Kerepes Tamás – Czinkóczki László

Adatbázisok

Az adatbázis logikai tervezése (Database Design) első rész

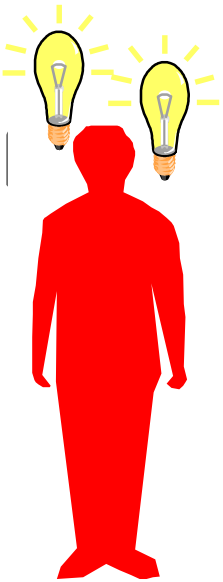
Database Design: miről is van szó?

- Különböztessük meg a fizikai adatbázist (az adatbázis fizikai szerkezetét) és a fogalmi (logikai) adatbázist
- A fizikai adatbázis:
  - A mai üzleti gyakorlatban leginkább relációs szerkezetű
  - Leginkább a számítógép lemezein elhelyezett fájlok
  - Ha relációs, akkor a fájlokban táblák, indexek, nézetek és egyéb fogalmak sokasága tárolódik
  - Egy komplex szoftver – az adatbáziskezelő rendszer (pl. az Oracle RDBMS) működése révén férnek hozzá a felhasználók az adatbázisban tárolt adatokhoz
- A fizikai adatbázist az adatbázis rendszergazdája (DBA) hozza létre és működteti - adminisztrálja. A DBA leginkább egy mérnök, aki „üzemeltet”. Ez is egy csodálatos és izgalmas feladat. De nem a mai „lecke”

A logikai (fogalmi) adatbázis

- A valós (vagy ritkábban a „kitalált”) világ adatait tárolja
- Az adatok részhalmazát tartjuk csupán tárolásra „alkalmasnak”
- Az adatoknak csak egy szűk körét tárolhatjuk
- Azért tároljuk őket, hogy később ebből információt nyerhessünk (mert könnyebb onnan, mint a valóságból megszerezni ugyanazt az információt)
- A tárolandó adatok kiválasztása klasszikus elemzési és modellezési feladat
- Ez a logikai adatbázis tervezés (Logical Database Design). Ma erről lesz szó.
- Aki ezt végzi, azt „elemző/tervezőnek”, adatmodellezőnek is nevezzük

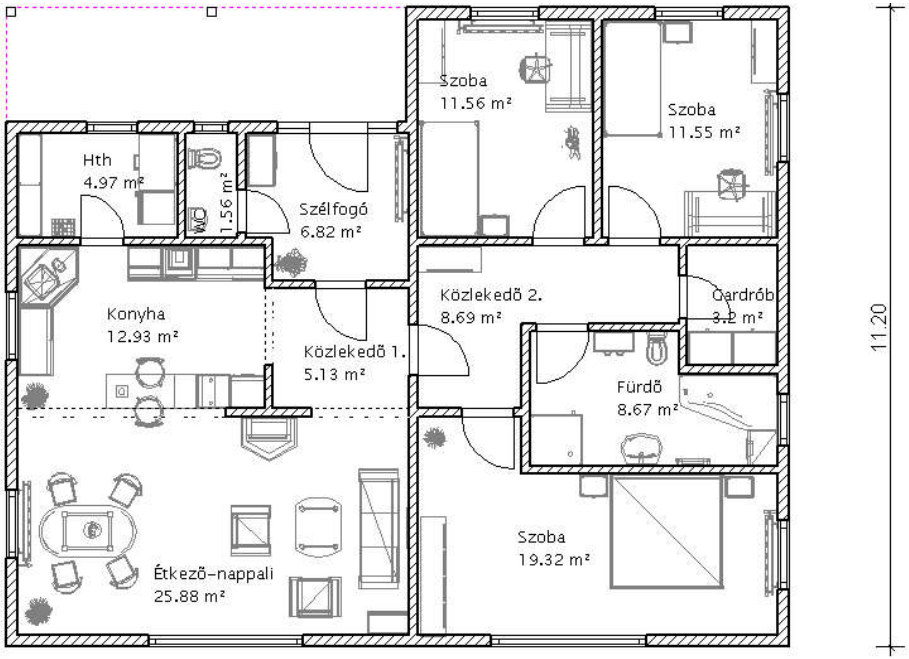
Kis kirándulás az „építészet” területére



