" princípoch matematickej analýzy
"F" matematickej algebre a teórii pravdepodobnosti
"T" teórii množín s využitím relačnej algebry a kalkulu

//podla mna posledne, to znie celkom logicky
//dal som ako T, podla zhrnutie na skuskuII.doc

15. Primárny kľúč v relačnej databáze je jednoznačný identifikátor, ktorý je reprezentovaný "**stĺpcom**" alebo skupinou "**stĺpcov**"
//George: nie atributom?
//kasi: riesilo sa uz niekde vyssie tusim..
//pd: riešilo, takáto odpoveď je v materiáloch
//tempus: tam ale bola otazka na primarny kluc tabulky databazy co su stlpce, tu by som dal // atributy

16. Doplňte formuláciu uvedených normálnych foriem.
- Relácia R je v 1. normálnej forme vtedy
a len vtedy, keď neobsahuje **"opakujúce sa"** podskupiny dát ,teda každý atribút je **"atomický"**.
- Relácia R je v 2. normálnej forme vtedy a len vtedy, keď je v 1NF a každý nekľúčový atribút je **"(úplne) funkčne"** závislý na primárnom kľúči.

-

- Relácia R je v 3. normálnej forme, ak je v 2NF a každý nekľúčový atribút nie je **"tranzitívne"** funkčne závislý na primárnom kľúči.
- Relácia R s relačnou schémou R(A,B,C) je v 4. normálnej forme, ak v každej **"multihodnotovej"** závislosti A↠B je atribút A primárnym kľúčom.

17. Cudzí kľúč je atribút alebo skupina atribútov, ktorého hodnota v relácii je buď prázdna alebo musí obsahovať hodnotu **"primárneho kľúča"** inej relácie

19. Transakcia reprezentuje postupnosť operácií Read a Write databázových objektov.
"T" YES
"F" NO

//pd: na základe kapitoly 10.2.4; odstránené duplicitné otázky

22. V relačnom distribuovanom databázovom systéme predstavuje lokálna databáza množinu relácií. Platí to aj pre globálnu databázu?
"" YES
"" NO

//mnicky: osobne si myslim ze ano, ale nemozem nicim podlozit ;-)
//pd: no zmysel to dáva

23. O celé riadenie databázy sa stará "**distribuovany**" systém riadenia bázy dát, ktorý spolupracuje s "**lokalnymi**" systémami riadenia databáz.
//tempus: to SRBD je tam odpoved? vyzera to celkom logicky
//mn : stacilo sa na to cele spytat googla

24. V distribuovanom databázovom systéme sa vykonáva horizontálna fragmentácia pomocou operácie **"výber"**
//mnicky: no neviem, v knihe kap. 16.2.2.2 sa pise: "**Horizontalna fragmentacia** je vykonana pomocou operacie relacnej algebry **VYBER** (selection)." A dalej sa pise, ze pomocou operacie 'zjednotenia' (union) sa tato horizontalne fragmentovana relacia akurat vie spojit dokopy.... Podobne pri **vertikalnej fragmentacii** (kap. 16.6.2.4) sa na fragmentaciu samotnu pouziva operacia '**projekcia**' (projection) a na opatovne rekonstruovanie povodnej, vertikalne nefragmentovanej relacie sa pouziva operacia spojenie (join).
A ked sa nad tym tak zamyslis a das si to do suvislosti s SQL JOIN a UNION, tak to aj sedi...

27. Zmenu uložených údajov v tabuľke na nové hodnoty umožní v jazyku SQL príkaz **"UPDATE"**

28. Vytvorenie relačnej databázy je možné pomocou dotazovacieho jazyka **"SQL"**.
Napíšte veľkými písmenami jeho skrátený názov.

//pd: existuje aj dotazovací jazyk QBE (query-by-example), ale tento je asi pravdepodobnejší

-

29. Do ktorej skupiny príkazov jazyka SQL patrí príkaz SELECT?
"F" DDL
"F" DIS
"F" DSD
"F" DAS
"F" DSL
"T" DML
"F" DDS
"F" DSS

30. Priraďte uvedené príkazy jazyka SQL do správnej skupiny
"DML" INSERT, UPDATE,DELETE, SELECT, ~~DROP~~
"DDL" CREATE, ALTER, DROP
"DIS" COMMIT, ROLLBACK
"DAS" GRANT, REVOKE

