Kulcs, rendezési kulcs, egyedi kulcs, elsődleges kulcs

Az adatbázis létrehozásának elsődleges célja az, hogy az adatokat meghatározott szempontok szerint gyorsan vissza tudjuk keresni, rendszerezni tudjuk. A gyors visszakeresés csak rendezett adathalmazból lehetséges. A telefonkönyvben azért találjuk meg pillanatok alatt a keresett személyt, mert az adatok név szerint abc-be vannak rendezve. A rendezési, keresési szempontot kulcsnak nevezzük.

A kulcs lehet a táblázat egy vagy több adatmezője, vagy a mezőkből alkotott kifejezés. Egy személyi nyilvántartásban például a lakóhely (település) lehet rendezési kulcs. Azt a kulcsot, amely egyértelműen azonosítja az egyedeket (rekordokat), egyedi kulcsnak, vagy egyedi azonosítónak nevezzük. A meghatározásból következik, hogy az egyedi kulcs a reláció minden sorában különböző értéket vesz fel. Ellenkező esetben ugyanis lennének a relációnak megegyező sorai, ami ellentmond a relációkra vonatkozó szabályoknak.

A személyi nyilvántartásban az egyedi kulcs lehet például a személyi szám. Egy táblában többféle egyedi azonosítót is választhatunk. A személyek egyedi azonosítója lehet a név, anyja neve, születési hely mezők együttese is. Az egyedi kulcs lehet egyszerű (például a TAJ szám), vagy összetett aszerint, hogy egy vagy több mezőből áll.

Az adatbázis szerkezetének kialakításakor az egyedi kulcsok közül legtöbbször kijelölünk egyet, amit azonosítóként használunk. Az aktuálisan kiválasztott azonosítót elsődleges kulcsnak nevezzük. Elsődleges kulcsként a mezőknek azt a legszűkebb kombinációját célszerű kiválasztani, amelyek együtt egyértelműen azonosítják a rekordot. Ha van a táblában egyetlen mező, amely alkalmas erre a célra, akkor legtöbbször azt érdemes elsődleges kulcsként megjelölni. Ha az egyedtípus például az autó, a legcélszerűbb elsődleges kulcs a rendszám. Azokban a táblákban, ahol az adatok jellegéből adódóan nincs megfelelő természetes azonosító, gyakran egy egyedi sorszámot képezünk elsődleges kulcsként. (Például egy videó kölcsönzőben a kazetták azonosítója lehet egy generált sorszám).

Azokat a tulajdonságokat, vagy tuladonság-halmazokat, amelyekből képezhetünk elsődleges kulcsot, kulcsjelölteknek nevezzük. Ezek közül elsődleges kulcsként azt kell kiválasztani, amely az egyednek "viszonylag" állandó tulajdonsága. (Lehet, hogy a személy neve, lakcíme és születési éve együttesen jó azonosító, mégsem jó választás, hiszen a lakcím változhat). Ilyen a személyi nyilvántartásban a TAJ szám, adószám, személyi szám.

Elsődleges, másodlagos tulajdonság

Az egyednek azt a tulajdonságát, amely része az elsődleges kulcsnak, elsődleges tulajdonságnak nevezzük. Azokat a tulajdonságokat, amelyek nem részei az elsődleges kulcsnak, másodlagos, vagy leíró tulajdonságnak nevezzük.

Például:

Személy(név, anyja neve, születési idő, születési hely, nem, szemszín, lakcím)

A személy tábla elsődleges kulcsa a négy aláhúzott mező együttese. A **név** elsődleges tulajdonság, mert része az elsődleges kulcsnak. A személy neme másodlagos tulajdonság, az elsődleges kulcs egyértelműen meghatározza. Az elsődleges kulcsot a reláció sémájában aláhúzással jelöljük. A relációban az elsődleges kulcsnak alapvetően fontos szerepe van, hiszen számos műveletnél (keresés, módosítás, törlés) ezzel hivatkozunk a rekordokra. Az adatbevitelnél éppen ezért az elsődleges kulcsként megjelölt mezőt vagy mezőket kötelező kitölteni. Az adatbázis tervezése során az egyik legfontosabb feladat az egyedtípusok elsődleges kulcsának célszerű megválasztása.

Redundancia

Az adatbázisban nem célszerű ugyanazt az adatot többször tárolni. Ha ez mégis előfordul, azt redundanciának (adatismétlődésnek) nevezzük. A redundanciát általában nem lehet teljes egészében kiküszöbölni, de célszerű erre törekedni. Amint azt látni fogjuk, a redundancia ugyanis nemcsak a tárhely

pazarlása miatt káros, lassítja az adatkezelést, gátolja az adathalmaz áttekinthetőségét és az adatbázisban ellentmondások (anomáliák) léphetnek fel. Vegyük példalánt a könyvesbolt adatbázisában a könyv egyedtípust.

