Tartalom

- Adatbázisok (relációs, objektum relációs, NoSQL)
- Adatbáziskezelő rendszerek
- Adatbázisok felépítése
- Adatbázisok tervezése

Adatbázisok

Az adatbázis fogalma

- Adatbázison valamely cél megvalósításának érdekében gyűjtött, feldolgozott és tárolt adatok a cél elérésének érdekében megfelelően rendszerezett és rendezett struktúrában tárolt változatát értjük.
- Az adatok ömlesztett tárolása nem tekinthető adatbázisnak, mivel hiányzik belőle a rendszerezés és a struktúra.

Adatbázisok tárolási formája

- Papíron, irattárban rendezett, rendszerezett formában
- Elektronikus úton, számítógépes rendszerekben

Adatbázisok

Példák

- Banki rendszerek
- Vállalati nyilvántartások
- Repülőgép helyfoglalási rendszerek
- Könyvtári nyilvántartások
- Irattári nyilvántartások

Alapvető hozzáférési igények

- Valós időben (azonnal)
- Párhuzamosan (egyszerre sok felhasználó)

Adattárolás

Történelmi háttér

- Adatokat azóta tarolunk, mióta létezik az írás
- Az első adatbázisok a kartotékok, irattárak voltak
- A számítógépek megjelenése és elterjedése lehetővé tette az adatok elektronikus tárolását
- Az első számítógépes adatbázisok tárolása lyukszalagon, lyukkártyán történt, amelyek nem voltak direkt módon elérhetők a számítógépek számára

Adattárolás

Történelmi háttér

- A mágneses adathordozók megjelenése és elterjedése egyszerűsítette az adatok tárolását és hozzáférését
- Ekkor még egyedi, speciális szoftverek kellettek az adatok kezeléséhez
- A számítástechnika gyors fejlődése (Moore törvény) lehetővé tette egyre nagyobb mennyiségű adatok feldolgozását és tárolását

Adattárolás

Történelmi háttér

- A számítógépek fejlődésével együtt fejlődtek a programozási nyelvek és egyre több lehetőségnyílt az adatok elektronikus feldolgozására és tárolására
- Az adatok mennyiségének növekedésével, az adatok egyedi feldolgozása és tárolása fárasztó és időrabló feladattá vált
- Megjelent az adatok egységes és általános módon való feldolgozásának igénye
- Ennek mentén fejlődtek ki az Adatbáziskezelő rendszerek (DBMS)

Adatbáziskezelő rendszerek

Adatkezelési elvárások

- Tegye lehetővé tetszőleges adatbázisok (adatszerkezetek) létrehozását
- Szabványos módon támogassa az adatok lekérdezését és módosítását (egységes lekérdező nyelv)
- Támogassa nagy mennyiségű adat tárolását és visszakereshetőségét hosszútávon

Adatbáziskezelő rendszerek

Biztonsági elvárások

- Garantálja az adatok biztonságát a meghibásodással és illetéktelen hozzáféréssel szemben
- Felügyelje az adatok hozzáférését az adatokhoz, olyan módon, hogy az egyes felhasználók műveletei ne legyenek hatással a többi felhasználó munkájára
- Az egyidejű hozzáférések ne okozhassanak adathibát vagy inkonzisztenciát

Adatbáziskezelő rendszerek célja

Absztrakciós réteg

- A fejlesztőnek ne kelljen foglalkozni a fizikai szintű adattárolással
- Rejtse el a felhasználó elől a számítógépek architektúráját
- Az eszközök típusától függetlenül az alkalmazás ugyanúgy használható legyen az adatbáziskezelő által támogatott minden platformon

Adatbáziskezelő rendszerek célja

Függetlenség az adatelérés módjától

- Az operációs rendszerek különböző adatelérési módokat támogatnak
- Rejtse el az adatelérés módját a felhasználó vagy fejlesztő elől akinek a lekérdezés megfogalmazása és ne az adatok előállítási módja legyen feladat
- Több adatelérési mód rendelkezésre állása esetén az optimális kiválasztása

