Adatbázisok tervezése egyed-kapcsolat és relációs adatmodellben

Készítette: Hampel György
Szegedi Tudományegyetem Mérnöki Kar

Lektorálta: Fabulya Zoltán
Szegedi Tudományegyetem Mérnöki Kar

Szeged
2024

Az oktatóanyag fejlesztését az RRF-2.1.2-21-2022-00012 azonosítószámú,

Komplex Digitális Modellváltás – intelligens fokozatváltás

projekt támogatta.

A projekt Magyarország Helyreállítási és Ellenállóképességi Tervének keretében valósul meg.

Tartalomjegyzék

1. lecke

Adatbázis, adatbázis-kezelő rendszer és adatbázis-tervezés

Elvárt tanulási eredmények

Tudás

- Ismeri az adatbázis fogalmát
- Átfogóan ismeri az adatbázisok fő típusait
- Ismeri az adatbázis-kezelő fogalmát
- Ismeri az adatbázis-tervezés lépéseit

Képesség

- Képes helyes sorrendbe rakni az adatbázis-tervezés lépéseit
- Képes meghatározni, hogy az egyed-kapcsolat modellnek és a relációs adatmodellnek hol a helye az adatbázis-tervezésben

Attitűd

- Fontosnak tartja az adatbázisok létrehozásának tervezéssel történő előkészítését

Autonómia, felelősség

- Önállóan és kreatív módon alkalmazza tanulmányai és munkája során az adatbázisokkal és azok tervezésével kapcsolatos ismereteket

Miről lesz szó ebben a fejezetben?

- Tisztázzuk, hogy miért is van szükségünk adatbázisokra.
- Megnézzük, hogy mi az adatbázis. Látni fogjuk, hogy sokféle meghatározás létezik rá.
- Áttekintjük az adatbázisok fő típusait különböző szempontok szerint.
- Az adatbázisokat adatbázis-kezelő rendszerek segítségével tudjuk használni. Röviden megnézzük, hogy mi ez és mi a feladata ennek a rendszernek.
- Röviden áttekintjük az adatbázis-kezelés fejlődését az 1960-as évektől napjainkig.
- Az adatbázisokat, különösen, ha nagyok és bonyolultak, célszerű átgondolni és megtervezni. A fejezet végén az adatbázis-tervezés lépéseiről lesz szó. Ebben kap szerepet a manapság népszerű egyed-kapcsolat modell, mint a koncepcionális tervezés egyik eszköze és a relációs adatmodell, amely a legelterjedtebb relációs adatbázis-kezelők által használt modell.

1 Adatbázis, adatbázis-kezelő rendszer és adatbázis-tervezés

1.1 Bevezetés

Környezetünk leírásához meg kell azt figyelnünk; adatokat kell gyűjtenünk az objektumairól, összegeznünk kell azok tulajdonságait, és meg kell figyelnünk a közöttük levő kapcsolatokat. Ezután juthatunk a rendelkezésre álló adatokból új ismeretekhez, információhoz.

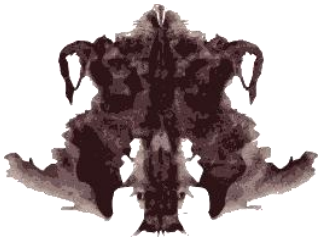
Amikor egy adatbázist magunk elé képzelünk, leginkább táblázato(ka)t látunk tele adatokkal. Talán innen jöhet a felhasználóktól néha hallható „**minek az adatbázis-kezelő, jó lesz a táblázatkezelő is**…”.

Vannak azonban **esetek, amikor egyetlen táblázatban nem tárolhatók, nem kezelhetők hatékonyan az adatok, mégpedig azok hatalmas mennyisége és a közöttük fennálló sokrétű és bonyolult kapcsolatrendszer miatt**. Erre lettek kitalálva az adatbázis-kezelők.

Az adatbázis-kezelés **alapvető szerepet játszik a modern információs rendszerekben**, melyek különböző alkalmazási területeken jelennek meg, mint például az üzleti intelligencia, az egészségügyi rendszerek, az e-kereskedelem és az oktatás. Az **adatbázis-kezelés olyan technológiák, eszközök és módszerek összessége, amelyek lehetővé teszik az adatok strukturált tárolását, kezelését és elérését**.

Az adatbázis-kezelés kulcsfontosságú a különböző szervezetek számára, mivel lehetővé teszi az adatok hatékony tárolását, kezelését és elemzését. Az adatvezérelt döntéshozatal, az üzleti intelligencia és a versenyképesség növelése érdekében az adatok gyors és pontos elérése elengedhetetlen.

1.2 Az adatbázis



Az adatbázisra vonatkozóan nincs egy egységesen elfogadott definíció. Néhány meghatározás: **Az adatbázis**…

- adatok összessége.
- a felhasználók által rugalmasan kezelhető adatok rendszere.
- olyan adathalmaz, amely több, különböző felépítésű, egymással kölcsönös kapcsolatban lévő adatok gyűjteménye.
- alatt az adatoknak kapcsolataikkal együtt való ábrázolását, tárolását értjük.
- olyan **adatállomány, amely egy adatbázis-kezelő rendszerrel hozható létre és érhető el**.
- összetartozó adatok azon rendszere, mely megosztott több felhasználó között, és az elérést egy központi vezérlőprogram szabályozza.
- **véges számú egyed-előfordulásnak, azok egyenként is véges számú tulajdonságértékének és kapcsolat-előfordulásainak az adatmodell szerint szervezett együttese**.

- egy olyan integrált adatszerkezet, amely több különböző objektum előfordulási adatait adatmodell szerint szervezetten, tartósan tárolja olyan segédinformációkkal, metaadatokkal (jelentése: adat az adattól) együtt, melyek a hatékonyság, integritásőrzés, adatvédelem biztosítását szolgálják.
- a **káros és felesleges redundancia nélkül közösen tárolt, egymással kapcsolatban lévő adatok halmaza**, amelynek alapvető célja egy vagy több alkalmazás optimális kiszolgálása. Az **adatokat az azokat kezelő programoktól függetlenül tároljuk**, így az adatokat és az azokat használó programokat is megváltoztathatjuk anélkül, hogy a másikat módosítanánk.

Az adatbázisok elvileg **tetszőleges méretűek** lehetnek: az adatok száma a nullától (az üres adatbázistól) a végtelen értékig terjedhet. Az elméletileg **végtelen kapacitást** a gyakorlatban **a rendelkezésre álló hely**, vagy az **adatbázis tárolási szerkezete korlátozza**.

1.3 Az adatbázisok típusai

Különböző feladatok különböző típusú adatbázist igényelhetnek.

Az információ feldolgozására készített számítógépes programok az adatokat különböző strukturáltságban tárolják. Az adatok lehetnek **lazább szerkezetben** vagy **szigorúbb struktúrában** tárolva:

- A **szövegszerű tárolásnál** a dokumentumok, könyvek, cikkek alkotják a legkisebb elérési egységet. A dokumentum fejezetekre, oldalakra, bekezdésekre, mondatokra, szavakra tagolva, de a számítógép számára ömlesztve tartalmazza az információt.
- Az **adatszerű tárolásnál** sokkal kisebb adatelemek, ún. objektumok, egyedek tulajdonságai is elérhetők és kezelhetők.

Az adatbázisokat sokféleképpen csoportosíthatjuk. Néhány **adatbázis típus**:

Az **ellátott feladat** szerint:

- **Operációs adatbázis**: Az elektronikus tranzakció-feldolgozás során alkalmazzák, ahol az adatok begyűjtése, módosítása és karbantartása napi rendszerességgel történik. Az ilyen adatbázisban tárolt adatok dinamikusak, ami azt jelenti, hogy folyamatosan változnak és mindig a friss állapotot tükrözik.
- **Analitikus adatbázis**: Elektronikus elemzési feldolgozásra használják. Ilyenkor az a fontos, hogy az adatok időbeli változását nyomon lehessen követni. Ez akkor hasznos, ha statisztikákat kell készíteni, trendeket kell elemezni, (stratégiai) döntéseket kell hozni. Az ilyen adatbázisok statikus adatokat tartalmaznak, ami azt jelenti, hogy tartalmuk csak nagyon ritkán változik, sok esetben csak bővül.

Az **alkalmazott módszertan** szerint:

- **Relációs adatbázis**: A adatbázisok legelterjedtebb fajtája. A relációs adatbázisban az információ az adatokról és a közöttük lévő kapcsolatokról strukturáltan, táblázatos formában tárolódik. A relációs adatbázisokat jellemzően akkor használjuk, ha fontos az adatintegritás. Az ilyen adatbázisok leggyakrabban a szabványos SQL (Structured Query Language, strukturált lekérdezőnyelv) segítségével kezelhetőek.
- **NoSQL adatbázis**: Ez a fajta adatbázis a fájl-mappa rendszerhez hasonló hierarchiával rendelkezik, és a benne lévő adatok strukturálatlanok vagy nem relációsak. Ez a struktúra hiánya teszi lehetővé, hogy nagyobb mennyiségű adatot dolgozzanak fel gyorsan. A szerkezete megkönnyíti a jövőbeni bővítést. A felhőalapú adatbázisok soksor NoSQL-adatbázisok. Az elnevezés annyit jelent, hogy az adatok kezeléséhez, lekérdezéséhez nem (csak) SQL használható.
- **Gráfadatbázis**: Ezek olyan adatbázisok, amelyek az adatokra és a köztük lévő kapcsolatokra összpontosítanak. Speciális adatbázisok, melyek olyan komplex adatstruktúrákat tárolnak, melyeket lehetetlen lenne például relációs adatbázisokban tárolni. A gráfadatbázisokat olyan esetben ajánlatos használni, amikor az adatok szoros kapcsolatban állnak egymással és a kapcsolatok száma is igen magas.
- **Objektumorientált adatbázis**: Itt az egyes adatbáziselemek (objektumok) „tudják”, hogy kik ők, mire használhatók, és miként kapcsolódnak a többi adatbáziselemhez. Az objektumorientált adatbázisok az objektumorientált programozási nyelvek térhódításával terjedtek el. Fő jellemzőjük öröklődés, azaz az alacsonyabb szinteken lévő objektumokból (szülő) levezetett magasabb szintű objektumok (gyerek) öröklik a szülők tulajdonságait.

A **központosítottság mértéke** szerint:

- **Központosított adatbázis:** Teljes egészében egyetlen helyen működik, egy központi számítógépen vagy adatbázis-rendszeren található. A központi számítógép végez el minden, az adatok kezelésével kapcsolatos műveletet, a felhasználók számítógépes hálózaton keresztül érhetik el az adatbázist.
- **Elosztott adatbázis:** Ahelyett, hogy az összes adatot egyetlen eszközön tárolnák, az elosztott adatbázisok több gépen működnek, akár ugyanazon a helyen vagy egy hálózat különböző számítógépein. Az adatbázis-kezelő rendszer feladata az egységesség látszatának fenntartása, a különböző helyeken tárolt adatbázisok összehangolása.

Hozzáférhetőség szerint:

- **Kereskedelmi adatbázis:** Ide tartozik minden üzleti vállalkozás által tervezett, díjfizetés ellenében hozzáférhető adatbázis. Az ezzel foglalkozó vállalkozások funkciókban gazdag adatbázisokat fejlesztenek, amelyeket eladnak ügyfeleiknek. A kereskedelmi adatbázisok összetételük és az általuk használt technológia tekintetében sokfélék.

- **Ingyenes adatbázis**: A felhasználók díjfizetés nélkül használhatják. Bár ingyenesek, de hiányozhatnak belőlük egyes – a kereskedelmi adatbázisokban található – fejlettebb szolgáltatások.
- **Végfelhasználói, személyes adatbázis:** A végfelhasználó számítógépén lévő, sok esetben a felhasználó által készített, egyszerű – egy vagy néhány táblázatból álló – adatbázis, amit jellemzően egy személy használ. Általában nem alkalmasak rendkívül összetett műveletekre, vagy nagy mennyiségű adat kezelésére.

1.4 Az adatbázis-kezelő rendszer

Az **adatbázis-kezelő rendszer** egy **több programból álló szoftvertermék**, amely biztosítja az adatbázisban tárolt adatok létrehozását, kezelését, valamint leírja és kezeli az adatok közötti komplex kapcsolatokat. Az adatbázis-kezelő rendszernek **támogatnia kell valamilyen adatmodellt**, hogy a valóságot le tudja képezni egy számunkra megfogható objektumra.

Az adatbázis-kezelő rendszerek központi szerepet játszanak az adatbázisok kezelésében. Ezek a rendszerek lehetővé teszik az adatbázisok létrehozását és szerkezetük kialakítását, az adatok bevitelét, kezelését és visszakeresését.

Az adatbázis-kezelő rendszernek **gondoskodnia kell az integritási** (sértetlenségi), **hatékonysági** és **védelmi feltételek megőrzéséről**. Az adatbázis-kezelő rendszer egy bonyolult programrendszernek tekinthető, mely sok funkcióját, összetettségét tekintve **leginkább az operációs rendszerekhez** hasonlítható. **Az integritási, hatékonysági és védelmi feltételek ellenőrzését és betartatását az adatbázis-kezelő rendszer a háttérben végzi** el a felhasználó közvetlen parancsa, tudta nélkül.

A legelterjedtebb adatbázis-kezelő rendszerek közé tartozik az Oracle Database, a MySQL, a Microsoft SQL Server és a PostgreSQL.

1.5 Az adatbázis-kezelés fejlődése

Az adatbázis-kezelés története az 1960-as évekig nyúlik vissza, amikor az első hierarchikus és hálózati adatbázisokat kifejlesztették.

Az 1970-es években jelentek meg az első relációs adatbázisok, amelyeket Edgar Frank Ted Codd munkássága alapozott meg (1970-ben megjelent Relational Model of Data for Large Shared Data Banks című írásával). A relációs modell forradalmasította az adatbázis-kezelést, mivel egyszerűsítette az adatok tárolását és lekérdezését. **Még ma is a relációs adatbázisok, illetve az azokat kezelő relációs adatbázis-kezelő rendszerek a legelterjedtebbek, legnépszerűbbek**.

A 2000-es években a NoSQL adatbázisok megjelenése újabb mérföldkő volt, különösen a nagy mennyiségű és változatos adatok kezelésében. A NoSQL adatbázisok rugalmas adatmodelleket

kínálnak, amelyek jobban alkalmazkodnak a modern alkalmazások igényeihez, mint amilyen például a az adatbányászat és a valós idejű webes alkalmazások kiszolgálása.

Az utóbbi években a felhőalapú adatbázis-kezelés is egyre népszerűbb. A felhőalapú szolgáltatások a korábbi módszereknél is hatékonyabban oldják meg az adatbázisok skálázhatóságát és rugalmasságát, miközben csökkentik a hardveres és karbantartási költségeket.

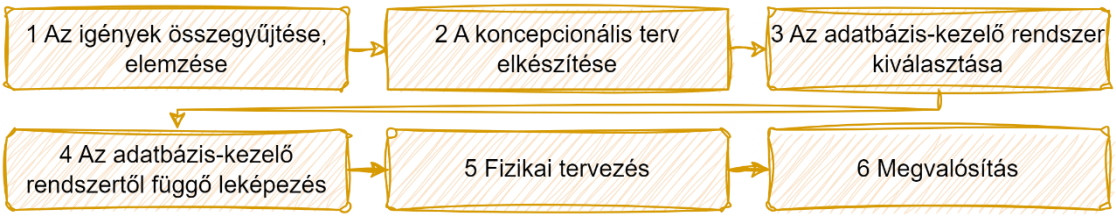
1.6 Az adatbázis-tervezés lépései

Egy adatbázis létrehozását – jó esetben – mindig az **adatbázis tervezése** előzi meg. Gondosan fel kell mérni az igényeket, meg kell fogalmazni a problémákat, elvárásokat, és fel kell tárnunk az adatok közötti összefüggéseket. Csak utána szabad megkezdeni az adatbázis tényleges, fizikai megvalósítását.

Az **alapos tervezés** azért is **fontos**, mert enélkül az elkészült rendszer nehezen átlátható, vagy akár átláthatatlan lesz, az utólagos módosítás pedig már nagyon körülményes és költséges lehet.

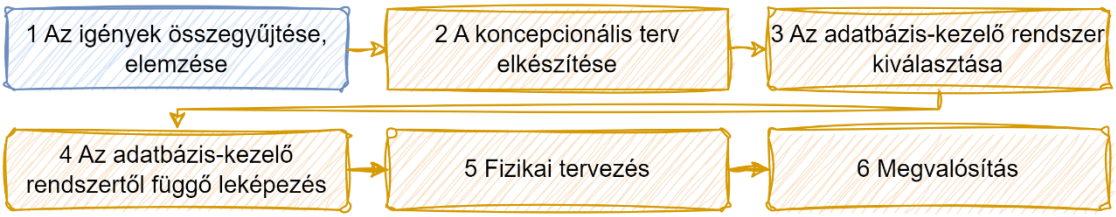
A **tervezés lépései**: (1-1. ábra)

1. Az igények összegyűjtése, elemzése;
2. A koncepcionális terv elkészítése;
3. Az adatbázis-kezelő rendszer kiválasztása;
4. Az adatbázis-kezelő rendszertől függő leképezés;
5. Fizikai tervezés;
6. Megvalósítás;



1-1. ábra. Az adatbázis-tervezés lépései. Forrás: Tímár et al. (1997) alapján a szerző szerkesztése.

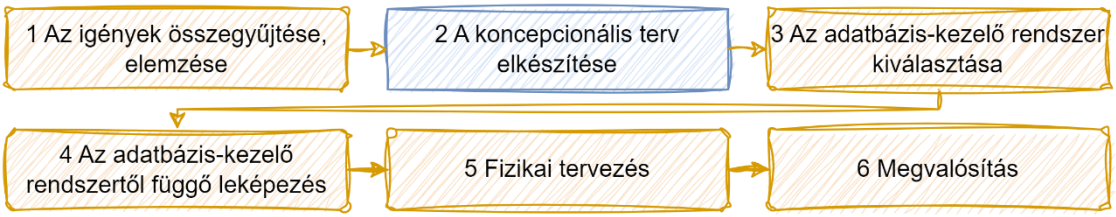
1. Az igények összegyűjtése, elemzése (a feladat specifikációja)



Ebben a szakaszban gondosan **fel kell deríteni a már meglévő alkalmazási területeket**. Tanulmányozni kell az adott területtel rokon, már meglévő alkalmazásokat és azok dokumentációit. **Meg kell vizsgálni a jelenlegi megvalósításokat** (még akár a nem számítógépes megoldásokat is), valamint azok alkalmazásának körülményit. A **felhasználói igények, elvárások összegyűjtése** érdekében célszerű a leendő felhasználókkal is elbeszélgetni.

Az adatok, információk összegyűjtése után olyan **specifikációt (leírást) kell készíteni**, amely tartalmazza a felhasználói igényeket kielégítő tárolandó adatokat és a szükséges feldolgozási műveleteket.

2. A koncepcionális terv elkészítése



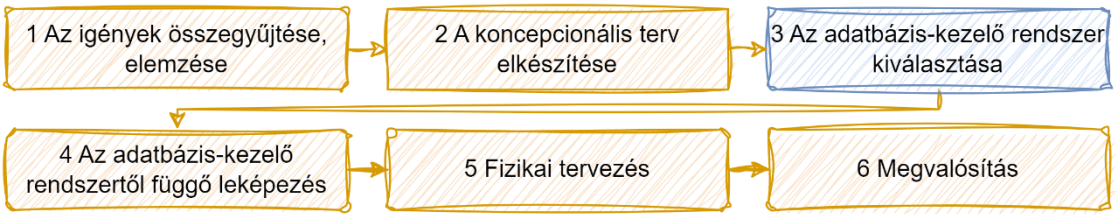
A koncepcionális terv készítése során kell **a magas szintű modellt kialakítani**. Ennek a tervnek az előnyei:

- olyan terv, leírás, amit az adatbázis-kezelő rendszer kiválasztásával, módosításával **nem kell megváltoztatni**,
- **könnyen érthető** formában mutatja az adatbázis szerkezetét, az adatcsoportokat és azok kapcsolatait, az esetleges korlátozásokat,
- a felhasználóknak és az adatbázis készítőjének is **újabb ötleteket adhat**,
- mivel könnyen megérthető, **segíti a** felhasználó és a programozó közötti **párbeszédet**.

A koncepcionális terv elkészítésére **kiválóan alkalmas az egyed-kapcsolat modell**, mivel (a) kifejező – az adatcsoportok (egyedtípusok) és tulajdonságaik mellett a kapcsolatok típusait is ábrázolja; (b) egyszerű – még a nem szakemberek is könnyen megérthetik; (c) kevés fogalmat használ – így ez az adatmodell rövid idő alatt megtanulható; (d) szemléletes ábrákkal dolgozik – könnyen áttekinthető, egyértelmű és némi gyakorlás után szinte nem lehet félreérteni.

Amennyiben hasonló igényeket fogalmaznak meg felhasználók egy-egy adatra, műveletre vonatkozóan, akkor **az igényeket össze kell fésülni**.

3. Az adatbázis-kezelő rendszer kiválasztása



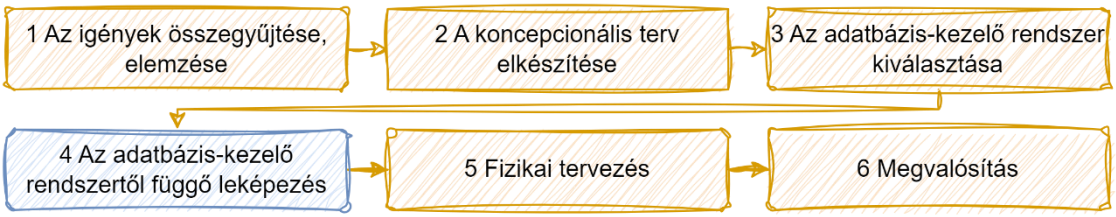
Az adatbázis-kezelő **rendszer kiválasztásában sok szempont** játszhat szerepet, így:

- a **feladat természete**: meghatározza, hogy milyen adatmodellre épülő rendszert célszerű használni; relációs, vagy esetleg más adatmodellre épülő adatbázis-kezelőt;
- **gazdaságossági megfontolások**: a hardver és szoftver beszerzési költségei, betanítási, működtetési, karbantartási és egyéb költségek;
- a **rendszer szolgáltatásai**, a felhasználóbarát felület, a hálózati működés, az optimálisan kezelhető adatmennyiség nagysága stb.

Egyszerűbb feladatok elvégzésére nem kell feltétlenül adatbázis-kezelő rendszert használni. Előfordulhat, hogy előnyösebb lehet egy táblázatkezelő, mint egy bonyolult – és az adott feladathoz mérten számos felesleges szolgáltatást biztosító – rendszer. **Adatbázis-kezelő rendszert akkor lehet érdemes használni**, ha:

- **több felhasználó** és program is használja ugyanazt az adatbázist,
- az **adatok gyorsan szaporodnak**, módosulnak,
- nagyon **sok adatot kell tárolni** az adatbázisban,
- a különböző adattípusok között **bonyolult kapcsolatrendszer** áll fenn és
- az adatellenőrzésre, **adatbiztonságra nagy az igény**.

4. Adatbázis-kezelő rendszertől függő leképezés



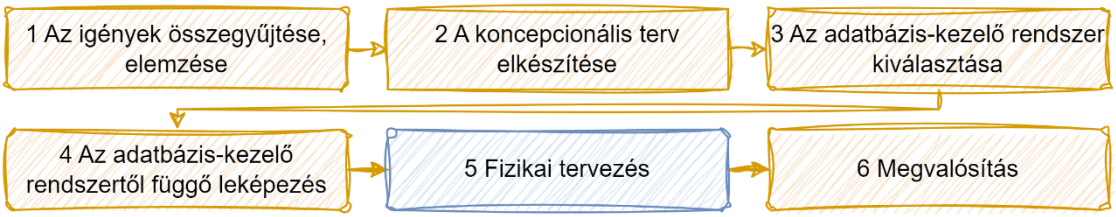
A leképezés **az adatmodelltől függő szabályok alkalmazása és konvertálása a kiválasztott adatbázis-kezelő által kezelt formára**. Például, ha a koncepcionális tervet egyed-kapcsolat modellben készítettük el, és relációs adatbázis-kezelőt kívánunk használni, akkor **ebben a szakaszban konvertáljuk át az egyed-kapcsolat modellünket relációs adatmodellre**, azaz **táblázatokká**.

A folyamat akár **automatizálható is** ún. CASE (Computer Aided Software Engineering) eszközök segítségével. Ezek az eszközök a magas szintű modellből kiindulva a kiválasztott

adatbázis-kezelő rendszer adatleíró nyelvén írják le az adatbázis szerkezetét. Ezt célszerű ellenőrizni!

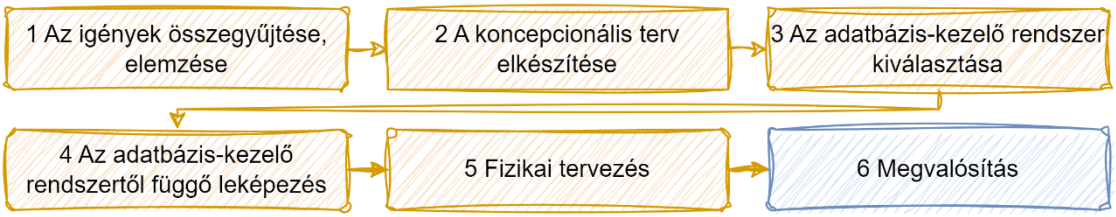
Ebben a fázisban történik a **lekérdezések megtervezése** is. Például relációs adatbázis-kezelő használata esetén a lekérdezések leírásához használhatunk relációs algebrát, vagy használhatunk SQL utasításokat.

5. Fizikai tervezés



Itt kell **dönteni a tárolási szerkezetről és a hozzáférési módokról**, amit az adatbázis-kezelő rendszer mellett az alkalmazott operációs rendszer is befolyásol. Döntő fontosságú lehet a lekérdező és aktualizáló – vagyis a beszúró, törlő és módosító – műveletek gyakorisága és egymáshoz való viszonya, az adatok gyors elérését biztosító, de ugyanakkor több tárhelyet foglaló megoldások alkalmazása stb.

6. Megvalósítás



Az előző fázisok végeredményeként **létrejön az üres szerkezetű adatbázis**. Ezt tölthetjük fel adatokkal.

A **programozók megírják a szükséges műveletek kódjait**, a felhasználói programokat, amelyek használhatják az adatbázis-kezelő nyelv parancsait. Korszerűbb rendszerek lehetővé teszik, hogy akár komolyabb **programozási ismeretek nélkül hozzunk létre programkódot** a grafikus felhasználói felületen – megtervezhetjük a menüket, képernyőket, űrlapokat, jelentéseket stb. egyetlen utasítás ismerete, begépelése nélkül (természetesen nem árt, ha ezeket is ismerjük).