Kis kirándulás az „építészet” területére

Családi ház (megrendelésre)

Kis kirándulás az „építészet” területére



Tervdokumentációk

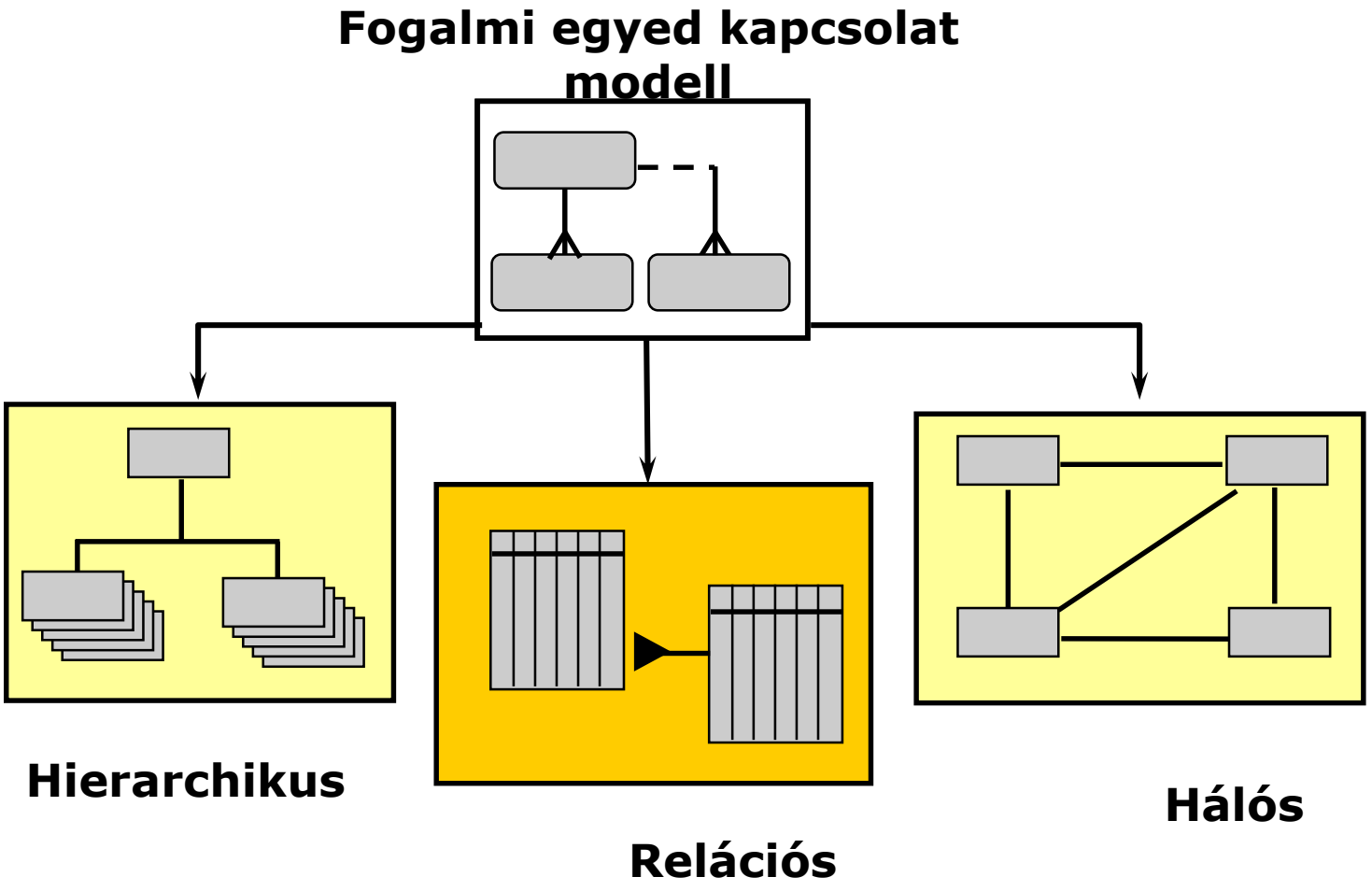
Az elemző/tervező

- Megrendelnek tőle egy „testreszabott” (nem dobozos) informatikai rendszert – illetve annak a megtervezését
- Elemzőként megérti az illető szakterület üzleti szabályait (pl. sok-sok interjún és beszélgetésen keresztül).
- Megállapítja, hogy adatbázis-centrikus a rendszer
- Kiválasztja azt, hogy milyen fogalmakról milyen tényeket kell tárolnunk az adatbázisban
- Megalkotja az adatmodellt. A folyamat végén CREATE TABLE utasítások születnek (kényszerek is)
- Ez egy hosszú, gyakran sok hónapos tervezői munka
- Néha többen – csapatban - végzik
- Ha ezt rosszul csináljuk, vagy nem csináljuk, akkor rendszerint kevésbé hasznos, rövid életű szoftverek keletkeznek (manapság ez gyakori hiba)

Adatmodellek, modellezés

- A tárolandó adatok kiválasztásakor klasszikus modellezési szempontok érvényesülnek
- A vizsgálat szempontjából fontosnak tartott „jellemzőket” tároljuk, a többit elhanyagoljuk
- Jellemző alatt egyaránt értünk „tulajdonságokat” és „kapcsolatokat”
- Az adatbázis a „világ egy darabjának” egy leegyszerűsített képét képes visszaadni (lásd „Gajdos Sándor: Adatbázisok”)