Adatbáziskezelő rendszerek célja

Függetlenség az adattárolástól

- Az adatok szerkezeti változásai minél kevésbé befolyásolják az alkalmazások működését
- Az adatok szerkezeti bővítése ne okozzon problémát az alkalmazás működésében

Adatbázis modellek (napjainkban)

Relációs adatbázis modell

- Az adatok táblázatos ábrázolása, megjelenítése
- Egy sor egy entitást ír le
- A táblázatokat nevezzük relációknak

Objektum-relációs modell

- A relációs adatmodell bővítésével állt elő
- Az objektum orientált megközelítésben használt objektum és öröklődési modellt alkalmazza a relációkat megvalósító táblákra
- Támogatja az adattípusok bővítését saját adattípusokkal (objektumok)

Adatbázis modellek (napjainkban)

NoSQL – Not only SQL

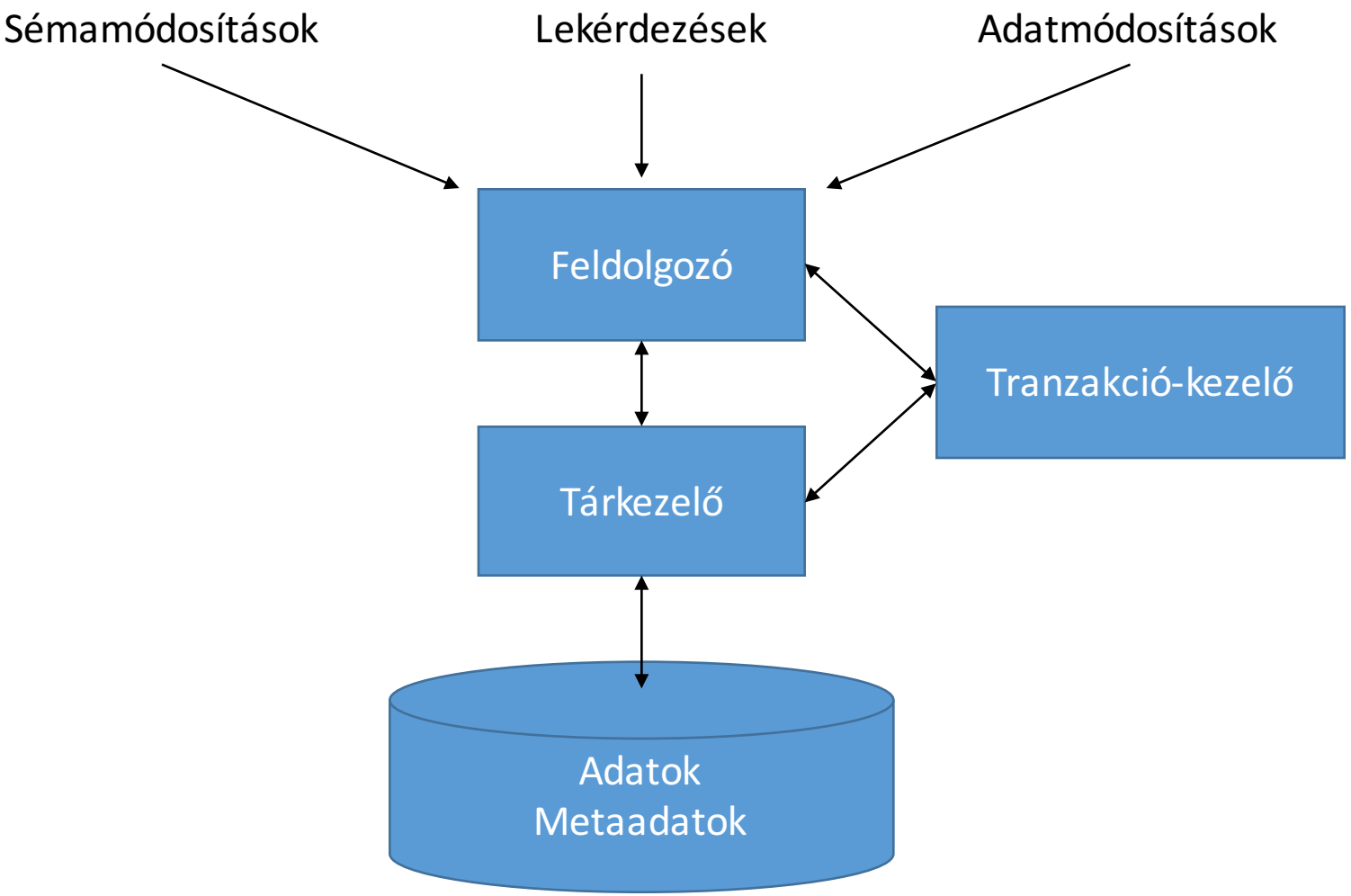
- Két fontos jellemző amelyek mentén fejlesztik: **performancia** és **skálázhatóság**
- A relációs adattárolástól eltérő tárolási módot használ
  - kulcs-érték párok
  - gráf struktúra