A tényleges adatok betöltése előtt **célszerű a kész adatbázist mintaadatokkal tesztelni**, hogy ne túl későn derüljenek ki az esetleges hibák. Ezt már csak azért is célszerű megtenni, mert még ekkor is előfordulhat, hogy mind a felhasználó, mind a programozó olyan újabb lehetőségeket fedez fel, amire eddig nem is gondolt.

Az adatbázis megvalósítása után, **használat közben is felmerülhetnek még hibák**, problémák, amiket utólag javítani kell. Emellett **egyéb módosítási igények is jelentkezhetnek** – részben

az új felhasználói igények, részben a változó külső körülmények következtében –, amelyeket utólag szintén be kell építeni az adatbázisba.

1.7 Összefoglalás

Amikor egy adatbázist magunk elé képzelünk, leginkább táblázato(ka)t látunk tele adatokkal. Talán innen jöhet a felhasználóktól néha hallható „minek az adatbázis-kezelő, jó lesz a táblázatkezelő is…”.

Vannak esetek, amikor egyetlen táblázatban nem tárolhatók, nem kezelhetők hatékonyan az adatok, azok hatalmas tömege és a közöttük fennálló sokrétű és bonyolult kapcsolatrendszer miatt. Erre lettek kitalálva az adatbázis-kezelők.

Az adatbázis-kezelés alapvető szerepet játszik a modern információs rendszerekben. Az adatbázis-kezelés olyan technológiák, eszközök és módszerek összessége, amelyek lehetővé teszik az adatok strukturált tárolását, kezelését és elérését.

Az adatbázis…

- olyan adatállomány, amely egy adatbázis-kezelő rendszerrel hozható létre és érhető el.
- véges számú egyed-előfordulásnak, azok egyenként is véges számú tulajdonságértékének és kapcsolat-előfordulásainak az adatmodell szerint szervezett együttese.
- a káros és felesleges redundancia nélkül közösen tárolt, egymással kapcsolatban lévő adatok halmaza, amelynek alapvető célja egy vagy több alkalmazás optimális kiszolgálása. Az adatokat az azokat kezelő programoktól függetlenül tároljuk, így az adatokat és az azokat használó programokat is megváltoztathatjuk anélkül, hogy a másikat módosítanánk.

…. és még sorolhatnák. Az adatbázisra vonatkozóan nincs egységesen elfogadott definíció.

Az adatbázisok elvileg tetszőleges méretűek lehetnek: az adatok száma a nullától (az üres adatbázistól) a végtelen értékig terjedhet. Az elméletileg végtelen kapacitást a gyakorlatban a rendelkezésre álló hely, vagy az adatbázis tárolási szerkezete korlátozza.

Az adatbázisokat csoportosíthatjuk

- az ellátott feladat szerint;
- az alkalmazott módszertan szerint;
- a központosítottság mértéke szerint és
- hozzáférhetőség szerint;

Különböző feladatok különböző típusú adatbázist igényelhetnek.

Az adatbázis-kezelő rendszer egy több programból álló szoftvertermék. Ezzel hozzuk létre, kezeljük, használjuk az adatbázisokat. Az adatbázis-kezelő rendszernek mindig támogatnia kell valamilyen adatmodellt, hogy a valóságot le tudja képezni. Tulajdonképpen az adatbázis-kezelő rendszer egy bonyolult programrendszernek tekinthető, mely sok funkcióját, összetettségét tekintve leginkább az operációs rendszerekhez hasonlítható.

Egy adatbázis létrehozását – jó esetben – mindig az adatbázis tervezése előzi meg. Gondosan fel kell mérni az igényeket, meg kell fogalmazni a problémákat, elvárásokat, és fel kell tárnunk az adatok közötti összefüggéseket. Csak utána szabad megkezdeni az adatbázis tényleges, fizikai megvalósítását.

Az adatbázis-tervezés lépései:

1. Az igények összegyűjtése, elemzése;
2. A koncepcionális terv elkészítése;
3. Az adatbázis-kezelő rendszer kiválasztása;
4. Az adatbázis-kezelő rendszertől függő leképezés;
5. Fizikai tervezés;
6. Megvalósítás;

1.8 Ellenőrző kérdések

1. Milyen esetben célszerű táblázatkezelő helyett adatbázis-kezelőt használni?
2. Mi az adatbázis?
3. Mekkora lehet egy adatbázis?
4. Mit jelent az adatok lazább, ill. szigorúbb szerkezetű tárolása?
5. Ellátott feladat szerint hogyan csoportosíthatók az adatbázisok?
6. Alkalmazott módszertan szerint hogyan csoportosíthatók az adatbázisok?
7. A központosítottság mértéke szerint hogyan csoportosíthatók az adatbázisok?
8. Hozzáférhetőség szerint hogyan csoportosíthatók az adatbázisok?
9. Mi az adatbázis-kezelő rendszer és mi a feladata?
10. Melyek az adatbázis-kezelés fejlődésének fontosabb állomásai az 1960-as évektől napjainkig?
11. Melyek az adatbázis-tervezés lépései (felsorolás)?
12. Mit jelent az adatbázis-tervezés során az igények összegyűjtése, elemzése?
13. Mit jelent az adatbázis-tervezés során a koncepcionális terv elkészítése?
14. Mit jelent az adatbázis-tervezés során az adatbázis-kezelő rendszer kiválasztása?
15. Mit jelent az adatbázis-tervezés során az adatbázis-kezelő rendszertől függő leképezés?
16. Mit jelent az adatbázis-tervezés során a fizikai tervezés?
17. Mit jelent az adatbázis-tervezés során a megvalósítás?

2. lecke

Az adatmodell és az egyed-kapcsolat modell

Elvárt tanulási eredmények

Tudás

- Ismeri a modell és az adatmodell fogalmát
- Átfogóan ismeri az adatmodellek fő típusait
- Ismeri az adatmodellek általános elemeit
- Ismeri az adatmodellezés alapelveit
- Ismeri az egyed-kapcsolat modell fő jellemzőit
- Ismeri az egyed-kapcsolat modell elemkészletét

Képesség

- Helyesen alkalmazza az egyed-kapcsolat elemkészletének elemeit
- Képes lerajzolni egyed-kapcsolat diagramot

Attitűd

- Törekszik arra, hogy koncepcionális tervei áttekinthetőek legyenek

Autonómia, felelősség

- Önállóan készít egyed-kapcsolat diagramot

Miről lesz szó ebben a fejezetben?

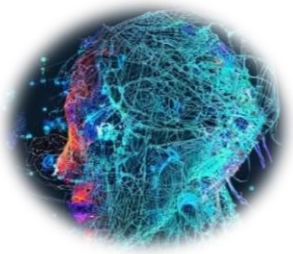
- Áttekintjük, hogy mi a modell és az adatmodell, mi a célja.
- Megbeszéljük az adatmodellek általános elemeit és az adatmodellezés általános elveit.
- Áttekintjük, hogy miért van szükség a tervezés során koncepcionális modell készítésére.
- Megbeszéljük az egyed-kapcsolat modell fő jellemzőit.
- Megtanuljuk az egyed-kapcsolat modell elemkészletének használatát.

2 Az adatmodell és az egyed-kapcsolat modell

A különböző **modellek alapvető szerepet játszanak** a környező **világ megértésében, leképzésében és alakításában**. A modellek teszik lehetővé a lényeg kiemelését és szemléltetését.

2.1 A modell fogalma

A **modell fogalma** alatt rendszerint két különböző dolgot szokás érteni:



- olyan rendszert, amely a **valóság egy vizsgált szeletével struktúrában vagy viselkedésben megegyezik**, vagy **hasonló jelleget mutat** és célja a vizsgálatán keresztül a valóság állapotára, viselkedésére vonatkozó következtetések levonása,
- modell kifejezéssel jelöljük azt az **eszközrendszert** is, amellyel az előző értelemben vett modell leírható, megadható.

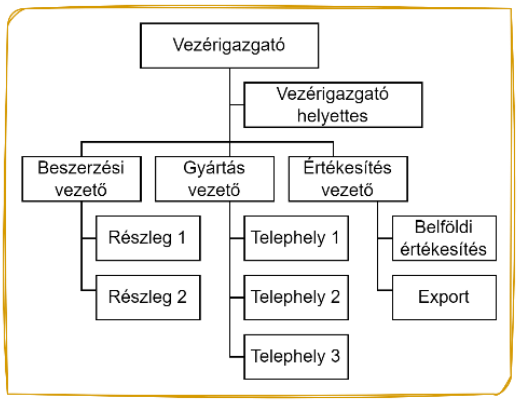
A modell tehát egyrészt egy **jelölésrendszert**, másrészt **egy elkészült leírást** is jelölhet.

2.2 Az adatmodell fogalma és célja

Azokat a modelleket, amelyek az **adatok struktúrájának leírására szolgálnak**, adatmodelleknek nevezzük. Az adatmodell központi szerepet játszik az adatbázis-kezelésben: Ezen keresztül adhatjuk meg a megvalósítandó rendszer leírását az adatbázis-kezelőnek. Ahhoz, hogy használni tudjuk az adatbázis-kezelőt, ismernünk kell a hozzá tartozó adatmodellt.

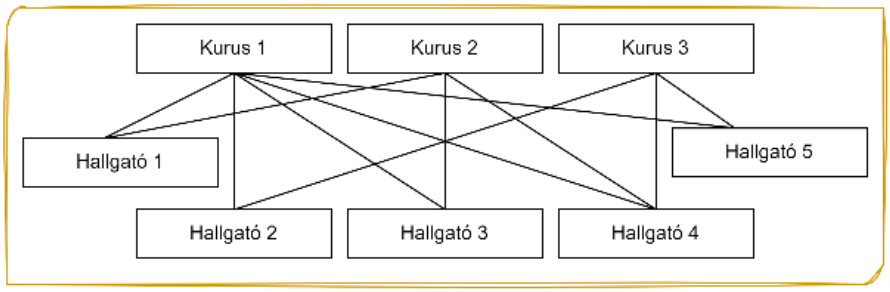
Az adatbázis-kezelés fejlődésével **újabb és újabb adatmodellek jöttek létre**, így létezik például hierarchikus, hálós, egyed-kapcsolat, relációs és objektum-orientált adatmodellek.

- A **hierarchikus adatmodellben** az adatok egy hierarchikus fastruktúrába szerveződnek. Minden elemnek egyetlen szülő eleme van, és egy elem több gyermek elemmel is rendelkezhet (2-1. ábra).



2-1. ábra. Hierarchikus adatmodell.

- A **hálós adatmodellben** egy elemnek több szülő eleme és több gyermek eleme is lehet, ami lehetővé teszi bonyolult kapcsolatok és összefüggések kezelését (2-2. ábra)

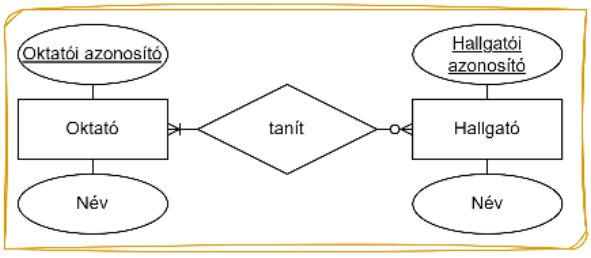


2-2. ábra. Hálós adatmodell.

- Az **objektum-orientált adatmodell** az objektum-orientált programozás elveit alkalmazza az adatbázisok tervezésében. A modellben fontos szerepet kap az öröklődés, azaz, egyes objektumok (adatok) más objektumoktól örökölhetnek tulajdonságokat (2-3. ábra).

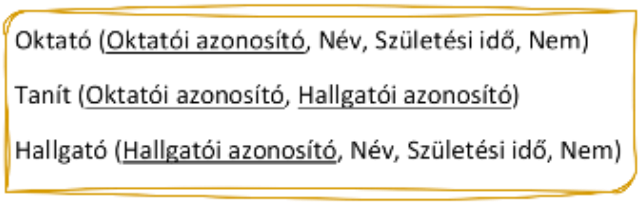
2-3. ábra. Objektum-orientált adatmodell.

- Az **egyed-kapcsolat modell** (angolul: Entity-Relationship model, ER-model) az adatbázisok (koncepcionális) tervezésének leginkább elterjedt, alapvető módszere, amelyben az adatokat egyedek és az azok közötti kapcsolatok formájában ábrázolják (2-4. ábra).



2-4. ábra. Egyed-kapcsolat modell.

- A **relációs adatmodell** az adatokat táblázatokban tárolja, és a táblázatok közötti kapcsolatokat ún. elsődleges és idegen kulcsokkal határozza meg. A ma alkalmazott legtöbb adatbázis és adatbázis-kezelő erre az adatmodellre épül (2-5. ábra).



2-5. ábra. Relációs modell tábláinak szerkezetét leíró séma

Az adatbázis-kezelő rendszerek egyik fő jellemzője, hogy mely adatmodellhez kapcsolódnak. Így például a relációs adatbázis-kezelő rendszer a relációs adatmodellen nyugszik.

Az adatmodell nem feltétlenül igényli egy adatbázis-kezelő rendszer létét. **Léteznek olyan adatmodellek, melyekhez nem létezik adatbázis-kezelő rendszer,** ilyen például az egyed-kapcsolat modell.

Az adatmodellezés tehát az adatállományok tervezésének korszerű módszere. Célja, hogy egy információs rendszer adatait és az adatok között fennálló kapcsolatokat következetesen ábrázolva, elősegítsük a számítógépes információ-feldolgozást.

2.3 Az adatmodellek általános elemei

Egy adatmodell a következő elemeket tartalmazza:

1. **Egyedek:** Ezek a valóság olyan dolgai, amelyek más dolgoktól megkülönböztethetők. Az egyed a valós világban létező, fogalmi vagy fizikai léttel rendelkező dolog, amelyet tulajdonságokkal akarunk leírni – személy, dolgozó, vásárló, áru, ingatlan, könyv, kurzus stb. Ugyanaz a dolog többféle egyednek is tekinthető – például egy személy lehet dolgozó és vásárló. A valamilyen szempontból összetartozó egyedek összessége egyedhalmazt (egyedtípust) alkot; így például a vásárló, mint egyedhalmaz az összes vevőt tartalmazza.
2. **Tulajdonságnak**, attribútumoknak nevezzük a valóság dolgainak azon jellemzőit, amelyekkel az egyes egyedeket leírjuk. Személyeknél tulajdonság lehet a név, születési hely és idő, nem, munkahely, beosztás stb. A név konkrét értékei lehetnek: Kovács János, Kiss Attila stb. Az egyes egyedek a tulajdonságértékeikkel azonosíthatók, például: Kovács János (név), Szeged (születési hely), 2002. április 8. (születési idő), férfi (nem), Könyvkiadó Kft. (munkahely), raktáros (beosztás).
3. **Kapcsolatnak** nevezzük az egyedek közötti viszonyok fogalmi tükörképeit. Egyedek közötti viszony lehet például az, hogy egy vásárló megvesz egy (vagy több) árut, egy hallgató részt vesz egy (vagy több) kurzuson, egy személy birtokol egy (vagy több) tárgyat. Ezeknek egy-egy konkrét előfordulása, ha Kovács János megveszi a cipőt, Nagy Anna részt vesz egy kurzuson, vagy Kiss Attila birtokolja az autót.

2.4 Az adatmodellezés alapelvei

Az adatbázis megtervezése során néhány **hasznos alapelvet** érdemes megfontolni, ezek:

1. **Valósághű modellezés**: Az egyedhalmazoknak, a tulajdonságoknak, valamint a kapcsolatoknak tükrözniük kell a valóságot.
   Példa: Nem lehet a Hallgató egyedeknek légköbméter vagy bőrszín tulajdonsága.
2. **Redundancia-mentesség** (vagy legalább arra való törekvés): Törekedni kell arra, hogy minden adat lehetőleg csak egyszer szerepeljen. Ezzel különböző – beviteli, tárolási, frissítési, és törlési – anomáliáknak nevezett, az adatbázis használatát és karbantartását nehezítő problémákat kerülhetünk el. Ugyanakkor előfordulhatnak olyan esetek, amikor célszerűségi okokból mégis rákényszerülhetünk a redundáns adattárolásra – ez az adott helyzettől, igényektől, adatbáziskezelő-rendszertől függ.
   Példa: A hallgatókat és kurzusaikat tartalmazó adatbázisban hallgatók nevét – mint tulajdonságot – ne tároljuk el a Hallgató és egyúttal a Kurzus egyedhalmazban is. Ugyanakkor nem árt, ha van biztonsági másolatunk.
3. **Egyszerűség**: Ne vegyünk fel több egyedhalmazt, tulajdonságot és kapcsolatot annál, mint amennyi feltétlenül szükséges.
   Példa: Hallgató esetén felesleges felvenni a szemszín attribútumot (bár kétségtelenül érdekes lenne annak megvizsgálása, hogy a hallgatói populációban leggyakrabban előforduló barna színhez képest van-e eltérés a tanulmányi eredményekben és ha igen, az milyen mértékű).
4. **Megfelelő kapcsolatok kiválasztása**: Az egyedhalmazokat – kiváltképpen, ha sok van belőlük – sokféleképpen kapcsolhatjuk össze. Általában nem célszerű az összes kapcsolatot létrehozni, mert az – beszúrási, módosítási, törlési – anomáliákhoz vezethet és csak megnehezíti (vagy akár el is lehetetlenítheti) az adatbázis kezelését. Azokat a kapcsolatokat célszerű feltűntetni, amelyek az adott feladat megoldása szempontjából szükségesek. Ehhez tudnunk kell előre, hogy mit várunk el az adatbázistól.
   Példa: Egy szerzők és cikkeik adatait tároló adatbázisban a Szerző és a Cikk között például több–a–többhöz kapcsolatot célszerű kialakítani, hiszen egy Szerző akár több tudományos cikket is írhat, illetve egy cikknek több társszerzője is lehet, de ugyanakkor a Folyóirat és a Cikk között csak egy–a–többhöz kapcsolat lehet, hiszen egy adott publikáció csak egyszer jelenhet meg a folyóiratban, másrészt egy folyóiratszám több cikket is tartalmazhat.
5. A **megfelelő típusú elem megválasztása**: Előfordulhat, hogy a valóság egy szeletét leírhatjuk egyedhalmazzal, tulajdonsággal (attribútummal) vagy kapcsolattal is. Általában jellemző, hogy attribútumokat könnyebb a modellbe bevinni, mint egyedhalmazt, vagy kapcsolatot. Ugyanakkor, mindent attribútummal megoldani nem célszerű, mivel a kész adatbázis kezelését nehézkessé teheti.
   Példa: A hallgatók tulajdonságai mellé (hallgatói azonosító, név, szak stb.) betehetjük tulajdonságként az általuk felvett kurzus tulajdonságait (kurzuskód, kurzusnév stb.),

sőt még hozzáírhatjuk tulajdonságként az oktató adatait is (oktatói azonosító, név, beosztás stb.), de így számos problémával kell megküzdenünk, ha aktualizálni szeretnénk akár a hallgatók, akár a kurzusok, akár az oktatók adatait. Célszerű ezért ezeket három különböző egyedhalmazként kezelni és közöttük kapcsolatokat kialakítani.

2.5 Miért van szükség a koncepcionális adatmodellre?

Az adatbázis megtervezése során **először egy elvont, „magas szintű" ún. koncepcionális modellt** készítünk, majd **ez alapján készül el** egy, a megvalósításhoz kiválasztott adatbázis-kezelő rendszerhez közelebb álló **logikai modell** és csak ezt követi a tényleges, fizikai megvalósítás.

A **koncepcionális adatmodell** olyan eszközök gyűjteménye, amellyel **leképezhetjük a valóság számunkra fontos részét** úgy, hogy az képes lesz a számunkra fontos kérdések megválaszolására. Éppen ezért, a koncepcionális adatmodell létrehozása elméletileg is és gyakorlatilag is jelentős szerepet tölt be az adatbázis-tervezési folyamatban. Ez a modell **még nem foglalkozik az adatok szerkezetével és a fizikai tárolás problémáival**.

A koncepcionális sémák **tartalmazzák az egyedek halmazát, azok jellemzőit, a kapcsolatokat és az esetleges korlátozásokat**. Emellett szemléletes módon, ábrákkal mutatják be az adatbázis logikai felépítését; nem hozzáértő számára is **érthető formában szemléltetik az adatbázist**.

Természetesen, mint mindent, ezt is **el lehet rontani**; a hibás tervezéshez hozzájárul a modellező eszközök nem megfelelő alkalmazása, a felhasználók és az adatbázis-tervezők képzelőerejének korlátai, a felhasználók és az adatbázis-készítő szakemberek eltérő látásmódja és problémamegközelítése. Bár a modell egyszerű – az fontos, hogy a modellező megtanulja és rendelkezzen azzal a képességgel, ami biztosítja számára a szükséges objektumok és kapcsolataik helyes azonosítását.

A modellezés során a **leggyakrabban elkövetett hibák**: A modell készítői

1. **nem képesek az egyedeket általánosítani** és egyedhalmazokat alkotni belőle,
2. **tulajdonságként szerepeltetnek egy jellemzőt**, holott azt szerencsésebb lenne egyed(halmaz)ként felvenni a modellbe,
3. **nem ellenőrzik** az egyedhalmazok közötti **kapcsolatokat az összes lehetséges irányból**, így a kardinalitás (vagyis az összekapcsolódó egyedek számossága) megállapítása téves,
4. esetleges **korlátozásokat figyelmen kívül hagynak**.

2.6 Az egyed-kapcsolat modell

Peter Pin-Shan Chen 29 évesen korában megjelentett „The Entity-Relationship Model – Toward a Unified View of Data” című cikkével megalkotta az egyed-kapcsolat (Entity-Relationship, röviden ER) modell alapjait és ezzel jelentős, máig tartó hatást gyakorolt az adatmodellezésre.

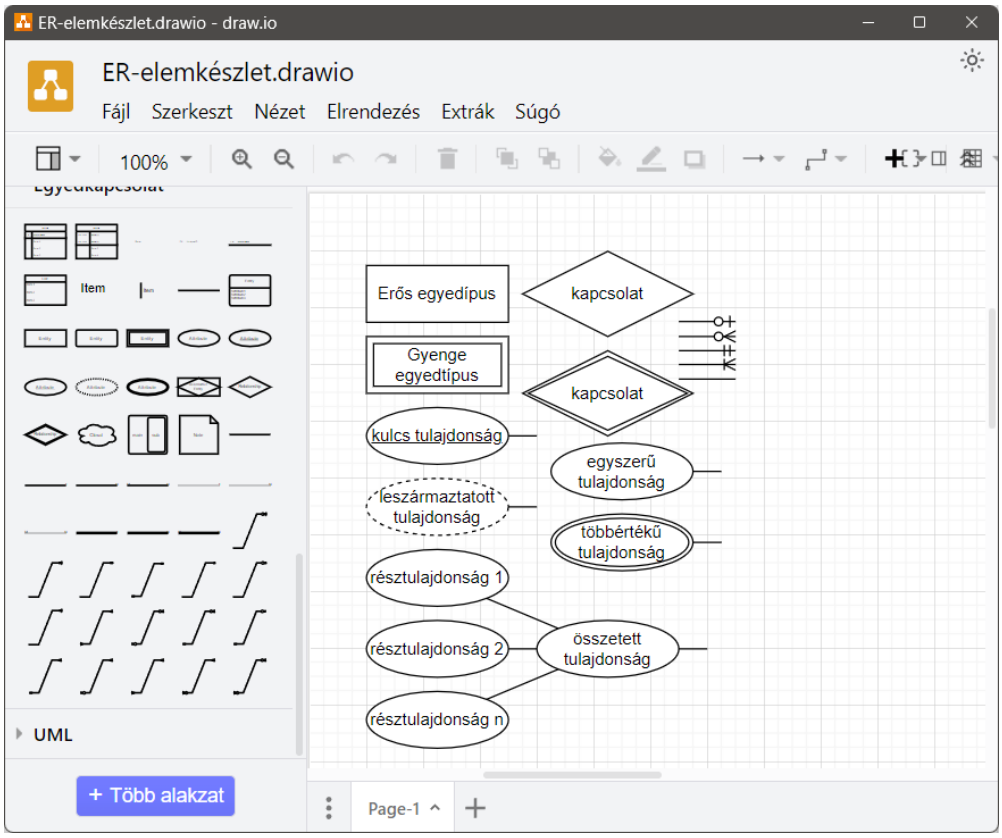
Az ER-modell a koncepcionális adatmodellezés legnépszerűbb eszköze. Népszerűsége az egyszerűség mellett az elméleti megalapozottságnak és annak is köszönhető, hogy a – mai adatbáziskezelő rendszerek által használt – relációs adatmodell előkészítésére kiválóan alkalmas.

A modellben az adatok szerkezetének ábrázolására ún. **egyed-kapcsolat diagramot (Entity-Relationship Diagram, ERD)** használunk, amely grafikus jelölésrendszert alkalmaz. Az ERD e**gy gráf**, ahol a csúcspontok a modell alapvető elemei (egyedhalmaz(ok), attributum(ok), azaz tulajdonságok és kapcsolatok). Az egyedhalmazokat az attribútumaikkal és a kapcsolataikkal vonalak segítségével kötjük össze, ezek a gráf élei.

Az ER-modell **kifejező, egyszerű** – nem szakemberek is viszonylag könnyen megérthetik –, kevés fogalmat használ – így **gyorsan megtanulható** –, **szemléletes ábrákkal dolgozik**.

Ennek a tananyagnak az ábrái az ingyenes Diagrams.net (korábban draw.io) diagramkészítő alkalmazással készültek (2-6. ábra). A program némi gyakorlással könnyen elsajátítható akár autodidakta módon is. Weboldala: https://www.drawio.com/. Akár itt is rajzolhatunk böngészőben, de offline is dolgozhatunk, ha telepítjük a programot a gépünkre. Windows, macOS és Linux operációs rendszerrel kompatibilis változat letölthető innen: https://get.diagrams.net. Itt találhatnak egy rövid, angol nyelvű ismertetőt a program használatáról: https://www.youtube.com/watch?v=bN6i6dsoZTs. A diagramok elkészítéséhez szükséges elemeket tartalmazó fájlt a tananyag csatolt fájlként tartalmazza. Szintén letölthető a tananyagban található összes diagram.

Természetesen nincs feltétlenül szükség számítógépre a diagramok elkészítéséhez, akár papírra is lerajzolhatjuk diagramokat.



2-6. ábra. A draw.io felhasználói felülete. Forrás: a szerző szerkesztése.

