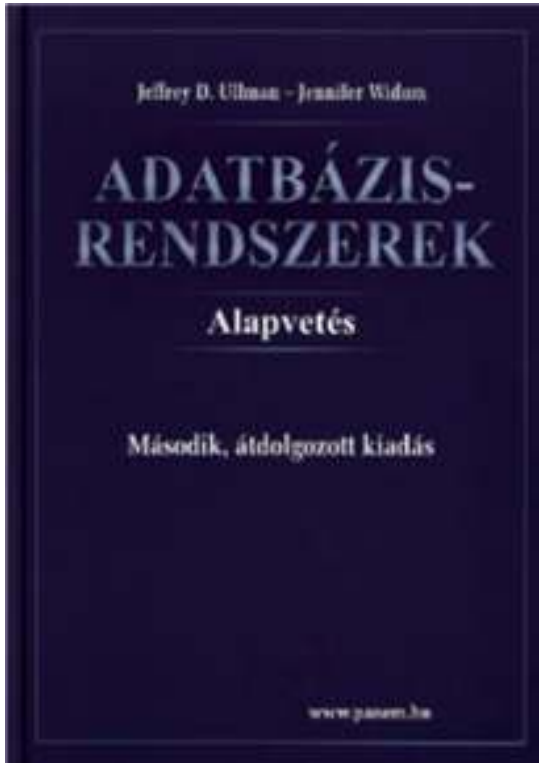
Bevezetés: Relációs adatmodell

Tankönyv: Ullman-Widom:
Adatbázisrendszerek Alapvetés
Második, átdolgozott kiadás,
Panem, 2009



2.1. Adatmodellek áttekintése

2.2. A relációs modell alapjai

-- Megjegyzés: A gyakorlat felépítése
-- miatt az SQL lekérdezésekkel és az
-- ahhoz szükséges alapokkal kezdünk.
-- Ezután lesz az 1.fejezet Az adatbázisrendszerek világáról.

Bevezető példa (reláció = tábla)

- Naponta találkozunk adatbázisokkal
  - 1960-as évektől a korai DBMS: banki rendszerek, repülőgép-helyfoglalás, vállalati nyilvántartások
  - Napi szinten: Google, Yahoo, Amazon.com, egyetemi tanulmányi rendszerek (ETR, Neptun)
- 1.példa: A jelentkezési adatok egyeztetésére táblázat (lásd papíron a jelenléti ív /Ez az első előadás példája)
- Reláció = tábla (a jelenléti ív)
- Séma = a reláció szerkezetének leírása (tába fejléce)
- Előfordulás vagy példány = egy adott időpontban a tábla aktuális tartalma (ebben a példában szereplő tábla tartalma a tantárgyfelvétel időszakában folyton változik)

Bevezető példa (folyt)

- Milyen műveleteket adhatunk meg erre a táblázatra?
- Melyik keresés könnyebb: ha jelentkezési dátum szerint rendezve vagy ha névsorban vannak az adatsorok?
- Melyik jobb, ha természetes azonosító (név) szerint vagy mesterséges azonosító (neptunkód) szerint keresünk?
- Lekérdezéseket milyen táblaműveletekkel tudjuk kifejezni (az első két hét erről szól: relációs algebra)

Mi az adatbázis?

- Mit várunk az adatbázis-kezelő rendszerektől? Implementáció kérdései:
  - nagy méretű (massive, több terabyte mennyiségű)
  - háttértárolón tárolt (persistent) adatokat
  - több-felhasználó számára (konkurencia-vezérlés)
  - biztonságos (konzisztens állapot biztosítsa, védve legyen a hardware, software és felhasználói hibáktól)
  - kényelmes (fizikai adatfüggetlenség, magas szintű deklaratív nyelv, mint például az SQL szabvány)
  - és hatékony legyen a megvalósítás.
- Erre később a lekérdezések (relációs algebra és SQL SELECT utasítás) után fogunk visszatérni.

Ki ismeri az SQL-t?

- Ki ismeri az SQL-t? Itt mi a különbség?

(1)
```
SELECT B FROM R
  WHERE A<10 OR A>=10;
```

(2)
```
SELECT B FROM R;
```

R

A	B
5	20
10	30
20	40
…	…

- Itt mi a helyzet ezzel?