Adatbázis modellek (napjainkban)

NoSQL előnyei

- Nagy mennyiségű adat kezelése
- Agilis módszertanok támogatása (iteráció és gyakori kód publikálás)
- Dinamikus adatbázis séma (nincs előre definiált struktúra, méret meghatározás) – pl. új attribútumok hozzáadása esetén nem kell a meglévő adatokat karbantartani
- Osztott feldolgozás támogatása – automatikus partícionálás – az alkalmazásnak nem kell tudnia a mögötte álló szerver hierarchiáról
- Automatikus adatreplikáció, nem kell ezzel foglalkoznia az alkalmazásnak

Adatbáziskezelők felépítése



Forrás: Jeffrey D. Ullman – Jenifer Widom: Adatbázisrendszerek

Adatbázis műveletek

Séma (meta-adat) módosítások

- Végrehajtásukhoz speciális jogosultságra van szükség
- Data Definition Language (DDL) segítségével történik
- Műveletek a meta-adatok szintjén:
  - Séma létrehozás
  - Séma törlés
- Struktúrák:
  - Létrehozása
  - Módosítása
  - Törlése

Adatbázis műveletek

Adatmanipuláció

- Data Manipulation Language (DML) segítségével történik
- Elvégezhető műveletek
  - Adatok létrehozása
  - Adatok mentése
  - Adatok keresése
  - Adatok módosítása
  - Adatok törlése

Relációs adatbázisok

Adatstruktúrák relációs adatbázisokban

- Adattáblák (relációk)
- Adattáblák sorokból és oszlopokból állnak
- Az oszlopokat attribútumoknak is nevezzük
- A sorok azonosítóval rendelkezhetnek (elsődleges kulcs)
- A táblák oszlopai (attribútumai) hivatkozhatnak másik táblák elsődleges kulcsaira – idegen kulcsok

Relációs adatbázisok

Szemelyek

ID	Név	Beosztás	Szervezet	Törölt
1	Kiss János	Könyvelő	1	Hamis
2	Nagy Béla	Informatikus	2	Hamis
3	…	…	…	…

Adatkapcsolatok: Idegen kulcs

Hivatkozások másik tábla elsődleges kulcsára

Szervezetek

ID	Név	Leírás	Törölt
1	Pénzügy	Pénzügyi és …	Hamis
2	Informatika	Informatikai …	Hamis
3	…	…	…