A **modell három alapvető eleme**:

1. egyedtípus (az egyedek halmaza, egyedhalmaz)
2. attribútum (az egyedek tulajdonságai)
3. kapcsolat (az egyedhalmazok közötti kapcsolatok)

2.7 Egyedtípus

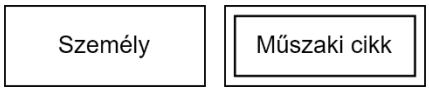
Az egyedtípus egy, a **külvilág többi részétől egyértelműen megkülönböztethető** dolgot, **objektumot** jelöl.

Az **egyedtípusok fajtái**:

- **Erős egyedtípus**: Önmagában is létezhet. Benne az egyes **egyedek egyértelműen azonosíthatók**, mivel azoknak van azonosító jellegű tulajdonsága (ún. kulcs tulajdonsága). Jele egy **téglalap**, amelynek **belsejében az egyedtípus azonosító neve** (főnév) áll, lásd: 2-7. ábra bal oldala.
  Példa: A személy egyedtípus erős, ha rendelkezik személyi igazolvány szám tulajdonsággal.
- **Gyenge egyedtípus**: Ez az egyedtípus **nem létezhet önmagában**, csak egy másik (erős) egyedtípussal (szülővel) együtt, azzal kapcsolatban állva. Jele egy **kettős téglalap**, melynek **belsejében az egyedtípus azonosító neve** (főnév), lásd: 2-7. ábra jobb oldala.

Egy-egy egyed beazonosításához szükség van az erős egyedtípus kulcs tulajdonságára is, mivel itt az egyes egyedeket a tulajdonságok nem határozzák meg egyértelműen, azok csak a kapcsolataik révén lesznek meghatározottak. Általában rendelkezik ún. parciális kulccsal, ami a szülőn belül azonosítja a gyenge egyedet. A szülő kulcs tulajdonsága és a gyenge egyedtípus parciális kulcsa már alkalmas a gyenge egyed azonosítására.

Példa: Egy szerviz adatbázisában az egyik egyedtípusunk személyek, a másik műszaki cikkek adatait tartalmazza. A műszaki cikkeket (gyenge egyedtípus) a műszaki cikk nevével, valamint a személy egyedtípusban (erős egyedtípus) található tulajdonos nevével azonosítjuk.



2-7. ábra. Erős egyedtípus (bal oldal) és gyenge egyedtípus (jobb oldal).

Minden egyedtípusnak kell legyen legalább egy tulajdonsága!

2.8 Attribútum (tulajdonság)

Az attribútum (vagy tulajdonság) az **egyedek egy meghatározott jellemzője**. Az attribútum, vagy inkább attribútumcsoport azonosítja az egyes egyedeket. **Ellipszisben** adjuk meg, **közepébe írva a tulajdonság azonosító nevét** (főnév). A tulajdonság **önmagában nem állhat**, ezért mindig meg kell adni, hogy mely egyedtípushoz (vagy kapcsolathoz) kötődik. A **kapcsolódást egy vonallal** jelöljük, ami a megfelelő tulajdonságot és az egyedtípust köti össze.

Példa: A személy egyedtípusnál tulajdonság lehet a személyi szám, név, születési idő, lakcím, végzettség, magasság; az autó egyedtípusnál tulajdonság lehet a rendszám, gyártó, típus, szín, ár.

A tulajdonságok **fajtái**:

- **Egyszerű** tulajdonság: Egy **elemi értékkel leírható** tulajdonságot ad meg. Az ellipszist ebben az esetben folytonos vonallal rajzoljuk meg. Lásd: 2-8. ábra.
  Példa: Az ember magassága egyszerű tulajdonság, hiszen egy skalár szám elegendő a megadásához.
- **Kulcs** tulajdonság: Egy-egy **egyed egyértelmű azonosítására** szolgáló tulajdonság az egyedhalmazban. Az egyedi azonosításra alkalmas tulajdonságok közül egyet kiválasztunk, ez lesz a kulcs tulajdonság és ezt úgy jelöljük, hogy a tulajdonság azonosító nevét **folytonos vonallal aláhúzzuk**. Lásd: 2-8. ábra. Ha nincs olyan tulajdonság, amely önmagában kulcs, akkor meg kell nézni, hogy két vagy több tulajdonság együtt alkothat-e kulcsot. Ilyen esetben összetett kulcsról beszélünk és valamennyi alkotóelemét aláhúzzuk.

Példa: A személy egyedtípusban a személyi szám, személyi igazolvány szám, TAJ szám, útlevélszám töltheti be ezt a szerepet. Ha az egyiket ezek közül kiválasztjuk és aláhúzzuk, onnantól kezdve az lesz a kulcs tulajdonság. Ha a személy egyedtípusban rendelkezésre áll a név, születési idő és anyja neve, akkor ezek együtt is alkothatnak kulcsot, ha nincs egyszerűbb megoldás.

- **Összetett** tulajdonság: Olyan tulajdonság, amely **több egyszerű tulajdonságra bontható**. Hogy egy ilyen résztulajdonságokra felbontható tulajdonságot felbontunk-e vagy egyszerű tulajdonságként kezeljük, az adatbázissal szembeni elvárásoktól függ. Az összetett tulajdonságot is **ellipszissel** jelöljük, amelyhez **hozzákötjük az illeszkedő résztulajdonságokat**. Egy összetett tulajdonságnak két vagy több résztulajdonsága kell legyen, hiszen ellenkező esetben nem lenne összetett. Lásd: 2-8. ábra.

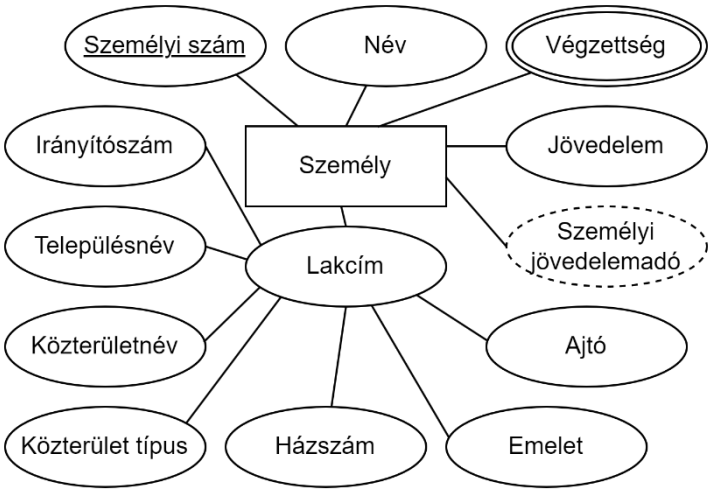
  Példa: A lakcím felbontható az irányítószám, település, közterületnév, közterület típus, házszám, emelet, ajtó résztulajdonságok együttesére (6724 Szeged, Mars tér 7., 1, emelet 216. ajtó). De akár a személyek neve is felbontható előtag, vezetéknév, utónév, utótag részekre (Dr. Kiss Edit Éva PhD).

- **Többértékű** tulajdonság: Olyan tulajdonság, amely nem csak egy elemi értéket, hanem több elemi értéket, **értékek egy tömbjét veheti fel**. A többértékű tulajdonságot egy **dupla keretű ellipszissel** jelöljük. Lásd: 2-8. ábra

  Példa: A személy egyedtípusban egy-egy személy végzettség tulajdonságának leírására több elemi értéket is meg lehet adni, hiszen előfordulhat, hogy valaki akár többféle végzettséggel is rendelkezik (Dr. habil. Gaál Jenő, közgazdász, logisztikai szakmérnök, fogászati asszisztens).

- **Leszármaztatott tulajdonság**: Olyan tulajdonság, amelynek értéke **más tulajdonságokból levezethető, kiszámolható**. A leszármaztatott tulajdonság jele egy **szaggatott vonallal határolt ellipszis**. Lásd: 2-8. ábra. Azt nem jelöljük, hogy mely más tulajdonságokból és mi módon származtatható az adott érték (a modell jelölésrendszere ezt nem támogatja), de a tulajdonságnak, amiből „kiszámoljuk” a leszármaztatott tulajdonságot, is szerepelnie kell a modellben.

  Példa: Egy személy személyi jövedelemadója (leszármaztatott tulajdonság) kiszámolható a jövedelem nagyságából (egyszerű tulajdonság) és az SZJA-kulcsból. Egy áru bruttó ára (leszármaztatott tulajdonság) meghatározható a nettó ár (egyszerű tulajdonság) és az ÁFA-kulcs ismeretében.



2-8. ábra. Erős egyed tulajdonságokkal. A személy normál egyedtípus a következő attribútumokkal rendelkezik: Személyi szám (kulcs tulajdonság), Név és jövedelem (egyszerű tulajdonság), Végzettség (többértékű tulajdonság), Lakcím (összetett tulajdonság, hozzá kapcsolva a résztulajdonságok: Irányítószám, Településnév, Közterületnév, Közterület típus, Házszám, Emelet, Ajtó), Személyi jövedelemadó (leszármaztatott tulajdonság).

2.9 Kapcsolat

A **kapcsolat** az egyedek között fennálló viszonyt mutatja. Jelölése **rombusszal** történik, amibe **beleírjuk a kapcsolatot leíró azonosító nevét** (ige). Az erős egyedtípusok közötti kapcsolat jelölésére folytonos vonalat, míg az erős és gyenge egyedtípus között kapcsolat létrehozásához kettős rombuszt használunk. A rombusz átellenes csúcsaiból **egy-egy vonalat húzunk az összekapcsolt egyedtípusokhoz**, amelyen **jelölhetjük a kapcsolat típusát** (a jelölésre többféle elterjedt megoldás létezik).

A legtöbb kapcsolatban két egyedtípus vesz részt, ez a **bináris kapcsolat**. Emellett létezhetnek kapcsolatok kettőnél több egyedtípus között is, ilyenkor **n-ed fokú kapcsolatról** beszélünk (ahol az n az egyedtípusok darabszáma), sőt olyan is előfordul, amikor egy egyedtípuson belüli egyedek között áll fenn, ez pedig a rekurzív (visszaható) **kapcsolat**.

A **kardinalitás a kapcsolatok számossága.** Meghatározza, hogy egy egyed hány másik egyedhez kapcsolódhat. Típusai: rekurzív, egy-az-egyhez, egy-a-többhöz, több-a-többhöz és n-ed fokú.

Megjegyzés: a következő ábrákon a kapcsolatra koncentrálunk, ezért a jobb áttekinthetőség kedvéért az egyedtípusok tulajdonságait nem jelöltük.

A kapcsolatban **két egyedtípus** vesz részt (bináris kapcsolat):

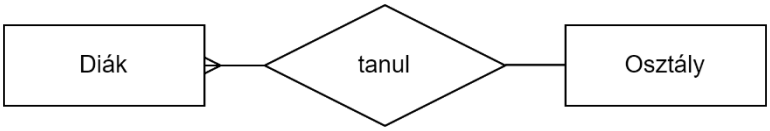
- **1:1 (egy-az-egyhez) kapcsolat:** A kapcsolatban **mindkét egyedtípus egyed-előfordulásai csak egyetlenegy egyed-előforduláshoz rendelődnek** a másik egyedtípusból. Lásd: 2-9. ábra.

Példa: A férfiak és nők külön egyedhalmazt alkotnak és egy férfi csak egy nőnek lehet a férje és egy nő is csak egy férfinak lehet a felesége.

2-9. ábra. 1:1 kapcsolat.

- **1:N (egy-a-többhöz) kapcsolat**: Abban különbözik az 1:1 kapcsolattípustól, **hogy az egyik egyedtípus előfordulásai több előfordulással tarthatnak kapcsolatot a másik egyedtípusból**, de a másik egyedtípus előfordulásai továbbra is csak egy előforduláshoz kapcsolódhatnak. Az 1:N kapcsolat ábrázolásánál azon egyedbe, **amelyből több is kapcsolódhat a másik egyedhez, egy „varjúláb”** vezet. Lásd: 2-10. ábra.

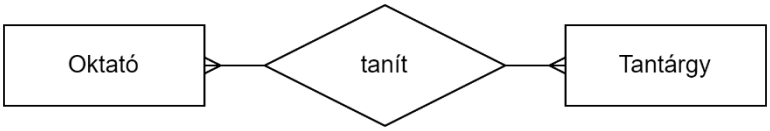
  Példa: Egy általános iskola tanulója egyszerre csak egy osztályba járhat, egy osztályban viszont több diák is tanulhat.



2-10. ábra. 1:N kapcsolat.

- **M:N (több-a-többhöz) kapcsolat**: Olyan kapcsolattípus, amelyben **mindkét egyedtípus egyed-előfordulásai több egyed-előfordulással is tarthatják a kapcsolatot** a másik egyedtípusból. A kapcsolat ábrázolásánál mindkét kapcsolódó egyedhalmaz felé „varjúláb” mutat. Lásd: 2-11. ábra.

  Példa: Egy oktató több tantárgyat taníthat és egy tantárgy oktatásában több oktató is részt vehet.

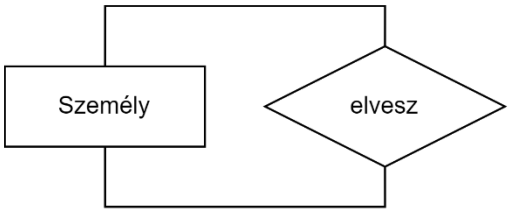


2-11. ábra. M:N kapcsolat.

A kapcsolatban **egyetlen egyedtípus** vesz részt:

- **Rekurzív** (visszamutató) kapcsolat: Egy egyedtípus egyed-előfordulásai **saját egyedtípusuk egyed-előfordulásaihoz** kapcsolódnak. A kapcsolat számossága itt is lehet 1:1, 1:N és M:N. Lásd: 2-12. ábra.

  Példa: Ha a férfiak és nők is a személy egyedtípushoz tartoznak, a közöttük rekurzív kapcsolatot alakíthatunk ki.

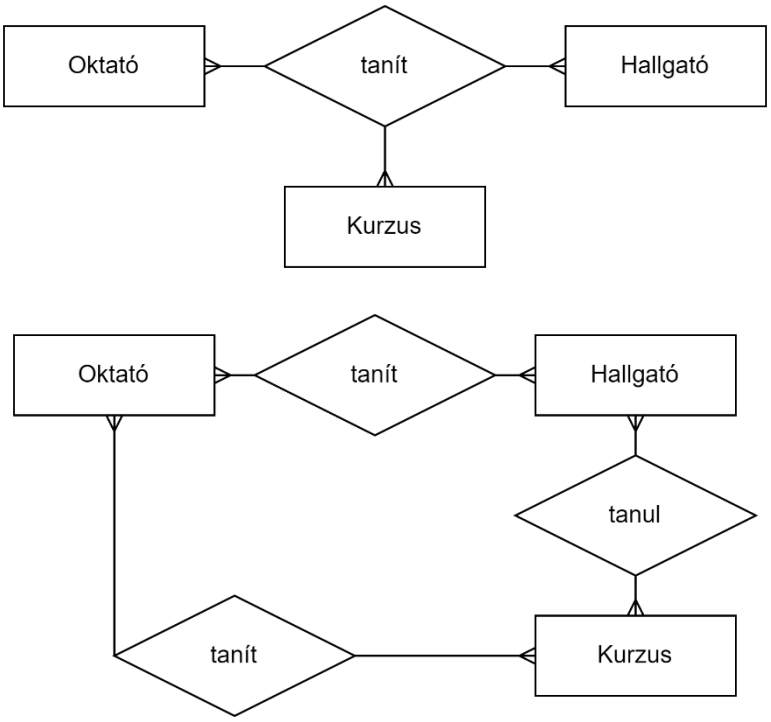


2-12. ábra. Rekurzív 1:1 kapcsolat.

A kapcsolatban **kettőnél több egyedtípus** vesz részt:

- **N-ed fokú kapcsolat**: A kapcsolatban **kettőnél több egyedtípus vesz részt**. Akárcsak a bináris és a rekurzív kapcsolat esetén, a kapcsolat számossága lehet 1:1, 1:N és M:N (illetve ezek kombinációja). Szükség esetén felbontható több bináris kapcsolatra. Lásd: 2-13. ábra.

  Példa: A tanul kapcsolatban a hallgató, a kurzus és az oktató kapcsolódik össze (3-ad fokú kapcsolat) minden irányban M:N kapcsolattal. Egy hallgató több kurzusra is járhat. Egy kurzusra több hallgató járhat. Egy kurzuson több oktató is oktathat. Egy oktató több kurzuson is oktathat. Egy hallgatót több oktató is taníthat. Egy oktató több hallgatót is oktathat.



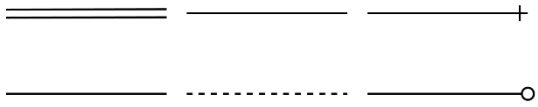
2-13. ábra. 3-ad fokú kapcsolat (fent) és bináris kapcsolatokra felbontva (lent)

Attól függően, hogy az egyedhalmazok egyedeinek mindegyike részt vesz-e kapcsolatban, a kapcsolat lehet:

- **Totális (teljes) kapcsolat**: Egy egyedtípus totálisan vesz részt a kapcsolatban, ha **minden egyed-előfordulása részt vesz egy kapcsolat előfordulásban**, azaz nincs olyan

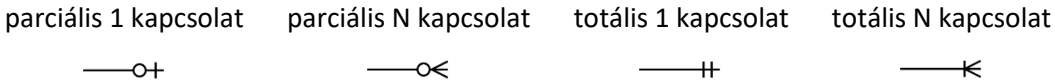
egyed-előfordulás, mely nem kapcsolódna egy másik egyedtípus valamely előfordulásához. Jelölése többféleképpen történhet: folytonos vonallal, kettős vonallal vagy a vonal végén arra merőleges szakasszal (2-14. ábra).

- **Parciális** (részleges, opcionális) **kapcsolat**: Azok az egyedtípusok, amelyek nem totálisan vesznek részt a kapcsolatban, parciális kapcsolatot alkotnak. **Ilyenkor van olyan egyed-előfordulás, amely nem kapcsolódok egy másik egyedtípus valamely előfordulásához**. Jelölése többféleképpen történhet: szaggatott vonallal, folytonos vonallal, a vonal végén egy 0-val (2-14. ábra).



2-14. ábra. A totális kapcsolat (fent) és a parciális kapcsolat (lent) jelölésének elterjedt módjai.

Ennek megfelelően a kapcsolatok számossága kiegészíthető a részvétel jelölésével is a 2-15. ábra szerint. A továbbiakban ezt a jelölést fogjuk alkalmazni.



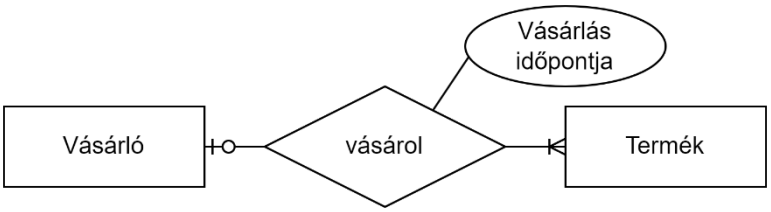
2-15. ábra. Parciális és totális 1 és N kapcsolatok jelölése.

Példa: Ha minden hallgatónak fel kell vennie legalább egy kurzust (egyébként nem kerülhet az adatbázisba), akkor a hallgató egyedtípus totális kapcsolatban van a kurzus egyedtípussal a felvesz kapcsolatban (2-16. ábra). Ha van olyan kurzus, amit nem vesz fel egyetlen hallgató sem, akkor a kurzus egyedtípus parciális kapcsolatban van a hallgató egyedtípussal a felvesz kapcsolatban.

2-16. ábra. Egyedtípusok totális és parciális kapcsolatban

Tulajdonságot nem csak egyedtípushoz, hanem **kapcsolathoz is hozzárendelhetünk** (2-17. ábra).

Példa: Egy vásárló legalább egy, vagy több terméket vásárol. Lehetnek olyan termékek, amelyeket még senki sem vásárolt meg és 1 terméknek legfeljebb 1 vásárlója lehet. A vásárlás kapcsolathoz hozzárendeltük a vásárlás időpontja attribútumot.

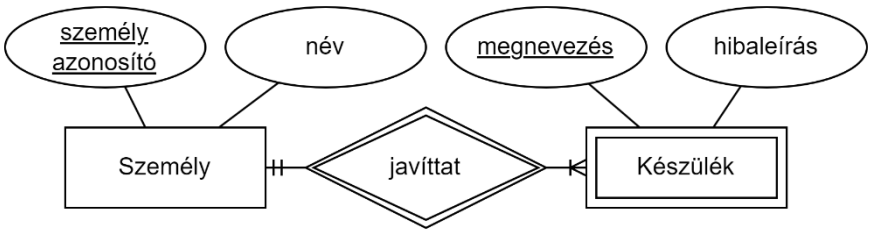


2-17. ábra. Kapcsolat tulajdonsággal.

2.10 A gyenge egyedtípus

A kapcsolatok megbeszélése után vissza kell térnünk a **gyenge egyedtípushoz**. Amint már volt szó róla, a gyenge egyedtípus csak egy másik, erős egyedtípussal együtt, ahhoz kapcsolódva létezhet, mivel a gyenge egyedek egyértelmű megkülönböztetéséhez nélkülözhetetlen az erős egyedtípus kulcsa. Az **erős és gyenge egyedtípus között** tehát **kapcsolatot kell létrehoznunk**, amit egy **kettős falú** (vagy szaggatott vonalú) **rombusszal jelölünk**. A közöttük lévő kapcsolat típusa megegyezik az előbb tárgyalt lehetőségekkel (2-18. ábra).

Példa: Egy készülék csak a megnevezésével (parciális kulcs) és hozzá tartozó személy azonosítójával (kulcs az erős egyedtípusban) azonosítható egyértelműen.



2-18. ábra. Gyenge egyedtípus az egyed-kapcsolat modellben.

Az egyed-kapcsolat **diagramok ábrázolásának többféle variánsa él egymás mellett**. A modellnek születtek bővített változatai is (Enhanced ER-model, EER-modell, kiterjesztett egyed-kapcsolat modell), amelyek **további lehetőségekkel egészítik ki a modell** valóságot leíró **képességét**, így például bevezetnek fő- és alosztályokat, az általánosítás és specializáció fogalmakat stb.

2.11 Összefoglalás

A különböző modellek alapvető szerepet játszanak a környező világ megértésében, leképzésében és alakításában.

Azokat a modelleket, amelyek az adatok struktúrájának leírására szolgálnak, adatmodelleknek nevezzük. Az adatmodell központi szerepet játszik az adatbázis-kezelésben. Ahhoz, hogy használni

tudjuk az adatbázis-kezelőt, ismernünk kell a hozzá tartozó adatmodellt.

Az adatbázis-kezelés fejlődésével újabb és újabb adatmodellek jöttek létre, így létezik például hierarchikus, hálós, egyed-kapcsolat, relációs és objektum-orientált adatmodellek.

Az adatmodellezés az adatállományok tervezésének korszerű módszere. Célja, hogy egy információs rendszer adatait és az adatok között fennálló kapcsolatokat következetesen ábrázolva, elősegítsük a számítógépes információ-feldolgozást.

Egy adatmodell a következő elemeket tartalmazza: (1) Egyedek a valóság olyan dolgai, amelyek más dolgoktól megkülönböztethetők. (2) Tulajdonságok (attribútumok) a valóság dolgainak azon jellemzői, amelyekkel az egyes egyedeket leírjuk. (3) A kapcsolatok az egyedek közötti viszonyokat adják meg.

Van néhány alapelv, amelyet érdemes megfontolni az adatbázis tervezése során: ezek az adott probléma, illetve megoldandó feladat szempontjából lényeges elemek megragadása. Azaz (1) a valósághű modellezés; (2) a redundancia-mentességre és (3) az egyszerűségre való törekvés; (4) a lényeges kapcsolatok feltárása, kiválasztása, a lényegtelenek figyelmen kívül hagyása; (5) a kezelés szempontjából optimális elemtípus kiválasztása.

Az adatbázis megtervezése során először koncepcionális modellt készítünk, majd ez alapján készíthetjük el az adatbázis-kezelő rendszerhez közelebb álló logikai modellt. Ezt követi a tényleges, fizikai megvalósítás.

A koncepcionális sémák – köztük az egyed-kapcsolat modell is – szemléletes módon, ábrákkal mutatják be az adatbázis logikai felépítését; nem hozzáértő számára is érthető formában szemléltetik az adatbázist.

Az egyed-kapcsolat diagramok ábrázolásának többféle variánsa él egymás mellett. A modellnek születtek bővített, kiterjesztett változatai is, amelyek további lehetőségekkel egészítik ki a modell leíró képességét.

Az egyed-kapcsolat modell három alapeleme:

1. Egyedtípus (az egyedek halmaza, egyedhalmaz), jelölése téglalappal, benne az egyedtípus nevével (főnév). Minden egyedtípusnak kell legyen legalább egy attribútuma. Fajtái:
    - erős egyedtípus;
    - gyenge egyedtípus (kettős falú téglalap);
2. Attribútum (az egyedek tulajdonságai), jelölése ellipszissel, benne a tulajdonság megnevezésével (főnév). Fajtái:
    - egyszerű tulajdonság;
    - kulcs tulajdonság (jelölés aláhúzással);
    - összetett tulajdonság (résztulajdonságok kapcsolódnak hozzá);
    - többértékű tulajdonság (jelölés kettős falú ellipszissel);

- leszármaztatott tulajdonság (jelölés szaggatott vonalú ellipszissel);

3. Kapcsolat (az egyedtípusok közötti viszony), jelölése rombusszal, benne a kapcsolat megnevezésével (ige). Fajtái:
   - 1:1 (egy-az-egyhez): két egyedtípus között
   - 1:N (egy-a-többhöz): két egyedtípus között
   - M:N (több-a-többhöz): két egyedtípus között
   - Rekurzív: az egyedtípus saját magához kapcsolódik
   - N-ed fokú: több egyedtípus között
   - Gyenge kapcsolat: gyenge egyedtípust kapcsol össze erős egyedtípussal (jelölés dupla falú rombusszal)

   A kapcsolat lehet totális vagy parciális.

   Rendelkezhet tulajdonsággal is.

2.12 Ellenőrző kérdések

1. Mi az adatmodell és mi a célja?
2. Milyen adatmodelleket ismer (felsorolás)?
3. Mik az adatmodellek általános elemei?
4. Mik az adatmodellezés alapelvei?
5. Miért van szükség a koncepcionális adatmodellre?
6. Mik az egyed-kapcsolat modell fő jellemzői?
7. Mik az egyedtípusok, mik a jellemzői, hogyan jelöljük?
8. Melyek az attribútumok fajtái az egyed-kapcsolat modellben, melyikre mi jellemző, hogyan jelöljük?
9. Milyen kapcsolatok léteznek az egyed-kapcsolat modellben, melyikre mi jellemző, hogyan jelöljük?

