Ebben a fejezetben beszélünk az adatmodellben való gondolkodás kialakulásáról, tisztázzuk az adatbázis fogalmát az alapelemek definiálásával: melyek az egyed, a tulajdonság és a kapcsolat. Ezek segítségével megkülönböztetjük az adatmodellek fajtáit.

Az első adatbázis-kezelő rendszert az IBM vezette be IMS (Information Management System) néven 1969-ben. Ebben a rendszerben az adattárolás faszerkezetekre vezethető vissza. Az adatkezeléshez szabványosított eljáráskészletet nyújtott az IBM.

A CODASYL (Committee on Data Systems and Languages) 1971-ben újfajta adatbáziselvet tett közzé, a hálós szerkezeten alapuló adatszerkezetet. Nyilvánvaló hátránya, hogy a logikai kapcsolat megváltoztatása a tárolás átszervezését okozta. Viszont értékes újítást jelentett az adatdefiníciók elválasztása az adatkezeléstől nyelvi eszközeiben. Híres megvalósítás: IDMS (Integrated Database Management System).

Az említett gondok leküzdését szem előtt tartva vázolta fel 1971-72-ben E. F. Codd a relációs adatmodellt. A logikailag hasonló egyedeket táblázatokba foglalta, melyek kezelését a halmazelmélet műveleteire alapozta. Az adategyedek azonosítására és kikeresésére annak kiválasztott adatmezőit használta, és azokat kulcsnak nevezte el.

P. Chen 1976-ban továbbfejlesztette Codd modelljét, és bevezette az adategyedek közti tartós kapcsolatokat jellemző kulcsok definiálását (ERM: Entity Relationship Model).

{Stolnicki Gyula (SQL kézikönyv, ComputerBooks 1995.) nyomán}

1. Formalista megközelítés

Ebben a modellben az adatok és kapcsolataik tárolása gráfokkal ábrázolható. Ebből fejlődött ki a hierarchikus és hálós adatmodell.

2. Szemantikai megközelítés

A modellt verbálisan, szavakkal írták le, ami kudarcot vallott. Akkor ugyanis még nem volt bizonyított, hogy nyelvi fordítóprogramot sem lehet készíteni.

3. Matematikai megközelítés

Matematikai struktúrák felhasználásával életképes modellt állított elő Codd, de csak jóval később nyert elismerést. A nevével is jelzett modell matematikai indoklásból a relációs adatmodell elnevezést kapta.

Adatbázis-kezelő rendszer (továbbiakban: ABKR):

Az a szoftver (DBMS - Database Management System), amellyel az adatbázist kezelni tudjuk.

Két fontos műveletet kell tudnia:

létrehozás (karbantartás)

visszakeresés (lekérdezés)

Ennek megfelelően adatdefiníciós nyelvre, illetve adatkezelési nyelvre bomlik. (Napjaink ABKR-eiben az adatbiztonsági utasítások további csoportot alkotnak.)

Az adatbázis-kezelő rendszerek két fajtája:

beépülő típus (IDMS, SQL)

ahol egy behívó nyelvvel együtt használható az adatfeldolgozó nyelv

valóságos programozási nyelv (dBase, Clipper, Paradox, Oracle, Informix...)

ahol önálló adatfeldolgozó nyelvként használható

Az adatbázis-kezelő rendszerek segédfeladatai:

Adatvédelem, adatbiztonság

Önállóan vagy az operációs rendszerrel együtt egyre nagyobb fokú biztonságra törekszenek az ABKR-ek gyártói.

Integritási feltételek

Az adatok közti kapcsolat és az egyes adatokra vonatkozó szabályok megőrzése nélkül ellentmondó információkhoz jutnánk.

Ebben a pillanatban kicsit korai a hivatkozási integritás taglalása, de modelltől függetlenül máris egyetértünk olyan, rendszerelemzés során található összefüggésekkel, mint:

nem szabad rendelt tételt addig felvinni, amíg a keretrendelés nem létezik,

nem szabad keretrendelést törölni addig, amíg vannak tételei,

nem szabad olyan cikkre hivatkozni, ami nincs a cikkek törzstárában,

nem szabad cikket addig kitörölni a törzstárból, amíg hivatkozik rá egy rendelési tétel, stb.

3. Szinkronizáció

Többfelhasználós esetben meg kell oldani az egyidejű hozzáférés anomáliáit.