//mnicky: to ze je DROP vo viacerych skupinach je vpohode? Nasiel som ho na wikipedii iba v DDL, tak ho skrtam.
pd: v materiáloch tiež len DDL

OT 2008/2009

6. Môže viacero atribútov tvoriť primárny kľúč?
"**T**" **ÁNO**
"F" NIE

//mnicky: samozrejme ze ano (zdroj: moja hlava ;-)
//pd: konečne vierohodný zdroj :D; pridal by som aj svoju hlavu ale kleslo by to na dôveryhodnosti
//mnicky: :-DDDD !!! Lol... tento tvoj comment ma prebral k zivotu....som skoro zaspaval tu nad dbs....
//pd: no ja to vidím, že zaspím aj na tej skúške už pri čítaní otázok, taká je to nuda; rozprávky brata Galbavého..

7. Doména je pomenovaná množina skalárnych hodnôt "**rovnakeho**" dátového typu

9. Môže mať tabuľka v relačnom databázovom systéme viac ako jeden primárny kľúč?
"F" áno
"**T**" **nie**
//mnicky: z mojej hlavy :-)

-

10.Množina hodnôt, ktoré môže nadobúdať atribút sa nazýva "**domena**".
//mnicky: kniha, kap. 4.4.5.3

11. Ktoré podmienky spĺňa primárny kľúč?
"T" jednoznačnosti
"T" unikátnosti
"F" opakovateľnosti
"" referenčnej integrity
"**T**" **neredukovateľnosti**
"" prepojenia relácií
"T" minimálnosti

//aj minimalnosti, suhlas?
//pd: súhlas, je to v materiáloch
//asi by malo byt oznacene aj jednoznacnosti
// aj podla mna... ved primarny kluc by mal jednoznacne urcovat zaznam

12. Integrita databázy súvisí so zabezpečením **"konzistencie"** údajov pri akýchkoľvek zmenách v databáze.

14. Ideálne spracovanie množiny transakcii { Ti, T 2, ,...Tn } je **"sériové"**

16. V kolkých fázach sa vykonáva spracovanie transakcie metódou kontrolného bodu?
"F" troch
"F" jednej
"**T**" **dvoch**
"F" viacerých

//mnicky: kniha, kap. 14.7.

17. Na zabezpečenie serializovateľnosti rozvrhov sa používa metóda časovej pečiatky. Časová pečiatka je "**unikatny identifikator vytvarany systemom riadenia bazy udajov**". Systém riadenia bázy dát vytvorí časovú pečiatku tak, aby sme mohli jednoznačne určiť transakciu.

//mnicky: z knihy (kap. 15.4.2):

- *casova peciatka* je unikatny identifikator vytvarany systemom riadenia bazy dat (udajov)
- *casova peciatka transakcie* je numericka hodnota priradena transakcii, umoznujuca usporiadat beziace transakcie
- *casova peciatka objektu* je numericka hodnota priradena objetku databazy, vyjadrujuca cislo transakcie, ktora s nou naposledy manipulovala

-

19. Pri poškodení média môžeme obnoviť databázu do stavu, v akom bola v okamihu havárie pomocou...
"T" metódy kontrolného bodu
"T" vytvorených archívnych kópií
"T" logických žurnálov
"F" metódy 2PhC
"F" metódy priameho zápisu

//mnicky: kniha, kap. 14.8

20. Napíšte veľkými písmenami v jazyku SQL príkaz na vytvorenie tabulky "**CREATE TABLE**"

21. Vytvorenie prázdnej relačnej databázy je možné v jazyku SQL pomocou príkazu "**CREATE DATABASE**"

22. Vyberte príkazy , ktoré patria do skupiny DDL jazyka SQL.
"F" DEFINE
"F" REVOKE
"**T**" **DROP**
"**T**" **CREATE**
"F" SELECT
"F" ASSUME
"F" INSERT
"F" UPDATE
"**T**" **ALTER**
"F" PROJECT
"F" DATA

23. Vyberte príkaz jazyka SQL, ktorý odstráni všetky riadky z jednej tabuľky
Počet správnych odpovéďi je 2.
"**T**" **DELETE from table**
"F" DELETE from table * rows
"F" DELETE in table * rows
"F" DELETE for columns from table * rows
"F" DELETE * rows from table
"T" DELETE * from table
"F" DELETE rows from table
"F" DELETE from tables