2.13 Feladat

Készítsen egy egyed-kapcsolat diagramot hallgatók, oktatók és kurzusok adatainak nyilvántartására!

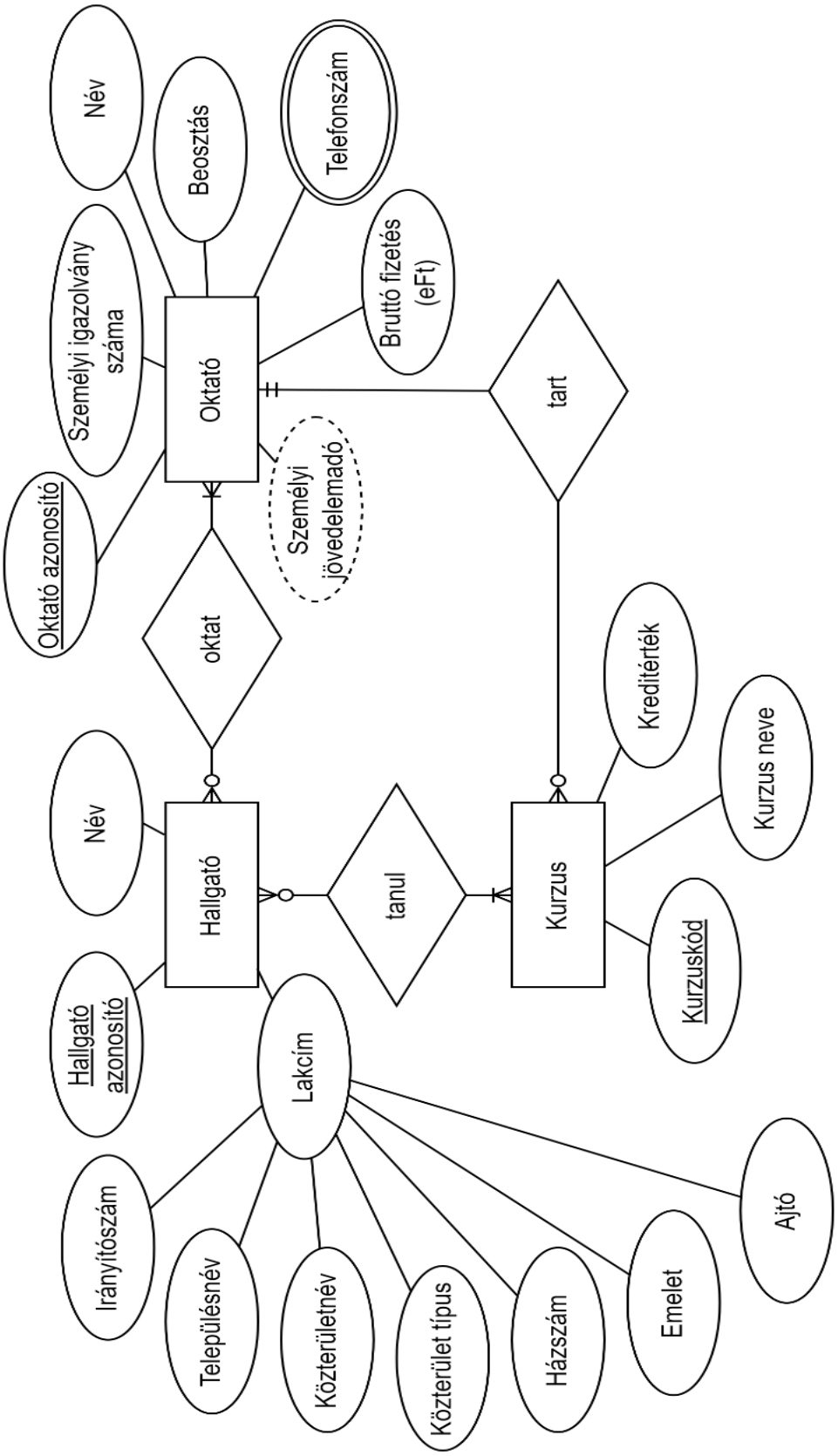
Az egyedtípusok és tulajdonságaik:

- A hallgatók tulajdonságai: hallgatóazonosító, név, lakcím.
- Az oktatók tulajdonságai: oktatóazonosító, személyi igazolvány száma, név, beosztás, telefonszám, bruttó fizetés (eFt), személyi jövedelemadó.
- A kurzusok tulajdonságai: kurzuskód, kurzus neve, kreditérték.
- Egy oktató több telefonszámon is hívható.
- A lakcím összetett tulajdonság.

Az egyedtípusok közötti kapcsolatok leírása:

- Egy kurzust több hallgató is felvehet.
- Egy kurzusra többen is járhatnak.
- Egy kurzust csak egy oktató oktat.
- Egy oktató több kurzuson is taníthat.
- Egy hallgatót több oktató is oktathat.
- Egy oktató több hallgatót is taníthat.
- Egy hallgatót több oktató is oktathat.
- Lehet olyan kurzus, amire nem jelentkezett senki.
- Egy hallgatónak legalább egy kurzust fel kell vennie.
- Lehet olyan oktató, akinek nincs kurzusa.
- Minden kurzusnak van oktatója.
- Egy hallgatót legalább egy oktató oktat.
- Lehet olyan oktató, aki nem oktat hallgatókat.

Egy lehetséges megoldás: (Lásd: 2-19. ábra)



2-19. ábra. Egyed-kapcsolat diagram.

3. lecke

A relációs adatmodell

Elvárt tanulási eredmények

Tudás

- Tudja, hogyan épül fel a relációs modell
- Ismeri a relációk szerkezetét
- Ismeri a relációs modellben található kulcsok típusait és azok szerepét
- Ismeri a relációk közötti kapcsolatok típusait és a kapcsolatok létrehozásának módját
- Ismeri a relációs sémák szerkezetét

Képesség

- Helyesen alkalmazza relációs modell elemkészletének elemeit
- Képes az adatot feladatot leképező relációs sémákat alkotni

Attitűd

- Törekszik arra, hogy relációs sémái áttekinthetőek legyenek

Autonómia, felelősség

- Önállóan hoz létre relációs sémákat

Miről lesz szó ebben a fejezetben?

- Megnézzük, hogy milyen alkotóelemekből és hogyan épül fel a relációs adatmodell.
- Áttekintjük a relációk (táblák) szerkezetét.
- Megismerkedünk a kulcsokkal és azok szerepével a relációs modell tábláiban.
- Megtanuljuk, hogy hogyan kell összekapcsolni táblákat.
- Megismerkedünk a relációs sémák, adatbázissémák szerkezetének formális leírásának módjával.

3 A relációs adatmodell

A **relációs adatmodell** napjaink **legelterjedtebb** – adatbázis-kezelők által támogatott – **adatmodellje**. Egyszerűségének következtében gyorsan népszerűvé vált a felhasználók körében és sok implementációja született. Az elméleti megalapozottság a kutatók, a szakemberek szimpátiáját is kiváltotta, ezért ez a modell számos fejlesztés alapját képezi.

3.1 A relációs modell felépítése

A relációs modell kétdimenziós **táblázatok rendszeréből** épül fel. A táblázatokat **relációnak is szokás nevezni**, mert szerkezetük és működésük megegyezik a matematikában használt reláció fogalmával.

Egy adatbázisban **minden táblázatot egyedi névvel** kell ellátni, így azok egyértelműen megkülönböztethetők egymástól. A **táblázatok megfelelnek** az egyedek halmazának, **az egyedtípusoknak** (lásd: egyed-kapcsolat modell). A **táblázat sorokból** – azaz **rekordokból** – és **oszlopokból** – azaz **mezőkből** – **áll** (3-1. ábra és 3-2. ábra). Egy **rekord** egy **cellája csak egy elemi (atomi) értéket** tartalmazhat.

A táblázat **első sora a fejrész**, amely **a mezőneveket**, a mezők elnevezését, azaz az egyedtípushoz tartozó tulajdonság megnevezését, azonosítóját **tartalmazza**. A **mezőneveknek** egy táblázaton belül **egyedieknek** kell lenniük.

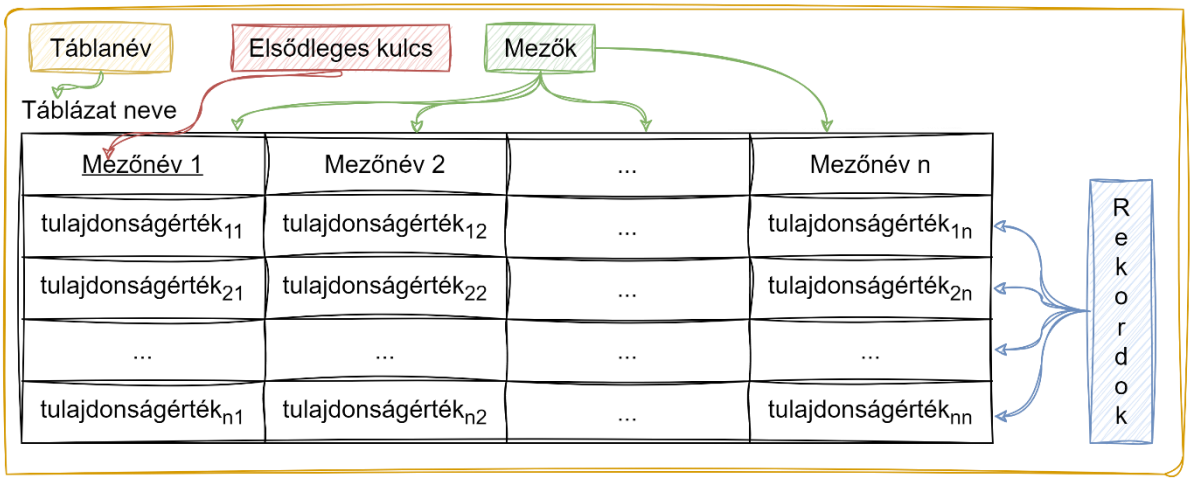
A táblázatban a fejléc alatt található sorok a rekordok, **egy sor az egy rekord**. A mezők (oszlopok) az **egyes egyedek tulajdonságértékeit** tartalmazzák.

Az **oszlopok száma** a táblázat **fokszáma**, a **sorok számát kardinalitásnak** nevezzük. Minden sor azonos számú oszlopból áll és fordítva, minden oszlop azonos számú sorból áll. Az **oszlopok**, illetve a **sorok sorrendje** (a fejlécet kivéve) **közömbös**, ezért azok tetszőlegesen felcserélhetők.

A mezők közül fontos szerepe van az **elsődleges kulcsnak**, amelyet **aláhúzással** jelölünk és általában a táblázat bal oldalán szerepeltetünk.

A relációs modellben az egyedtípusok és azok tulajdonságai mellett a **kapcsolatokat is táblázatban** jelenítjük meg.

Az adatokat tartalmazó táblázatokat bármilyen erre alkalmas programmal elkészíthetjük. Használhatunk táblázatkezelőt, vagy akár szövegszerkesztőt is. A tananyagban szereplő táblázatok OpenDocument-táblázat (ODS) fájlként letölthetők.



3-1. ábra. A táblázat felépítése a relációs modellben.

Személy

Egyedi azonosító	Név	Születési idő	Nem	Jövedelem (eFt)
27	Alma Alfonz	1960.10.05.	Férfi	310
48	Birs Bea	1995.08.28.	Nő	410
52	Citrom Cecil	1995.08.28.	Nő	410
67	Alma Alfonz	1982.04.07.	Férfi	390
80	Dinnye Dénes	1982.04.07.	Férfi	390

3-2. ábra. Táblázat mintaadatokkal relációs modellben.

3.2 Relációs séma és adatbázisséma

A **relációs séma** a **táblázat** (reláció) logikai felépítését határozza meg. Ez tábla **szerkezetének formális leírása**, ami tartalmazza a **táblázat** (reláció) **nevét** és a táblázatban található **attribútumok** (oszlopok, mezők) **nevét**. Az **elsődleges kulcsot aláhúzzuk**, a többi kulcsot viszont nem jelöljük meg. A **séma** tehát csak a táblázat szerkezetét mutatja be, de a **táblázatban található adatokat nem jeleníti meg**!

A relációs séma **általános alakja**: Reláció neve (oszlop$_1$ neve, oszlop$_2$ nev, … oszlop$_n$ neve)

A 3-2. ábra táblázatának relációs sémája: Személy (egyedi azonosító, név, születési idő, nem, jövedelem)

Egy adatbázis relációs modellje egy vagy több relációs sémát is tartalmazhat. A **relációs sémákból álló halmazt** relációs **adatbázissémának**, vagy egyszerűen adatbázissémának **nevezzük**.

3.3 A kulcs fajtái és szerepe a táblázatokban

A modell fontos eleme a **kulcs, amelynek fajtái**:

- szuperkulcs,
- jelölt kulcs,
- elsődleges kulcs,
- alternatív kulcs,
- összetett kulcs,
- idegen kulcs.

3.3.1 Szuperkulcs

A szuperkulcs egy vagy több mező kombinációja, amely **egyedileg azonosítja a táblázat minden sorát**. Minden szuperkulcs biztosítja, hogy nincs a táblázatban két olyan sor, amelynek értékei az összes tulajdonságkombinációban megegyeznek. A szuperkulcs állhat egyetlen mezőből, vagy mezők kombinációkából, **tartalmazhat felesleges mezőket** is.

Példa: Személyek adatait tartalmazó táblázatban a személyi szám, név, születési idő tulajdonságokat tároljuk. A minimális szuperkulcs személyi szám, ami egyben elsődleges kulcs is. Egyéb szuperkulcsok: személyi szám, név; személyi szám, születési idő; személyi szám, név, születési idő – ezek mindegyike tartalmaz felesleges tulajdonságokat.

3.3.2 Jelölt kulcs

A jelölt kulcs a minimális szuperkulcs, ami azt jelenti, hogy nincs benne felesleges tulajdonság. Egy táblázatban akár több jelölt kulcs is előfordulhat.

Példa: Személyek adatait tartalmazó táblázatban a személyi szám és a személyi igazolványszám is alkalmas a személyek azonosítására, tehát jelölt kulcsok.

3.3.3 Alternatív kulcs

Ha **több alkalmas mezőnk is van**, akkor szabadon választhatjuk meg, hogy közülük melyik legyen az elsődleges kulcs, a többit **alternatív kulcsnak** nevezzük.

Példa: Személyek adatait tartalmazó táblázatban kulcsok lehetnek a személyi szám, a személyi igazolvány szám, lakcímkártya száma vagy egy mesterségesen generált sorszám. Ezek tetszőlegesen választhatók. Amelyiket ezekből kiválasztjuk a személyek azonosítására, elsődleges kulcs lesz, a többi pedig alternatív kulcs.

3.3.4 Elsődleges kulcs

Elsődleges kulcs vagy elemi kulcs a jelölt kulcsok közül kiválasztott kulcs, amely **alkalmas a rekordok** (sorok) **egyértelmű megkülönböztetésére**, azonosítására. A többi tulajdonságtól **aláhúzással különböztetjük meg**. A kulcs nem veheti fel kétszer ugyanazt az értéket egy táblázaton belül. Két fajtája a **természetes** kulcs és a **mesterséges** (vagy helyettesítő) kulcs.

Példa: Személyek adatait tartalmazó táblázatban elsődleges természetes kulcs az adott személyhez „természetesen tartozó” azonosítók, ami lehet a személyi szám, személyi igazolvány szám, lakcímkártya száma stb. – de csak az egyik ezek közül! Elsődleges mesterséges kulcs lehet egy minden sorra – mesterségesen – generált egyedi tetszőleges sorszám, sorozatszám.

Az **elsődleges kulcs jelenléte** a relációs modell szerint **kötelező** minden relációban. Ugyanakkor a relációs adatbázis-kezelő rendszerek nem ennyire szigorúak: megengedik elsődleges kulcs nélküli táblázatok létrehozását is.

Az elsődleges kulcsot úgy kell kiválasztani, hogy a **benne lévő mezők száma minimális** legyen.

Példa: Személyek adatait tartalmazó táblázatban a személy neve, az anyja neve és a személy születési dátuma alkalmas lehet egy konkrét személy beazonosítására, de ha rendelkezésre áll a személyi szám is, akkor azt kell választani elsődleges kulcsnak.

3.3.5 Összetett elsődleges kulcs

Előfordulhat, hogy egy tulajdonság önmagában nem elegendő egy adott egyed egyértelmű beazonosításához. Ha **egynél több mezőre van szükségünk** ehhez, akkor **összetett elsődleges kulcsról** beszélünk.

Példa: Személyek adatait tartalmazó táblázatban a személy neve, az anyja neve és a személy születési dátuma alkalmas lehet egy konkrét személy beazonosítására, ha nincs lehetőségünk kevesebb mezőt tartalmazó kulcs alkalmazására.

3.3.6 Idegen kulcs

Két (vagy több) táblázat összekapcsolását idegen kulcs segítségével oldhatjuk meg. Az idegen kulcs egy vagy több mező, amely hivatkozik egy másik táblázat elsődleges kulcsára.

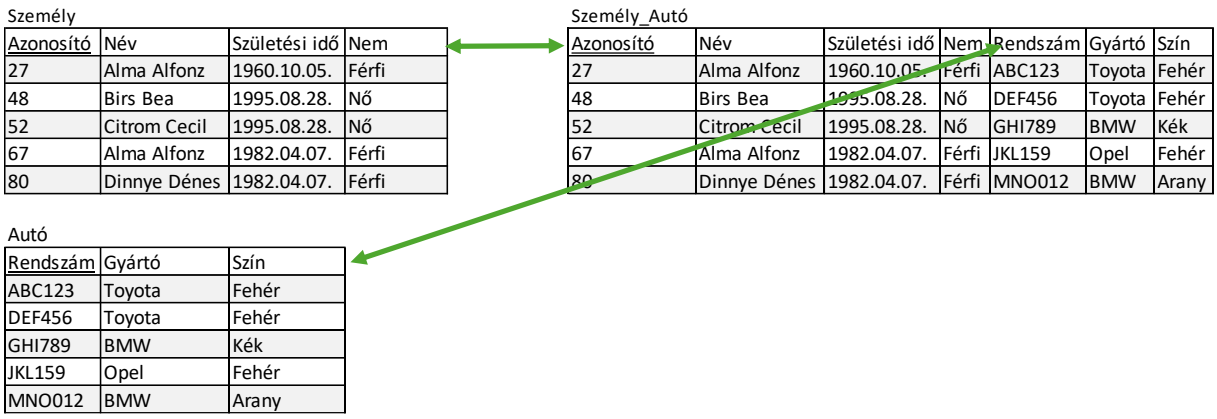
Példa: Lásd a kapcsolatoknál!

3.4 Kapcsolatok létrehozása a táblák között és az 1:1 kapcsolat

Egy táblából a táblával logikai kapcsolatban lévő **másik tábla egy meghatározott sorára az idegen kulcs segítségével** hivatkozhatunk. Az idegen kulcsnak megfelelő érték abban a táblában, amelyiknek rekordjára hivatkozunk, elsődleges kulcs.

Az 1:1 kapcsolat esetén – ami a legegyszerűbb eset – eljárhatunk úgy, hogy **a két táblát összevonjuk, egyesítjük egyetlen táblába**. Az egyik tábla elsődleges kulcsát tetszőlegesen választjuk az új tábla elsődleges kulcsának. Ha vannak azonos nevű mezők, azok átnevezéséről is gondoskodnunk kell. Ezt a megoldást totális kapcsolat esetén célszerű választani, ellenkező esetben sok üres cellánk lesz.

Példa: Autók és tulajdonosaik adatairól vezetünk nyilvántartást. Feltételezzük, hogy mindenkinek egyetlen autója van, illetve minden autó egyetlen tulajdonossal rendelkezik (3-3. ábra).



Személy

Azonosító	Név	Születési idő	Nem
27	Alma Alfonz	1960.10.05.	Férfi
48	Birs Bea	1995.08.28.	Nő
52	Citrom Cecil	1995.08.28.	Nő
67	Alma Alfonz	1982.04.07.	Férfi
80	Dinnye Dénes	1982.04.07.	Férfi

Személy_Autó

Azonosító	Név	Születési idő	Nem	Rendszám	Gyártó	Szín
27	Alma Alfonz	1960.10.05.	Férfi	ABC123	Toyota	Fehér
48	Birs Bea	1995.08.28.	Nő	DEF456	Toyota	Fehér
52	Citrom Cecil	1995.08.28.	Nő	GHI789	BMW	Kék
67	Alma Alfonz	1982.04.07.	Férfi	JKL159	Opel	Fehér
80	Dinnye Dénes	1982.04.07.	Férfi	MNO012	BMW	Arany

Autó

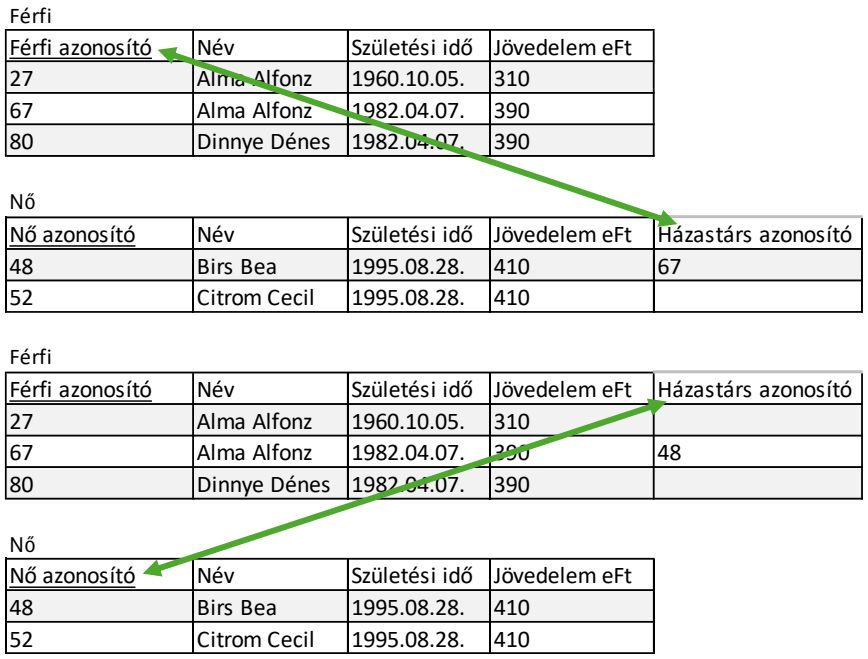
Rendszám	Gyártó	Szín
ABC123	Toyota	Fehér
DEF456	Toyota	Fehér
GHI789	BMW	Kék
JKL159	Opel	Fehér
MNO012	BMW	Arany

3-3. ábra. 1:1 kapcsolat létrehozása táblák egyesítésével.

A modell relációs sémái:
Személy (azonosító, név, születési idő, nem)
Autó (rendszám, gyártó, szín)
Személy_Autó (azonosító, név, születési idő, nem, rendszám, gyártó, szín)

Megtehetjük azt is, hogy az egyik táblát (mindegy, melyiket) **kiegészítjük a logikailag hozzá kapcsolódó másik tábla elsődleges kulcsa felhasználásával**. Az lényegtelen, hogy hogyan nevezzük el az új mezőt – azzal a megkötéssel, hogy nem használhatunk már létező mezőnevet. A fontos az, hogy a másik táblából tartalmazza a megfelelő kulcsértékeket.

Példa: A Férfi és a Nő táblázata között 1:1 kapcsolat van (mindenkinek csak egy házastársa lehet). Az egyik lehetséges megoldás, hogy a Nő táblához hozzáírjuk a Férfi tábla elsődleges kulcsát idegen kulcsként. A másik lehetséges megoldás az előző fordítottja: A Férfi táblához írjuk hozzá a Nő tábla elsődleges kulcsát idegen kulcsként. Mindkét megoldásnál kiolvasható a táblákból, hogy a 48-as nő és az 67-es férfi házastársak, a többieknek nincs házastársa. (3-4. ábra)



3-4. ábra. 1:1 kapcsolat létrehozása táblák között idegen kulcs hozzáírásával.

A modell relációs sémái:
Férfi (férfi azonosító, név, születési idő, jövedelem)
Nő (nő azonosító, név, születési idő, jövedelem, házastárs azonosító)
vagy
Férfi (férfi azonosító, név, születési idő, jövedelem, házastárs azonosító)
Nő (nő azonosító, név, születési idő, jövedelem)

Az 1:1 kapcsolatnál eljárhatunk úgy is, hogy **létrehozunk egy kapcsolat táblát** és ebbe az új táblába **beleírjuk a két összekapcsolandó táblából az elsődleges kulcsokat**, amelyek a kapcsolat táblában idegen kulcsok lesznek. Ez a járható út akkor is, ha szeretnénk olyan tulajdonságot is felvenni, amit az egyik táblázatban sem lenne szerencsés tárolni. A kapcsolat táblát célszerű egy beszédes névvel ellátni – ez lehet például az egyed-kapcsolat modellben már megismert ige, vagy képezhetjük a táblanevet az összekapcsolt táblák nevének összerakásából is. A kapcsolat táblában tetszőlegesen választhatjuk meg, hogy melyik lesz az elsődleges kulcs, hiszen 1:1 kapcsolatnál bármelyik érték csak egyszer fordulhat elő.

Példa: A Férfi és a Nő táblázata között 1:1 kapcsolat van (mindenkinek csak egy házastársa lehet). A házasságkötés időpontját egyik meglévő táblában sem lenne szerencsés eltárolni. Létrehozunk egy kapcsolat táblát, amibe beleírjuk az összekapcsolandó táblákból az elsődleges kulcsokat idegen kulcsként, az egyiket tetszőlegesen elsődleges kulcsnak választjuk. A választás azért lehet tetszőleges, mert ha valakinek az azonosítója többször szerepelne itt, az azt jelentené, hogy vagy több házastársa van, vagy tévedésből többször szerepel a házastársa azonosítójával együtt. A kapcsolat táblába beleírjuk a kapcsolathoz tartozó tulajdonságot is. A kapott táblákból kiolvasható, hogy a 48-as nő és az 67-es férfi házastársak, továbbá látható a házasságkötésük időpontja is. (3-5. ábra)

Férfi

Férfi azonosító	Név	Születési idő	Jövedelem eFt
27	Alma Alfonz	1960.10.05.	310
67	Alma Alfonz	1982.04.07.	390
80	Dinnye Dénes	1982.04.07.	390

Elvesz

Férfi	Nő	Házasságkötés ideje
67	48	2018.07.08.

Nő

Nő azonosító	Név	Születési idő	Jövedelem eFt
48	Birs Bea	1995.08.28.	410
52	Citrom Cecil	1995.08.28.	410

3-5. ábra. 1:1 kapcsolat létrehozása táblák között kapcsolat táblával.