Könyv (ISBN, szerző, cím, a kiadó neve, kiadás éve, a kiadó címe, téma, ár)

A leggyakoribb tervezői hiba az, hogy minden adatot egy táblába szeretnénk belezsúfolni. A redundancia ekkor elkerülhetetlen.

ISBN	Szerző	Cím	A kiadó neve	A kiadó telefonszáma	Kiadás éve	A kiadó címe	Téma	Ár
963 618 1764	Bócz, Tamás	A világháló lehetőségei	Panem	3 123 454	2016	Budapest, Varga u. 20.	informatika	3200
985 561 1234	Rödel, Egmar	A világ fővárosai	Európa	2 444 555	2013	Szeged, Fő tér 1.	földrajz	1190
666 999 3333	Ullman, Jeffrey	A programozás alapjai	Panem	3 123 454	2018	Budapest, Varga u. 20.	informatika	2500
777 231 1231	Kollár, István	A filozófia története	Gondolat	2 432 454	2018	Debrecen, Csapó u.12.	filozófia	2340
123 456 7721	Balla, Béla	Programozás C nyelven	Panem	3 123 454	2017	Budapest, Varga u. 20.	informatika	1900

A kiadók nevét, telefonszámát, stb. a táblában többször tároltuk. A többszörös tárolás hátrányai:

- pazarlás a tárhellyel
- fölösleges többletmunka az adatbevitelnél
- módosítási anomália
- törlési anomália
- bővítési anomália

Módosítási anomália

Tegyük fel, hogy a Panem könyvkiadónak megváltozik a telefonszáma. A módosítást csak úgy lehet következetesen végig vinni, ha a tábla minden rekordját egyenként megvizsgáljuk. Ez nyilvánvalóan olyan koncepcionális hiba, amit az adatbázis szerkezetének kialakítása során meg kell kísérelni kiküszöbölni.
Ugyanez a helyzet áll elő akkor is, ha a témakörök valamelyikét kell módosítani (például, ha az "Informatika" témakört "Informatika és számítástechnika" témakörre szeretnénk javítani.

Törlési anomália

Tegyük fel, hogy "A filozófia története" című könyv elfogyott, és nem is fogják újra kiadni. Ha töröljük a táblából, a Gondolat kiadó telefonszámával együtt eltűnik a nyilvántartásunkból. A hibás adatszerkezet eredménye az, hogy egy adat törlésekor olyasmi is elvész, amire még később szükség lenne.

Bővítési anomália

Tegyük fel, hogy a könyvesbolt felveszi a kapcsolatot az Akadémia kiadóval. Mindaddig, amíg a kiadónak nincs könyve a raktárban, a kiadó adatait sem lehet bevinni.

Az anomáliák elkerülésének módja az, hogy a táblázatot felbontjuk több olyan táblázatra, amelyekben nem keverednek a különböző egyedtípusok. A **Könyv** egyedtípusba ugyanis olyan tulajdonságokat is belevettünk, amelyek valójában más egyedtípus jellemzői. A tábla nem egy, hanem három egyedtípus adatait tartalmazza: Könyv, Kiadó, Témakör, Szerző. A Kiadó egyedtípus tartalmazza azokat a tulajdonságokat, amelyek szigorúan csak a kiadóra vonatkoznak. Ebben a táblában az azonosító legyen egy egyedi sorszám.

Kiadó azonosító	A kiadó neve	A kiadó címe	A kiadó telefonszáma
1	Panem	Budapest, Varga u. 20.	3 123 454
2	Európa	Szeged, Fő tér 1.	2 444 555
3	Gondolat	Debrecen, Csapó u.12.	2 432 454

A **Témakör** egyedtípus

Téma azonosító	Téma
1	informatika
2	földrajz
3	filozófia

A módosított Könyv tábla

ISBN	Szerző	Cím	Kiadó	Kiadás éve	Téma	Ár
963 618 1764	Bócz Tamás	Az Internet világa	1	2016	1	3200
985 561 1234	Rödel, Egmar	A világ fővárosai	2	2013	2	1190
666 999 3333	Ullman, Jeffrey	A programozás alapjai	1	2018	1	2500
777 231 1231	Kollár, István	A filozófia története	3	2018	3	2340
123 456 7721	Balla, Béla	Programozás C nyelven	1	2017	1	1900

Ha a könyv adatait szeretnénk kilistázni, a kiadó nevét, és a témakört elsődleges kulcsuk alapján a Kiadó és Témakör táblából vesszük elő. A módosítási anomália ezzel megszünt, hiszen ha a változik egy kiadó valamely adata, akkor azt egyetlen helyen kell módosítani, a Kiadó táblában.