(3)
```
SELECT A FROM R, S
  WHERE R.B = S.B;
```

(4)
```
SELECT A FROM R
  WHERE B IN (SELECT B FROM S);
```

Mi az adatmodell?

- (Később lesz bővebben az E/K-modellnél) Az adatmodell a valóság fogalmainak, kapcsolatainak, tevékenységeinek magasabb szintű ábrázolása
- Kettős feladat: az adatmodell megadja, hogy a számítógép számára és a felhasználó számára hogy néznek ki adatok.
- Az adatmodell: adatok leírására szolgáló jelölés.
  Ez a leírás általában az alábbi három részből áll:

1. Az adat struktúrája
2. Az adaton végezhető műveletek (lekérdezések, módosítások, feldolgozások legyenek megfogalmazhatók)
3. Az adatokra tett megszorítások (milyen adatokat engedélyezünk, milyen megszorításokat teszünk?)

A fontosabb adatmodellek

- Hálós, hierarchikus adatmodell (gráf-orientált, fizikai szintű, ill. apa-fiú kapcsolatok gráfja, hatékony keresés)
- Relációs adatmodell (táblák rendszere, könnyen megfogalmazható műveletek), magában foglalja az objektumrelációs kiterjesztést is (strukturált típusok, metódusok), SQL/Object, SQL/CLI, SQL/PSM (PL/SQL)
- Objektum-orientált adatmodell (az adatbázis-kezelés funkcionalitásainak biztosítása érdekében gyakran relációs adatmodellre épül), ODMG: ODL és OQL
- Logikai adatmodell (szakértői rendszerek, tények és következtetési szabályok rendszere)
- Dokumentum típusú adatok, félig-strukturált adatmodell (XML-dokumentum), további modellek: gráf adatbázisok

Relációs adatmodell története

- E.F. Codd 1970-ban publikált egy cikket
  A Relational Model of Data for Large Shared Data Banks
  Link: http://www.seas.upenn.edu/~zives/03f/cis550/codd.pdf

  amelyben azt javasolta, hogy az adatokat táblázatokban, **relációkban** tárolják. Az elméletére alapozva jött létre a relációs adatmodell, és erre épülve jöttek létre a relációs adatmodellen alapuló relációs adatbázis-kezelők.
- Relációs (objektum-relációs) **adatbázis-kezelők** például:

  ORACLE , INFORMIX , SYSBASE , INGRES, DB2, stb
- Adatbázisok-1 gyakorlaton ORACLE adatbázis-kezelő rendszert használunk.

Relációs adatmodell előnyei

Miért ez a legelterjedtebb és legkifinomultabb?

- Az adatmodell egy egyszerű és könnyen megérthető strukturális részt tartalmaz. A természetes táblázatos formát nem kell magyarázni, és jobban alkalmazható.
- A relációs modellben a fogalmi-logikai-fizikai szint teljesen szétválik, nagyfokú logikai és fizikai adatfüggetlenség. A felhasználó magas szinten, hozzá közel álló fogalmakkal dolgozik (implementáció rejtve).
- Elméleti megalapozottság, több absztrakt kezelő nyelv létezik, például relációs algebra (ezen alapul az SQL automatikus és hatékony lekérdezés optimalizálása).
- Műveleti része egyszerű kezelői felület, szabvány SQL.

Relációs lekérdező nyelvek

Három nyelv szerepel (Adatbázisok-1 gyakorlaton is lesz)

- Relációs algebra: algebrai megközelítés (az 1.előadáson elkezdjük) itt megadjuk a kiértékelési eljárásokat, többféle lehetőség összevetése, hatékonysági vizsgálatok.
- SQL szabvány relációs lekérdező nyelv: folyt.2.előadáson elkezdjük az SQL-t: a története, szabványok, felépítése. Lekérdezések SELECT-utasítás, SQL DDL, DML, DCL, és SQL/PSM illetve PL/SQL (Oracle)
- Datalog: logika alapú megközelítés (korábban elsőrendű logikának megfelelő kalkulus típusú lekérdezések voltak, erről az MSc-n lesz majd bővebben). Itt az Adatbázisok-1 keretében a Datalog az összetett lekérdéseknél segítség, például a rekurzív lekérdezéseknél.

