Feladatok

Egyed:	A valós világnak az az eleme (tárgy, jelenség, elképzelés, személy, fogalom stb.), amely a modellezés tárgyát képezi (X nevű Y éves, Z szakos hallgató)
Egyedtípus:	Az azonos tulajdonságtípusokkal rendelkező egyedek absztrakciója (hallgató)
Tulajdonság:	Az egyednek a modellezés szempontjából lényeges jellemzője. (Kovács Péter név, második évfolyam, PTI szak)
Tulajdonságtípus:	Az azonos szerepű tulajdonságok absztrakciója. Más néven attribútum. (hallgató neve, évfolyama, szakja)
Kapcsolat:	A két vagy több egyedtípus egyedei között fennálló viszony. (XY felvette a Z tárgyat)
Kapcsolattípus:	Két vagy több egyedtípus közötti jól meghatározott viszony. (hallgató és tárgy közötti felvételi viszony)
Koncepcionális adatmodell (séma):	Véges számú tulajdonságtípussal megadott véges számú egyedtípus és a közöttük fennálló véges számú kapcsolattípus összessége.
Adatbázis:	Az adatmodell, valamint az egyed-előfordulások, tulajdonság-előfordulások és kapcsolat-előfordulások együttese.
NULL:	nem alkalmazható / nem értelmezett / létezik, de hiányzik / nem tudjuk, hogy létezik-e
Reláció:	Értékek egy táblázata, amely sorok egy halmazából áll.
Sor:	a modellezett kisvilág egy egyed-előfordulásáról vagy egy kapcsolat-előfordulásáról tartalmaznak adatokat
Oszlop:	fejléccel (címmel) rendelkezik, amely az illető oszlopban lévő adatok jelentéséről ad információt
Relációséma:	Relációséma alatt az $R(A_1, A_2, \ldots, A_n)$ jelölést értjük, ahol R a relációséma neve, $A_1, A_2, \ldots, A_n$ pedig attribútumok. Minden A_i attribútum egy szerepkör neve, amelyet valamely Di tartomány játszik. D_i-t az A_i attribútum tartományának nevezzük, és $dom(A_i)$-vel is jelöljük, n a reláció foka.
Reláció:	Az $R(A_1, A_2, \ldots, A_n)$ relációséma egy r relációja - amit szokás r(R)-rel is jelölni - elem n-eseknek (rekordoknak) egy halmaza: $r = \{t_1, t_2, \ldots, t_m\}$. Minden t_i elem n-es ($1 \le i \le m$) n darab értéknek egy rendezett listája: $t_i = \langle v_{i1}, v_{i2}, \ldots, v_{in} \rangle$, ahol minden v_{ij} érték ($1 \le j \le n$) vagy $dom(A_j)$-nek az eleme, vagy egy speciális NULL érték.
Szuperkulcs:	Az R relációsémának létezik egy olyan attribútumhalmaza, amely olyan tulajdonságú, hogy tekintve R bármelyik r relációját, az adott relációban nincs két olyan rekord, amelynek az értékei azonosak lennének ezen attribútumokra vonatkozóan. Az attribútumoknak egy ilyen részhalmazát SK-val jelölve, bármely két különböző t_1 és t_2 rekordot kiválasztva R egy r relációjából: $t1[SK] \neq t2[SK]$. Minden ilyen SK attribútumhalmaz az R relációséma szuperkulcsa
Kulcs:	Egy R relációséma K kulcsa R-nek egy olyan szuperkulcsa, amelyből bármelyik A attribútumot elhagyva, az így kapott $K' = K \setminus A$ attribútumhalmaz már nem szuperkulcsa R-nek. Egy kulcs kielégíti a következő két feltételt: Bármilyen relációt tekintve, a reláció két különböző rekordjának nem lehetnek azonosak a kulcsban szereplő attribútumokhoz tartozó értékei. ***Minimális szuperkulcs***, azaz egy olyan szuperkulcs, amelyből nem tudunk úgy eltávolítani egyetlen attribútumot sem, hogy az egyediségre vonatkozó feltétel továbbra is fennálljon.