A törlési anomália is megszűnt, hiszen azzal, hogy a "A filozófia története" c. könyvet töröljük a Könyv táblából, a Gondolat kiadó adatai még megmaradnak a Kiadó táblában.

Azt a mezőt, amely egy másik táblában elsődleges kulcs, az adott táblában idegen kulcsnak nevezzük. Egy elsődleges és az idegen kulcs-pár alapján a táblákat összekapcsolhatjuk. Az összetartozó egyedek megfeleltetése a relációk közötti természetes összekapcsolás (natural join) műveletével történik. A kapcsoló mező az egyik táblában elsődleges kulcs, a másikban idegen kulcs.

A táblák rekordjai közötti kapcsolat tükrözi a valóság egyedei közötti kapcsolatokat. Például:

- minden könyvnek van egy egyértelműen meghatározható kiadója
- a könyvek témakör szerint csoportosíthatók

A **Könyv** táblában a Kiadó és Téma mezőkben csak az adott téma azonosítóját tároltuk. Az első rekordban a **Kiadó** mező értéke: 1. Ez egyértelműen azonosítja a Kiadó táblában azt a rekordot, amelyben megtalálható a kiadó neve, mivel a Könyv tábla **Kiadó** mezője a Kiadó táblában elsődleges kulcs.

Az egyedek közötti kapcsolatok

Az adatbázis akkor jó, ha a vizsgált rendszer objektumai (egyedei) között fennálló kapcsolatokat képes modellezni. Az első leckében említettük, hogy a hierarchikus és a hálós adatmodellekben az egyedek közötti kapcsolat megvalósítása meglehetősen bonyolulttá tette az adatkezelést. A relációs adatmodell egyik legfontosabb előnye az, hogy a kapcsolatok könnyen kezelhetővé válnak azáltal, hogy egyedek közötti kapcsolatok a relációk (táblák) közötti kapcsolatokkal modellezhetők.

Az alábbiakban a kapcsolatok típusait mutatjuk be.

Egy a többhöz kapcsolat

Jele: 1:N

Ha az egyik egyedtípus valamely egyedéhez a másik egyedtípus több egyede is tartozhat, de a másik egyedtípus egyedeihez az első egyedtípusnak pontosan egy egyede tartozik, akkor azt mondjuk, hogy a két egyed között egy a többhöz kapcsolat van.

Feltételezzük, hogy a könyvesboltban tárolt könyvek mindegyikének egy kiadója van (egy adott időszakban ez biztosan igaz, ha nem így volna, az adatmodellünk nem tükrözné a valóságot), de egy kiadó sok könyvet ad ki. A Kiadó oldaláról nézve ez egy a többhöz kapcsolat (egy kiadó több könyvet ad ki).
A meghatározás így is helyes:
Ha az egyik tábla valamely rekordjához a másik tábla több rekordja tartozhat, akkor azt mondjuk, hogy a két tábla között egy a többhöz kapcsolat van.

Egy az egyhez kapcsolat

Jele: 1:1
Ha az egyik tábla egy rekordjához a másik táblának pontosan egy rekordja tartozik és fordítva, akkor azt mondjuk, hogy a két tábla között egy az egyhez kapcsolat van.
A városok nyilvántartásában a város és a polgármesterek között egy az egyhez kapcsolat van, egy adott pillanatban egy városnak pontosan egy polgármestere van, és megfordítva.

Több a többhöz kapcsolat

Jele: N:M

Akkor mondjuk, hogy két tábla között több a többhöz kapcsolat van, ha az egyik tábla valamely rekordjához a másik tábla több rekordja is tartozhat, és megfordítva.