- Manapság leginkább az Egyed-Kapcsolat (Entity-Relationship, E-R) modellt alkalmazzuk erre a célra

Különböző típusú adatbázisok, mint megvalósítási környezet



Az egyed-kapcsolat (E-R) modell „egyed” része

- **Egyed** („Entity”): a valós világban létező, logikai vagy fizikai szempontból saját léttel rendelkező dolog, amelyről adatokat tárolunk
- **Tulajdonság** (attribútum, „attribute”): az, ami az entitásokat jellemzi. Ez, vagy ezek segítségével különböztetjük meg az entitásokat.
- Különbség van az „**Entity**” és az „**Entity Set**” között, de ennek a különbségnek itt csupán elméleti a jelentősége. (A programozásban ez a különbség sokkal fontosabb.)
- Az elemző/tervező választja ki mind az egyedeket, mind az egyedek tulajdonságait
- Ideális esetben ez később (tehát már az implementálás után) ritkán változik

Az E-R modellek alapelemei: Egyedek

EGYED	A rendszer szempontjából jelentős objektum (valós, vagy képzeletbeli). Az egyed neve szerepel a diagramon a dobozban nagy betűkkel. Nem adatbázistáblák láthatók ezen a szinten, hanem a valós fogalmak!
EGYED * ATTRIBÚTUM1 o ATTRIBÚTUM2	Attribútum: Az egyedek tulajdonságai vagy jellemzői, amelyekkel • azonosítjuk, • leírjuk, • minősítjük, • osztályozzuk, vagy • állapotát fejezzük ki az egyednek.