Egyedintegritási megszorítás:	Kimondja, hogy egyetlen elsődlegeskulcs-érték sem lehet NULL érték. Ha az elsődleges kulcs összetett, akkor annak egyik komponense sem lehet NULL érték.
Hivatkozási integritási megszorítás:	Egy R_1 relációséma FK-val jelölt attribútumhalmaza külső (idegen) kulcsa R_1-nek, amely hivatkozik az R_2 relációsémára, ha eleget tesz a következő feltételeknek: Az FK-beli attribútumoknak és az R_2 PK-val jelölt elsődleges kulcsattribútumainak páronként azonos a tartománya; ekkor azt mondjuk, hogy az FK attribútumok hivatkoznak az R_2 relációsémára. Bármely $r_1(R_1)$ aktuális állapotának egy t_1 rekordjában egy FK-beli érték vagy megjelenik egy $r_2(R_2)$ aktuális állapotának valamely t2 rekordjában PK értékeként, vagy az értéke NULL. Az előbbi esetben $t_1[FK] = t_2[PK]$, ekkor azt mondjuk, hogy a t_1 rekord hivatkozik a t_2 rekordra.
Relációs adatbázisséma:	Egy S relációs adatbázisséma az $S = \{R_1, R_2, \ldots, R_m\}$ relációséma-halmaz, valamint integritási megszorítások (IC jelölésű) halmazának az együttese.
Relációs adatbázis:	S egy DB relációs adatbázisa olyan $DB = \{r_1, r_2, \ldots, r_m\}$ relációk halmaza, ahol minden r_i az R_i séma egy relációja, és minden r_i reláció kielégíti az IC-ben megadott integritási megszorításokat
Szelekció:	σ ⟨szelekciós feltétel⟩ (R) ahol R azt a relációt jelöli, amelyből a ⟨szelekciós feltétel⟩-nek eleget tevő rekordokat válogatjuk ki.
Projekció:	π ⟨attribútumlista⟩ (R) ahol az ⟨attribútumlista⟩ az R reláció lekérdezni kívánt attribútumainak listája.
Uniókompatibilitás:	két relációnak ugyanannyi attribútuma van, és attribútumaik tartományai páronként megegyeznek egymással
Equijoin:	Az az általános összekapcsolási művelet, amelynek összekapcsolási feltételében csak az egyenlőségjel (=) szerepel összehasonlító műveleti jelként
Funkcionális függés:	Az R két attribútumhalmaza, X és Y között, X → Y -nal jelölt funkcionális függés előír egy megszorítást azokra a lehetséges rekordokra, amelyek egy R fölötti r relációt alkothatnak. A megszorítás az, hogy bármely két, r-beli t_1 és t_2 rekord esetén, amelyekre $t_1[X] = t_2[X]$ teljesül, teljesülnie kell $t_1[Y] = t_2[Y]$-nak is.
Attribútumhalmaz lezártja:	Minden egyes X attribútumhalmazra meghatározzuk az attribútumoknak egy olyan XF halmazát, amelyet X funkcionálisan meghatároz F alapján; XF-et X F alatti lezártjának nevezzük.
Gyenge egyedtípus:	Azon egyedtípusok, melyek nem rendelkeznek saját kulcsattribútumokkal
Erős egyedtípus:	Azon egyedtípusok, melyeknek van kulcsattribútumuk
Diszkriminátor:	A gyenge egyedtípusoknak részleges kulcsuk (diszkriminátoruk) van, amely azon attribútumok halmaza, amelyek egyértelműen azonosítják azokat a gyenge egyedeket, amelyek ugyanazon tulajdonos egyed(ek)hez kapcsolódnak.
Specializáció:	$Z = \{S_1, S_2, \ldots, S_n\}$ specializáció olyan alosztályoknak egy halmaza, amelyeknek ugyanaz a G a szuperosztálya, azaz i = 1, 2, . . . , n esetén G/S_i egy szuperosztály/alosztály kapcsolat. (Példa: Dolgozó szuperosztály Mérnök, Vezető, Titkárnő alosztályai)
Generalizáció:	G-t generalizált egyedtípusnak (vagy a specializáció szuperosztályának, olykor pedig az $\{S_1, S_2, \ldots, S_n\}$ alosztályok generalizációjának) nevezzük. (Példa: Autó és Tehergépkocsi egyedtípus, Jármű szuperosztály az Autó és Tehergépkocsi alosztályokkal)