25. Kvalitu databázového systému vo všeobecnosti určuje
"T" zdieľanie dát
"F" dotazovací jazyk
"F" kvalita používateľských kategórií

-

"T" bezpečný prístup k údajom
"F" počet serverov
"T" oddelenie definície dát a príkazov na manipuláciu s nimi
"F" počet používateľov
"F" počet transakcií
"F" velkosť úložného priestoru
"F" paralelizmus transakcií
"T" ochrana proti nekonzistencii dát
"T" procedurálne a neprocedurálne rozhranie
"T" nezávislosť dát
"F" organizovanie údajov do multidimenzionálnych štruktúr
"F" výber typu servera
"T" integrita údajov
"T" minimalizácia redundancie dát
"F" databázový administrátor

//podla zhrnutie_na_skuskuII.doc

27. Systém riadenia bázy údajov je množina "**programov**", ktorá zabezpečuje definíciu a manipuláciu s dátovými Štruktúrami.

28. Určite základné triedy osôb, ktoré manipulujú s databázou
"F" správca, programátori, databázový špecialista
"T" **koncoví používatelia, aplikační programátori, databázový administrátor**
"F" sekretárky, matematici, programátori

29. Hlavnými reprezentantmi súborových systémov je "**hierarchicky**" databázový systém a "**sietovy**" databázový systém.

//podla zhrnutie_na_skuskuII.doc, ale nie som isty
//pd: je to dobre

32. Všetky výstupné údaje získané z databázy sú perzistentné
"F" ÁNO
"T" **NIE**

34. Každá úroveň architektúry databázového systému je špecificky opísaná. Tomuto opisu hovoríme.... (Vyberte len jednu z následujucich možných odpovedi.)

"F" dátový model
"F" funkcionálny pohľad
"T" schéma
"F" definícia
"F" logický pohľad

-

"F" koncept
"F" mapa
"F" záznam

35. V "**klient/server**" architektúre sa aplikácia spracováva v počítačovej sieti a funkčnosť aplikácie je rozdelená do dvoch vrstiev.
//George: klient-server?
//pd: vyzerá že áno, hneď prvý dokument str. 21, kapitola 3.1

36. Najdôležitejšie operácie s kurzorom sú :
"F" zmena
"F" priradenie k nejakému SQL prikazu
"T" **otvorenie**
"F" pridanie
"T" **uzatvorenie**
"F" transformácia
"F" výber
"F" prenesenie
"T" **deklarácia**
"T" **dealokovanie**
"T" **prechádzanie**
"F" zrušenie

//mnicky: je to v materialoch kurzory.doc

38. Microsoft ActiveX Data Objects je označenie pre
"F" hlásenie o chybe v aplikácii
"F" spracovanie aktívnych dátových objektov v jazyku XML
"F" súbor aktívnych dát
"F" knižnicu objektov
"T" databázové rozhranie

//podla mna takto
//pd: yep, môže byť

39. Embedded SQL je upravený SQL jazyk, ktorý sa vkladá priamo do kódu "**hostiteľského**" jazyka.

//podla mna “C” jazyka
//pd: podľa wiki je najpopulárnejšie C, ale sú tam aj COBOL, Pascal, FORTRAN...
//programovacieho? :D
//pd: ale to vôbec nie je blbý nápad :D
//dle mne hostitelskeho jazkya :) podla zhrnutie_na_skuskuII.doc
//pd: ja o tom embedded neviem nič v jeho materiáloch nájsť, ale vôbec. no keď je v tom docu

-

napísaný hostiteľský, asi to bude ono, dávam ho aj do odpovede, ak by niečo píšte.

DALSIE OTAZKY CO ESTE VYSSIE NIE SU
//pd: bude aj kapitola s otazkami, ktore uz vyssie su? :D

8. Atributom budeme rozumiet funkciu priradujucu entitam alebo vztahom hodnotu, urcujucu niektoru **x** vlastnost entity alebo vztahu.
Vyberte za x spravnu moznost:

"**T**" **podstatnu**
"F" typicku
"F" nepodstatnu
"F" presne neurcenu
"F" docasne definovanu