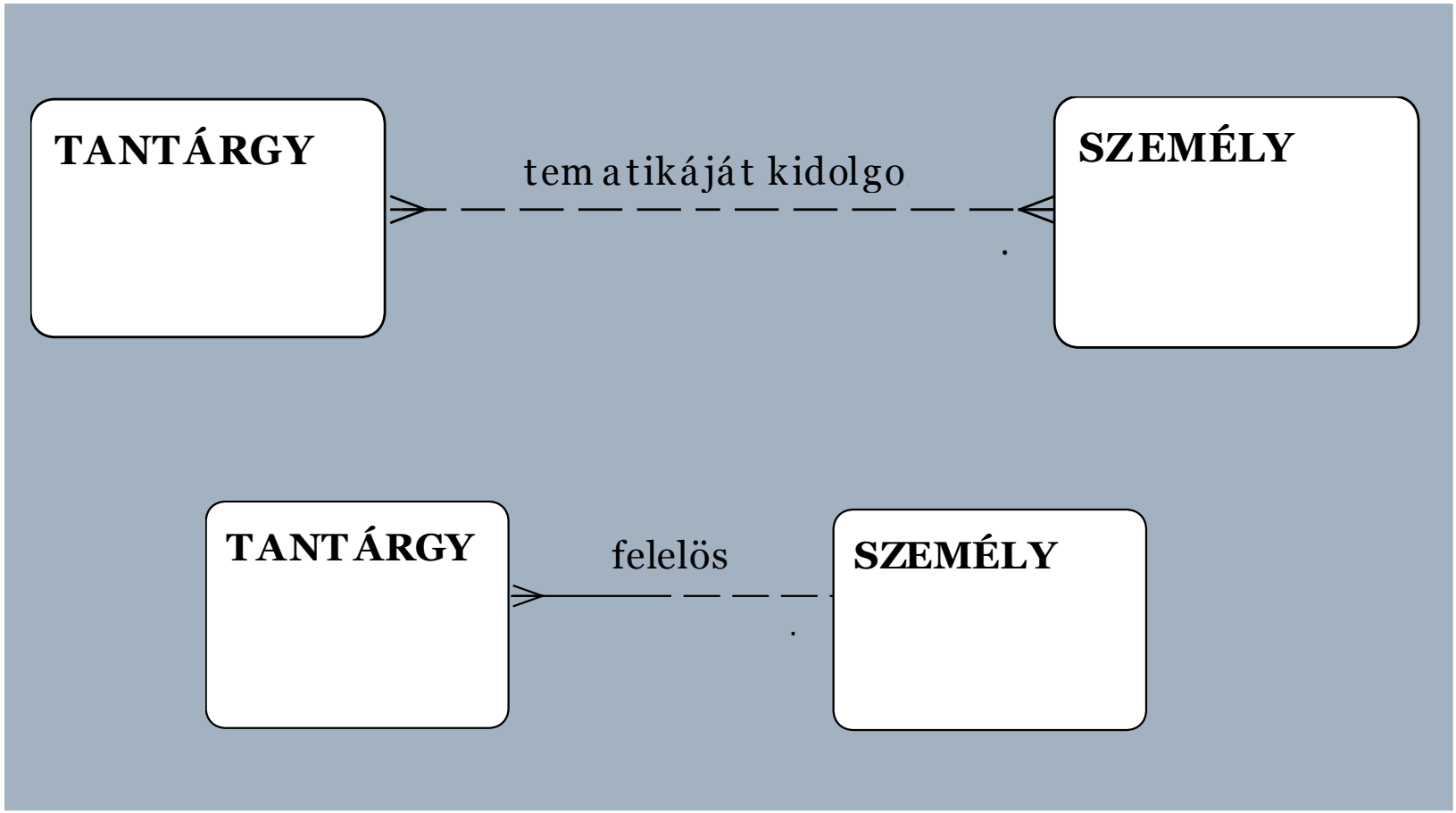
Az Egyed: példák

- Az **entitás** vagy egyed (és az egyedek halmaza) lehet:
  - **ország**: COUNTRIES
  - **dolgozó**: WORKERS
  - ...
- Egy másik rendszerben:
  - **szerződés**: SZERZODESEK
  - vagy akár **szeretet**, vagy pl. **avatar**,
- A **tulajdonság** (attribútum „attribute”) az entitás néhány kiválasztott jellemzője
- Sokszor igen nehéz előrelátni, melyik jellemzőket érdemes kiválasztani. Pl. egy bérszámfejtő rendszerben régen a „**gyerekek_száma**” nem kellett.
- Új entitást „költséges” utólag bevezetni, új attribútumot általában jóval könnyebb