Kategória	T kategória egy osztály, amely n definiáló szuperosztály (D_1, D_2, …, D_n, n > 1) uniójának egy részhalmaza. Formálisan: $T \subseteq (D_1 \cup D_2 \cup \ldots \cup D_n)$
Első normálforma:	Tiltja az összetett attribútumokat, a többértékű attribútumokat, a beágyazott relációkat (az olyan attribútumokat, amelyek értékei a különálló rekordokban nem atomiak).
Második normálforma:	Egy R relációséma második normálformában (2NF-ben) van, ha R minden másodlagos (leíró) attribútuma teljesen funkcionálisan függ R elsődleges kulcsától. Egy R relációséma második normálformában (2NF-ben) van, ha R-nek nincs olyan másodlagos (leíró) attribútuma, amely részlegesen függne R bármely kulcsától.
Harmadik normálforma:	Egy R relációséma harmadik normálformában (3NF-ben) van, ha 2NF-ben van, és nincs R-nek olyan másodlagos (leíró) attribútuma, amely tranzitívan függne az elsődleges kulcstól. Egy R relációséma harmadik normálformában (3NF-ben) van, ha valahányszor egy $X \rightarrow A$ nemtriviális funkcionális függés fennáll R-en, akkor vagy (a) X egy szuperkulcsa R-nek, vagy (b) A egy elsődleges attribútuma R-nek.
Boyce-Codd-féle normálforma:	Egy R relációséma Boyce-Codd-féle normálformában (BCNF-ben) van, ha valahányszor egy $X \rightarrow A$ nemtriviális funkcionális függés fennáll R-en, akkor X egy szuperkulcsa R-nek.
Tranzakció:	A tranzakció egy végrehajtás alatt álló program, amely az adatbázis-feldolgozás egy logikai egységét alkotja. Egy tranzakció egy vagy több adatbázis-hozzáférési műveletből (beszúrás, törlés, módosítás és lekérdezés) áll.
ACID tulajdonságok:	• ***Atomosság*** (atomicity): A tranzakció a feldolgozás atomi egysége; vagy teljes egészében végrehajtódik, vagy egyáltalán nem. • ***Konzisztenciamegőrzés*** (consistency preservation): Egy tranzakció konzisztenciamegőrző, ha teljes és önálló végrehajtása az adatbázist konzisztens állapotból konzisztens állapotba viszi át. • ***Elkülönítés*** (isolation): Egy tranzakciónak látszólag más tranzakcióktól elkülönítve kell végrehajtódnia. Ez azt jelenti, hogy a tranzakció végrehajtása nem állhat kölcsönhatásban semelyik másik konkurensen végrehajtott tranzakcióval sem. • ***Tartósság*** vagy ***állandóság*** (durability vagy permanency): Egy véglegesített tranzakció által az adatbázison véghezvitt módosításoknak meg kell őrződniük az adatbázisban. Ezeknek a módosításoknak semmilyen hiba miatt nem szabad elveszniük.
Adattárház:	Az adattárház adatok téma-orientált, integrált, nemváltozó, időbélyeggel rendelkező összessége a menedzsment döntéseinek támogatására.
Típus konstruktorok ORDBMS-ben:	• rekord (row type), mely megfelel a rekord (tuple, struct) konstruktornak • tömb (array type), mellyel kollekciókat hozhatunk létre • a további kollekció típusokkal, ld. halmaz, lista, zsák, később bővült a szabvány
CAP tétel:	Állítás: Egy elosztott rendszer az alábbi három alapvető képesség közül legfeljebb kettőt tud megvalósítani: • ***konzisztencia*** (consistency), minden csomópont egy adott pillanatban ugyanazt az adatot látja • ***rendelkezésre*** állás (availability), minden kérésre érkezik válasz arról, hogy a kérés végrehajtása sikeres vagy sikertelen volt-e

	• ***particionálástűrés*** (partition tolerance), a rendszer egy tetszőleges üzenet elvesztése vagy a rendszer egy részének hibája esetén is tovább működik csak a teljes rendszer hibája, pl. egy generális hálózati hiba, okozhatja a működés hibáját

Osztályozza a tulajdonságtípusokat (attribútumokat)!
a tulajdonság-előfordulás szerkezete (***összetettsége***) szerint:

- egyszerű vagy atomi (pl. születési hely vagy évszám)
- összetett (pl. lakcím: irányítószám, település, utca, házszám)

a tulajdonság-előfordulás ***hány értéket vehet föl*** egyszerre

- egyértékű (pl. születési dátum)
- halmazértékű vagy többértékű (pl. felvett kurzusok)

a tulajdonság-előfordulás minden esetben ***megjelenik-e a háttértárolón*** (a fizikai adatbázisban)

- tárolt
- származtatott

Osztályozza a kapcsolattípusokat (fogalom és definíció)!
A kapcsolat ***foka***: meghatározza, hogy hány egyedtípus vesz részt a kapcsolatban.

- bináris (másodfokú)
- ternáris (harmadfokú)
- magasabb fokú

A (bináris) kapcsolat ***számossága***: meghatározza, hogy legfeljebb hány kapcsolat-előfordulásban vehet részt egy egyed-előfordulás.

- 1:1
- 1:N
- M:N

A (bináris) kapcsolat ***szorossága***: meghatározza, hogy a kapcsolatban részt vevő egyedtípusok minden egyedének részt kell-e vennie legalább egy kapcsolat-előfordulásban.

- kötelező
- félig kötelező
- opcionális

Ismertesse az adatbázis séma és állapot fogalmakat!
Adatbázis séma: az adatbázis leírása. Az adatbázis szerkezetének, az adattípusoknak és a megszorításoknak a leírását tartalmazza.
Adatbázis állapota: egy időpillanatban az adatbázisban tárolt aktuális adatok összessége.

Ismertesse a háromséma-architektúra szintjeit!

- ***Belső séma*** belső szinten a szerkezet és az elérési utak (pl. indexek) fizikai tárolásának leírására. Jellemzően fizikai adatmodellt használ.
- ***Koncepcionális séma*** koncepcionális szinten a teljes adatbázis szerkezetének és megszorításainak leírására a felhasználók közössége számára. Jellemzően koncepcionális vagy implementációs adatmodellt használ.

- ***Külső sémák*** külső szinten a különböző felhasználói nézetek leírására. Rendszerint ugyanazt az adatmodellt használja, mint a koncepcionális séma.

Definiálja a program-adat függetlenséget!
Definiálja a kétféle adatfüggetlenséget!
Ismertesse a logikai és a fizikai adatfüggetlenséget!