Elsődleges kulcs

Egyedi azonosító

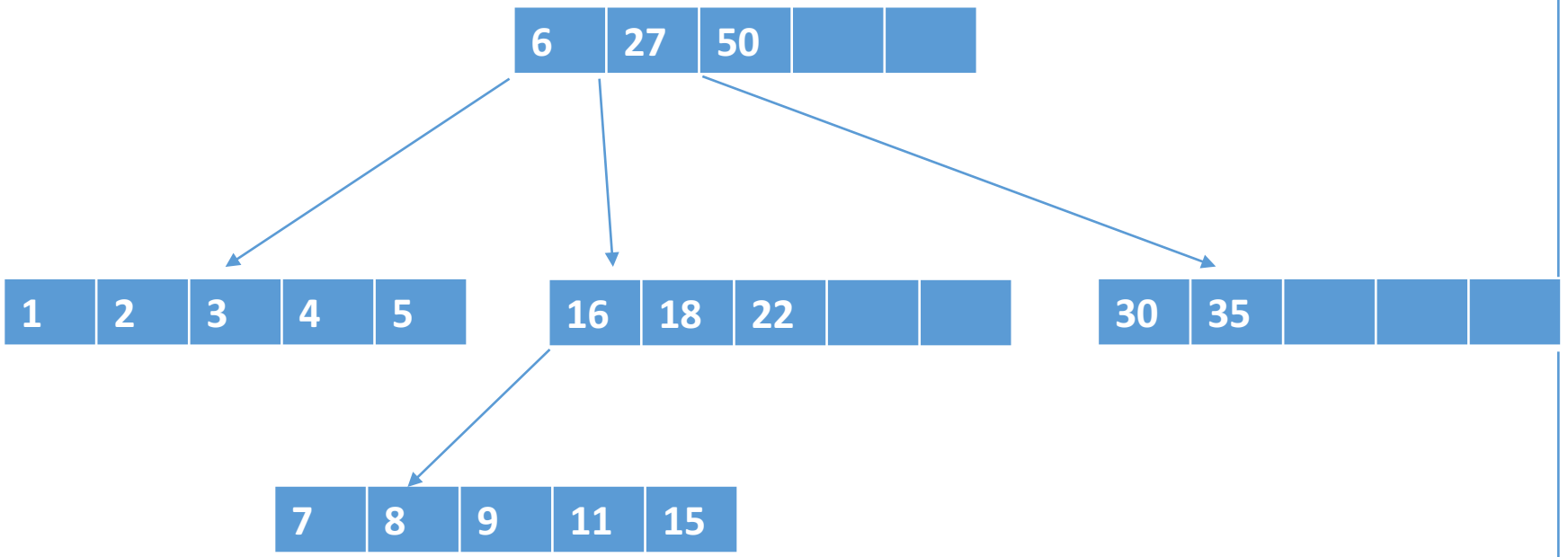
Relációs adatbázisok

Indexek

- Adatok gyors hozzáférését biztosítják
- Különböző típusú indexek léteznek, az elterjedtebbek:
  - B-fa index
  - Bittérkép index
  - teljes szövegindexelés
- A megfelelő indexek létrehozása felgyorsítja az adatok kezelését
- Túl sok index létrehozása lassítja az adatok létrehozását és módosítását (minden adatot fel kell vennie a rendszernek az indexekbe is)

Relációs adatbázisok

B-fa indexek



Relációs adatbázisok

Bittérkép indexek

Tábla
Osztályzat oszlop kevés fix (előre rögzített) értékkel

ID	Név	Osztályzat
1	Kiss Pista	1
2	Nagy Béla	5
3	Kovács András	4
4	Laza Kati	4
5	Erős Pista	5

Bittérképek
Minden értékhez van egy bittérkép

1	2	3	4	5
1	0	0	0	0
0	0	0	0	1
0	0	0	1	0
0	0	0	1	0
0	0	0	0	1

Relációs adatbázisok

További adatbázis objektumok

- Gyári függvények és eljárások
  - Támogatják a felhasználót tetszőleges eredmények előállításában
  - Lehetővé teszik az adatbázisban tárolt adatok tetszőleges átalakítását
- Felhasználói adattípusok
  - Lehetővé teszik az adatbázis-kezelő rendszerek adattípusainak bővítését