Az egyed-kapcsolat modell „kapcsolat” része

- **Kapcsolat** („relationship”): entitások névvel ellátott viszonya
- Az egyedek ritkán léteznek elszigetelten. Tipikus az, hogy kapcsolatban állnak egymással.
- Pl. Emberek cégeknél dolgoznak. Itt a **dolgoznak** a kapcsolat az **ember** és a **cég** az entitások.
- Pl. a **házasság** a reláció egy **ember** és egy másik **ember** között ☺
- Pl. a **tanár** „**tanit**”-ja a **diákot**. Itt a „tanít” a reláció

Fogalmi adatmodellek alapelemei : Kapcsolatok



Fogalmi adatmodellek alapelemei : Kapcsolatok

Két egyed között létező logikai kapcsolat.

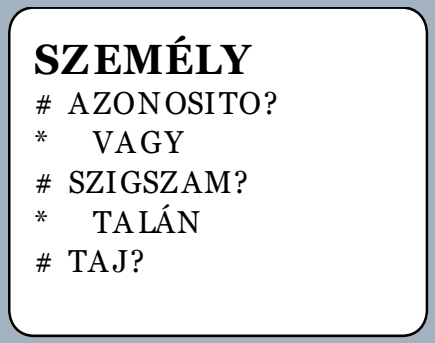
A E-R modellben a kapcsolatokat a következőkkel jellemezzük:

- a kapcsolat neve
- a kapcsolat foka: 1:N, M:N, 1:1
- a kapcsolat típusa lehet kötelező vagy opcionális,

Fogalmi adatmodellek : Egyedi azonosító

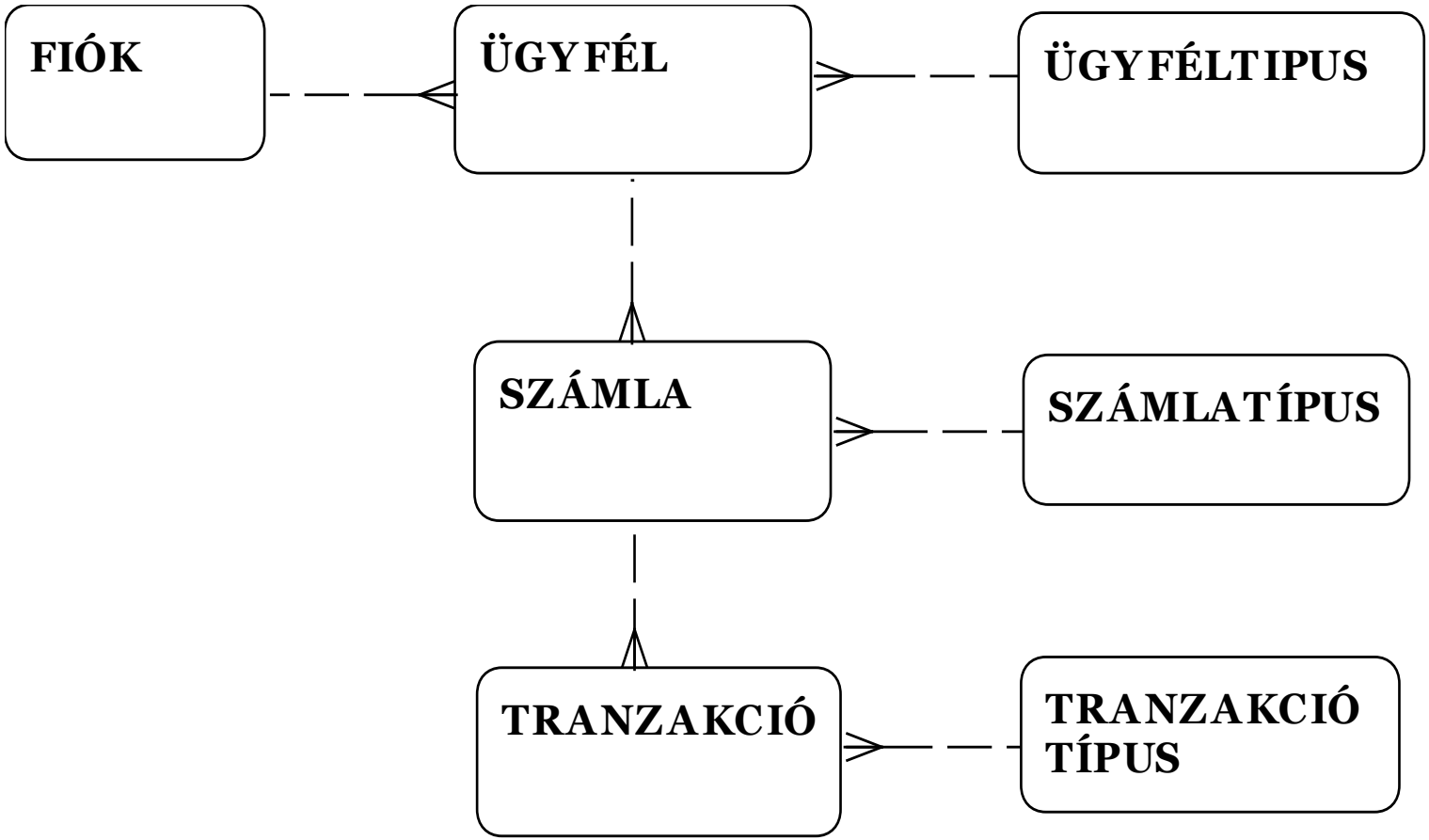
Ugyanazon egyedtípus előfordulásainak megkülönböztetése egyedi azonosító(k) segítségével.