- ***Logikai adatfüggetlenség***: Annak képessége, hogy a koncepcionális séma anélkül változzon meg, hogy a külső sémáknak és a hozzájuk rendelt alkalmazói programoknak meg kellene változni.
- ***Fizikai adatfüggetlenség***: Annak képessége, hogy a belső séma anélkül változzon meg, hogy a koncepcionális sémának meg kellene változnia. Pl. a belső séma megváltozhat azáltal, hogy bizonyos fájl szerkezeteket átszervezünk vagy új indexeket hozunk létre az adatbázis hatékonyságának a javítása miatt.
- Amikor egy alacsonyabb szintű séma megváltozik, akkor csak ez és az eggyel magasabb szintűsémák közötti leképezésnek kell változnia.
- A magasabb szintű sémák változatlanok maradnak. Ezért az alkalmazói programoknak nem szükséges módosulniuk, mivel azok a külső sémákra hivatkoznak.

Sorolja fel a relációs modell sémaalapú megszorításait!

- tartománymegszorítás
- kulcsmegszorítás NULL értékre vonatkozó megszorítás
- egyedintegritási megszorítás
- hivatkozási integritási megszorítás

Ismertesse a relációs modell megszorításait, azon belül sorolja fel a sémaalapú megszorításokat!

- Az adatmodellben benne rejlő megszorítások: ***modellalapú*** vagy ***implicit megszorítások*** (pl. az adatok egy matematikai relációba szerveződnek holott alkothatnának más struktúrát is).
- Az adatmodell sémáiban közvetlenül kifejezett megszorítások: ***sémaalapú*** vagy ***explicit megszorítások***:
  - tartománymegszorítás
  - kulcsmegszorítás NULL értékre vonatkozó megszorítás
  - egyedintegritási megszorítás
  - hivatkozási integritási megszorítás
- Olyan megszorítások, amelyeket nem lehet közvetlenül az adatmodell sémáiban kifejezni, és ezért az alkalmazói programokkal kell kifejezni és érvényre juttatni őket: ***alkalmazásalapú*** vagy ***szemantikus megszorítások,*** vagy üzleti szabályok

Ismertesse a lehetséges integritási megszorítási sérüléseket!

- INSERT művelet:
  - tartomány megszorítás: ha az új rekord egyik attribútum értéke nem a megadott tartományba esik
  - kulcs megszorítás: ha az új rekord kulcs attribútum értéke már létezik a reláció egy másik rekordjánál
  - hivatkozási integritás: ha a külső kulcs érték az új rekordban egy olyan elsődleges kulcs értékre hivatkozik, amely nem létezik a hivatkozott relációban
  - egyedintegritás: ha az elsődleges kulcs érték NULL az új rekordban
- DELETE művelet:

- csak a hivatkozási integritási megszorításnál okozhat sérülést: olyan elsődleges kulcs értékkel bíró rekordot törlünk, amelyre más relációból hivatkozás van.

- UPDATE művelet:
  - a tartomány megszorítás és a NULL érték megszorítást sértheti meg.

Ismertesse a relációalgebra unáris műveleteit (definíció és tulajdonságok)!
Ismertesse a relációalgebra három unáris műveletét (definíció és tulajdonságok)!
Ismertesse a relációalgebra szelekció műveletét és annak tulajdonságait!
Ismertesse a relációalgebra projekció műveletét és annak tulajdonságait!

- **Szelekció**:
  - σ ⟨szelekciós feltétel⟩ (R), ahol R azt a relációt jelöli, amelyből a ⟨szelekciós feltétel⟩-nek eleget tevő rekordokat válogatjuk ki.
  - A szelekció megvalósítása SQL-ben: SELECT * FROM R WHERE szelekciós feltétel;
  - unáris művelet
  - Az eredményül kapott reláció foka és sémája megegyezik R fokával, illetve sémájával
  - Az eredményül kapott reláció számossága mindig kisebb vagy egyenlő R számosságánál, azaz bármely f feltétel esetén
    $|\sigma_f (R)| \leq |R|$
  - Két egymásba ágyazott szelekciós művelet végrehajtási sorrendje felcserélhető:
    $\sigma_{felt1} (\sigma_{felt2} (R)) = \sigma_{felt2} (\sigma_{felt1} (R))$
  - Minden többszörösen egymásba ágyazott (kaszkádolt) szelekció átírható egyetlen szelekcióvá, amelynek a feltétele az eredeti feltételek konjunkciója:
    $\sigma_{felt1} (\sigma_{felt2} (\ldots(\sigma_{feltn} (R))\ldots)) = \sigma_{felt1 \text{ AND } felt2 \text{ AND} \ldots \text{ AND } feltn} (R)$
- **Projekció**:
  - π ⟨attribútumlista⟩ (R), ahol az ⟨attribútumlista⟩ az R reláció lekérdezni kívánt attribútumainak listája.
  - unáris művelet.
  - Az eredményül kapott reláció fokát és sémáját az attribútumlistában szereplő attribútumok határozzák meg:
    - az eredmény sémájában az attribútumok sorrendje megegyezik a listában megadott attribútumok sorrendjével,
    - a fokszám a listában megadott attribútumok darabszáma lesz.
  - Ha az attribútumlista nem tartalmaz kulcs attribútumot, akkor az eredményül kapott reláció számossága kisebb lehet R számosságánál, ugyanis az eredményben nem jelenhetnek meg duplikált rekordok.
  - Ha az attribútumlista R szuperkulcsa, akkor az eredmény számossága megegyezik R számosságával.
  - Két egymásba ágyazott projekciós művelet eredménye megegyezik a külső projekció eredményével:
    $\pi_{lista1} (\pi_{lista2} (R)) = \pi_{lista1} (R)$
  - ha $_{lista2} \supseteq _{lista1}$, egyébként a baloldal nem értelmezhető
- **Átnevezés**:
  - Általános alakja:

    $\rho S(B_1,B_2\ldots,B_n) (R)$ vagy $\rho S (R)$ vagy $\rho(B_1,B_2\ldots,B_n) (R)$

  - Az S a reláció jelölésére használt új szimbólum, B_1, B_2, …, B_n az új attribútumnevek.
  - unáris művelet.