//mnicky: kniha, kap. 4.4, str. 49

11. N-tici v relacii zodpoveda v relacnej databaze "**riadok**" tabulky

//mnicky: knih,a kap.5.2., str. 91

14. Napíšte základné dva typy domén: **"jednoducha", "kompozitna"**

//podla zhrnutie_na_skuskuII.doc

9. Entita je
"F" inštancia
"T" reálny objekt, ktorý je jednoznačne odlišný od ostatných objektov
"F" objekt reálneho sveta, ktorého existencia závisí od existencie ostatných objektov
//Fillky: overene podla materialov (ERModel str. 5)

17 . Čím sa odlišuje databázová relácia od matematickej? Vyberte 2 správne
"T" obsahuje schému relácie
"F" využíva štatistické metódy
"T" prvky domen, z ktorych sa beru hodnoty atributov su atomicke hodnoty
"F" pracuje s mnozinovymi operaciami
"F" reprezentuje relaciu tabulkou

-

//podla zhrnutie_na_skuskuII.doc

18. Vysledky potvrdenej transakcie "**sú**" perzistentne ulozene v databaze.

//co tym asi myslia? ja by som tam dal "nie su", ale to len vystrel naslepo
//jm:Ak transakcia končí úspešne, hovoríme, že transakcia je potvrdená a toto potvrdenie je zabespečené príkazom COMMIT. V opačnom prípade transakcia môže skončiť bez toho, aby transakcia splnila svoje úlohy, v tom prípade hovoríme, že končí príkazom ABORT (abnormaly termination). Príkaz COMMIT signalizuje SRBD, že všetky zmeny vykonané transakciou sú zaznamenané v databáze a môžu sa stať viditeľnými, alebo prístupnými iným transakciám, ktoré chcú používať dáta doposiaľ používané touto transakciou. Bod, v ktorom začína potvrdzovanie ukončenia transakcie, je bodom, z ktorého niet návratu. Výsledky potvrdenej transakcie sú týmto **perzistentne uložené v databáze** a nemôžu byť vrátené späť. -

19. Transakcia v databazovom systeme zabezpeci spracovanie jednoho ci viac " " prikazov.

//zeby "transakcnych"?
//kudlohlavec: mozno by tam mohlo byt aj zapuzdrenych
//jm: tiez by som povedal zapuzdrenych

20. Uspesne ukoncenie transakcie byva oznacene prikazom
"F" ABORT WORK
"F" END WORK
"F" ROLLBACK WORK
"**T**" **COMMIT WORK**
"F" VYSTUP WORK
//mnicky: kniha, pr. 14.1, str. 434

22. Vyberte vlastnosti, ktore by mal splnat idealny distribuovany databazovy system
"F" zavislost na technickom vybaveni
"T" lokalna autonomnost
"T" nezavislost na operacnom systeme
"F" zavislost na fragmentacii dat
"F" zavislost na lokalnych databazovych systemoch
"T" suvisla prevadzka
"T" nezavislost na centralnom uzle
"T" spracovanie distribuovanych dotazov
"F" zavislost na replikacii dat
"F" paralelne riadenie transakcii
"T" nezavislost na sieti
"T" nezavislost na umiestneni dat

//podla zhrnutie_na_skuskuII.doc, to 3. odspodu ma byt distribuovane riadenie transakcii, vtedy by to bolo True

-

23. Distribuovany databazovy system mnozina navzajom prepojenych “**uzlov**” pocitacovej siete, prisom kazdy z “**uzlov**” je samostatny databazovy system, ale tieto “**uzly**” navzajom spolupracuju tak, ze z kazdeho uzla je mozne spristupnit udaje ulozene na inom “**uzle**” akoby boli umiestnene na vlastnom “**uzle**”.
//sa mi pacia tie otazky, ze ked si vsimnes tak uz v tej vete je v podstate skryta odpoved :DD
//mnicky: kniha, zac. kap. 16

24. V distribuovanom databazovom systeme sa vykonava fragmentacia podobne ako proces normalizacie relacii, ale okrem toho je nutne zabezpecit tri podmienky

“F“ integritu
“F“ nezavislost uzlov siete
“F“ zavislost na replikacii dat
“F“ konzistentnost
“**T**“ **rekonstruovatelnost**
“F“ zabezpecenie paralelizmu
“**T**“ **nespojitelnost**
“F“ jednoznacnost
“F“ zalohovanie
“**T**“ **kompletnost**
//mnicky: kniha, kap. 16.6.2.1