Példák:



Fogalmi adatmodellek : Szupertípus

Példa: Egyedek és kapcsolatok



Hogyan tervezzük meg a relációs „sémáinkat”

- A többször tárolt adatok feleslegesek lehetnek. (De nem mindig azok – pl. adattárház)
- Ha egy relációban valamely attribútum értékét a relációban található más attribútom(ok) értékéből ki lehet következtetni, akkor a relációt redundánsnak nevezzük
- A redundáns adatok tárolásakor egy sor kellemetlen jelenség állhat elő. Ezeket anomáliáknak nevezzük.

Kikövetkeztető értékek: példa

DIVISIONS

DIVISION_NAME	DIVISION_TASK	Num of employees	COUNTRY
Sybozu	developer	250	Japan
Velebit	operator	100	Hungary
Masala	operator	100	Montenegro
Velebit	developer	70	Hungary
Sybozu	web design	150	Japan

Módosítási anomália

- Előző dián a DIVISIONS tábla második sorában (VELEBIT) a COUNTRY oszlop értékét megváltoztatjuk: HUNGARY helyett CROATIA lesz.
- Ezután a második és negyedik sor egymásnak „ellentmondanak”

Beszúrási anomália

- Nem tudunk egy új adatot nyilvántartásba venni, ha nem ismert egy másik adat.
- Pl. tegyük fel, hogy a „num_of_employees” attribútum nem lehet NULL érték az előző dián. Ekkor egy új részleget nem tudunk nyilvántartásba venni, ha még nem tudjuk a részlegben dolgozók számát.
- Ennél a példánál kellemetlenebb esetek is előfordulnak a nem megfelelő adatmodell választása esetén.

Törlési anomália

- Ha csak egy attribútum értékét szeretnénk törölni, akkor előfordulhat, hogy ez nem lehetséges (pl. mivel része valamely kulcsnak)
- Ilyenkor az egész sort törölhetjük, de akkor meg más információt (adatot) is elveszíthetünk.