- Az eredményül kapott reláció foka és számossága megegyezik R fokával, illetve számosságával.
- Az eredményül kapott reláció sémája
  - a B_1, B_2, …, B_n attribútumokkal meghatározott séma lesz, ha megadtuk őket,
  - megegyezik az R sémájával, ha a B_1, B_2, …, B_n attribútumokat nem soroltuk fel.

Definiálja az uniókompatibilitás fogalmát és ismertesse a relációalgebra halmazműveleteit és azok tulajdonságait!

- **Uniókompatibilitás**: két relációnak ugyanannyi attribútuma van, és attribútumaik tartományai páronként megegyeznek egymással
- **Unió:**
  - $R \cup S$
  - eredménye: azok a rekordok, amelyek szerepelnek valamelyik relációban
  - bináris művelet
  - kommutatív: $R \cup S = S \cup R$
  - asszociatív: $R \cup (S \cup T) = (R \cup S) \cup T$
- **Metszet:**
  - $R \cap S$
  - eredménye: azok a rekordok, amelyek szerepelnek valamelyik relációban
  - bináris művelet
  - kommutatív: $R \cap S = S \cap R$
  - asszociatív: $R \cap (S \cap T) = (R \cap S) \cap T$
- **Különbség:**
  - $R - S$
  - eredménye: azok a rekordok, amelyek szerepelnek az első relációban, de nem szerepelnek a másodikban
  - bináris művelet
  - általában nem kommutatív: $R - S \neq S - R$

Ismertesse a relációalgebra általános összekapcsolási (theta join) műveletét és tulajdonságait!

- $R \bowtie_{\langle \text{összekapcsolási feltétel} \rangle} S$
- bináris művelet
- operandusai $R(A_1, A_2, \ldots, A_n)$ és $S(B_1, B_2, \ldots, B_m)$ sémájú relációk
- Az eredményül kapott Q egy n + m fokszámú reláció, melynek sémája: $Q(A_1, A_2, \ldots, A_n, B_1, B_2, \ldots, B_m)$
- Az eredményül kapott relációban benne lesz az R és az S relációk rekordjainak minden olyan kombinációja, amely kielégíti az összekapcsolási feltételt.
- Az általános összekapcsolás SQL-ben: SELECT * FROM R, S WHERE összekapcsolási feltétel;

Ismertesse a relációalgebra egyenlőségen alapuló összekapcsolási műveletét!

- Azt az általános összekapcsolási műveletet, amelynek összekapcsolási feltételében csak az egyenlőségjel (=) szerepel összehasonlító műveleti jelként, egyenlőségen alapuló összekapcsolásnak vagy más szóval ***equijoin*** műveletnek nevezzük.
- Az egyenlőségen alapuló összekapcsolás eredményeként kapott reláció (tábla) minden rekordjában van legalább egy pár azonos érték.
- Az egyenlőségen alapuló összekapcsolás SQL-ben: SELECT * FROM R [INNER] JOIN S ON R.ID = S.ID;

Definiálja az Armstrong-axiómákat és segítségükkel bizonyítsa be az XY alábbi szabályát!

- a ***reflexivitás*** szabálya:
  Ha X ⊇ Y, akkor X → Y
  - A reflexivitás szabálya szerint egy attribútumhalmaz mindig meghatározza önmagát, vagy saját maga bármelyik részhalmazát.

A reflexivitás bizonyítása

Tegyük fel, hogy $X \supseteq Y$, és hogy léteznek t_1 és t_2 rekordok R valamely r relációjában úgy, hogy $t_1[X] = t_2[X]$. Ekkor $t_1[Y] = t_2[Y]$, mivel $X \supseteq Y$; ezért $X \to Y$-nak teljesülnie kell r-ben.

- az ***augmentivitás*** szabálya:
  { X → Y } |= XZ → YZ
  - Az augmentivitás szabálya szerint egy funkcionális függés mindkét oldalának ugyanazzal az attribútumhalmazzal történő bővítésé újabb érvényes funkcionális függést eredményez.

Az augmentivitás bizonyítása (indirekt módon)

Tegyük fel, hogy $X \to Y$ fennáll R egy r relációjában, de $XZ \to YZ$ nem áll fenn. Ekkor léteznie kell t_1 és t_2 rekordoknak úgy, hogy

1. $t_1[X] = t_2[X]$,
2. $t_1[Y] = t_2[Y]$,
3. $t_1[XZ] = t_2[XZ]$ és
4. $t_1[YZ] \neq t_2[YZ]$.