26. Interna uroven architektury databazoveho systemu sa vseobecne tyka

“?“ pristupovych metod k ulozenym udajom
“ “ programovacich jazykov
“F“ suborovych systemov
“?“ transformacie konceptualnej urovne
“**T**“ **fyzickej podoby modelu**
//fillky: pristupove metody tam podla mna patria tiez??(vid materialy 2.2.3 -z “uvod.doc”)
ak uvazujeme len o tom coho sa tyka tak to bude asi len posledna pripadne predposl. Ostatne moznosti len opisuju alebo su vyuzivane...
Shit zavadzajuca otazka pretoze interna schema sa tyka pristupovych metod... Takze co navrhujete??
//mnicky: kniha kap. 2.2.3. Len neviem ci dat True aj druhu moznost, lebo interna uroven na svojpopis vyuziva prog. jazyky.....

//liho: pôvodná diskusia k RT 08/09 otázka číslo 16 :

-

//tiez by som bol za ROLLBACK
//mnicky: podla knihy, kap. 14.3.2 aj ROLLBACK - ak to chapem spravne (davam to aj ako spravnu odpoved. ak nesuhlasite, vyjadrite sa ;-)

//pd: ja som proti rollbacku, kapitola 10.2.2 v materiáloch (09_8transakcie.pdf) neobsahuje ani slovko o rollbacku - okrem examplu; môžeš dať nejakú citáciu z tej 14.3.2?

//mnicky: tu je: “Ked je transakcia zrusena, jej vykonavanie sa zastavi a vsetky nou vykonane cinnosti su neplatne a budu vratene (UNDO) tak, aby databaza bola v stave ako na zaciatku vykonavania transakcie. Tato operacia je znama ako operacia ROLLBACK” (kniha 14.3.2.)
- ked to teraz znova citam, neviem ci to chapat tak, ze ROLLBACK aj ukoncuje spracovanie transakcie, alebo ju iba vrati db do povodneho stavu. Je to dost nejednoznacne. Neda sa povedat, na ktoru cast prvej vety sa vztahuje ta druha. Menim to teda na “?”....

//kasi: podla prednasky 8transakcie citujem “Označme Ni ako ukončovací príkaz transakcie Ti, kde: Ni leží {Abort, Commit}”
//kasi: rollback a abort je zjavne to iste http://www.sql.org/sql-database/postgresql/manual/sql-abort.html
//mnicky: tak teraz uz ozaj neviem ze co :-P
//mnicky: inak, ked si vsimnete ot. c. 9 (vid. vyssie), tak tam sa pise, ze ked sa pouzije roll back, tak transakcia je uz zrusena...

//liho: označujem ROLLBACK za F
-- tu je v mytable 50 riadkov
BEGIN WORK;
SELECT * FROM mytable WHERE 1;
ROLLBACK WORK; -- tu sa to vrati do stavu hned po begine
DELETE FROM mytable WHERE 1; -- zmaze to 50 zaznamov
COMMIT WORK; -- AZ TU SA SKONCI TRANSAKCIA
-- tu je mytable prazdna
//Kveri: to nie je pravda, ABORT existuje len z historickych dovodov a je identicky s ROLLBACK. To WORK je zbytocne pouzivat:
sql-92 standard hovori: “Execution of a COMMIT Statement or ROLLBACK Statement completes the current transaction”
postgres: skus spravit commit po rollbacku a dostanes warning, ze tranzakcia nebezi
mysql: spravil som ten delete co si napisal a v druhom okne som uz samozrejme nevidel ziadne riadky (tj. nebezalo to v tranzakcii). mysql evidentne nedava warningy ked das commit/rollback mimo tranzakcie
//oznacujem ROLLBACK za T, kedze je identicky s ABORT
//liho: jo, sorry, uz som si to overil, fakt to funguje tak, ze ROLLBACK ukonci transakciu a dalsie querycka tykajuce sa transakcie su proste odignorovane, napr:
BEGIN WORK;
INSERT INTO `test` (`t`, `id`) VALUES ('a', '1');
ROLLBACK WORK; -- toto rollbackne to acko a ukonci transakciu

-

```
INSERT INTO `test` (`t`, `id`) VALUES ('b', '2'); -- vlozi b
ROLLBACK WORK; -- tu sa nic nestane, proste je to odignorovane
COMMIT WORK; -- tiez odignorovane
-- tu ostalo v tabulke test b
```

… btw to WORK je fakt fail, ale tak v tych doc-och z dokumentoveho to tam vsade pisal, tak budem pisat aj ja...

-