3. tétel

A relációs adatmodell alapfogalmai.

Az adatbázis-kezelő rendszerek mindig valamilyen adatmodellre épülnek. Az adatmodellben találhatóak a különböző típus-leírások, a köztük fennálló kapcsolatok, összefüggések, adatvédelmi eljárások. Az adatmodellben egyedtípusokkal foglalkozunk, míg a feldolgozásban a rekordtípusok konkrét előfordulásával.

Már említettük, hogy az adatmodellnek 3 független szintje van, ezt most vázolni fogjuk:

- külső szint: vagy alsémák, leírja, hogy egyes felhasználók mit látnak az adatbázisból
- fogalmi szint: vagy koncepcionális szint, mely az adatbázisban lévő összes adat szerkezetét, és az adatokhoz való hozzáférés logikáját, hozzáférési jogosultságokat ír le
- belső szint: az adatok fizikai tárolását, az adat-elérés mechanizmusát adja meg

Az adatmodell három fő típusa:

- hierarchikus (az adatok között alá-, fölérendeltségi viszony van)
- hálós (egy fő csoport köré gyűjti az oda tartozókat)
- relációs modell (azonos tulajdonságú kapcsolatok vannak összefűzve)

A relációs adatmodell:

A relációs adatmodell azonos tulajdonságú kapcsolatok összefűzésén alapul. Az adathalmazt reláció tartja egyben, melyet táblázattal tudunk ábrázolni. A relációk (táblák) közötti kapcsolatokat relációs műveletekkel tudjuk létrehozni.

Az adatmodell egyik előnyei:

- az adatokat táblázatokkal tudjuk ábrázolni, így szemléletes, követhető
- matematikailag is részletesen kidolgozták
- ebben a modellben valósítható meg legnagyobb mértékben a logikai és fizikai adatfüggetlenség

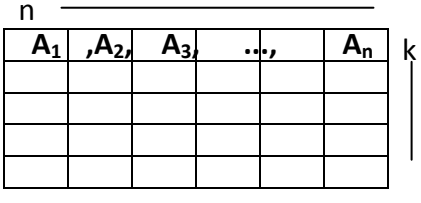
A **reláció** (táblázat) tartalmazza a mini-world egyedeit. (Mini-world: a világ adatainak leképezése (redukálása) a számunkra hasznos adathalmazba.)

A reláció nem más, mint sorok, azaz **n-esek (tuple)** halmaza.

Az **attribútumok** az oszlopnevek.

A **relációnév** a táblanév.

Táblaszerkezet = **relációséma** : **$R(A_1,A_2,A_3,..,A_n)$**
ahol **R** a relációnév, **$A_1,A_2,A_3,..,A_n$** pedig az attribútumok.



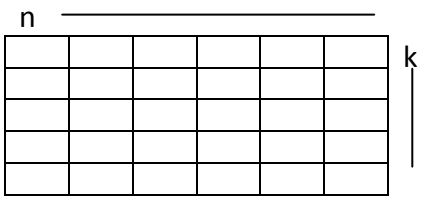
A relációra a relációnévvel, azaz a táblanévvel tudunk hivatkozni.

Táblaszerkezet: relációséma, pl.: **R(A_1,A_2, …. A_n) // HALLGATO(HAZON,LAKHELY,SZDATUM)**
ahol **R** a relációnév, **A_1,A_2, …. A_n** az attribútum.

Minden attribútumhoz tartozik egy domain, ami megmondja, hogy honnan veheti fel az értékeket. Például a fenti alapján **SZDATUM → DOM = {dátum}.** Azaz a **SZDATUM** attribútumai csak olyan elemek lehetnek, melyek megfelelnek annak a logika tulajdonságnak, hogy dátum.

n a reláció foka
k a reláció mélysége

t_i= < a_{i1}, a_{i2}, … ,a_{in} >
Minden $a_{ij} \in DOM_j \cup \{NULL\}$



Minden tartományhoz (domainhez) hozzárendelünk egy típust, és egy formátumot.

A relációs adatmodell használati lehetőségei:

- önálló nyelvű — saját nyelve van, és annak segítségével lehet megvalósítani az adatbázis-kezelést is
- befogadó nyelvű — nem komplett programozási nyelv, csupán az adatok kezeléséhez kapcsolódó utasításokat tartalmazza

Integritási megszorítások a relációs modellben.

Kulcsmegszorítások

Mi a kulcs? A **kulcs** nem más, mint egy rekordok halmazához rendelt azonosító, mely egyértelműen meghatározza a halmazt, vagyis azonosít egy sort a relációban.