Relációs adatmodell: relációs séma

- Adatok gyűjteményét kezeli (gyűjtemény azonosítása: név)
  A gyűjtemény - R reláció (tábla, táblázat) megadása
- A gyűjtemény milyen típusú adatokat gyűjt?
  adattípus: sor-típus. A sor-típus (egy n-es) megadása:
  <Attribútumnév$_1$: értéktípus$_1$, … , Attrnév$_n$: értéktípus$_n$>
  röviden <A_1, … , A_n>
- Relációséma: Relációnév (sortípus) (itt: kerek zárójelben!)
  vagyis R(Anév$_1$: értéktípus$_1$, …, Anév$_n$: értéktípus$_n$)
  röviden R(A_1, … , A_n) ill. U = {A_1, … , A_n} jelöléssel R(U)
- PÉLDA: jelenléti ív fejléce – relációséma: megadjuk a tábla szerkezetét, oszlopnevek és típusuk. Milyen megszorításokat (pl. kulcs) tudunk megadni a sémán?

Relációs adatmodell: előfordulás

- Mit jelent egy konkrét sor? sor $<A_1$: érték$_1$, …, A_n: érték$_n>$
- Reláció előfordulás (példány, instance)

  A sor-típusnak megfelelő véges sok sor (sorok halmaza).

  $\{t_1, \ldots, t_m\}$ ahol t_i (tuple, sor, rekord) i= 1, …, m (véges sok)

  $t_i = <v_{i1}, \ldots, v_{in}>$ (vagyis egy sor n db értékből áll)

  m - számosság (sorok száma)

  n - dimenzió (attribútumok száma)
- Értéktartományok: A reláció minden attribútumához tartozik egy értéktartomány (adott értéktípusú értékek halmaza) (1normálforma, 1NF feltétel: atomi típusú)
- PÉLDA: Jelenléti ív táblázat (a táblázatban az adatok)

Szemléltetés: táblázatos forma

- Szemléltetése: a táblázatos forma (reláció, tábla)

R	A_1	…	A_j	…	A_n
t_1	v_{11}		…		v_{1n}
…			…		
t_i	…		v_{ij}		…
…			…		
t_m	v_{m1}		…		v_{mn}

A_j - attribútumnév

t_i - itt csak szimbolikusan vezetem be, hogy tudjak a sorokra hivatkozni (Oracle-ben: rowid)

Szemléltetés: függvénnyel

- A táblázatos szemléltetésből áttérhetünk a sorok egy másik szemléltetésére:
- $t \in R$ sor felfogható függvényként is
- $t_i : U \rightarrow$ értékek, ahol $U = \{A_1, \ldots, A_n\}$

  Ezzel a jelöléssel $t_i (A_j) = v_{ij}$ ekvivalens jelöléssel $t_i [A_j]$ vagy $t_i .A_j$ (objektum-orientált/metódus típusú jelöléssel)
- Előfordulás: sor-függvények véges halmaza
- Korlátozom a függvényt: $X \subseteq U = \{A_1, \ldots, A_n\}$

  Ha $X = \{A_{j_1}, \ldots, A_{j_k}\}$ attr.halmaz, akkor t[X] típusa:

  $t[X] = \langle A_{j_1}: t(A_{j_1}), \ldots, A_{j_k}: t(A_{j_k})\rangle$
- Függvény szemléltetéssel könnyen tudom képezni a sorok egy részét és így állítom elő a megfelelő sort.

Relációs adatbázis felépítése

- Az adatbázis tulajdonképpen relációk halmaza.
- a megfelelő relációsémák halmaza adja az adatbázissémát (jelölése dupla szárú $\mathbb{R}$)

  $$\mathbb{R} = \{R_1, \ldots, R_k\}$$
- a hozzá tartozó előfordulások az adatbázis-előfordulás
- Előfordulás tartalma: egyes relációk előfordulásai

- Ez a koncepcionális szint, vagyis a fogalmi modell.
- Fizikai modell: a táblát valamilyen állományszerkezetben jeleníti meg (például szeriális állományban). A relációs adatbázis-kezelők indexelnek, indexelési mód: pl. B+ fa.