A könyvesbolt adatbázisában az adatmodellt egy kicsit leegyszerűsítettük. Ugyanis azzal, hogy a Szerző mezőt felvettük a Könyv egyedtípus tulajdonságai közé, feltételeztük, hogy egy könyvnek csak egy szerzője van. Valójában egy könyvnek több szerzője is lehet, és egy szerző több könyvet is írhat. Tipikus példa a több a többhöz kapcsolatra. A fenti megoldásban a könyvet csak egy szerző szerint tudjuk visszakeresni, hiszen a rekordokban csak egy szerzőt tüntettünk fel. Másrészt egy-egy szerzőt többször is tárolnunk kell, pontosan annyiszor, ahány könyvet írt. Ha a Könyv egyedtípusból kiemeljük a **Szerző** mezőt, és a szerzők adatait egy külön táblában tároljuk, a két tábla között több a többhöz kapcsolat lesz.
A N:M kapcsolatot fel kell bontani két 1:N típusú kapcsolatra, így a kapcsolat modellezésére be kell ikatatni egy úgynevezett kapcsoló táblát, amely a mi példánkban a "Melyik szerző, melyik könyv szerzője?" kérdésre ad választ. A Könyv táblából emeljük ki a **Szerző** mezőt.

ISBN	Cím	Kiadó	Kiadás éve	Téma	Ár
963 618 1764	Az Internet világa	1	2016	1	3200
985 561 1234	A világ fővárosai	2	2013	2	1190
666 999 3333	A programozás alapjai	1	2018	1	2500
777 231 1231	A filozófia története	3	2018	3	2340
123 456 7721	Programozás C nyelven	1	2017	1	1900

Hozzunk létre egy új táblát a szerzők adataival: Szerző (sz_id, szerző)

(Az "id" rövidítés az identify angol szó első két betűje, magyarul: azonosít)

Sz_id	Szerző
1	Bócz, Tamás
2	Rödel, Egmar
3	Ullman, Jeffrey
4	Kollár, István
5	Balla, Béla
6	Kovács, Pál

A "Melyik könyvnek, ki a szerzője?" kapcsolatot modellezhetjük egy harmadik táblával:

Könyv_Szerző

ISBN	Sz_id
963 618 1764	1
963 618 1764	6
985 561 1234	2
666 999 3333	3
777 231 1231	4
123 456 7721	5

A kialakult kapcsolatok:
Könyv --- Könyv_Szerző: egy a többhöz kapcsolat, kapcsoló mező az ISBN. Az ISBN a Könyv táblában elsődleges, a Könyv_Szerző táblában idegen kulcs.

Könyv_Szerző---Szerző: több az egyhez kapcsolat, kapcsoló mező az Sz_id. Az Sz_id a Szerző táblában egyedi, a Könyv_Szerző táblában idegen kulcs.

Ezzel a megoldással rögzíthetjük az a tényt, hogy "Az Internet" c. könyvnek két szerzője van. Ha pontosan szeretnénk tudni, egy könyv szerzőinek nevét, akkor

1. a Könyv táblában megkeressük a könyv ISBN számát.
2. az Könyv_Szerző táblából kigyűjtjük azokat a sorokat, ahol ez az ISBN szám szerepel.
3. a kigyűjtött sorokban lévő Sz_id alapján kiiratjuk a Szerző táblából a szerzők nevét.

Az első adatbázis-kezelő programokban (dBASE, Clipper) ezt a feladatot pontosan a megadott lépések szerint kellett elvégezni. A későbbiekben látni fogjuk, hogy a feladat az SQL-nyelvben reláció-műveletekre vezethető vissza.

Megjegyzés:

Ha a könyv címét és szerzőit együtt szeretnénk látni, az alábbi reláció-műveleteket kell elvégeznünk.
Képezzük a Könyv, a Könyv_Szerző és a Szerző táblák egymás utáni természetes összekapcsolását az ISBN illetve a Sz_id mezők alapján, majd a kapott tábla projekcióját az Cím és Szerző mezőkre. Az eredmény a következő tábla lesz:

Szerző	Cím
Bócz Tamás	Az Internet
Kovács, Pál	Az Internet

Rödel, Egmar	A világ fővárosai
Ullman, Jeffrey	A programozás alapjai
Kollár, István	A filozófia története
Balla, Béla	Programozás C nyelven

A redundancia és az anomáliák megszüntetéséhez felbontottuk az eredeti Könyv relációt négy relációra. Így már valódi adatbázist hoztunk létre, egy reláció ugyanis még nem alkot adatbázist, négy már inkább. Egy komolyabb adatbázis többszáz táblából is állhat. Az adatbázis hatékonysága legtöbbször nem az adatbázis-kezelő program hatékonyságán múlik, hanem azon, hogy mennyire pontosan írja le a tervezés során kialakított adatmodell a vizsgált rendszert. A tervezés legfontosabb lépése a rendszer egyedtípusainak és az egyedek közötti kapcsolatoknak a feltárása. A következő leckében bemutatjuk a tervezés során használatos, úgynevezett normalizálás módszerét, amelynek célja a logikailag jól felépített, anomáliáktól és redundanciától mentes adatbázis-séma kialakítása.