A modell relációs sémái:
Férfi (férfi azonosító, név, születési idő, jövedelem)
Nő (nő azonosító, név, születési idő, jövedelem)
Elvesz (férfi, nő, házasságkötés ideje)

3.5 1:N kapcsolat

Az 1:N kapcsolatnál is alkalmazhatjuk azt a megoldást, mely szerint **az egyik tábla elsődleges kulcsát hozzáírjuk** a másik táblához **idegen kulcsként**. Itt viszont a választásunk már nem lehet tetszőleges: **az N-oldali táblához** kell hozzáírnunk az idegen kulcsot.

Példa: Személyek és autók adatairól vezetünk nyilvántartást. Feltételezzük, hogy egy tulajdonosnak több autója lehet, egy autónak viszont csak egy tulajdonosa. Lehetnek olyan személyek, akiknek nincs autója, ill. olyan autók, amelyeknek nincs tulajdonosa. A 3-6. ábra látható, hogy az N-oldali táblához, az Autóhoz írtuk hozzá a tulajdonosok azonosítóját idegen kulcsként (1 személynek több autója lehet). Az 52 azonosítójú személynek 2 autója van, ezért ő többször is szerepel. Ha valamelyik autónak nincs tulajdonosa, a Tulajdonos mező üres. Miért nem lehet az 1 oldali részhez (a Személyhez) hozzáírni az autók rendszámát idegen kulcsként? Azért, mert ha valakinek több autója van, annak több rendszámot is bele kellene írni ugyanabba a cellába, de ez nem elemi (atomi) érték lenne, ezért nem szabad (3-7. ábra).

Személy

Azonosító	Név	Születési idő	Nem
27	Alma Alfonz	1960.10.05.	Férfi
48	Birs Bea	1995.08.28.	Nő
52	Citrom Cecil	1995.08.28.	Nő
67	Alma Alfonz	1982.04.07.	Férfi
80	Dinnye Dénes	1982.04.07.	Férfi

Autó

Rendszám	Gyártó	Szín	Tulajdonos
ABC123	Toyota	Fehér	52
DEF456	Toyota	Fehér	52
GHI789	BMW	Kék	67
JKL159	Opel	Fehér	

3-6. ábra. 1:N kapcsolat létrehozása táblák között.

A modell relációs sémái:
Személy (azonosító, név, születési idő, nem)
Autó (rendszám, gyártó, szín, tulajdonos)

Személy

Azonosító	Név	Születési idő	Nem	Rendszám
27	Alma Alfonz	1960.10.05.	Férfi	
48	Birs Bea	1995.08.28.	Nő	
52	Citrom Cecil	1995.08.28.	Nő	ABC123 DEF456
67	Alma Alfonz	1982.04.07.	Férfi	GHI789
80	Dinnye Dénes	1982.04.07.	Férfi	

Autó

Rendszám	Gyártó	Szín
ABC123	Toyota	Fehér
DEF456	Toyota	Fehér
GHI789	BMW	Kék
JKL159	Opel	Fehér

3-7. ábra. Hibás 1:N kapcsolat létrehozása táblák között.

Az 1:N kapcsolatnál is alkalmazhatjuk a **kapcsolat táblát**, amibe **beleírjuk a két összekapcsolandó táblából az elsődleges kulcsokat**, amelyek a kapcsolat táblában idegen kulcsok lesznek. A kapcsolat táblában viszont nem választhatjuk meg tetszőlegesen, hogy melyik lesz **az elsődleges kulcs**: az **N-oldali táblából érkező mezőt** kell választani. Ezt a megoldást célszerű választani akkor is, amikor szeretnénk felvenni olyan tulajdonságot, amit egyik táblázatban sem lenne szerencsés tárolni.

Példa: Személyek és autók adatairól vezetünk nyilvántartást. Feltételezzük, hogy egy tulajdonosnak több autója lehet, egy autónak viszont csak egy tulajdonosa. Lehetnek olyan személyek, akiknek nincs autója, ill. olyan autók, amelyeknek nincs tulajdonosa. Tároljuk a vásárlás dátumát is. A 3-8. ábra látható, hogy az 52 azonosítójú személynek 2 autója van, ezért ő többször is szerepel a kapcsolat táblában – ezért nem lehet a személy azonosítója ott

elsődleges kulcs. Mivel egy autónak csak egyetlen tulajdonosa lehet, a rendszám csak egyszer szerepelhet a kapcsolat táblában, ezért az lehet elsődleges kulcs.

Személy

Azonosító	Név	Születési idő	Nem
27	Alma Alfonz	1960.10.05.	Férfi
48	Birs Bea	1995.08.28.	Nő
52	Citrom Cecil	1995.08.28.	Nő
67	Alma Alfonz	1982.04.07.	Férfi
80	Dinnye Dénes	1982.04.07.	Férfi

Birtokol

Rendszám	Azonosító	Vásárlás dátuma
ABC123	52	2020.03.10.
DEF456	52	2022.08.10.
GHI789	67	2022.08.10.

Autó

Rendszám	Gyártó	Szín
ABC123	Toyota	Fehér
DEF456	Toyota	Fehér
GHI789	BMW	Kék
JKL159	Opel	Fehér

3-8. ábra. 1:N kapcsolat létrehozása táblák között kapcsolat táblával.

A modell relációs sémái:
Személy (azonosító, név, születési idő, nem)
Autó (rendszám, gyártó, szín)
Birtokol (rendszám, azonosító, vásárlás dátuma)

3.6 M:N kapcsolat

Az M:N kapcsolatnál mindenképpen egy **új, kapcsolatot biztosító táblát kell létrehoznunk**. A kapcsolat tábla oszlopai most is a két összekapcsolt tábla elsődleges kulcsai lesznek (továbbá a kapcsolathoz esetlegesen felvett tulajdonságok). A kapcsolatot biztosító táblában M:N kapcsolatnál **a két idegen kulcs együtt alkot elsődleges kulcsot**, azaz összetett kulcs.

Példa: Személyek és autók adatairól vezetünk nyilvántartást. Egy tulajdonosnak több autója lehet, egy autónak pedig több tulajdonosa. A kapcsolatot biztosító Birtokol táblában a Személy táblából az azonosító, az Autó táblából a rendszám az idegen kulcs. Látható, hogy az 52-es Citrom Cecilnek 2 autója van, ezért ő egynél többször szerepel, a GHI789 BMW-nek 2 tulajdonosa van, ezért ez a rendszám is egynél többször szerepel; tehát külön-külön egyik sem lehet elsődleges kulcs, a kettő együtt viszont lehet (3-9. ábra).

Személy

Azonosító	Név	Születési idő	Nem
27	Alma Alfonz	1960.10.05.	Férfi
48	Birs Bea	1995.08.28.	Nő
52	Citrom Cecil	1995.08.28.	Nő
67	Alma Alfonz	1982.04.07.	Férfi
80	Dinnye Dénes	1982.04.07.	Férfi

Birtokol

Rendszám	Azonosító
ABC123	52
DEF456	52
GHI789	67
GHI789	48

Autó

Rendszám	Gyártó	Szín
ABC123	Toyota	Fehér
DEF456	Toyota	Fehér
GHI789	BMW	Kék
JKL159	Opel	Fehér

3-9. ábra. M:N kapcsolat létrehozása táblák közötti kapcsolat táblával.

A modell relációs sémái:
Személy (azonosító, név, születési idő, nem)
Autó (rendszám, gyártó, szín)
Birtokol (rendszám, azonosító)

3.7 N-ed fokú kapcsolat

N-ed fokú kapcsolat esetén **kettőnél több táblát kapcsolunk** össze kapcsolat táblával, amelybe **beleírjuk az összekapcsolandó táblák elsődleges kulcsait** idegen kulcsként és kapcsolat táblához tartozó egyéb tulajdonságokat. A táblák közötti kapcsolat megvalósítható a táblák között párosával létrehozott kapcsolat táblákkal is.

Példa: Személyek, autók és garázsok adatairól vezetünk nyilvántartást. Minden irányban M:N kapcsolat áll fenn: Egy személy több autót is használhat, egy autót többen is használhatnak; Egy garázsban többen is tárolhatnak autót, illetve egy autót több garázsban is tárolhatnak; Egy személy több garázst is használhat, illetve egy garázst akár több személy használhat. A Használ nevű kapcsolat táblában idegen kulcsként szerepel az összekapcsolt táblák elsődleges kulcsa (3-10. ábra). Mivel mindhárom mező értékei többször is előfordulhatnak – csak a három együtt egynél többször nem –, csak együtt alkothatnak kulcsot. A feladat megoldható úgy is, hogy párosával kapcsoljuk össze a táblákat (3-11. ábra).

Személy

Azonosító	Név	Születési idő	Nem
27	Alma Alfonz	1960.10.05.	Férfi
48	Birs Bea	1995.08.28.	Nő
52	Citrom Cecil	1995.08.28.	Nő
67	Alma Alfonz	1982.04.07.	Férfi
80	Dinnye Dénes	1982.04.07.	Férfi

Használ

Rendszám	Személy azonosító	GarázsID
ABC123	52	G01
DEF456	52	G01
GHI789	67	G02
GHI789	48	G03

Garázs

Garázs azonosító	Település	Terület m2
G01	Szeged	30
G02	Szeged	20
G03	Makó	15

Autó

Rendszám	Gyártó	Szín
ABC123	Toyota	Fehér
DEF456	Toyota	Fehér
GHI789	BMW	Kék
JKL159	Opel	Fehér

3-10. ábra. Harmadfokú kapcsolat létrehozása táblák között kapcsolat táblával (minden irányban M:N kapcsolat esetén).

A modell relációs sémái:

Személy (azonosító, név, születési idő, nem)

Autó (rendszám, gyártó, szín)

Garázs (garázs azonosító, település, terület)

Használ (rendszám, személy azonosító, garázsID)

Személy

Azonosító	Név	Születési idő	Nem
27	Alma Alfonz	1960.10.05.	Férfi
48	Birs Bea	1995.08.28.	Nő
52	Citrom Cecil	1995.08.28.	Nő
67	Alma Alfonz	1982.04.07.	Férfi
80	Dinnye Dénes	1982.04.07.	Férfi

Használ

Személy azonosító	Garázs azonosító
52	G01
67	G02
48	G03

Birtokol

Rendszám	Személy azonosító
ABC123	52
DEF456	52
GHI789	67
GHI789	48

Tárol

Rendszám	GarázsID
ABC123	G01
DEF456	G01
GHI789	G02
GHI789	G03

Autó

Rendszám	Gyártó	Szín
ABC123	Toyota	Fehér
DEF456	Toyota	Fehér
GHI789	BMW	Kék
JKL159	Opel	Fehér

Garázs

Garázs azonosító	Település	Terület m2
G01	Szeged	30
G02	Szeged	20
G03	Makó	15

3-11. ábra. Harmadfokú kapcsolat létrehozása táblák között párosával kialakított kapcsolat táblával (minden irányban M:N kapcsolat esetén).

A modell relációs sémái:
Személy (azonosító, név, születési idő, nem)
Autó (rendszám, gyártó, szín)
Garázs (garázs azonosító, település, terület)
Birtokol (rendszám, személy azonosító)
Használ (személy azonosító, garázsID)
Tárol (rendszám, GarázsID)

3.8 Rekurzív kapcsolat

Rekurzív kapcsolatról akkor beszélünk, ha **egy egyedtípus egyed-előfordulásai saját egyedtípusuk egyed-előfordulásaihoz kapcsolódnak**. A **táblához hozzáadhatjuk az elsődleges kulcsot idegen kulcsként is** (vagy ebben az esetben is létrehozhatunk kapcsolat táblát). (Relációs adatbázis-kezelőben rekurzív kapcsolatot úgy is létrehozhatunk, hogy mintegy „megkettőzzük” a táblát úgy, hogy két ún. aliasnevet, becenevet adunk a táblának és ezután kapcsoljuk össze önmagával a két különböző néven 1:1, vagy 1:N kapcsolattal.)

Példa: Személyek, nevelőszülők és gyerekeik adatait tároljuk ugyanabban a táblában. Szeretnénk létrehozni a nevelőszülők és gyerekeik közötti kapcsolatot. A 3-12. ábra olyan megoldást láthatunk, ahol a személyek egyedi azonosítóját idegen kulcsként újra hozzáadtuk a táblához. Ez a mező a megfelelő gyerek sorában tartalmazza a nevelőszülő egyedi azonosítóját. A 3-13. ábra mutatja azt a megoldást, amikor kapcsolat táblát alkalmaztunk. A kapcsolat táblában a nevelőszülők és mellettük a gyerek azonosítója szerepel idegen kulcsként. Mivel egy szülő több gyereket is nevelhet (a nevelőszülő azonosítója többször szerepelhet), illetve egy gyereket többen is nevelhetnek (a gyerek azonosítója is többször szerepelhet), külön-külön nem, csak együtt alkothatnak elsődleges kulcsot.

Személy

Egyedi azonosító	Név	Születési idő	Nem	Szülő azonosító
27	Alma Alfonz	1960.10.05.	Férfi	
48	Birs Bea	1995.08.28.	Nő	
52	Citrom Cecil	1995.08.28.	Nő	27
67	Alma Alfonz	1982.04.07.	Férfi	27
80	Dinnye Dénes	1982.04.07.	Férfi	

3-12. ábra. Rekurzív kapcsolat.

A modell relációs sémája:
Személy (egyedi azonosító, név, születési idő, nem, szülő azonosító)

Személy

Egyedi azonosító	Név	Születési idő	Nem
27	Alma Alfonz	1960.10.05.	Férfi
48	Birs Bea	1995.08.28.	Nő
52	Citrom Cecil	1995.08.28.	Nő
67	Alma Alfonz	1982.04.07.	Férfi
80	Dinnye Dénes	1982.04.07.	Férfi

Nevel

Szülő	Gyerek
27	52
27	67

3-13. ábra. „Rekurzív” kapcsolat egy kapcsolat tábla segítségével.

A modell relációs sémái:
Személy (egyedi azonosító, név, születési idő, nem)
Nevel (szülő, gyerek)

3.9 Összefoglalás

A relációs adatmodell napjaink legelterjedtebb adatmodellje. A modell kétdimenziós táblázatok – relációknak is nevezett – rendszeréből épül fel.

A táblázatok megfelelnek az egyedek halmazának, az egyedtípusoknak. Egy adatbázisban minden táblázatot egyedi névvel kell ellátni, hogy azok egyértelműen megkülönböztethetők legyenek egymástól. A táblázat sorokból – azaz rekordokból – és oszlopokból – azaz mezőkből – áll. Egy rekord egy cellája csak egy elemi (atomi) értéket tartalmazhat.

A táblázat első sora a fejrész, amely a mezőneveket, a mezők elnevezését, azaz az egyedtípushoz tartozó tulajdonság megnevezését tartalmazza. A mezőneveknek egy táblázaton belül egyedieknek kell lenniük.

A táblázatban a fejléc alatt található sorok a rekordok, egy sor az egy rekord. A mezők (oszlopok) az egyes egyedek tulajdonságértékeit tartalmazzák.

A modell fontos eleme a kulcs, amelynek fajtái:

- szuperkulcs,
- jelölt kulcs,
- elsődleges kulcs,
- alternatív kulcs,
- összetett kulcs,
- idegen kulcs.

Ezek közül a legfontosabb az elsődleges kulcs, amely alkalmas a rekordok (sorok) egyértelmű megkülönböztetésére, azok azonosítására. A többi tulajdonságtól aláhúzással különböztetjük meg. Az elsődleges kulcs jelenléte a relációs modell szerint kötelező minden relációban.

Ugyanakkor a relációs adatbázis-kezelő rendszerek nem ennyire szigorúak: megengedik elsődleges kulcs nélküli táblázatok létrehozását is.

A relációs modellben az egyedtípusok és azok tulajdonságai mellett a kapcsolatokat is táblázatban jelenítjük meg.

A táblák közötti kapcsolatok kialakítására a következő általános lehetőségek állnak rendelkezésre:

- egyesíthetjük a táblákat,
- az egyik tábla elsődleges kulcsát hozzáírhatjuk a másik táblához idegen kulcsként vagy
- kapcsolat táblát hozunk létre, amely tartalmazza az összekapcsolandó táblák elsődleges kulcsait idegen kulcsként.

Azt, hogy a fentiek közül melyik megoldást célszerű választani, a kapcsolat típusa, azaz az dönti el, hogy egy-az-egyhez (1:1), egy-a-többhöz (1:N), több-a-többhöz (N:M), n-ed fokú, vagy rekurzív kapcsolatról van-e szó. Emellett azt is célszerű figyelembe venni, hogy a kapcsolat totális vagy parciális-e.

A relációs séma a táblázat (reláció) logikai felépítését határozza meg. Ez a tábla szerkezetének formális leírása, ami tartalmazza a táblázat (reláció) nevét és a táblázatban található attribútumok (oszlopok, mezők) nevét, az elsődleges kulcs jelölésével. Egy adatbázis relációs modellje egy vagy több relációs sémát is tartalmazhat. A relációs sémákból álló halmazt relációs adatbázissémának, vagy egyszerűen adatbázissémának nevezzük. A relációs séma általános alakja: Reláció neve (<u>oszlop$_1$ neve</u>, oszlop$_2$ neve, … oszlop$_n$ neve)

3.10 Ellenőrző kérdések

1. Miből épül fel a relációs modell?
2. Mi a relációs séma és adatbázisséma?
3. Mi a kulcs szerepe a táblázatokban és milyen fajtái vannak (felsorolás)?
4. Mi a szuperkulcs?
5. Mi a jelölt kulcs?
6. Mi az alternatív kulcs?
7. Mi az elsődleges kulcs?
8. Mi az összetett elsődleges kulcs?
9. Mi az idegen kulcs?
10. Hogyan hozható létre 1:1 kapcsolat táblák között?
11. Hogyan hozható létre 1:N kapcsolat táblák között?
12. Hogyan hozható létre M:N kapcsolat táblák között?
13. Hogyan hozható létre n-ed fokú kapcsolat táblák között?
14. Hogyan hozható létre rekurzív kapcsolat táblák között?

3.11 Feladat

Készítsen relációs modellt hallgatók, oktatók és kurzusok adatainak nyilvántartására!

- A hallgatók tulajdonságai: hallgatóazonosító, név, lakcím.
- Az oktatók tulajdonságai: oktatóazonosító, személyi igazolvány száma, név, beosztás, telefonszám, bruttó fizetés (eFt), személyi jövedelemadó.
- A kurzusok tulajdonságai: kurzuskód, kurzus neve, kreditérték.
- Egy kurzust több hallgató is felvehet és egy kurzusra többen is járhatnak. Egy kurzust csak egy oktató oktathat és egy oktató több kurzuson is taníthat.
- Egy oktató több telefonszámon is elérhető, illetve egy telefonszámon több oktató is hívható.

Írja le a relációs modell sémáját és készítsen a sémának megfelelő táblázatokat legalább 5 rekordot tartalmazó mintaadattal!

A feladat egy lehetséges megoldása:

Létrehozzuk a Hallgató relációt a feladatban felsorolt tulajdonságokkal. Egy olyan tulajdonságunk van, amelyik alkalmas a hallgatók egyedi azonosítására. Ezt aláhúzzuk:

R1: Hallgató (hallgatóazonosító, név, lakcím)

Létrehozzuk az Oktató relációt a feladatban felsorolt tulajdonságokkal. Az egyedi azonosításra alkalmas 2 kulcs közül az egyiket elsődleges kulcsnak választjuk és aláhúzzuk:

R2: Oktató (oktatóazonosító, személyi igazolvány száma, név, beosztás, telefonszám, bruttó fizetés (eFt), személyi jövedelemadó)

Ha egy oktató több telefonszámon is elérhető, akkor hozzá a telefonszám oszlop egyetlen cellájába akár több számot is be kellene írni (többértékű tulajdonság), ami nem szerencsés. Célszerű ezért ezt az oszlopot külön táblába tenni. R2 helyett létrehozzuk R3 és R4-et:

R3: Oktató (oktatóazonosító, személyi igazolvány száma, név, beosztás, bruttó fizetés (eFt), személyi jövedelemadó)

Ahhoz, hogy ezt a telefonszámokat tartalmazó táblát hozzá tudjuk kapcsolni az oktatói táblához, szükségünk van az Oktató tábla elsődleges kulcsára is idegen kulcsként. Mivel ebben a táblában mind az oktatóazonosító, mind a telefonszám többször is előfordulhat (M:N kapcsolat van közöttük), a két mező együtt alkothat elsődleges kulcsot.

R4: Telefon (oktatóazonosító, telefonszám)

A bruttó fizetésből kiszámolható a személyi jövedelemadó összege, így azt nem feltétlenül szükséges tárolnunk a táblában. R3 helyett létrehozzuk R5-öt:

R5: Oktató (oktatóazonosító, személyi igazolvány száma, név, beosztás, bruttó fizetés (eFt))

Létrehozzuk a Kurzus relációt a feladatban felsorolt tulajdonságokkal. Mivel az oktató és a kurzus között 1:N kapcsolat áll fenn, az N-oldali táblához hozzáírhatjuk a másik tábla elsődleges kulcsát idegen kulcsként.

R6: Kurzus (kurzuskód, kurzus neve, kreditérték, oktatóazonosító)

A hallgató és a kurzus között M:N kapcsolat van, ezért azok összekapcsolására egy kapcsolat táblát hozunk létre, amelybe beleírjuk a két összekapcsolandó tábla elsődleges kulcsát idegen kulcsként. A kapcsolat tábla így kapott oszlopai együtt alkotnak elsődleges kulcsot.

R7: HallgatóKurzus (hallgatóazonosító, kurzuskód)

Így megkapjuk az adatbázissémát (R1, R4, R5, R6, R7 alapján):

Hallgató (hallgatóazonosító, név, lakcím)

Oktató (oktatóazonosító, személyi igazolvány száma, név, beosztás, bruttó fizetés (eFt))

Telefon (oktatóazonosító, telefonszám)

Kurzus (kurzuskód, kurzus neve, kreditérték, oktatóazonosító)

HallgatóKurzus (hallgatóazonosító, kurzuskód)

Táblázatok mintaadatokkal: (3-14. ábra)

Hallgató

Hallgatóazonosító	Név	Lakcím
H1	Alma Alfonz	6722 Szeged, Petőfi Sándor sugárút 156.
H2	Birs Bea	8200 Veszprém, Óváros tér 196.
H3	Citrom Cecil	6720 Szeged, Kossuth Lajos sugárút 305.
H4	Alma Alfonz	9700 Szombathely, Fő tér 53.
H5	Dinnye Dénes	6600 Szentes, Ady Endre utca 126B.

Oktató

Oktatóazonosító	személyi igazolvány száma	Név	Beosztás	Bruttó fizetés (eFt)
O1	AA123456	Alma Alfonz	egyetemi docens	700
O2	BB654321	Birs Bea	egyetemi docens	700
O3	CC234567	Citrom Cecil	főiskolai docens	590
O4	DD765432	Dinnye Dénes	egyetemi tanár	1000
O5	EE345678	Egres Emil	főiskolai tanár	800

Telefon

Oktatóazonosító	Telefonszám
O1	+36 30 555 1111
O1	+36 62 123 001
O1	+36 30 555 1112
O2	+36 30 555 1112
O3	+36 70 555 1113
O4	+36 20 333 1114
O5	+36 62 123 001

Kurzus

Kurzuskód	Kurzus neve	Kreditérték	Oktatóazonosító
K1	Adatbázis-kezelés	4	O1
K2	Marketing	3	O1
K3	Közgazdaságtan	4	O2
K4	Vállalati információs rendszerek	2	O3
K5	Vállalatgazdaságtan	2	O4

HallgatóKurzus

Hallgató	Kurzuskód
H1	K1
H1	K2
H1	K3
H1	K4
H1	K5
H2	K1
H2	K2
H2	K3
H3	K3
H3	K4
H4	K1
H4	K3
H4	K5
H5	K2

3-14. ábra. Táblázatok mintaadatokkal.

4. lecke

Egyed-kapcsolat modell leképezése relációs adatmodellre

Elvárt tanulási eredmények

Tudás

- Tudja, milyen megoldást kell alkalmazni egyedtípusok táblázattá konvertálásakor
- Ismeri a tulajdonságok átkonvertálásának módjait
- Ismeri a kapcsolatok leképezésének lehetséges megoldásait
- Tudja, mire kell odafigyelni a leképezés során
- Tudja, hogy milyen esetekben melyik leképezési megoldás alkalmazása a legcélszerűbb

Képesség

- Átalakít egyedtípusokat táblázatokká
- Helyesen konvertálja az attribútumokat mezőkké
- Képes a kapcsolat típusa szerinti relációs sémákat kialakítani

Attitűd

- Törekszik az optimális megoldásra az egyed-kapcsolat modell relációs modellre történő leképezése során

Autonómia, felelősség

- Önállóan alakít át egyed-kapcsolat modellt relációs modellre

Miről lesz szó ebben a fejezetben?

- Az előző fejezetek ismeretanyagát felhasználva áttekintjük az egyed-kapcsolat modell relációs modellre történő leképezésének, átalakításának módjait.
- Megnézzük, hogyan lesz az egyedtípusból táblázat.
- Megtanuljuk, hogy hogyan kell különböző típusú tulajdonságokat táblázatoszlopokká konvertálni.
- Megismerkedünk a kapcsolattípusok átalakításakor használható megoldásokkal.

4 Egyed-kapcsolat modell leképezése relációs adatmodellre

Az előző fejezetekben megismerkedtünk az adatbázis koncepcionális tervének elkészítéséhez használható egyed-kapcsolat modellel. Tudjuk azt is, hogy bár ez az eszköz egyszerű, könnyen megtanulható és használható, de sajnos az adatbázis-kezelő rendszerek nem tudnak mit kezdeni ezzel a magas szintű tervvel. Egy implementációs tervet, egy logikai sémát kell készítenünk, amely közelebb áll az adatbázis-kezelőhöz. A ma használatos adatbázis-kezelők többsége a relációs adatmodellre épül, ezért az egyed-kapcsolat modellünket át kell konvertálnunk relációs modellre. A relációs modellel már szintén megismerkedtünk.

Ebben a fejezetben azt nézzük meg, hogy hogyan képezzük le az egyed-kapcsolat modellünket relációs adatmodellre. A leképezés tulajdonképpen a két modell ismeretének birtokában „józan ésszel” végig gondolható és kitalálható, heurisztikus szabályok alkalmazását jelenti.

4.1 Az egyedtípusok leképezése

Minden egyes **egyedtípust egy-egy relációnak**, azaz **táblázatnak kell megfeleltetni**. Az erős és gyenge egyedtípusok kezelése kissé eltér egymástól.