Az adatmodellek alapelemei

  A képen szöveg látható

Automatikusan generált leírásAz egyed: minden olyan dolog (objektum), ami minden más dologtól (objektumtól) megkülönböztethető.

Előzetes tanulmányokra támaszkodva az egyedet nevezzük fájlnak.

Az egyed egy konkrét értéke: előfordulás.

Például: DOLGOZÓ, AUTÓ, BETEG, ÜGYFÉL, CIKK, BIZONYLAT...

Tehát én, mint Kupcsikné Fitus Ilona egy előfordulás leszek a Számalk Rt. Dolgozó egyedében vagy egy balesetet követően az egyik kórház Beteg-egyedében. Már most érezzük, hogy a két egyedben más-más jellemzőimmel fogok szerepelni.

A tulajdonság: az egyed belső szerkezete.

Az egyedeket tulajdonságokkal (attribútumokkal) írjuk le.

A tulajdonság értékeivel egy adott egyed konkrét értékét határozzuk meg.

A dolgozó nevű egyed tulajdonságai például: név, lakcím, dolgozószám, fizetés, végzettség, osztály,...

A beteg nevű egyed tulajdonságai: név, életkor, TAJ-szám, diagnózis,...

Amennyiben egy tulajdonság vagy tulajdonságok egy csoportja egyértelműen meghatározza, hogy az egyed melyik értékéről van szó, akkor ezeket együtt kulcsnak nevezzük.

Pl. hallgatókód a HALLGATOban

A hallgató jelentkezéskor előírás szerint megadja azt az 5 adatát, amely őt egyértelműen azonosítja. Ezek: neve, lakcíme, születési helye, ideje, anyja neve. Miután ez az 5 tulajdonság nagyon hosszú kulcsnak, ezért a nyilvántartásba vételkor azonnal generálnak neki egy rövid azonosítószámot, amit hallgatókódnak nevezhetnek el.

A kapcsolat: az egyed külső szerkezete.

A kapcsolat az egyedek közötti viszony.

A konkrét értékek a kapcsolat előfordulásai.

Példa:

VEVŐ-RENDELÉS, SZÁMLA-CIKK ...

A példákban már az egyedek neve is sugallja a kapcsolatok fajtáját: 1 vevőhöz több rendelés tartozhat, több számlán szerepelhet 1 bizonyos cikk. Egyelőre csak sejtjük, hogy azért van több egyedünk az adatbázisban, mert ugyanazt az adatot feleslegesen többször nem fogjuk tárolni. Egy vevőnek van rengeteg olyan tulajdonsága, ami egyértelmű: a törzsadatai (pl. név, születési hely, idő, különféle azonosítószám stb.) és aktuális adatai (mint pillanatnyi lakcíme, telefonja, utolsó vásárlásának dátuma stb.). Ezen kívül vannak a forgalmi adatai, melyek minden rendelése alkalmával tárolásra kerülnek (pl. rendelés dátuma, fizetési határidő, összérték stb.). Ha belegondolunk, hogy egy rendelés több tételt is tartalmazhat, akkor minden egyes rendelési adathoz több rendeléstétel is tartozhat, vagyis új egyed szükséges, melyben a konkrét rendelésbeli cikk és az abból megrendelt mennyiség kerül tárolásra.

Később meglátjuk, hogy milyen algoritmus szerint járnak el az egyedek kialakításánál relációs modell esetén.

A kapcsolatok fajtái adatmodelltől függetlenül:

1:1 kapcsolat

1:N kapcsolat

N:M kapcsolat

Az egy-egy típusú kapcsolat:

Az egyik egyedhalmaz mindegyik eleméhez a másik egyedhalmaznak pontosan egy eleme kapcsolódik.

Például DOLGOZÓK : KÜLSŐK

A példa az „alegyed" esetére (az összes dolgozónak egy részhalmaza a külsők halmaza) vonatkozik, ugyanis ekkor ajánlott az ilyen kapcsolat-fajta. Ez azt jelenti, hogy a minden egyes dolgozóról tárolt törzsadatot egyben tartunk, és azokról, akiket további tulajdonságokkal kell jellemezni, új egyedben tartjuk ezekkel a megkülönböztetett tulajdonságokkal.

Az egy-több típusú kapcsolat:

Az A egyedhalmaz mindegyik eleméhez a B egyedhalmaznak több eleme is tartozik.