Ez nem lehetséges, mert (3)-ból kapjuk, hogy

5. $t_1[Z] = t_2[Z]$,

míg (2)-ből és (5)-ből kapjuk, hogy

6. $t_1[YZ] = t_2[YZ]$,

ami ellentmond (4)-nek.

- az ***additivitás*** szabálya:
  { X → Y , X → Z } |= X → YZ
  - Az additivitás szabálya szerint funkcionális függések egy {X → A1, X → A2…, X → An} halmazát összevonhatjuk egyetlen X → {A1, A2…, An} funkcionális függéssé.

Az additivitás bizonyítása

1. $X \to Y$ adott.
2. $X \to Z$ adott.
3. $X \to XY$, alkalmazva az augmentivitás szabályát (1)-re, azt X-szel bővítve; megjegyezve, hogy $XX = X$.
4. $XY \to YZ$, alkalmazva az augmentivitás szabályát (2)-re, azt Y-nal bővítve.
5. $X \to YZ$, alkalmazva a tranzitivitás szabályát (3)-ra és (4)-re.

- a ***tranzitivitás*** szabálya:
  { X → Y , Y → Z } |= X → Z
  - A tranzitivitás szabálya szerint a funkcionális függések tranzitívak

A tranzitivitás bizonyítása

Tegyük fel, hogy

1. $X \to Y$ és
2. $Y \to Z$

fennáll egy r relációban. Ekkor tetszőleges t_1 és t_2 r-beli rekordokra, melyekre igaz, hogy $t_1[X] = t_2[X]$, (1) miatt kapjuk, hogy

3. $t_1[Y] = t_2[Y]$;

így (3)-ból és a (2)-es feltevésünkből azt is kapnunk kell, hogy

4. $t_1[Z] = t_2[Z]$;

ezért $X \to Z$-nek fenn kell állnia r-ben.

- a ***dekompozíció*** szabálya:
  { X → YZ } |= X → Y
  - A dekompozíció szabálya azt mondja, hogy egy funkcionális függés jobb oldaláról eltávolíthatunk attribútumokat.

A dekompozíció bizonyítása

1. $X \to YZ$ adott.
2. $YZ \to Y$, felhasználva a reflexivitás szabályát, és tudva, hogy $YZ \supseteq Y$.
3. $X \to Y$, alkalmazva a tranzitivitás szabályát (1)-re és (2)-re.

- a ***pszeudotranzitivitás*** szabálya:
  { X → Y , WY → Z } |= WX → Z

A pszeudotranzitivitás bizonyítása

1. $X \to Y$ adott.
2. $WY \to Z$ adott.
3. $WX \to WY$, alkalmazva az augmentivitás szabályát (1)-re, azt W-vel bővítve.
4. $WX \to Z$, alkalmazva a tranzitivitás szabályát (3)-ra és (2)-re.

Sorolja fel az ER séma leképezésének lépéseit relációs sémává?

1. Erős egyedtípusok leképezése
2. Gyenge egyedtípusok leképezése

3. Bináris 1:1 számosságú kapcsolattípusok leképezése
    (a) külső kulcs használata
    (b) összevonás
    (c) kereszthivatkozás v. kapcsoló reláció használata
4. Bináris 1:N számosságú kapcsolattípusok leképezése
5. Bináris M:N számosságú kapcsolattípusok leképezése
6. Többértékű attribútumok leképezése
7. N-edfokú kapcsolattípusok leképezése

Ismertesse az 1:1 kapcsolatok leképezésének módjait az ER séma relációs sémává leképezésénél?

- Külső kulcs használata: Válasszuk ki az egyik relációt (mondjuk S-t) és vegyük fel S külső kulcsaként T elsődleges kulcsát. Célszerű S-nek azt a relációt választani, amelyiket abból az egyedtípusból képeztünk le, amelyik totális résztvevője az R kapcsolatnak. Vegyük fel továbbá R egyszerű attribútumait, illetve R összetett attribútumainak egyszerű komponenseit S attribútumaiként. (Azt is megtehetnénk, hogy S (a totális résztvevő) elsődleges kulcsát vesszük fel T külső kulcsaként, de ebben az esetben T minden olyan rekordjánál, amelyik nem vesz részt a kapcsolatban, ehhez a külső kulcshoz null értéket kellene rendelni.)
- Összevonás: a két egyedtípust és a kapcsolatot egyetlen relációba vonjuk össze. Ezt akkor tehetjük meg, ha mindkét egyedtípus totális résztvevője a kapcsolatnak.
- Kereszthivatkozás vagy kapcsoló reláció használata: felveszünk egy harmadik R relációt abból a célból, hogy kereszthivatkozással lássuk el a két egyedtípusból képzett S és T relációk elsődleges kulcsait. Ahogy látni fogjuk, ezt a megközelítést alkalmazzuk a bináris M:N kapcsolatoknál is. Az R relációt kapcsoló relációnak nevezzük, mert R minden rekordja egy kapcsolat-előfordulást reprezentál, amely S egy rekordját T egy rekordjával kapcsolja össze.

Sorolja fel az EER séma leképezésének lépéseit relációs sémává?