Mik és miért kellenek a normálformák

- Ahhoz, hogy az előzőleg ismertetett anomáliákat elkerülhessük, a relációink sémái meghatározott feltételeket kell hogy teljesítsenek
- Ezeket normálformáknak nevezzük
- Létezik elég sok normálforma: ezek közül a legfontosabbak az 1NF, 2NF, 3NF, BCNF (Boyce-Codd normálforma)
- Az ennél magasabb rendűek egy piciny túlzással csupán az elméleti szakértőket érdeklik.

Első normálforma (1NF)

- Egy relációs séma 1NF alakú (1NF normalizált) ha benne csak atomi attribútum-értékek szerepelnek
- Köznyelven mondva: a tábla sorai és oszlopai „metszetében” lévő „mezők” csak 1 értéket tárolnak.
- Érdekesség, hogy az Oracle adatbáziskezelő rendszer az objektumorientáltság kedvéért megengedi ennek az alapszabálynak a megszegését is akár

Első normálforma (1NF)

COUNTRIES in 0NF

countrie_id	country_name	capital_city	continent
36	Hungary	Budapest	Europe
90	Turkey	Ankara	Europe; Asia

COUNTRIES in 1NF

countrie_id	country_name	capital_city	continent
36	Hungary	Budapest	Europe
90	Turkey	Ankara	Europe
90	Turkey	Ankara	Asia

Második normálforma (2NF)

- Elsődleges attribútum az, amely valamelyik kulcs része.
- Másodlagos attribútum az, amely nem elsődleges, tehát olyan attribútum, amely nem része egyik kulcsnak sem.
- Egy 1NF relációs séma 2NF alakú, ha egyik másodlagos attribútuma sem függ a kulcs egy részhalmazától, hanem kizárólag a teljes kulcstól függ

Második normálforma (2NF)

DIVISIONS (not in 2NF)

DIVISION_NAME	DIVISION_TASK	Num of employees	COUNTRY
Sybozu	developer	250	Japan
Velebit	operator	100	Hungary
Masala	operator	100	Montenegro
Velebit	developer	70	Hungary
Sybozu	web design	150	Japan

DIV_EMPLOYEE_NUMBER

DIVISION_NAME	DIVISION_TASK	Num of empl
Sybozu	developer	250
Velebit	operator	100
Masala	operator	100
Velebit	developer	70
Sybozu	web design	150

DIVISIONS

DIVISION_NAME	COUNTRY
Sybozu	Japan
Velebit	Hungary
Masala	Montenegro

Mi a baj az előző dia felső részén?

- Ha a fenti séma (DIVISIONS not in 2NF) kulcsa a (DIVISION_NAME, DIVISION_TASK) páros, akkor az a baj, hogy a COUNTRY csupán a DIVISION_NAME-től függ

Harmadik normálforma (3NF)

- Egy 1NF relációs séma 3NF alakú, ha egyetlen másodlagos attribútuma sem függ tranzitívan egyetlen kulcstól sem
- Ami nem 3NF alakú, abban rejtett függőségek vannak, és ezeket meg kell bontani több sémára

Harmadik normálforma (3NF)

DIVISIONS not in 3NF

DIVISION_ID	DIVISION_NAME	CITY	COUNTRY	CONTINENT
11	developer	Tokyo	Japan	Asia
12	operator	Debrecen	Hungary	Europe
13	developer	Szeged	Hungary	Europe