A **táblázatokat** – először is – el kell **neveznünk**. A legegyszerűbb, ha ez megegyezik az egyedtípus nevével, ugyanakkor ez nem kötelező. Eldöntendő az egyes vagy többes szám használata, a több szóból álló elnevezések használatának módja (a szóközök kezelése), illetve az ékezetes betűk és egyéb speciális karakterek használata. Logikai sémáról lévén szó, itt még nem kell feltétlenül figyelembe vennünk egy konkrét adatbázis-kezelő (és az operációs rendszer) által megszabott korlátokat, de nem árt. Általában célszerű **rövid, beszédes elnevezéseket** használni; szóközök helyett például _ (aláhúzás karakter) alkalmazni – vagy akár egybe is írhatjuk a többszavas neveket. Nincs kötelezően alkalmazandó szabály. A lényeg, hogy a **táblák elnevezése következetes legyen**.

Példa: Szemelyek_adatai, SzemelyekAdatai, Személyek, Személy, Szemely stb.

4.1.1 Erős egyedtípus

Az **erős egyedtípusból** (vagy egyszerűen csak egyedtípusból) **képzett táblázat oszlopai** az **egyed-kapcsolat modell attribútumai** lesznek. Vannak kivételek, amelyekről a tulajdonságok leképezésénél lesz szó. Az **egyedtípus kulcs tulajdonsá**ga lesz a **táblázat elsődleges kulcsa** 4-1. ábra).

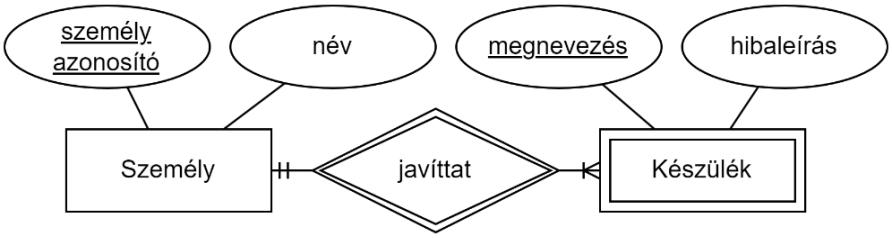
4.1.2 Gyenge egyedtípus

A **gyenge egyedtípus táblázattá alakítása hasonló** az erős egyedtípushoz, azonban itt figyelembe kell venni, hogy ennek a fajta egyedtípusnak nincs saját elsődleges kulcsa,

legfeljebb parciális kulccsal rendelkezik. Ezért **szükségünk van az erős egyedtípusból képzett táblázatra** és hozzá **egy kapcsolatra is** annak érdekében, hogy az erős egyedtípus elsődleges kulcsából (és a gyenge egyedtípus parciális kulcsából) képzett kulccsal azonosítani tudjuk a gyenge egyedtípusból képzett tábla rekordjait (4-1. ábra).

Példa: Egy szerviz adatbázisában személyek és javításra leadott készülékeik adatai találhatók. Az egyes műszaki cikkek csak a személyek egyedi azonosítójával együtt azonosíthatók be. A Személy erős egyedtípus tulajdonságai: személyID, név. A Készülék gyenge egyedtípus tulajdonságai: megnevezés, hiba. Egy személy legalább egy, de egyszerre akár több eszközt is javíttathat és minden készüléknek van egy és csak egy tulajdonosa.

Az egyed-kapcsolat modell:



A relációs séma: Személy (<u>személyID</u>, név)
Készülék (<u>személyID</u>, <u>megnevezés</u>, hiba)

4-1. ábra. Erős egyedtípus (diagram bal oldalán) és gyenge egyedtípus (diagram jobb oldalán) leképezése.

4.1.3 Egyed

Egy-egy egyedet (egyed-előfordulást) a tulajdonságértékei azonosítanak. A relációs séma nem tartalmazza az egyes egyedek konkrét tulajdonságértékeit (ahogyan az egyed-kapcsolat modell sem!), hiszen a séma csak a táblázat szerkezetét írja le. Ha megvalósul a táblázat a kiválasztott relációs adatbázis-kezelőben, akkor az adatfeltöltés után lesznek láthatók a táblázat egyes sorai, azaz rekordjai és ezek adják meg egy-egy egyed tulajdonságait.

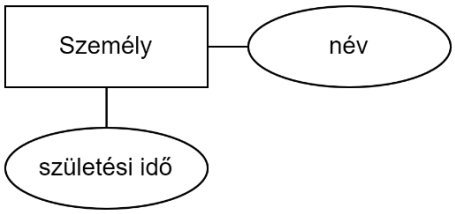
4.2 Egyszerű tulajdonság leképezése

Az **attribútumok**, tulajdonságok **elnevezésével** kapcsolatban ugyanazt lehet elmondani, mint a táblázatok esetében. Alkalmazhatjuk az egyed-kapcsolat modellben használt elnevezést. Itt is eldöntendő az egyes vagy többes szám használata, a több szóból álló elnevezések használatának módja, az ékezetes betűk és egyéb speciális karakterek használata. Célszerű rövid, beszédes elnevezéseket használni. Nincs tehát kötelezően alkalmazandó szabály, **legyünk következetesek** az elnevezésekben.

Egy egyedtípus **egyszerű tulajdonságai** az egyedtípus alapján létrehozott **táblázat oszlopai** (mezői) lesznek (4-2. ábra).

Példa: A Személy egyedtípus egyszerű tulajdonságai: név, születési idő.

Az egyed-kapcsolat modell:



A relációs séma: Személy (név, születési_idő)

Minta:

Személy

név	születési_idő
Alma Alfonz	1991.11.05.
Alma Alfonz	1995.05.03.
Birs Bea	1980.03.05.
Citrom Cecil	1970.10.05.
Dinnye Dénes	1991.11.05.
Egres Emil	2003.11.05.

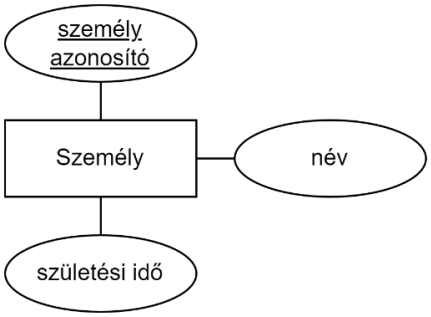
4-2. ábra. Egyszerű tulajdonság leképezése és táblázat mintaadatokkal.

4.3 Kulcs tulajdonság leképezése

A **kulcs tulajdonság** a relációs sémában **elsődleges kulcs** lesz, amit **aláhúzással jelölünk**. Nem kötelező ugyan, de az áttekinthetőség kedvéért célszerű a mezők felsorolását az **elsődleges kulccsal kezdeni** (4-3. ábra, 4-4. ábra).

Példa: A Személy egyedtípus egyszerű tulajdonságai: személy azonosító név, születési idő. A személy azonosító a kulcs tulajdonság.

Az egyed-kapcsolat modell:



A relációs séma: Személy (<u>személyID</u>, név, születési_idő)

Minta:

Személy

<u>személyID</u>	név	születési _idő
27	Alma Alfonz	1991.11.05.
48	Birs Bea	1980.03.05.
52	Citrom Cecil	1970.10.05.
67	Alma Alfonz	1995.05.03.
80	Dinnye Dénes	1991.11.05.
92	Egres Emil	2003.11.05.

4-3. ábra. Kulcs tulajdonság leképezése és táblázat mintaadatokkal.

Példa: A Személy egyedtípus egyszerű tulajdonságai: név, anyja neve, születési idő, foglalkozás. Az első három tulajdonság együtt alkot kulcsot.

Az egyed-kapcsolat modell:

A relációs séma: Személy (<u>név</u>, <u>anyja neve</u>, <u>születési_idő</u>, foglalkozás)

Minta:

Személy

<u>név</u>	<u>születési_idő</u>	<u>anyja_neve</u>	foglalkozás
Alma Alfonz	1991.11.05.	Nagy Natália	tűzoltó
Alma Alfonz	1995.05.03.	Szabó Edit	tanár
Birs Bea	1980.03.05.	Kis Andrea	orvos
Citrom Cecil	1970.10.05.	Varga Katalin	orvos
Dinnye Dénes	1991.11.05.	Szabó Edit	tanár
Egres Emil	2003.11.05.	Nagy Natália	tűzoltó

4-4. ábra. Kulcs tulajdonság leképezése összetett kulcs esetén és táblázat mintaadatokkal.

4.4 Összetett tulajdonság leképezése

Az **összetett tulajdonságot** úgy képezzük le, hogy **csak a részattribútumait vesszük fel a táblázatba**, minden résztulajdonságot egy-egy mezőnek feleltetünk meg (4-5. ábra).

Példa: A Személy egyedtípus tulajdonságai: személy azonosító, név, lakcím. A lakcím összetett tulajdonság, résztulajdonságai az irányítószám, településnév, közterület.

Az egyed-kapcsolat modell:

A relációs séma: Személy (<u>személyID</u>, név, irányítószám, településnév, közterület)
Minta:

Személy

<u>személyID</u>	név	irányítószám	településnév	közterület
27	Alma Alfonz	1024	Budapest	Széchenyi utca
48	Birs Bea	6000	Kecskemét	Rákóczi út
52	Citrom Cecil	9022	Győr	Kossuth tér
67	Alma Alfonz	6720	Szeged	Ady Endre utca
80	Dinnye Dénes	8200	Veszprém	Petőfi utca
92	Egres Emil	1024	Budapest	Széchenyi utca

4-5. ábra. Összetett tulajdonság leképezése és táblázat mintaadatokkal.

4.5 Leszármaztatott tulajdonság leképezése

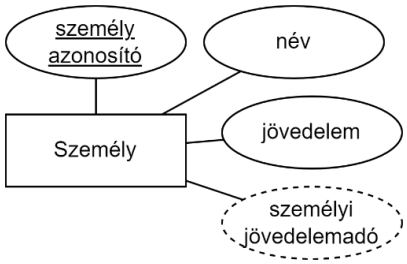
A **leszármaztatott tulajdonság** olyan tulajdonság, amit **más tulajdonság értékéből** valamilyen módon ki tudunk számolni, le tudunk vezetni minden egyes alkalommal, amikor szükségünk van rá. Éppen ezért nem kell fizikailag is eltárolnunk az adatbázisban, így **nem része a relációs sémának** (4-6. ábra).

Abban az esetben viszont, ha a leszármaztatott tulajdonság tulajdonságértékeire gyakran van szükség, a gyakorlatban mégis célszerű tárolni azt az adatbázisban egyszerű tulajdonságként. Az adott követelmények, elvárások döntik tehát el, hogy nem szerepeltetjük (alapeset) és ezzel tárhelyet spórolunk, vagy szerepeltetjük és – mivel csak egyszer kell kiszámoltatni és eltároltatni a számítógéppel – processzoridőt takarítunk meg.

Alapesetben tehát nem vesszük fel a relációs modellbe! Arra viszont ügyeljünk, hogy az a tulajdonság, **amiből leszármaztatjuk, szerepeljen a sémában**!

Példa: A Személy egyedtípus tulajdonságai: személy azonosító, név, bruttó személyi jövedelem, személyi jövedelemadó. Ez utóbbi leszármaztatott tulajdonság, kiszámolható a jövedelemből.

Az egyed-kapcsolat modell:



A relációs séma: Személy (<u>személyID</u>, név, jövedelem)
Minta:

Ezt az oszlopot nem tartalmazza a táblázat. Ha szükséges, számítódik.

Személy

<u>személyID</u>	név	jövedelem (ezer Ft)	SZJA (ezer Ft): jövedelem*0,15
27	Alma Alfonz	400	60
48	Birs Bea	350	52,5
52	Citrom Cecil	310	46,5
67	Alma Alfonz	600	90
80	Dinnye Dénes	650	97,5
92	Egres Emil	400	60

4-6. ábra. Leszármaztatott tulajdonság „leképezése” és táblázat mintaadatokkal.

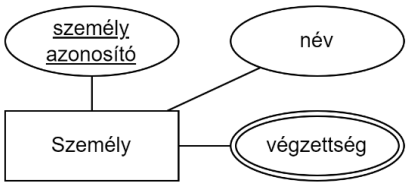
4.6 Többértékű tulajdonság leképezése

A **többértékű tulajdonság** első ránézésre többféleképpen konvertálható. A következő lehetőségek állnak rendelkezésre:

A többértékű attribútumot **egy attribútumnak tüntetjük fel** a relációsémában, mintha egyszerű tulajdonság lenne, de egy-egy cellába több érték bevitelét is megengedjük. Ezzel viszont lehetővé tesszük, hogy egy cella ne csak atomi értéket tartalmazzon. Az így bevitt adatértékekre később sokkal nehezebb lesz rákeresni. Mindezek miatt ez **nem egy szerencsés megoldás, kerülendő**! (4-7. ábra)

Példa: A Személy egyedtípus tulajdonságai: személy azonosító, név, végzettség. Egy személynek akár többféle végzettsége is lehet.

Az egyed-kapcsolat modell:



A relációs séma: Személy (<u>személyID</u>, név, végzettség)

Minta:

Személy

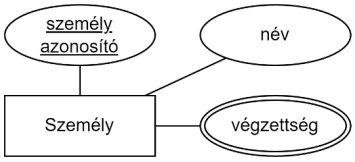
<u>személyID</u>	név	végzettség
27	Alma Alfonz	orvos, közgazdász
48	Birs Bea	jogász
52	Citrom Cecil	pedagógus, fogászati aszisztens, állatorvos
67	Alma Alfonz	közgazdász, informatikus
80	Dinnye Dénes	mérnök, informatikus, jogász
92	Egres Emil	orvos

4-7. ábra. Többértékű tulajdonság leképezése egyszerű tulajdonságként és táblázat mintaadatokkal – kerülendő megoldás.

A többértékű tulajdonság értékeinek felvételéhez **több oszlopot hozhatunk létre** és a táblázat minden egyes cellájába csak egy atomi érték kerül. Ez sem egy optimális megoldás, mert speciális esetektől eltekintve valószínűleg nem fogjuk tudni előre megmondani, hogy egy rekord (egyed) esetében hány különböző értéke lehet a többértékű tulajdonságnak. Ha például csak három oszlopot veszünk fel, nem fogjuk tudni hova beírni az esetleges negyedik értéket. Ha többségében csak egy vagy két érték tartozik egy rekord többértékű tulajdonságához, sok felesleges, üres cellánk lesz. Így ez a megoldás is általában **kerülendő**. (4-8. ábra)

Példa: A Személy egyedtípus tulajdonságai: személy azonosító, név, végzettség. Egy személynek akár többféle végzettsége is lehet.

Az egyed-kapcsolat modell:



A relációs séma: Személy (<u>személyID</u>, név, végzettség1, végzettség2, végzettség3)

Minta:

Személy

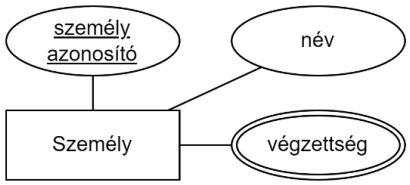
<u>személyID</u>	név	végzettség 1	végzettség 2	végzettség 3	végzettség 4
27	Alma Alfonz	orvos	közgazdász		
48	Birs Bea	jogász			
52	Citrom Cecil	pedagógus	fogászati asszisztens	állatorvos	
67	Alma Alfonz	közgazdász	informatikus		
80	Dinnye Dénes	mérnök	informatikus	jogász	könyvelő
92	Egres Emil	orvos			

4-8. ábra. Többértékű tulajdonság leképezése n darab mező hozzáadásával és táblázat mintaadatokkal – kerülendő megoldás.

A többértékű attribútumot **egy attribútumként tüntetjük** fel, **egyszerű tulajdonságként**, és egy-egy cellába csak egy elemi értéket viszünk be. **Több érték beviteléhez újabb sort kell felvennünk**, ahol csak a „többértékű mező” értéke különbözik, a többi mező értékét megtöbbszörözzük. Ez felesleges adatismétlődéssel, redundanciával jár. Az eredeti elsődleges kulcs értékét is meg kell többszöröznünk, így az nem maradhat elsődleges kulcs, tehát egy új oszlopot is fel kell vennünk elsődleges kulcsnak. Mindezek miatt ez sem tökéletes megoldás, **kerülendő**. (4-9. ábra)

Példa: A Személy egyedtípus tulajdonságai: személy azonosító, név, végzettség. Egy személynek akár többféle végzettsége is lehet. A személy azonosító nem lehet elsődleges kulcs, mert többször is előfordulhat, ha valakinek több végzettsége van, ezért felveszünk egy újabb mezőt (sorszám), amelynél gondoskodunk arról, hogy minden sorban más értéket vegyen fel.

Az egyed-kapcsolat modell:



A relációs séma: Személy (<u>sorszám</u>, személyID, név, végzettség)

Minta:

Személy

<u>Sorszám</u>	személyID	név	végzettség
1	27	Alma Alfonz	orvos
2	27	Alma Alfonz	közgazdász
3	48	Birs Bea	jogász
4	52	Citrom Cecil	pedagógus
5	52	Citrom Cecil	fogászati asszisztens
6	52	Citrom Cecil	állatorvos
7	67	Alma Alfonz	közgazdász
8	80	Dinnye Dénes	mérnök
9	80	Dinnye Dénes	informatikus
10	80	Dinnye Dénes	jogász
11	80	Dinnye Dénes	könyvelő
12	92	Egres Emil	orvos

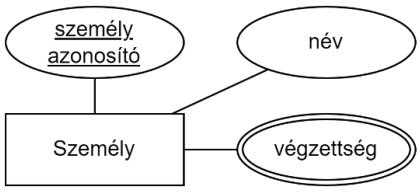
4-9. ábra. Többértékű tulajdonság leképezése egyszerű tulajdonságként, rekordtöbbszörözéssel és táblázat mintaadatokkal – kerülendő megoldás.

Többértékű tulajdonság leképezésénél a **helyes, optimális megoldás** az, hogy a **többértékű attribútumot az eredeti táblából kivesszük** és számára egy **külön relációs sémát hozunk létre**, amelyben feltüntetjük a tulajdonságot az eredeti tábla elsődleges kulcsával együtt (4-10. ábra).

Példa: A Személy egyedtípus tulajdonságai: személy azonosító, név, végzettség. Egy személynek akár többféle végzettsége is lehet. A végzettség tulajdonságot külön táblába

tesszük és ebbe a táblába beletesszük az eredeti táblánk elsődleges kulcsát is. Ha valakinek több végzettsége van, akkor a végzettséget tartalmazó táblában többször is elő fog fordulni az azonosítója. Ha egy bizonyos végzettséggel többen is rendelkeznek, a végzettség is többször fordulhat elő. Ezért a végzettséget tartalmazó táblában az elsődleges kulcs a személy azonosítót és a végzettséget is tartalmazó összetett kulcs. A két tábla között 1:N kapcsolat lesz.

Az egyed-kapcsolat modell:



A relációs séma: Személy (<u>személyID</u>, név)
Végzettség (<u>személyID</u>, <u>végzettségmegnevezés</u>)

Minta:

Személy

<u>személyID</u>	név
27	Alma Alfonz
48	Birs Bea
52	Citrom Cecil
67	Alma Alfonz
80	Dinnye Dénes
92	Egres Emil

Végzettség

<u>személyID</u>	<u>végzettségmegnevezés</u>
27	orvos
27	közgazdász
48	jogász
52	pedagógus
52	fogászati asszisztens
52	állatorvos
67	közgazdász
80	mérnök
80	informatikus
80	jogász
80	könyvelő
92	orvos

4-10. ábra. Többértékű tulajdonság leképezése külön táblával és táblázatok mintaadatokkal – az optimális megoldás.

4.7 1:1 kapcsolat leképezése

A **kapcsolatok leképezése** többféleképpen történhet **kardinalitás típusától függően**. Nem mindegy az sem, hogy a kapcsolat **totális vagy parciális**. Egyedtípusok közötti kapcsolatokat **leképezhetünk**

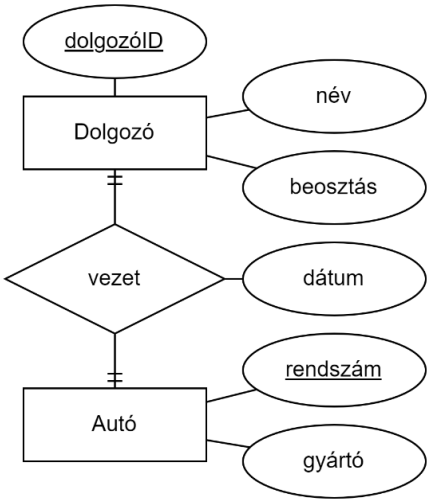
- egyedtípusok egy táblába történő összevonásával,
- bizonyos esetekben úgy, hogy az **egyik táblát kiegészítjük a kapcsolódó tábla elsődleges kulcsával** (speciális esetekben nem kulcs attribútumokat is összekapcsolhatunk, de ez nem ajánlott)
- és minden esetben kapcsolat táblák létrehozásával.

Ha **mindkét oldalon totális az 1:1 kapcsolat**, célszerű a két kapcsolódó egyedtípust úgy leképezni, hogy **egyetlen táblázatot alakítunk ki**, amely az **összes tulajdonságot tartalmazza** a két egyedhalmazból. Ha a két egyedtípusban különböző a kulcs, akkor a kettő közül tetszőlegesen választunk elsődleges kulcsot (4-11. ábra).

Abban az esetben, ha a két egyedtípus közül valamelyik részt vesz más kapcsolatban is, akkor viszont érdemes megtartani a két táblát és közöttük úgy létrehozni a kapcsolatot, hogy az egyik tábla elsődleges kulcsát hozzáírjuk a másik táblához idegen kulcsként.

Példa: Dolgozókról és szolgálati autóikról vezetünk nyilvántartást. A dolgozókról az azonosítójukat, nevüket és beosztásukat, az autókról a rendszámot és a gyártót tároljuk. Minden dolgozó egy autót kap és minden autóhoz egy dolgozó van hozzárendelve. Tároljuk azt is, hogy mióta vezeti egy adott dolgozó az autóját.

Az egyed-kapcsolat modell:



A relációs séma: Dolgozó (dolgozóID, név, beosztás, rendszám, gyártó, dátum)
Minta:

Dolgozó

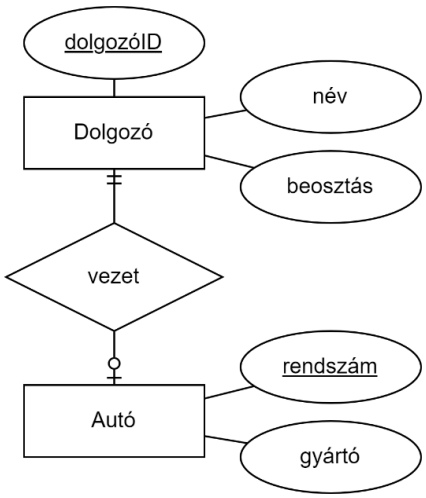
dolgozoID	név	beosztás	rendszám	gyártó	dátum
27	Alma Alfonz	könyvelő	ABC111	Opel	2023.10.05.
48	Birs Bea	jogász	DEF222	BMW	2024.01.15.
52	Citrom Cecil	titkárnő	GHI333	Opel	2023.05.08.
67	Alma Alfonz	irodavezető	JKL444	Opel	2023.10.05.
80	Dinnye Dénes	informatikus	MNO555	Suzuki	2023.10.05.
92	Egres Emil	üzemorvos	PQR666	Skoda	2024.01.15.

4-11. ábra. 1:1 totális-totális kapcsolat leképezése és táblázat mintaadatokkal.

Ha az **egyik oldalon totális, a másik oldalon parciális az 1:1 kapcsolat**, akkor az az optimális megoldás, hogy a **totális részvételű oldal tábláját kiegészítjük a parciális részvételű oldal táblájának elsődleges kulcsával**, mint idegen kulccsal. Abban az esetben, ha mindkét táblában azonosak a kulcsok, a parciális oldali tábla elsődleges kulcsa idegen kulcs is lesz (4-12. ábra).

Példa: Dolgozókról és szolgálati autóikról vezetünk nyilvántartást. A dolgozókról az azonosítójukat, nevüket és beosztásukat, az autókról a rendszámot és a gyártót tároljuk. Egy dolgozó legfeljebb egy autót kaphat (lehet olyan dolgozó, aki nem kap autót), és minden autóhoz hozzá van rendelve egy dolgozó.

Az egyed-kapcsolat modell:



A relációs séma: Dolgozó (dolgozóID, név, beosztás)
Autó (rendszám, gyártó, személyID)

Minta:

Dolgozó

dolgozoID	név	beosztás
27	Alma Alfonz	könyvelő
48	Birs Bea	jogász
52	Citrom Cecil	titkárnő
67	Alma Alfonz	irodavezető
80	Dinnye Dénes	informatikus
92	Egres Emil	üzemorvos

Autó

rendszám	gyártó	dátum	dolgozoID
ABC111	Opel	2023.10.05.	27
DEF222	BMW	2024.01.15.	48
GHI333	Opel	2023.05.08.	52

4-12. ábra. 1:1 totális-parciális kapcsolat leképezése és táblázat mintaadatokkal.

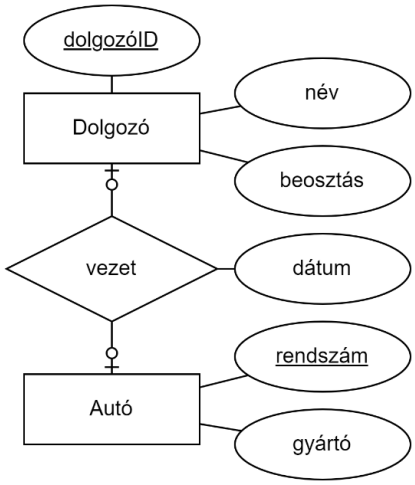
Ha **mindkét oldalon parciális az 1:1 kapcsolat**, akkor célszerű egy új, **kapcsolat táblát létrehozni**, beleírva a két összekapcsolandó tábla elsődleges kulcsait idegen kulcsként (4-13. ábra). A kapcsolat táblában az **egyik idegen kulcsot tetszőlegesen elsődleges kulcsként** jelöljük. **Ugyanez** a megoldás **ajánlott totális-parciális kapcsolat esetén is, ha a kapcsolatnak is van tulajdonsága**.

Lehetne alkalmazni azt a megoldást is, hogy kapcsolat tábla nélkül, úgy hozzuk létre a kapcsolatot, hogy az egyik tábla elsődleges kulcsát hozzáírjuk a másik táblához, de így számos

nem kitöltött, üres cellát kapnánk az idegen kulcsban, ami tárolási szempontból nem szerencsés.

Példa: Dolgozókról és szolgálati autóikról vezetünk nyilvántartást. A dolgozókról az azonosítójukat, nevüket és beosztásukat, az autókról a rendszámot és a gyártót tároljuk. Egy dolgozó legfeljebb egy autót kaphat (lehet olyan dolgozó, aki nem kap autót). Egy autóhoz legfeljebb egy dolgozót rendelünk hozzá, és lehetnek olyan autók is, amelyeket még nem használ senki.