1. Erős egyedtípusok leképezése
2. Gyenge egyedtípusok leképezése
3. Bináris 1:1 számosságú kapcsolattípusok leképezése
    a. külső kulcs használata
    b. összevonás
    c. kereszthivatkozás v. kapcsoló reláció használata
4. Bináris 1:N számosságú kapcsolattípusok leképezése
5. Bináris M:N számosságú kapcsolattípusok leképezése
6. Többértékű attribútumok leképezése
7. N-edfokú kapcsolattípusok leképezése
8. Specializációk és generalizációk leképezése
9. Unió típusok (kategóriák) leképezése

Ismertesse a specializációk osztályozási lehetőségeit?

- Z specializáció ***totális***, ha mindig teljesül, hogy $G = (S_1 \cup S_2 \cup \ldots \cup S_n)$
- Z specializáció ***részleges***, ha Z specializáció nem totális.
- Z specializáció ***diszjunkt,*** ha minden $i \neq j$ esetén teljesül, hogy $S_i \cap S_j = \emptyset$ (üres halmaz)
- Z specializáció ***átfedő,*** ha nem diszjunkt.

Ismertesse a relációs adatbázis-tervezés nem hivatalos irányelveit!

1. Egy reláció minden egyes rekordja egy egyed-előfordulást vagy kapcsolat-előfordulást reprezentáljon. (Az egyes relációkra és azok attribútumaira külön-külön vonatkozik.)

2. Olyan sémát tervezzünk, amelyben nem jelennek meg beszúrási, törlési és módosítási anomáliák. Ha mégis előfordulnak anomáliák, akkor jegyezzük fel azokat, hogy az alkalmazások számításba vehessék őket.
3. A relációkat úgy kell megtervezni, hogy a rekordjaik a lehető legkevesebb NULL értéket tartalmazzák. Azok az attribútumok, amelyek gyakran vesznek fel NULL értéket, külön relációkba tehetők (az elsődleges kulccsal).
4. A relációkat úgy kell megtervezni, hogy kielégítsék a veszteségmentes összekapcsolás feltételét. Egy tetszőleges, relációkon végrehajtott természetes összekapcsolás nem állíthat elő álrekordokat.

Sorolja fel a karbantartási anomáliákat és mondjon rájuk példákat!

DOLG_PROJ(Dszsz, Pszám, Dnév, Pnév, Órák) relációt tekintve:

- ***Beszúrási anomália***: Nem tudunk új projektet beszúrni, ha nincs hozzárendelve egyetlen dolgozó sem, valamint nem tudunk új dolgozót beszúrni, ha nincs hozzárendelve egyetlen projekthez sem.
- ***Törlési anomália***: Ha törlünk egy projektet, akkor az összes olyan dolgozó is törlődik, aki az adott projekten dolgozik, valamint ha egy dolgozó egyedüliként dolgozik egy projekten, akkor a dolgozó törlése a szóban forgó projekt törlését is maga után vonja.
- ***Módosítási anomália***: ha a P1 számú projekt nevét megváltoztatjuk, akkor azt minden olyan dolgozó esetén végre kell hajtani, aki a P1 projekten dolgozik.

Sorolja fel egy tranzakció lehetséges állapotait!

- aktív állapot
- részlegesen véglegesített állapot
- véglegesített (commit) állapot
- hibás állapot
- megszakított állapot

Sorolja fel a tranzakciós műveleteket!

- begin_transaction
- read vagy write
- end_transaction
- commit_transaction
- rollback vagy abort

Definiálja egy tranzakció véglegesítési (commit) pontját!

Egy tranzakció akkor éri el a véglegesítési (commit) pontját, ha az összes adatbázis-hozzáférési művelete sikeresen végrehajtódott és ezen műveletek hatása kiírásra került a log-fájlba. A véglegesítési pontja után a tranzakciót véglegesítettnek nevezzük és feltételezzük, hogy összes hatása állandó bejegyzésre került az adatbázisban. A tranzakció ezután egy [commit,T] bejegyzést tesz a log-ba.

Ismertesse azokat a problémákat, amelyek tranzakciók egyidejű feldolgozásakor léphetnek fel!

- ***Az elveszett frissítés problémája***. Akkor fordul elő, amikor két tranzakció, amely ugyanazokat az adatbázis elemeket éri el, úgy fésülődik össze, hogy egyes adatbázis elemek hibásakká válnak.

- ***Az időleges frissítés (dirty read) problémája.*** Akkor fordul elő, amikor egy tranzakció frissít egy adatbázis elemet, ami után valamilyen oknál fogva a tranzakció hibásan fejeződik be. Ezt a frissített elemet más tranzakció is eléri mielőtt az még visszaállna az eredeti értékére.
- ***A helytelen összegzés problémája.*** Amikor egy tranzakció rekordok egy összegző függvényét számolja, amíg egy másik tranzakció ezen rekordok közül néhányat frissít. Ekkor az összegző függvény olyan értékekkel számolhat, amelyek még a frissítés előtt vannak, míg mások már a frissítés után.

Mit ért a tranzakció ütemezés alatt és milyen fajtáit ismeri?