Relációs adatbázisok

Tárolt eljárások és függvények

- Felhasználó által definiált programcsomagok
- Megírásuk adatbáziskezelőbe integrált programozási nyelven történik
- Lehetővé teszik az adatok helybeli feldolgozását az adatbáziskezelő rendszeren belül
- Nem kell az adatokat kiolvasni és átemelni a kliensre és visszaírni az adatbázisba
- Kiváló batch adatfeldolgozási feladatok elvégzésére
- Alkalmas komplex triggerek megírására

Relációs adatbázisok

Tárolt eljárások és függvények

- A különböző adatbáziskezelő rendszerek más-más programozási nyelvet kínálnak a cél megvalósítására
- Alkalmazásuk nagyobb performanciát eredményez, főleg nagy mennyiségű adat feldolgozásakor – nincs adatmozgatás hálózaton, csak egy kontextusváltás a szerveren belül
- Legismertebb programozási nyelvek
  - Oracle – PL/SQL
  - MSSQL – Tanzasct-SQL
  - pgSQL – PL/pgSQL
  - …

Relációs adatbázisok

Triggerek

- Adatok írása és olvasása közben meghívott eseménykezelő eljárások
- Az adattáblákba beírt adatok ellenőrzésére, kiegészítésére, átalakítására használják, például üres oszlop tartalmának feltöltése, adat érvényességének ellenőrzése, stb.
- Lehet soronként (for each row) és utasításonként végrehajtott
- Lehet adatírás előtt vagy után végrehajtott (before insert, before update, after insert, after update)
- Megírásuk adatbáziskezelő specifikus

Relációs adatbázisok

Megszorítások

- Primary key (elsődleges kulcs)
  - Egyedi érték az oszlopon belül
  - Egyedi index készül hozzá
- Foreign key (idegen kulcs)
  - Másik tábla elsődleges kulcsára való hivatkozás
  - Referencia integritás kikényszerítésére
  - Csak olyan kulcsérték szerepelhet a táblában, amely létezik a hivatkozott tábla elsődleges kulcsértékei között
- Ellenőrzési megszorítás
  - Oszlop értékkészletének a megadása

Relációs adatbázisok

Partícionálás

- Adatok tárolásának szegmentálása az adattáblákban tárolt adatok alapján
- Leginkább nagy mennyiségű adat gyorsabb lekérdezése esetén használják
- Például: számlák szétosztása tábla partíciók között kiállítási hónap vagy év alapján
- Lekérdezések végrehajtásának felgyorsítása: ha a partícionálásban érintett adat mentén keresünk, a rendszer csak azokat a partíciókat veszi figyelembe, amelyek releváns adatot tartalmazhatnak
- Megvalósítása adatbáziskezelő függő

Adatok lekérdezése (SQL)

Standard Query Language

- Szabványos lekérdező nyelv
  - Minden relációs adatbáziskezelő támogatja
  - Lehetővé teszi az adatok alatti adatbáziskezelő motor cseréjét a programkód cseréje nélkül
- Gyártói kiegészítések
  - Kényelmesebbé teszik a lekérdezések használatát
  - Nagyobb performanciát nyújtó opciók
  - Platformfüggővé teszik az alkalmazást

Adatok lekérdezése (SQL)

SQL utasítások

- Insert – adatsorokok beszúrása adattáblákba
- Update – adatsorok attribútumainak módosítása adott feltételek mentén
- Delete - adatsorok törlése adott feltételek mentén
- Select – adatok lekérdezése adattáblákból

Adatok lekérdezése (SQL)

Select utasítás

- **SELECT** kifejezés1 as oszlop, kifejezés2 as oszlop, …
- **FROM** tábla1 alias1 [**INNER**| **LEFT OUTER**| **RIGHT OUTER**] **JOIN** tábla2 alias 2 ON feltétel2 … [**INNER**| **LEFT OUTER**| **RIGHT OUTER**] JOIN táblaX aliasX **ON** feltételX
- **WHERE** feltétel
- **GROUP BY** kifejezések
- **HAVING** feltétel
- **ORDER BY** kifejezések
- **LIMIT** kezdő adat, adatmennyiség