Szuperkulcs: rekordok halmazát egyértelműen azonosítja. (nem lehet 2 különböző sornak ugyanaz a szuperkulcsa). Ha egy szuperkulcshoz hozzáveszünk még egy elemet, úgy is szuperkulcsot kapunk.

Kulcs: minimális szuperkulcs. Ha egyetlen elemet elhagyunk belőle, nem tud egyértelműen azonosítani.

SK= szuperkulcs, K=kulcs.
SK $\subset$ { **A_1,A_2, …. A_n** }=R , vagyis szuperkulcs lehet akár egy egész sor is. Ha t[SK] = u[SK], akkor t[R] = u[R]

Például: **R(A_1,A_2, …. A_n) =** Autók(állam, rendszám,alvázszám, típus)
ekkor szuperkulcs lehet az R,
SK= {állam, rendszám}.
K= {alvázszám}.

Egyed integritási megszorítás

Előírja, hogy az elsődleges kulcs attribútumai nem lehetnek NULL értékek, mindenképpen adottnak kell lenniük.. (Hogyan is azonosíthatna valamit úgy, hogy nem ismerjük az értékét!)

Hivatkozási integritási megszorítás

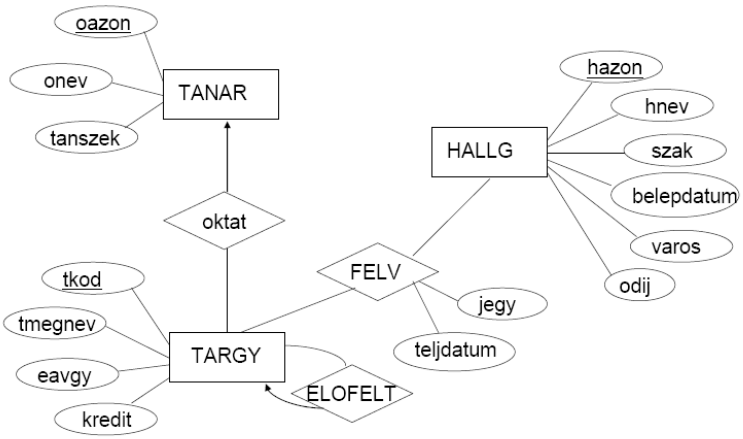
Először, jöjjön a magyarázat. Tegyük fel hogy van két táblánk. Az egyik táblában vannak a **tantárgyak**, a másikban a **hallgatók által felvett tárgyak**. Ha egy tantárgy – hallgató páros benne van a második táblában, akkor a tantárgynak benne kell lennie az tantárgyakat felsorakoztató táblában is.

Legyen R_1, R_2 relációk (táblák). $R_1(t,n)$, $R_2(t,h)$. Ha az R_2–ben van olyan sor vagy n-es hogy (x,y), ahol az x egy t-beli attribútum, akkor az R_1–ben x-nek benne kell lenni a t helyen. Ekkor x-et az R_2 reláció külső, vagy idegen kulcsának nevezzük. Ez az első táblában primary key-ként, vagyis elsődleges kulcsként szerepel.

E/K modell átírása relációs adatbázissémára

1. Példa: Egyetemi tantárgyfelvétel (E/K)

Itt egy hallgatói tantárgy-felvétel egyed-kapcsolat diagramja látható. Nézzük meg, hogyan lehet átírni relációs adatbázissémára.



TANAR(<u>oazon</u>, onev, tanszek)
TARGY(<u>tkod</u>,tmegnev, kredit, eavgy)
HALLG(<u>hazon</u>, hnev, szak …..)

kapcsolatok leírása (táblákkal):

OKTAT(<u>oazon</u>,<u>tkod</u>);

Ekkor lehetőségünk van arra, hogy egy tárgynak több oktatója is legyen. Ha ez nem lehetséges, akkor ez is helyes megoldás:
TARGY(<u>tkod</u>,<u>oazon</u>,tmegnev, kredit, eavgy)

FELV(). Mi a szerepe a kapcsolatnak? Hogy megtudjuk mely hallgatók milyen tárgyat vettek fel, milyen jegyet szereztek, mikor teljesítették a tárgyat.
FELV(<u>hazon</u>, <u>tkod</u>, jegy, tejdatum)

Valósítsuk meg az előfeltétel relációt is. (fontos tudni, melyik tárgynak mi az előfeltétele)
Amelyik tárgy előfeltétele egy másiknak, annak benne kell lennie a tantárgyak táblában.
ELOFELT(<u>tkod1</u>,<u>tkod2</u>) (tkod1 előfeltétele a tkod2 tárgy)