- Tranzakció ütemezés: A különböző tranzakciókban lévő műveletek sorrendje, amikor a tranzakciókat összefésülve egy szálon hajtjuk végre.
- Típusok:
    - Visszaállítható ütemezés
    - Kaszkádmentes ütemezés
    - Szeriális ütemezés
    - Szerializálható ütemezés

Definiálja az osztályhierarchiát és öröklődést, valamint ismertesse az öröklődés típusait?
Mi a különbség a többszörös és a szelektív öröklődés között?

- ***Osztályhierarchia***: az összes szupertípus (típus, amiből szubtípust származtatunk) és szubtípus (új típus, amely egy már definiált típus összes függvényét tartalmazza) kapcsolatrendszere
- ***Öröklődés típusai***:
    - ***Többszörös öröklődés***: akkor beszélünk róla, ha egy altípus kettő vagy több típus altípusa és így értelemszerűen örökli mindkettő vagy az összes függvényét (attribútumait és metódusait).
    - ***Szelektív öröklődés***: amikor egy altípus csak egy típus bizonyos függvényeit örökli, amelyeket nem, azokat az EXPECT klózzal jelezzük

Mit jelent a tranziencia és perzisztencia az OO adatbázisokban?

- Tranziencia: ideiglenesség
- Perzisztencia: állandóság

Ismertesse az adattárházak és a hagyományos adatbázisok közötti különbségeket!

- Az adattárházakat főként a gyors adatelérésre optimalizálják. A hagyományos adatbázisok tranzakciósak és egyaránt optimalizáltak az adatelérési mechanizmusok és a konzisztencia biztosítása tekintetében.
- Az adattárházak nagyobb hangsúlyt helyeznek a históriku adatokra mivel fő céljuk idősorok és trend elemzések támogatása.
- A tranzakciós adatbázisokkal szemben az adattárházak nem változnak abban az értelemben, hogy ha egy adat egyszer oda bekerült, akkor az ott is marad változatlan formában az „idők végezetéig”.
- A tranzakciós adatbázisokban a tranzakció az a mechanizmus, amely megváltoztatja az adatbázist. Ezzel szemben az adattárházakban az információ durván szemcsézett és a frissítési politika alaposan megválasztott, általában inkrementális jellegű.

Ismertesse az SQL objektum-relációs kiterjesztésében a felhasználó által definiált típust (UDT)!

- Célja:

    - összetett szerkezetű (a relációs modell rekordjainál bonyolultabb) objektumok létrehozása
    - egy típus deklarációjának elválasztása a tábla (reláció) létrehozásától
    - rekord típusú konstruktor a ROW kulcsszóval rekord típusú attribútumok létrehozására
    - 4-féle kollekció típus: ARRAY, MULTISET, LIST és SET
- Létrehozásának módja: CREATE TYPE típus_neve AS (komponensek deklarációja)
- Objektumok azonosítása egyértelmű, rendszer által generált OID-vel referencia típus útján. Emellett használható a relációs modell hagyományos kulcsa is.
- A példányosítható (INSTANTIABLE kulcsszó) UDT-khez táblákat (relációkat) is létrehozhatunk.
- Az UDT-khez műveleteket (metódusokat) is definiálhatunk.
- Attribútumok és műveletek három fajtája:
    - PUBLIC - látható az UDT interfészen
    - PRIVATE - nem látható az UDT interfészen
    - PROTECTED - csak az altípusok számára látható

Hogyan örököltethet felhasználó által definiált típust és mik ennek a tulajdonságai?

- A NOT FINAL kulcsszót kell használni, ha egy UDT-nek további altípusát szeretnénk deklarálni.
- Minden attribútum öröklődik.
- A szupertípusok sorrendje az UNDER kulcsszó után határozza meg az öröklődési sorrendet.
- Egy altípus példánya minden olyan kontextusban használható, ahol szupertípusának példánya használható.
- Egy altípus minden a szupertípuson definiált függvényt újradefiniálhat feltéve, hogy a szignatúra nem változhat.
- Egy függvény hívásakor a legjobban illeszkedő implementáció kerül alkalmazásra az összes argumentum típusát figyelembe véve.
- Dinamikus kötéskor a paraméterek futáskori típusait veszi figyelembe.

Definiálja a gyenge és az erős konzisztenciát NoSQL adatbázisokban!

- ***Erős konzisztencia***: Miután a módosítás végrehajtódott, minden (akár A, akár B, akár C által végzett) hozzáférés a módosított értéket adja eredményül.
- ***Gyenge konzisztencia***: A rendszer nem garantálja, hogy a későbbi hozzáférések a módosított értéket adják eredményül. Több feltételnek is teljesülnie kell, mielőtt az érték visszaadásra kerül. A módosítás megtörténte és azon pillanat közötti időszakot, amelyre már garantált, hogy minden megfigyelő mindig a módosított értéket látja, inkonzisztenciaablaknak (inconsistency window) nevezzük.

Sorolja fel a NoSQL adatmodelleket!

- NoSQL adatmodellek
- Kulcs–érték modell
- Rendezett kulcs–érték modell
- Oszlopcsalád modell
- Dokumentum modell
- Gráfmodel