Az egyed-kapcsolat modell:



A relációs séma: Dolgozó (dolgozóID, név, beosztás) Autó (rendszám, gyártó)
Vezet (dolgozóID, rendszám, dátum) vagy
Vezet (rendszám, dolgozóID, dátum)

Minta:

Dolgozó

dolgozoID	név	beosztás
27	Alma Alfonz	könyvelő
48	Birs Bea	jogász
52	Citrom Cecil	titkárnő
67	Alma Alfonz	irodavezető
80	Dinnye Dénes	informatikus
92	Egres Emil	üzemorvos

Autó

rendszám	gyártó
ABC111	Opel
DEF222	BMW
GHI333	Opel

Vezet

dolgozoID	rendszám	dátum
27	ABC111	2023.10.05.
48	DEF222	2024.01.15.
52	GHI333	2023.05.08.

4-13. ábra. 1:1 parciális-parciális kapcsolat leképezése és táblázat mintaadatokkal.

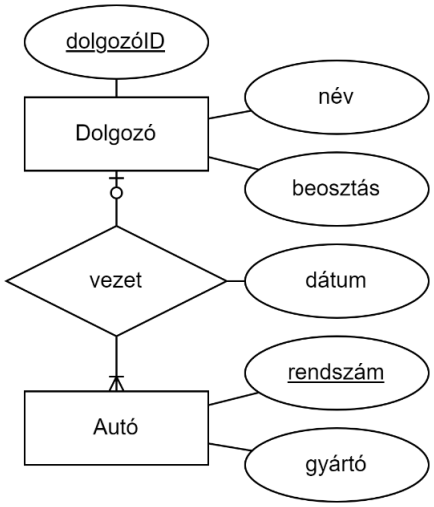
4.8 1:N kapcsolat leképezése

A kapcsolat leképezésekor itt is célszerű figyelembe venni, hogy totális, vagy parciális az N-oldal részvétele.

Totális N-oldali kapcsolat esetén az ajánlott megoldás, hogy **az N-oldali tábla mezőit kiegészítjük az 1-oldal elsődleges kulcsával,** mint idegen kulccsal. Ha a kapcsolatnak vannak esetleg tulajdonságai, akkor azok is az N-oldali táblába kerülnek (4-14. ábra).

Példa: Dolgozókról és szolgálati autóikról vezetünk nyilvántartást. A dolgozókról az azonosítójukat, nevüket és beosztásukat, az autókról a rendszámot és a gyártót tároljuk. Minden dolgozó kap legalább egy szolgálati autót, de kaphat akár többet is. Egy autót viszont csak egy dolgozó használhat.

Az egyed-kapcsolat modell:



A relációs séma: Dolgozó (dolgozóID, név, beosztás)
Autó (rendszám, gyártó, dolgozóID, dátum)

Minta:

Dolgozó

dolgozoID	név	beosztás
27	Alma Alfonz	könyvelő
48	Birs Bea	jogász
52	Citrom Cecil	titkárnő
67	Alma Alfonz	irodavezető
80	Dinnye Dénes	informatikus
92	Egres Emil	üzemorvos

Autó

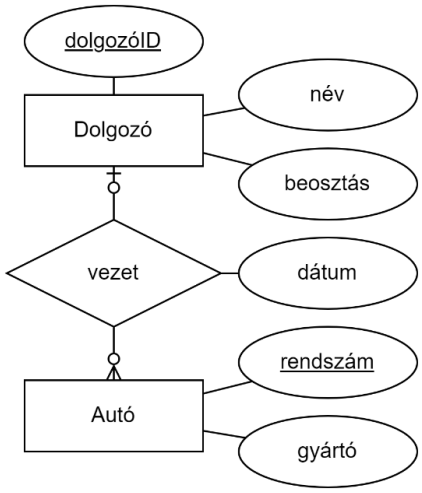
rendszám	gyártó	dolgozoID	dátum
ABC111	Opel	27	2023.10.05.
DEF222	BMW	48	2024.01.15.
GHI333	Opel	52	2023.05.08.
JKL444	Opel	67	2023.10.05.
MNO555	Suzuki	67	2023.10.05.
PQR666	Skoda	80	2024.01.15.

4-14. ábra. 1:N kapcsolat leképezése totális N-oldali kapcsolatnál és táblázat mintaadatokkal.

Parciális N-oldali kapcsolatnál az az ideális megoldás, hogy **kapcsolat táblát hozunk létre**, amelybe beletesszük a két összekapcsolandó táblából származó elsődleges kulcsokat idegen kulcsként. Ebben a táblában **az elsődleges kulcs az N-oldali táblából származó kulcs** lesz (4-15. ábra).

Példa: Dolgozókról és szolgálati autóikról vezetünk nyilvántartást. A dolgozókról az azonosítójukat, nevüket és beosztásukat, az autókról a rendszámot és a gyártót tároljuk. A dolgozók kaphatnak egy, vagy akár több szolgálati autót, illetve lehet olyan dolgozó, aki nem kap autót. Egy autót viszont csak egy dolgozó használhat.

Az egyed-kapcsolat modell:



A relációs séma: Dolgozó (dolgozóID, név, beosztás)
Autó (rendszám, gyártó)
Vezet (rendszám, dolgozóID, dátum)

Minta:

Dolgozó

dolgozoID	név	beosztás
27	Alma Alfonz	könyvelő
48	Birs Bea	jogász
52	Citrom Cecil	titkárnő
67	Alma Alfonz	irodavezető
80	Dinnye Dénes	informatikus
92	Egres Emil	üzemorvos

Autó

rendszám	gyártó
ABC111	Opel
DEF222	BMW
GHI333	Opel
JKL444	Opel
MNO555	Suzuki
PQR666	Skoda

Vezet

rendszám	dolgozoID	dátum
ABC111	27	2023.10.05.
JKL444	67	2023.10.05.
MNO555	67	2023.10.05.

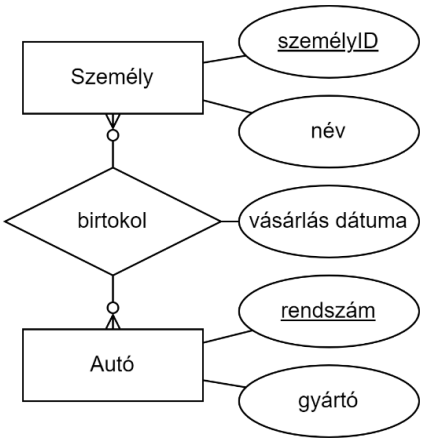
4-15. ábra. 1:N kapcsolat leképezése parciális N-oldali kapcsolatnál és táblázat mintaadatokkal.

4.9 M:N kapcsolat leképezése

M:N kapcsolat leképezésénél – függetlenül attól, hogy totális, vagy parciális kapcsolatról van-e szó – az egyetlen megoldás egy **kapcsolat tábla létrehozása**, benne az **összekapcsolt táblák elsődleges kulcsaival**, mint idegen kulcsokkal, amelyek **összetett elsődleges kulcsot alkotnak**. Ha van a kapcsolatnak tulajdonsága, az is ebbe a kapcsolat táblába kerül (4-16. ábra).

Példa: Személyek és autók adatait tároljuk. A személyekről az azonosítójukat, nevüket, az autókról a rendszámot és a gyártót tároljuk. Egy személynek lehet akár több autója, egy autónak lehet akár több tulajdonosa.

Az egyed-kapcsolat modell:



A relációs séma: Személy (<u>személyID</u>, név)
Autó (<u>rendszám</u>, gyártó)
Birtokol (<u>személyID</u>, <u>rendszám</u>, vásárlásdátum)

Minta:

Személy

<u>személyID</u>	név
27	Alma Alfonz
48	Birs Bea
52	Citrom Cecil
67	Alma Alfonz
80	Dinnye Dénes
92	Egres Emil

Autó

<u>rendszám</u>	gyártó
ABC111	Opel
DEF222	BMW
GHI333	Opel
JKL444	Opel
MNO555	Suzuki
PQR666	Skoda

Birtokol

<u>személyID</u>	<u>rendszám</u>	vásárlásdátum
27	ABC111	2022.01.08.
48	ABC111	2022.01.08.
67	GHI333	2024.03.12.
67	JKL444	2024.03.12.
67	PQR666	2023.05.10.
92	PQR666	2023.05.10.

4-16. ábra. M:N kapcsolat leképezése és táblázat mintaadatokkal.

4.10 N-ed fokú kapcsolat leképezése

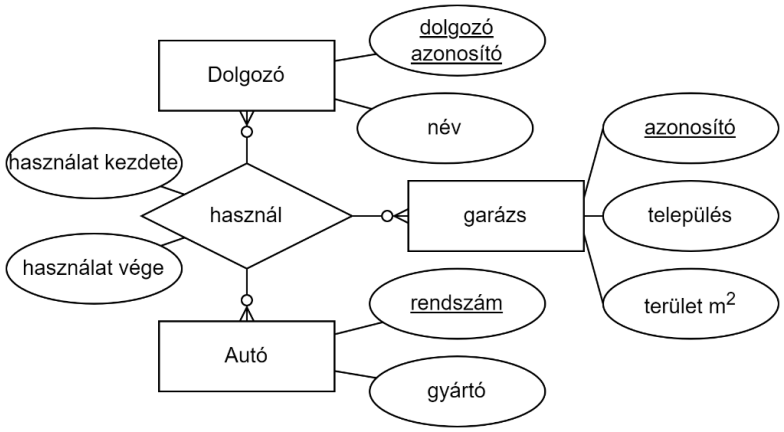
Eljárhatunk úgy, **mint az M:N kapcsolat estében**. A táblák összekapcsolásához létrehozott kapcsolat táblába beleírjuk a táblák elsődleges kulcsait idegen kulcsként, ami itt összetett kulcs lesz. A kapcsolat esetleges tulajdonságai is ide kerülnek (4-17. ábra).

Ha nincs olyan tulajdonságunk, amelyik minden összekapcsolt egyedtípustól függ, **megtehetjük** azt is, hogy **párosával hozzuk létre a kapcsolatokat** az egyedtípusokból

létrehozott táblák között. A párosával létrehozott kapcsolatok ilyenkor – a kapcsolat típusától függően – **visszavezethetők az 1:1, 1:N és M:N kapcsolatok leképezésére**.

Példa: Nyilvántartást vezetünk dolgozók, szolgálati autók és céges garázsok adatairól. A dolgozókról az azonosítójukat, nevüket, az autókról a rendszámot és a gyártót, a garázsokról az azonosítót, a települést és a területet tároljuk. Egy dolgozó akár több autót is használhat, egy autót akár többen is használhatnak. Egy garázst több dolgozó is használhat, egy garázsban több autó is állhat. Nyilvántartjuk azt is, hogy melyik személy melyik autót melyik garázsban mettől meddig tárol.

Az egyed-kapcsolat modell:



A relációs séma: Dolgozó (dolgozóID, név)
Autó (rendszám, gyártó)
Garázs (garázsID, település, terület)
Használ (dolgozóID, rendszám, garázsID, használatkezdete, használatvége)

Minta:

Dolgozó

dolgozoID	név
27	Alma Alfonz
48	Birs Bea
52	Citrom Cecil
67	Alma Alfonz
80	Dinnye Dénes
92	Egres Emil

Autó

rendszám	gyártó
ABC111	Opel
DEF222	BMW
GHI333	Opel
JKL444	Opel
MNO555	Suzuki
PQR666	Skoda

Garázs

garázsID	település	terület
1	Szeged	50
2	Szeged	35
3	Budapest	80
4	Budapest	40
5	Budapest	50
6	Debrecen	60

Használ

dolgozoID	rendszám	garázsID	használatkezdete	használatvége
27	ABC111	1	2023.01.01.	2023.04.30.
27	DEF222	2	2023.02.15.	2024.05.30.
67	JKL444	1	2024.01.01.	2024.03.15.
67	MNO555	3	2023.01.01.	2023.12.31.
80	PQR666	4	2024.01.01.	2024.06.30.

4-17. ábra. 3-ad fokú kapcsolat leképezése és táblázatok mintaadatokkal.

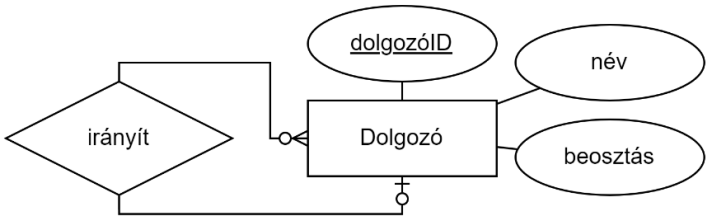
4.11 Rekurzív kapcsolat leképezése

Rekurzív a kapcsolat, ha az egyedtípuson belül van kapcsolat az egyes egyed-előfordulások között.

Rekurzív **1:1 és 1:N kapcsolatnál az egyedtípusból kialakított táblázatot kiegészítjük a táblázat elsődleges kulcsával azonos, de eltérő nevű idegen kulccsal** (4-18. ábra).

Példa: Dolgozók azonosítója, neve, beosztása mellett azt is tároljuk, hogy ki kit irányít. Egy dolgozónak több beosztottja lehet, egy beosztottat legfeljebb egy dolgozó irányíthat.

Az egyed-kapcsolat modell:



A relációs séma: Dolgozó (dolgozóID, név, vezetőID)

Minta:

Dolgozó

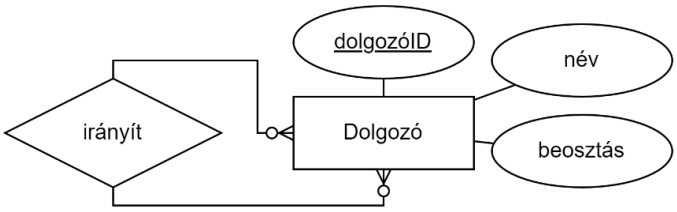
dolgozoID	név	beosztás	főnökID
27	Alma Alfonz	könyvelő	67
48	Birs Bea	jogász	67
52	Citrom Cecil	titkárnő	27
67	Alma Alfonz	irodavezető	
80	Dinnye Dénes	informatikus	67
92	Egres Emil	üzemorvos	

4-18. ábra. 1:N rekurzív leképezése és táblázatok mintaadatokkal.

Rekurzív M:N kapcsolatnál egy **olyan új, kapcsolat táblát hozunk létre**, amely tartalmazza a kapcsolatban részt vevő egyedek elsődleges kulcsait, amelyek itt – a kapcsolat táblában – összetett kulcsot alkotnak (4-19. ábra).

Példa: Dolgozók azonosítója, neve, beosztása mellett azt is tároljuk, hogy ki kit irányít. Egy dolgozó alá több beosztott tartozhat, egy dolgozót több másik dolgozó irányíthat.

Az egyed-kapcsolat modell:



A relációs séma: Dolgozó (dolgozóID, név, beosztás)
Irányít (dolgozóID, vezetőID)

Minta:

Dolgozó

dolgozoID	név	beosztás
27	Alma Alfonz	könyvelő
48	Birs Bea	jogász
52	Citrom Cecil	titkárnő
67	Alma Alfonz	irodavezető
80	Dinnye Dénes	informatikus
92	Egres Emil	üzemorvos

Irányít

dolgozoID	főnökID
52	27
52	48
52	67
52	80
27	67
48	67
80	67

4-19. ábra. M:N rekurzív leképezése és táblázatok mintaadatokkal.

4.12 A leképezési szabályok összefoglalása

A leképezési, konvertálási szabályok összefoglalását tartalmazza a 4-1. táblázat.

4-1. táblázat. Egyed-kapcsolat modell leképezése relációs modellre.

Ami az egyed-kapcsolat modellben…	**az a relációs modellben…**
erős egyedtípus	táblázat (reláció)
gyenge egyedtípus	táblázat (reláció), amelyben szerepel az azonosító reláció elsődleges kulcsa és – ha van, akkor a gyenge reláció parciális kulcsa
egyed	rekord (sor)
egyszerű tulajdonság	egy mező (oszlop)
kulcs tulajdonság	elsődleges kulcs
összetett tulajdonság	résztulajdonságonként egy mező (oszlop)
leszármaztatott tulajdonság	nem szerepel egyik táblázatban sem, mivel szükség esetén kiszámolható
többértékű tulajdonság	egy táblázat, benne a többértékű tulajdonsággal és az eredeti táblázatból származó elsődleges kulcs (mint idegen kulcs)
1:1 kapcsolat	egy egyesített tábla az eredeti táblák mezőivel, vagy két tábla, az egyikben a másik tábla elsődleges kulcsa idegen kulcsként, vagy két tábla egy kapcsolat táblával kiegészítve, amelyben az összekapcsolni kívánt táblák elsődleges kulcsai vannak idegen kulcsként
1:N kapcsolat	két tábla, az egyikben a másik tábla elsődleges kulcsa idegen kulcsként, vagy két tábla egy kapcsolat táblával kiegészítve, amelyben az összekapcsolni kívánt táblák elsődleges kulcsai vannak idegen kulcsként
M:N kapcsolat	két tábla egy kapcsolat táblával kiegészítve, amelyben az összekapcsolni kívánt táblák elsődleges kulcsai vannak idegen kulcsként
N-ed fokú kapcsolat	n darab tábla egy kapcsolat táblával kiegészítve, amelyben az összekapcsolni kívánt táblák elsődleges kulcsai vannak idegen kulcsként, vagy n darab tábla táblapáronként létrehozott kapcsolat táblákkal kiegészítve, amelyekben az összekapcsolni kívánt táblák elsődleges kulcsai vannak idegen kulcsként
rekurzív kapcsolat	egy tábla, amelyben az elsődleges kulcs még egyszer szerepel másik néven idegen kulcsként, vagy két tábla, az egyikben az egyedek tulajdonságai, a másik táblában az összekapcsolni kívánt rekordok elsődleges kulcsai összetett kulcsként és idegen kulcsként

Forrás: Tímár et al. (1997) alapján a szerző szerkesztése.

4.13 Összefoglalás

Az adatbázis-tervezés során az igények összegyűjtését és a specifikációt követően a koncepcionális tervezés keretében készül el a koncepcionális terv, amelyre az egyed-kapcsolat modell, illetve egyed-kapcsolat diagram kiválóan megfelel.

Bár ezzel a modellezési módszerrel gyorsan papírra vethetjük az adatbázissal kapcsolatos elképzeléseinket – nincs olyan adatbázis-kezelő rendszer, amelyik az elkészült diagramot értelmezni tudná. Ezért szükséges egy logikai modell, amely segít a terv implementálásában, megvalósításában.

Abban az esetben, ha relációs adatbázis-kezelő rendszer mellett döntünk, az egyed-kapcsolat modellünket relációs adatmodellre kell átalakítanunk. A leképezés tulajdonképpen a két modell ismeretének birtokában „józan ésszel” végiggondolható és kitalálható, heurisztikus szabályok alkalmazását jelenti.

Az egyedtípusok és a tulajdonságok leképezése relációs adatmodellre a következőképpen történik:

- Az egyedtípusokból egyedi névvel ellátott táblázatokat alakítunk ki.
- Az egyedtípusok tulajdonságai a táblázatok oszlopai, mezői lesznek, amelyeket – táblázat szinten – szintén egyedi nevekkel látunk el.
- Minden egyes táblázatban kell legyen aláhúzással jelölt elsődleges kulcs, ami megfelel az egyed-kapcsolat modell egyedhalmazaiban a kulcs tulajdonságoknak.
- Az összetett tulajdonságoknál csak a résztulajdonságokat vesszük fel a táblázatba.
- A leszármaztatott tulajdonságokat (általában) kihagyjuk a táblázatból, hiszen azok más tulajdonságok értékeiből kiszámolhatók.
- A többértékű tulajdonságból külön táblát képezünk, amelyet hozzákapcsolunk az eredeti táblához annak elsődleges kulcsa felhasználásával.

A kapcsolatok leképezése többféleképpen történhet a kardinalitás típusától függően, illetve attól, hogy totális vagy parciális kapcsolatról van-e szó:

- Összevonjuk a kapcsolódó táblákat egyetlen táblába. Ez a módszer ajánlott 1:1 totális kapcsolatnál.
- Megtartjuk a két táblát és az egyik tábla elsődleges kulcsát hozzáírjuk a másik táblához idegen kulcsként. Ez a megoldás 1:1 totális-parciális kapcsolatnál és 1:N totális N-oldali kapcsolat esetén ajánlott.
- Kapcsolat táblát alakítunk ki a kapcsolódó táblák között. Ezt a megoldást 1:1 parciális-parciális kapcsolatnál célszerű alkalmazni vagy ha 1:1 kapcsolatnak – legyen az totális vagy parciális – tulajdonsága is van. Ez a használható megoldás 1:N parciális N-oldali kapcsolat, M:N kapcsolat, N-ed fokú kapcsolat és rekurzív M:N kapcsolat esetén is.

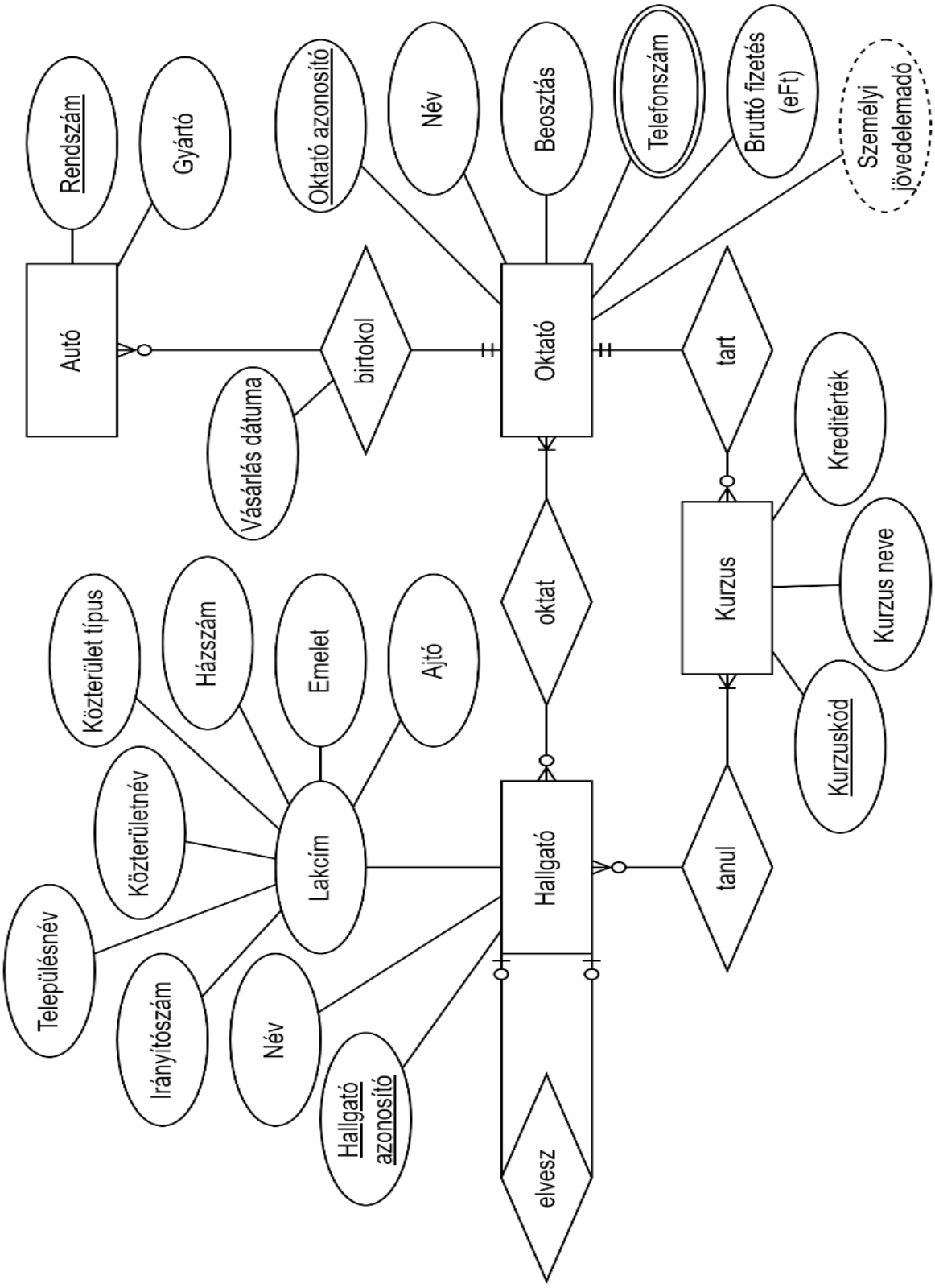
- Kiegészíthetjük ugyanazt a táblát az elsődleges kulcsával idegen kulcsként. Ezt a leképezést rekurzív 1:1 és rekurzív 1:N kapcsolat esetén alkalmazhatjuk.

4.14 Ellenőrző kérdések

1. Miből mi lesz az egyed-kapcsolat modellből a relációs modellre konvertálás során?
2. Hogyan történik az egyedtípusok leképezése?
3. Hogyan alakítható át az erős egyed?
4. Hogyan alakítható át a gyenge egyed?
5. Hogyan konvertálható az egyszerű tulajdonság?
6. Hogyan konvertálható a kulcs tulajdonság?
7. Hogyan konvertálható az összetett tulajdonság?
8. Hogyan konvertálható a leszármaztatott tulajdonság?
9. Hogyan konvertálható a többértékű tulajdonság?
10. Hogyan alakítható át 1:1 kapcsolat?
11. Hogyan alakítható át 1:N kapcsolat?
12. Hogyan alakítható át M:N kapcsolat?
13. Hogyan alakítható át az N-ed fokú kapcsolat?
14. Hogyan alakítható át a rekurzív kapcsolat?

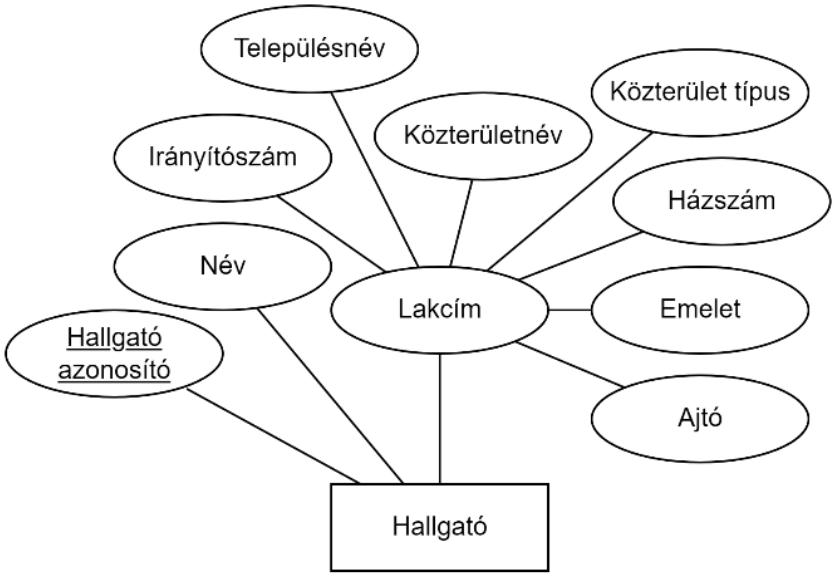
4.15 Feladat

Értelmezze az alábbi egyed-kapcsolat modellt (4-20. ábra) és alakítsa át relációs adatmodellre!