Adatok lekérdezése (SQL)

Select – Elővett adatok listája

- **SELECT** kifejezés1 as oszlop, kifejezés2 **AS** oszlop, …
- Kifejezéseket adhatunk meg, amelyek az
  - adattábla oszlopainak értékeiből,
  - konstansokból,
  - függvényhívásokból (gyári és saját)
  - műveletekből állnak
- Az előállított értékhez tetszőleges elnevezést kapcsolhatunk

Adatok lekérdezése (SQL)

FROM – Honnan vesszük elő

- **FROM** tábla1 alias1 [INNER| LEFT OUTER| RIGHT OUTER] JOIN tábla2 alias 2 ON feltételek2 … [INNER| LEFT OUTER| RIGHT OUTER] JOIN táblaX aliasX ON feltételekX
- Egy lekérdezésben egy vagy több táblából is vehetünk elő adatokat
- DESCARTES szorzatot (minden sor – minden sorral)
- Adhatunk meg összekapcsolási feltételeket JOIN

Adatok lekérdezése (SQL)

FROM – Honnan

- Kiválogathatjuk:
  - csak az összekapcsolási feltételeknek megfelelő sorokat (INNER JOIN)
  - a bal oldali tábla sorait hozzákapcsolva a jobb oldali tábla sorait amelyek megfelelnek az összekapcsolási feltételnek (LEFT OUTER JOIN)
  - a jobb oldali tábla sorait hozzákapcsolva a jobb oldali tábla sorait amelyek megfelelnek az összekapcsolási feltételnek (RIGHT OUTER JOIN)

Adatok lekérdezése (SQL)

WHERE – Milyen feltételeknek mentén

- **WHERE** feltétel
- Az adattáblákból tetszőleges feltételek mentén válogathatunk sorokat
- A **feltétel** eredménye egy logikai érték (igaz/hamis), amely további kifejezések tetszőleges összekapcsolásából állhat
- Például: **WHERE tabla1.attributum2=10 and tabla2.attributum3> tabla1.attributum2**

Adatok lekérdezése (SQL)

GROUP BY - csoportosítás

- **GROUP BY** kifejezések
- Összegzések, statisztikák előállítása
- A következő kérdésekre adott válasz:
  - Hány olyan sor van … **COUNT**
  - Mennyi az összege a … **SUM**
  - Mennyi az átlaga … **AVG**
  - További statisztikai függvények
- A kifejezések által alkotott csoportonként kapjuk a választ
- A **SELECT** után csak statisztikai függvények és a **GROUP BY** kifejezési szerepelhetnek

Adatok lekérdezése (SQL)

HAVING – csoportok szűrése

- **HAVING** feltételek
- A csoportosítás során előállított adatokból válogathatunk a feltétel mentén
- Az eredményben azok a csoportok jelennek meg, amelyek eleget tesznek a HAVING után szereplő feltételeknek
- A feltétel tipikusan a csoportosítással előállított valamely statisztikai értékre vonatkozik
- Példa: Azon csoportok megjelenítése amely több mint 5 sorból állt össze: **HAVING COUNT(ID)>5**

Adatok lekérdezése (SQL)

ORDER BY – eredmény sorba állítása

- **ORDER BY** kifejezések
- Adatok sorba állítása kifejezések mentén
- Az egyes kifejezések esetében megadhatjuk, hogy melyek mentén állítjuk növekvő (ASC) illetve csökkenő (DESC) sorrendbe az adatokat
- Példa: **ORDER BY SZERVEZET ASC, FIZETES DESC**

Adatok lekérdezése (SQL)