DIVISIONS

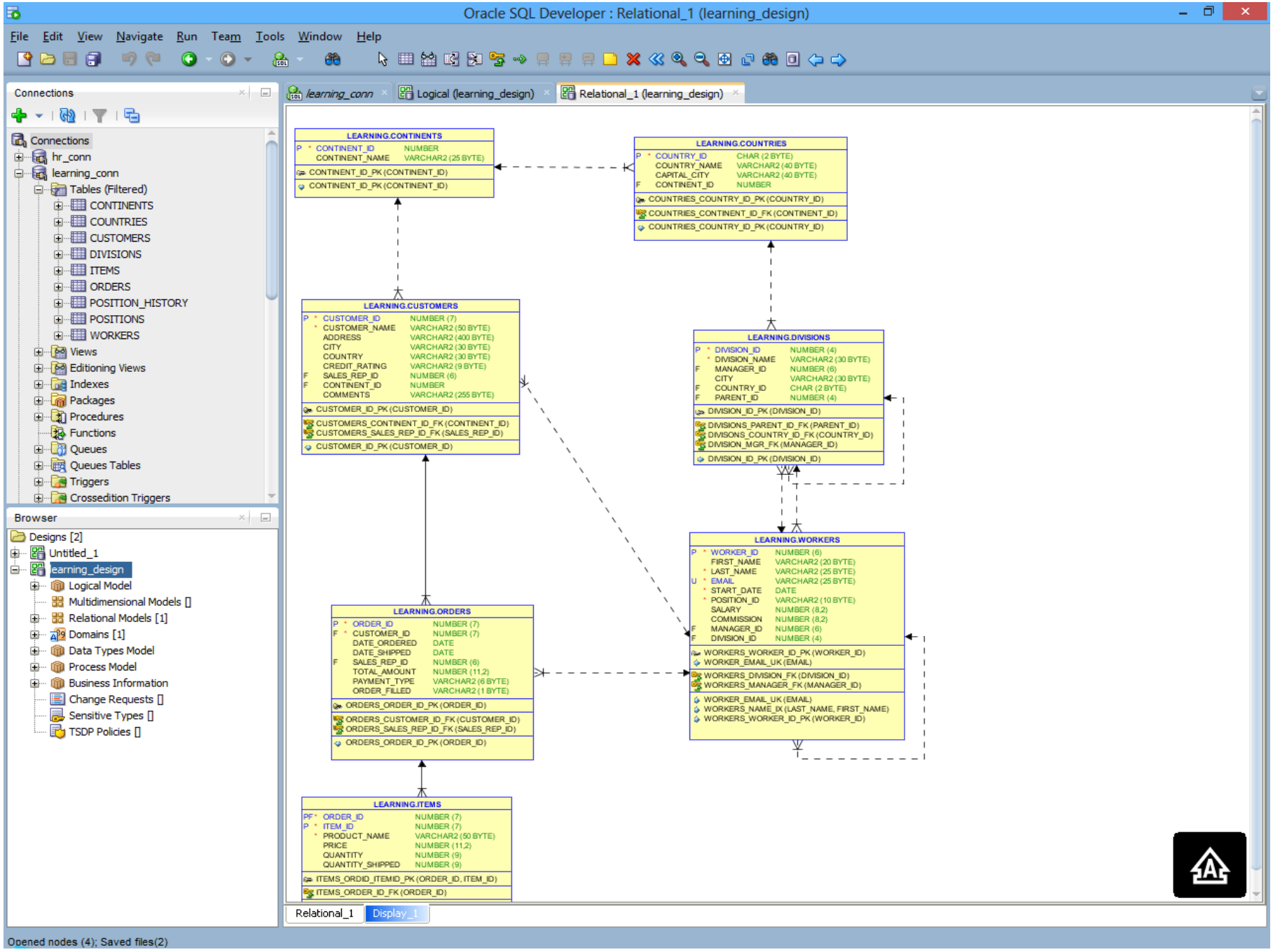
DIVISION_ID	DIVISION_NAME	CITY	COUNTRY
11	developer	Tokyo	Japan
12	operator	Debrecen	Hungary
13	developer	Szeged	Hungary

COUNTRIES

countrie_id	country_name	continent
36	Hungary	Europe
81	Japan	Asia

A folytatásban (a jövő héten) a következő diákon lévő entitásokat és relációkat fogjuk megvizsgálni:

A LEARNING séma



A LEARNING séma egy részlete