4-20. ábra. Egyed-kapcsolat diagram.

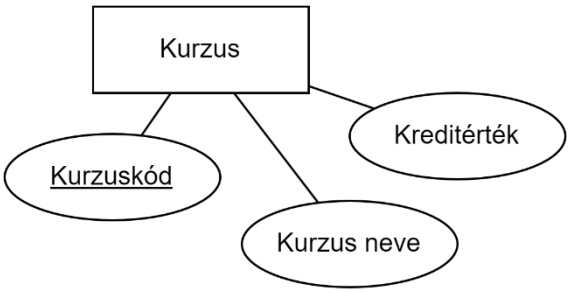
Megoldás:



A Hallgató egyedtípus tulajdonságai:

- hallgató azonosító: kulcs tulajdonság, elsődleges kulcs a relációs modellben
- név: egyszerű tulajdonság
- lakcím: összetett tulajdonság, nem kerül be a relációs modellbe
- irányítószám, településnév, közterületnév, közterület típus, házszám, emelet, ajtó: a lakcím résztulajdonságai, egyszerű tulajdonságokként kerülnek be a relációs modellbe

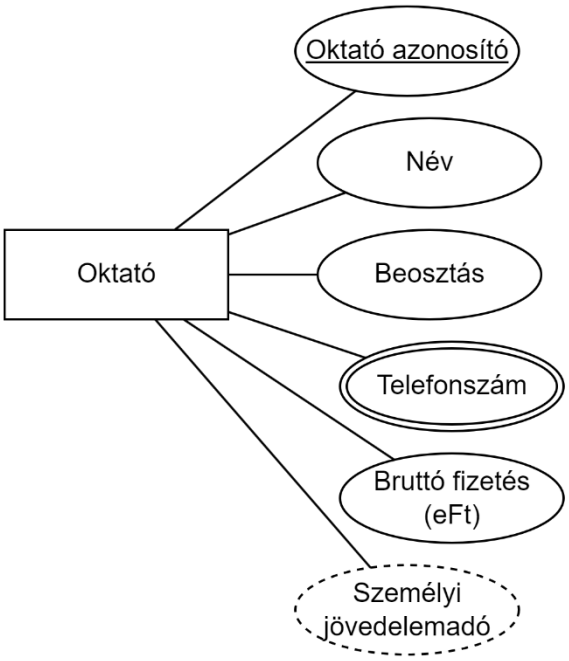
R1: Hallgató (hallgató azonosító, név, irányítószám, településnév, közterületnév, közterület_típus, házszám, emelet, ajtó)



A kurzus egyedtípus tulajdonságai:

- kurzuskód: kulcs tulajdonság, elsődleges kulcs a relációs modellben
- kurzus neve: egyszerű tulajdonság
- kreditérték: egyszerű tulajdonság

R2: Kurzus (kurzuskód, kurzus_neve, kreditérték)

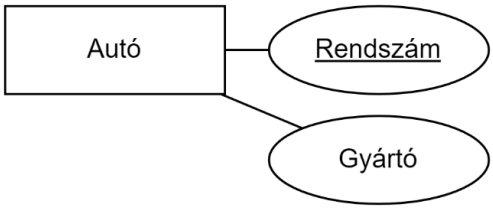


Az oktató egyedtípus tulajdonságai:

- oktató azonosító: kulcs tulajdonság, elsődleges kulcs a relációs modellben
- név: egyszerű tulajdonság
- beosztás: egyszerű tulajdonság
- bruttó fizetés: egyszerű tulajdonság
- telefonszám: többértékű tulajdonság, külön tábla lesz a relációs modellben
- személyi jövedelemadó: leszármaztatott tulajdonság, nem kerül be a relációs modellbe

R3: Oktató (oktató_azonosító, név, beosztás,bruttó_fizetés)

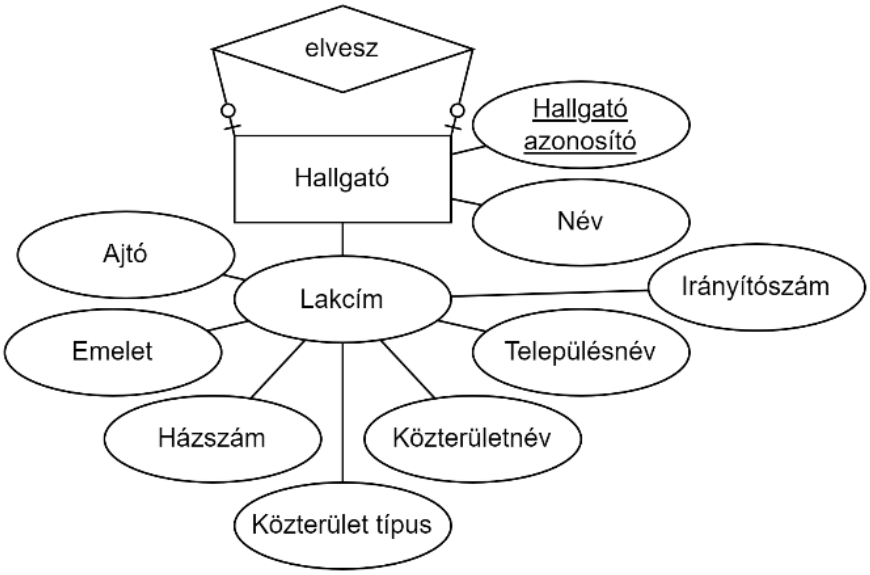
R4: Telefon (oktató_azonosító, telefonszám)



Az Autó egyedtípus tulajdonságai:

- rendszám: kulcs tulajdonság, elsődleges kulcs a relációs modellben
- gyártó: egyszerű tulajdonság

R5: Autó (rendszám, gyártó)

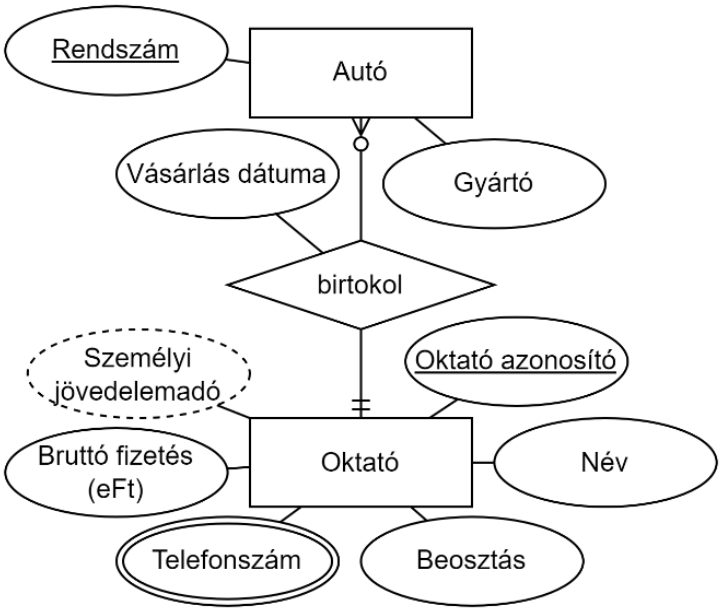


Elvesz kapcsolat:

- 1:1 rekurzív parciális kapcsolat hallgatók között
- egy hallgatónak legfeljebb egy hallgató házastársa lehet (parciális kapcsolat)

A Hallgató relációhoz hozzáírjuk a házastárs azonosítóját (hallgató azonosító más néven, idegen kulcsként)

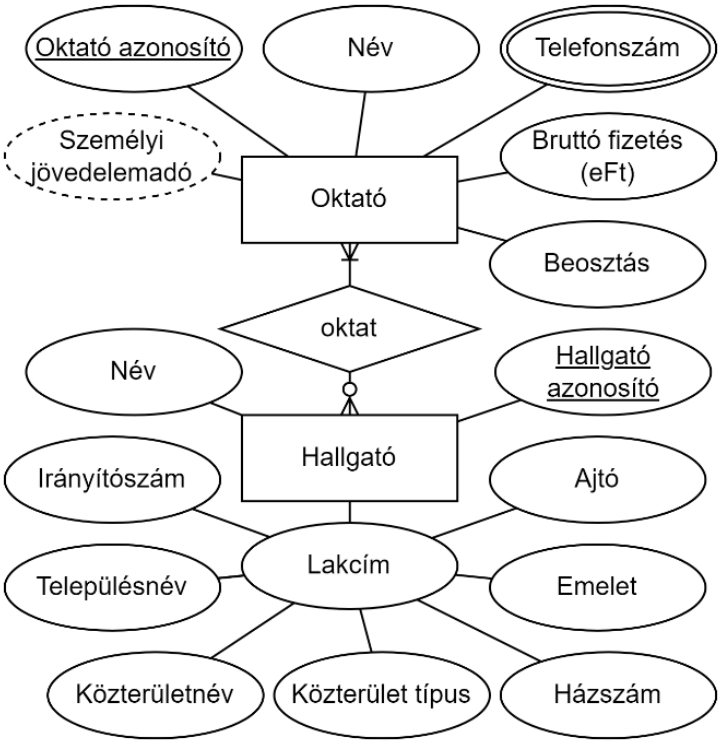
R1 -> **R6**: Hallgató (hallgató azonosító, név, irányítószám, településnév, közterületnév, közterület_típus, házszám, emelet, ajtó, házastárs_azonosító)



Birtokol kapcsolat:

- 1:N kapcsolat az Oktató és az Autó között
- egy oktatónak több autója lehet (parciális kapcsolat)
- egy autónak legalább 1 és csakis egy tulajdonosa van (totális kapcsolat)
- a kapcsolatnak van egy egyszerű tulajdonsága

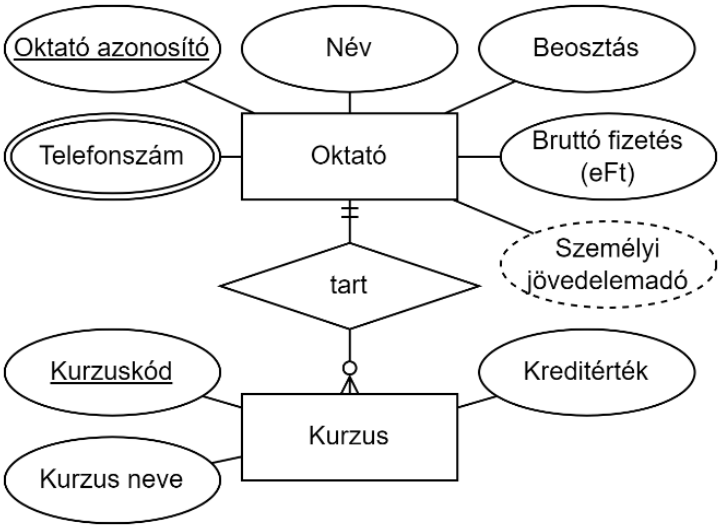
R7: Birtokol (<u>rendszám</u>, oktató_azonosító, vásárlás_dátuma)



Oktat kapcsolat:

- M:N kapcsolat az Oktató és a Hallgató között
- egy oktató több hallgatót taníthat (parciális kapcsolat)
- egy hallgatót legalább egy, de akár több oktató is taníthat (totális kapcsolat)

R8: Oktat (<u>oktató_azonosító</u>, <u>hallgató_azonosító</u>)

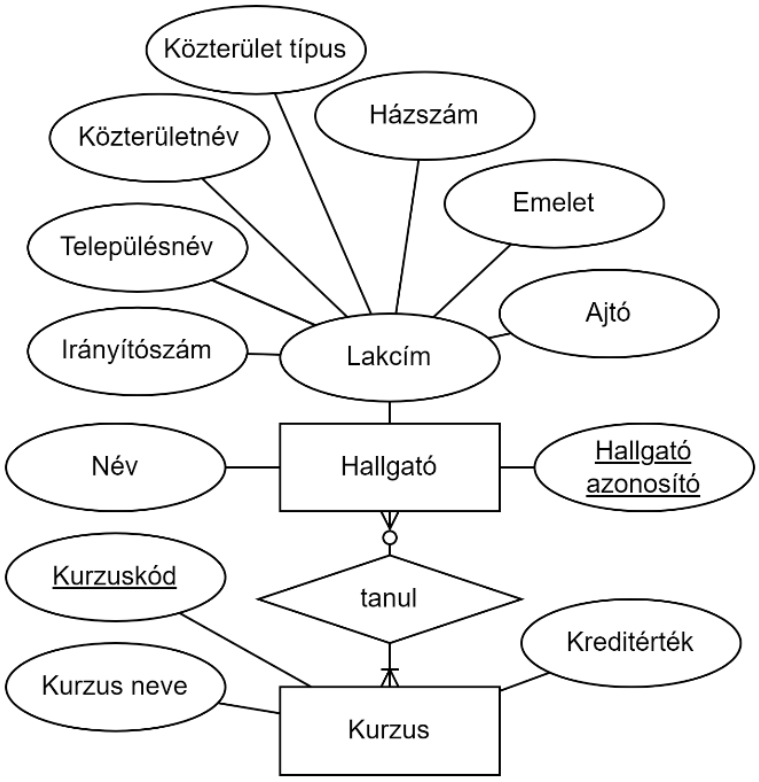


Tart kapcsolat:

- 1:N kapcsolat az Oktató és a Kurzus között
- egy oktató akár több kurzust is tarthat (parciális kapcsolat)
- egy kurzusnak legalább egy és csakis egy oktatója van (totális kapcsolat)

R9: Tart (kurzuskód, oktató_azonosító) De, mivel egy kurzusnak csak egy oktatója van, nem kell külön tábla, a kurzuskódot betehetjük az Oktató táblába idegen kulcsként. Így R9 és R2 relációk helyett:

R9, R2 -> **R10**: Kurzus (kurzuskód, kurzus_neve, kreditérték, oktató_azonosító)



Tanul kapcsolat:

- M:N kapcsolat a Hallgató és a Kurzus között
- egy hallgató legalább egy, de akár több kurzuson is tanulhat (totális kapcsolat)
- egy kurzuson akár több hallgató is részt vehet (parciális kapcsolat)

R11: Tanul (hallgató_azonosító, kurzuskód)

A következő relációs sémák alkotják az adatbázissémát (**R6**, **R3**, **R4**, **R5**, **R7**, **R8**, **R10**, **R11**):

Hallgató (hallgató_azonosító, név, irányítószám, településnév, közterületnév, közterület_típus, házszám, emelet, ajtó, házastárs_azonosító)

Oktató (oktató_azonosító, név, beosztás,bruttó_fizetés)

Telefon (oktató_azonosító, telefonszám)

Autó (rendszám, gyártó)

Birtokol (rendszám, oktató_azonosító, vásárlás_dátuma)

Oktat (oktató_azonosító, hallgató_azonosító)

Kurzus (kurzuskód, kurzus_neve, kreditérték, oktató_azonosító)

Tanul (hallgató_azonosító, kurzuskód)

Fogalomtár

Fogalom	Magyarázat
Adatbázis Database	Általában véve olyan adatállomány, amely egy adatbázis-kezelő rendszerrel hozható létre és érhető el. Véges számú egyed-előfordulásnak, azok egyenként is véges számú tulajdonságértékének és kapcsolat-előfordulásainak az adatmodell szerint szervezett együttese. Olyan integrált adatszerkezet, amely több különböző objektum előfordulási adatait adatmodell szerint szervezetten, tartósan tárolja olyan segédinformációkkal, metaadatokkal együtt, melyek a hatékonyság, integritásőrzés, adatvédelem biztosítását szolgálják. Káros és felesleges redundancia nélkül közösen tárolt, egymással kapcsolatban lévő adatok halmaza, amelynek alapvető célja egy vagy több alkalmazás optimális kiszolgálása.
Adatmodell Data model	Egy formális rendszer, amely az adatok szerkezetét és az azok közötti kapcsolatokat írja le. Célja, hogy megkönnyítse az adatok szervezését, tárolását, visszakeresését és kezelését. Alkalmazásával az adatokat logikus és konzisztens formában lehet ábrázolni. Az adatmodellek közé tartozik az egyed-kapcsolat és a relációsadatmodell.
Adatbázis-kezelő rendszer Database Management System (DBMS)	Olyan szoftver, amely lehetővé teszi az adatok tárolását, visszakeresését, frissítését és kezelését egy strukturált és szervezett módon.
Adatbázisséma Database schema	Leírja az adatbázis teljes struktúráját és a különböző elemek közötti kapcsolatokat. Relációs sémák összessége. Lásd: Relációs séma
Attribútum Attribute	A valóság dolgainak azon jellemzői, amelyekkel az egyes egyedeket leírjuk.
Egyed Entity	A valós világban létező, fogalmi vagy fizikai léttel rendelkező dolog, amelyet tulajdonságokkal akarunk leírni.
Egy-a-többhöz kapcsolat (1:N) One-to-many relationship	Egyed-kapcsolat modellben az egyik egyedtípus egyedei több egyeddel tarthatnak kapcsolatot a másik egyedtípusból, de a másik egyedtípus egyedei csak egy egyedhez kapcsolódhatnak. Relációs modellben az egyik táblázat rekordjai több rekorddal tarthatnak kapcsolatot a másik táblából, de a másik tábla rekordjai csak egy rekordhoz kapcsolódhatnak.
Egy-az-egyhez kapcsolat (1:1) One-to-one relationship	Egyed-kapcsolat modellben a kapcsolatban mindkét egyedtípus egyed-előfordulásai csak egyetlenegy egyed-előforduláshoz rendelődnek a másik egyedtípusból. Relációs modellben a kapcsolatban mindkét tábla rekordjai csak egyetlenegy rekordhoz rendelődnek a másik táblából.
Egyedhalmaz Entity set	-> Egyedtípus
Egyedtípus Entity type	Valamilyen szempontból összetartozó egyedek összessége.
Egyed-kapcsolat modell Entity-relationship model (ER-model)	Az adatbázisok logikai struktúrájának tervezésére szolgál. Grafikus megjelenítéssel teszi lehetővé az adatok és az azok közötti kapcsolatok érthető ábrázolását.
Egyed-kapcsolat diagram Entity-relationship diagram (ERD)	-> Egyed-kapcsolat modell

Fogalom	Magyarázat
Elsődleges kulcs Primary key	A jelölt kulcsok közül kiválasztott kulcs, amely alkalmas a rekordok (sorok) egyértelmű megkülönböztetésére, azonosítására. Szokás elemi kulcsnak is nevezni.
Idegen kulcs Foreign key	Olyan mező vagy mezőcsoport, amely egy másik táblázat elsődleges kulcsára hivatkozik, így biztosítva a táblák összekapcsolását.
Jelölt kulcs Candidate key	Minimális szuperkulcs, amelyben nincs felesleges tulajdonság.
Kapcsolat Relationship	Az egyedek közötti viszonyok fogalmi tükörképei. Meghatározza az egyedek közötti viszonyokat.
Kardinalitás Cardinality	Egyed-kapcsolat modellben a kapcsolatok számossága. Meghatározza, hogy egy egyed hány másik egyedhez kapcsolódhat. Lehet rekurzív (visszaható), 1:1 (egy-az-egyhez), 1:N (egy-a-többhöz), N:M (több-a-többhöz), vagy N-ed fokú, totális (teljes), vagy parciális (opcionális). Relációs modellben a táblázat sorainak száma.
Koncepcionális modell Conceptional model	Az adatbázis-tervezés során használt eszköz az adatbázis logikai szerkezetének magasabb szintű, absztrakt ábrázolása. A felhasználó és a fejlesztő számára egyaránt érthető nyelven fogalmazza meg az adatbázis szerkezetét és működését.
Kulcs tulajdonság Key attribute	Egy-egy egyed azonosítására szolgáló tulajdonság. Relációs modellben számos fajtája van: szuperkulcs, jelölt kulcs, alternatív kulcs, elsődleges kulcs, összetett (elsődleges) kulcs, idegen kulcs.
Logikai adatmodell Logical model	Az adatbázis-tervezésben az adatbázis szerkezetének koncepcionális modellnél részletesebb és formálisabb ábrázolása, amely már figyelembe veszi az adatbázis-kezelő rendszer specifikus követelményeit és korlátait, de még mindig független a fizikai tárolási megvalósítástól.
Metaadat Metadata	Olyan adat, amely más adatokról szolgáltat információkat. Célja az, hogy leírja, azonosítsa és könnyebben kezelhetővé tegye az adatokat. Adatbázisban metaadatok például az adatbázis szerkezetét leíró adatok.
Mező Field	A táblázat egy oszlopa. Minden mező egy adott típusú adatot tartalmazhat (szöveg, szám, dátum, logikai érték stb.).
Modell Model	Olyan rendszert jelöl, amely a valóság egy vizsgált szeletével struktúrában vagy viselkedésben megegyezik, vagy hasonló jelleget mutat. Egyúttal egy eszközrendszer is, amellyel a modell leírható, megadható.
N-ed fokú kapcsolat N-ary relationship	Kettőnél több egyedtípus, illetve tábla közötti kapcsolat.
Opcionális kapcsolat Optional relationship	-> Parciális kapcsolat
Összetett (elsődleges) kulcs Composite (primary) key	Egynél több mezőből álló elsődleges kulcs.
Oszlop Column	-> Mező

Fogalom	Magyarázat
Parciális kapcsolat Optional relationship	Egyed-kapcsolat modellben egy egyedtípus nem minden egyede vesz részt a kapcsolatban. Relációs modellben egy tábla nem minden rekordja kapcsolódik egy másik rekordhoz.
Parciális kulcs Partial key	A gyenge egyedtípust részlegesen azonosító kulcs. Egyed-kapcsolat modellben használt fogalom.
Redundancia Redundancy	Olyan helyzetet jelöl, amikor ugyanaz az adat több helyen is tárolásra kerül és emiatt egy adott információ többszörösen jelen van az adatbázis különböző részeiben.
Relációs séma Relational schema	A tábla (reláció) szerkezetének formális leírása. Tartalmazza a táblázat (reláció) nevét, a táblázatban található attribútumok nevét, megjelöli az elsődleges kulcsot.
Rekurzív kapcsolat Recursive relationship	Olyan kapcsolat, amelyben egy egyedtípus vagy tábla önmagára hivatkozik, azaz egy egyed másik egyedhez kapcsolódik ugyanabban az egyedhalmazban, illetve a rekord sorai más sorokra utalnak ugyanabban a táblában.
Relációs adatmodell Relational model	Az egyik legelterjedtebb adatbázis-modell, amely az adatokat táblázatok (relációk) formájában ábrázolja.
Táblázat Table	Adatokat tároló alapvető struktúra, amely egyedi névvel, legalább egy sorra és legalább egy oszloppal rendelkezik.
Tábla Table	-> Táblázat
Több-a-többhöz kapcsolat (M:N) Many-to-many relationship	Egyed-kapcsolat modellben mindkét egyedtípus egyedei több egyeddel is tarthatják a kapcsolatot a másik egyedtípusból. Relációs modellben mindkét tábla rekordjai több rekorddal is tarthatják a kapcsolatot a másik táblából.
Tulajdonság Attribute	-> Attribútum
Reláció Relation	-> Táblázat
Rekord Record	A táblázat fejléctől különböző sora. Egy egyed tulajdonságait (attribútumait) tárolja.
Sor Row, tuple	-> Rekord
Szuperkulcs Super key	Egy vagy több mező kombinációja, amely egyedileg azonosítja a táblázat minden sorát. Tartalmazhat felesleges mezőket is.
Visszaható kapcsolat Recursive relationship	-> Rekurzív kapcsolat
Totális kapcsolat Mandatory relationship	Egyed-kapcsolat modellben egy egyedtípus minden egyede részt vesz a kapcsolatban. Relációs modellben egy tábla minden rekordja kapcsolódik egy másik rekordhoz.

Ajánlott és felhasznált irodalom

Chen P. P.-Sh. (1976): The Entity-Relationship Model – Toward a Unified View of Data. ACM Transactions on Database Systems, 1 (1): 9–36. https://doi.org/10.1145/320434.320440

Chua, C. E. H., Storey, V. C. (2011): Issues and Guidelines in Modeling Decomposition of Minimum Participation in Entity-Relationship Diagrams. Communications of the Association for Information Systems, 29 (9): 159-184. https://doi.org/10.17705/1CAIS.02909

Codd, E. F. (1970): A relational model of data for large shared data banks. Communications of the ACM, 13: (6): 377-387. https://doi.org/10.1145/362384.362685

Elmasri, R., Navathe, S. B. (2016): Fundamentals of Database Systems, 7th edition. Pearson. 1273 p. 3. fejezet (59-106. oldal), 5. fejezet (149-176 oldal). ISBN: 978-0-13-397077-7 http://debracollege.dspaces.org/bitstream/123456789/168/1/Fundamentals-of-Database-Systems-Pearson-2015-Ramez-Elmasri-Shamkant-B.-Navathe.pdf

Halassy B. (2000): Adatmodellezés. Elmélet és gyakorlat. Budapest. 298 p. 2. fejezet (22-34. oldal). https://mek.oszk.hu/11100/11144/11144.pdf

Szabó B. (2013): Adatbázis fejlesztés és üzemeltetés I. Eszterházy Károly Főiskola, Eger. 187. p. 2. lecke (19-34.oldal, 3. lecke (35-47. oldal), 4. lecke (55-66. oldal), 5. lecke (84-92. oldal)

Timár L., Vígh K., Tátrai J., Szigeti J., Vathy Á., Telekesi É., Vass I., Kocsis T. (1997): Építsünk könnyen és lassan adatmodellt! Veszprémi Egyetem - Műszertechnika-Veszprém Kft. Veszprém. 251 p. 15-100. oldal., 222-235. oldal. ISBN: 9637332685

Ullman, J. D., Widom, J. (2009): Adatbázis-rendszerek: alapvetés. Panem Kiadó. Budapest. 600 p.15-30. oldal, 133-173. oldal. ISBN: 9789635454815

Az oktatóanyag fejlesztését az RRF-2.1.2-21-2022-00012 azonosítószámú,

Komplex Digitális Modellváltás – intelligens fokozatváltás

projekt támogatta.

A projekt Magyarország Helyreállítási és Ellenállóképességi Tervének keretében valósul meg.