LIMIT – szegmentálás

- **LIMIT** kezdő adat, adatmennyiség
- Gyakran fordul elő, hogy az eredmény nagy mennyiségű adatot tartalmaz, amit egyszerre nem lehet megjeleníteni a képernyőn, ilyenkor lapozási technikát alkalmazunk
- Megadhatjuk, hogy az eredmény valamely sorától adott sormennyiséget tartalmazzon az eredmény
- Adatbáziskezelő szervertől függően ennek szintaktikája változhat (Oracle esetében ROWNUM)

Relációs adatbázisok

Lekérdezések optimalizálása

- Nagy adatmennyiség esetében nem mindegy, hogy milyen sorrendben hajtja végre a rendszer az eredmény összeállítását
- Alapvetően kétféle optimalizálási módot különböztetünk meg
  - Szabály alapú
  - Költség alapú
- Végrehajtási tervek készítése, kiértékelése

Lekérdezések optimalizálása

Az optimalizálás technikákra támaszkodik

- Minden adatbázis háttérműveletnek megvan a prioritása
- Indexekre támaszkodik (nem kell végigolvasni a táblákat)
  - Full table scan (teljes tábla olvasás – nincs megfelelő index)
  - Index scan (index olvasás majd csak az indexben megtalált sorok elővétele)
- A táblák összekapcsolásához többféle technikát is használhat
  - NESTED LOOP (összefésüli a két tábla sorait)
  - SORT-MERGE (a táblák sorba állított sorait fésüli össze)
  - HASH JOIN (kihasználja a SORT-MERGE algoritmus karakterisztikáit, az azonos értékek egymás melletiségét, csoportosíthatóságát)

Lekérdezések optimalizálása

Szabály alapú optimalizálás

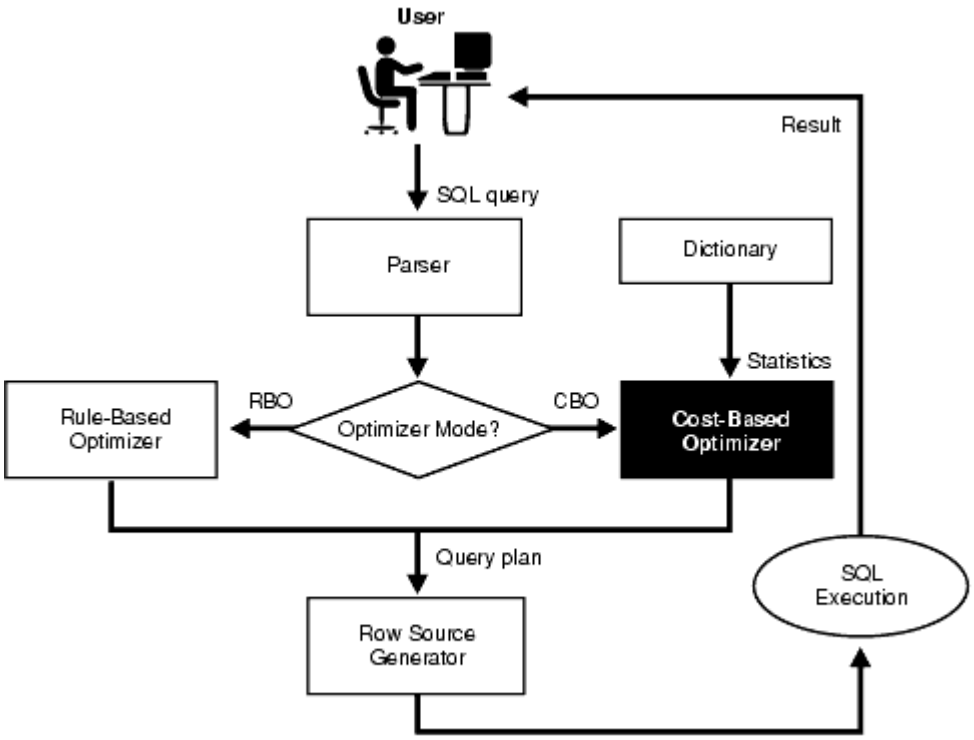
- Minden adatbázis műveletnek megva a prioritása
- A lekérdező motor meghatározza az eredmény előállításához szükséges feldolgozási műveleteket
- A műveletek prioritása alapján az adatok minőségétől függetlenül állítja sorba a különböző végrehajtási terveket