  (Ez az Adatbázis-2 kurzuson lesz a fizikai megvalósítás)

Logikai szinten: táblázatos szemléltetés

- A relációk táblákban jelennek meg. A tábláknak egyedi neve van.
  A relációk oszlopait az attribútumok címzik. A tábla sorait tetszőlegesen megcserélhetjük, sorok sorrendje lényegtelen (a halmazszemlélet miatt)

Mivel attribútumok halmazáról van szó, a Példa 1 és Példa 2 relációk nevüktől eltekintve azonosak.

Példa 1

A	B	C
a	b	c
d	a	a
c	b	d

Példa 2

B	C	A
b	c	a
a	a	d
b	d	c

Mivel sorok halmazáról van szó, a Példa 1 és Példa 3 relációk nevüktől eltekintve azonosak.

Példa 3

A	B	C
c	b	d
d	a	a
a	b	c

Példa 4

A	B	C
c	b	d
c	b	d
a	b	c

Ebben a modellben Példa 4 nem reláció, de a valóságban megengedünk mutihalmazokat lásd később SQL

Példa: Filmek séma

Tankönyv példája, hibás fordítás:
title=(film)cím és address=(lak)cím

Mit jelentenek az aláhúzások?

Filmek(
cím:string,
év:integer,
hossz:integer,
műfaj:string,
stúdióNév:string,
producerAzon:integer)

FilmSzínész(
név:string,
cím:string,
nem:char,
születésiDátum:date)

SzerepelBenne(
filmCím:string,
filmÉv:integer,
színészNév:string)

GyártásIrányító(
név:string,
cím:string,
azonosító:integer,
nettóBevétel:integer)

Stúdió(
név:string,
cím:string,
elnökAzon:integer)

Tervezéssel később foglalkozunk,
a fenti példa hibás, az elnevezések

Példa megszorításokra: Kulcs

- Előző példában: attribútumok aláhúzása mit jelent?
- Filmek: elvárjuk, hogy ne legyen a megengedett előfordulásokban két különböző sor, amelyek megegyeznek cím, év attribútumokon.
- Egyszerű kulcs egy attribútumból áll, de egy kulcs nem feltétlenül áll egy attribútumból, ez az összetett kulcs. Például a **Filmek** táblában a cím és év együtt alkotják a kulcsot, nem elég a cím, ugyanis van például (King Kong, 1933), (King Kong, 1976) és (King Kong, 2005).
- A kulcsot aláhúzás jelöli: Filmek (cím, év, hossz, …)

Kulcsra vonatkozó megszorítások

- Az attribútumok egy halmaza egy kulcsot alkot egy relációra nézve, ha a reláció bármely előfordulásában nincs két olyan sor, amelyek a kulcs összes attribútumának értékein megegyeznének.
- Formális megadása:

  $R(U)$, $X \subseteq U$, $U = \{A_1, \ldots, A_n\}$, $X = \{A_{j1}, \ldots, A_{jk}\}$

  $t \in R$, $t[X] < A_{j1} : t(A_{j1}), \ldots, A_{jk} : t(A_{jk}) >$

  ezzel a jelöléssel mit jelent, hogy X kulcs elvárás?

  ha $t_1 \in R$, $t_2 \in R$ és $t_1[X] = t_2[X]$ akkor $t_1 = t_2$

Idegen kulcs és hivatkozási épség

- SzerepelBenne táblában ha van egy sor filmCím=c, filmÉv=é értékkel, akkor Filmek táblában is legyen c,é kulccsal rendelkező sor
- Idegen kulcs, FOREIGN KEY

  $R(A_1, \ldots, A_m)$ séma, $X = \{A_{i_1}, \ldots, A_{i_k}\}$ kulcs,

  $S(B_1, \ldots, B_n)$ séma, $Y = \{B_{j_1}, \ldots, B_{j_k}\}$ idegen kulcs,

  ami az X-re hivatkozik a megadott attribútum sorrendben:

  B_{j_1} az A_{i_1} -re, …, B_{j_k} az A_{i_k} -re
- Hivatkozási épség, REFERENCES

  megszorítás a két tábla együttes előfordulására:

  Ha $s \in S$ sora, akkor létezik $t \in R$ sor,

  hogy $s[B_{j_1}, \ldots, B_{j_k}] = t[A_{i_1}, \ldots, A_{i_k}]$

  Ekkor az S-en Y idegen kulcs, ami hivatkozik az R kulcsára