Például VEVŐ : RENDELÉS

Ahogy fentebb írtuk, 1 vevőhöz több rendelés is tartozhat, míg fordítva nem igaz: egy rendelés kizárólag egy vevőtől jön.

A több-több típusú kapcsolat:

Az A egyedhalmaz minden eleméhez a B egyedhalmaz több eleme tartozhat, és fordítva.

Például TERMÉK : ALKATRÉSZ

Ha végiggondoljuk, hogy 1 termék több alkotóból állhat, és 1 alkatrész is több terméknek lehet az alkotója, akkor rádöbbenünk, hogy van ilyen kapcsolat-fajta. Figyelem, amint modellt választunk (a logikai modell fizikai megvalósításra kerül), a kapcsolatfajták csökkenhetnek.

Három adatmodell létezik az alapelemek fizikai tárolásától függően

|  |  |  |  |
| --- | --- | --- | --- |
|  | egyed | tulajd | kapcs |
| **hálós, hierarchikus** | + | - | + |
| **relációs** | + | + | - |
| **objektumos** | + | + | + |

1. ábra

objektum-relációs: vegyes adatmodell

A hálós modellben gráffal ábrázolható az adatbázis: a gráf csúcsain az egyed-előfordulások, az éleken a köztük lévő kapcsolat helyezkedik el. A legkisebb címezhető egység a rekord, tehát a tulajdonságok leírása csak a dokumentációban található meg.

A relációs modellben, ahol az egyed egy táblázat, óriási szerepet kapnak a tulajdonságok. Viszont fizikailag nem tárolódnak a kapcsolatok, csak a dokumentáció tartalmazza. (Amennyiben a külső kulcsok megszorításként való definiálására gondolnak, az bizony logikai utalás: csak egy program segítségével lehet az értékek egyezőségét megkeresni.) A PC-k elterjedésével monopolhelyzetbe kerültek az ilyen modellt kezelő adatbázis-kezelők, és így kifejlődött a szabványos lekérdező nyelvük (az SQL).

A tiszta objektumos modellben a tulajdonság lehet felsorolt típusú, és egy ilyenben a vele kapcsolatban álló másik egyed rekordjainak fizikai címeit tartjuk. Ez a kapcsolat fizikai tárolása. Meg kell jegyezni, hogy az ilyen adatbázis-kezelők nem tudnak elterjedni, mert még nincs szabványos lekérdező nyelvük, vagyis programozni kell minden funkciót.

Viszont gyorsan fejlődnek az öszvér-megoldások, amikor a robosztus relációs adatbázis-kezelőket javítják fel az objektumos lehetőségekkel.

Miután a nagy relációs adatbázis-kezelő rendszerek gyártói igyekeznek az SQL nyelvet állandóan bővíteni, ez az újabb szabványok bevezetését eredményezi, de az eredeti szabványt mindegyik termék érti. Idén megismerjük az első szabvány SQL nyelvet, jövőre pedig az újabb szabványok lehetőségeit.

Senki se higgye, hogy ezek szerint a legfejlettebb modell a relációs, mert minden feladat meghatározza saját alkalmas modelljét, csak a körülmények (pl. valamely adatbázis-kezelő megléte) befolyásolják a rendszerfejlesztési választást. Az igazsághoz az is hozzátartozik, hogy hálós adatbázis-kezelő csak nagygépes operációs rendszerekre létezik, a tiszta objektumos adatbázisok kezeléséhez pedig programozói tehetség kell.

Ezek után a definíciók:

Az adatmodell véges sok egyednek, azok véges számú tulajdonságainak és kapcsolataiknak a halmaza.

Az adatbázis véges sok egyed-előfordulásnak, azok véges sok tulajdonságértékének és kapcsolat-előfordulásának az adatmodell szerint szervezett együttese.

A PC-k elterjedésével helyzetbe került modell kezelésének fejlesztése Codd 1971-es eredményeire alapozva állandó fejlődésben van, és számos szoftvercég érdekeit tiszteletben tartva folyik a feldolgozás nyelvének szabványosítása.

A korai modellekkel szemben a relációs modellt egy magasszintű lekérdező nyelv támogatja, amely nemcsak a lekérdezést, hanem az adatdefiniálást és az adatmanipulálást is ellátja. Ez a nyelv az SQL. Az első ilyen adatbázis az R adatbázis (IBM, 1976) volt, melynek kidolgozói - Codd, Chen és Date - megvalósították azt a törekvést, hogy az adatok tárolási módja független legyen a logikai adatszerkezettől.