Lekérdezések optimalizálása

Költség alapú optimalizálás

- Alkalmazásához adatminőségi statisztikákra van szükség (mennyiség, eloszlás, stb.)
- Minden adatbázis műveletnek becsülhető a költsége statisztikák alapján
- A lekérdező motor meghatározza az eredmény előállításához szükséges feldolgozási műveleteket
- Az optimalizáló
  - meghatározza a műveletek költségét az adatok minőségétől függően
  - összegezi a bekerülési költségeket és ez alapján állítja sorba a végrehajtási terveket

Lekérdezések optimalizálása



Forrás: Oracle.com

Lekérdezések optimalizálása

```
EXPLAIN PLAN FOR
SELECT e.employee_id, j.job_title, e.salary, d.department_name
    FROM employees e, jobs j, departments d
    WHERE  e.employee_id < 103
       AND e.job_id = j.job_id
       AND e.department_id = d.department_id;
```

```
-------------------------------------------------------------------------------------
| Id  | Operation                     |  Name        | Rows  | Bytes | Cost (%CPU)|
-------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT              |              |     3 |   189 |    10  (10)|
|   1 |  NESTED LOOPS                 |              |     3 |   189 |    10  (10)|
|   2 |   NESTED LOOPS                |              |     3 |   141 |     7  (15)|
|*  3 |    TABLE ACCESS FULL          | EMPLOYEES    |     3 |    60 |     4  (25)|
|   4 |    TABLE ACCESS BY INDEX ROWID| JOBS         |    19 |   513 |     2  (50)|
|*  5 |     INDEX UNIQUE SCAN         | JOB_ID_PK    |     1 |       |            |
|   6 |   TABLE ACCESS BY INDEX ROWID | DEPARTMENTS  |    27 |   432 |     2  (50)|
|*  7 |    INDEX UNIQUE SCAN          | DEPT_ID_PK   |     1 |       |            |
-------------------------------------------------------------------------------------
```

Forrás: Oracle.com

Relációs adatbázisok tervezése

Adattáblák meghatározása

Szemelyek és szervezetek

ID	Név	Beosztás	Szervezet	Szervezet leírása	Törölt
1	Kiss János	Könyvelő	Pénzügy	Pénzügyi és …	Hamis
2	Nagy Béla	Informatikus	Informatika	Informatikai …	Hamis
3	…	…	…		…

Normalizálás — **Redundanciák kiszűrése**

Szemelyek

ID	Név	Beosztás	Szervezet	Törölt
1	Kiss János	Könyvelő	1	Hamis
2	Nagy Béla	Informatikus	2	Hamis
3	…	…	…	…

Szervezetek

ID	Név	Leírás	Törölt
1	Pénzügy	Pénzügyi és …	Hamis
2	Informatika	Informatikai …	Hamis
3	…	…	…

Relációs adatbázisok tervezése

Indexek meghatározása

- Az adat lekérdezési igények határozzák meg az alkalmazandó indexeket
  - adattartalom
  - csoportosítás
  - sorba állítás
- Egy oszlopos vagy több oszlopos index?
  - Egy adatelővételi művelet esetében egy index alkalmazható
  - Több oszlopos index akkor alkalmazandó, ha több oszlopra történik a szűrés
  - Ha több oszlop adatai szerepelnek a szűrési feltételekben, az optimalizáló a legmegfelelőbbnek talált indexet alkalmazza
- Minél több az index annál lassúbb az adatlétrehozás és módosítás

Köszönöm a figyelmet!