Nem-relációs adatbáziskezelés (No-SQL, BigData)

Gajdos Sándor

2019. ápr. 30.

Motiváció

- A feladat pontosabb értelmezése: Hogyan lehet adatokat minél nagyobb hatékonysággal kezelni?
- Mit jelent az „adatkezelés”?
- Meddig lehet a rendszer funkcionalitását egyszerűsíteni/csökkenteni?
- Mitől lesz ez adatbáziskezelés?

Történelmi előzmények

- Hierarchikus adatbáziskezelés (IBM, 1960-)
- Hálós adatbáziskezelés (~1970-85)
- Ami mindkettőben volt:
  - Imperatív lekérdezések
    - A programozó dolgozik, nem a számítógép
    - Deklaratív lekérdezések hiánya, nem kell lekérdezés optimalizálás
  - Kapcsolatok megvalósítása direkt linkekkel
    - V.ö: relációsnál adatok értékegyezése alapján – keresés
- Következmény: eleve lényegesen gyorsabbak lehetnek.

A legnagyobb weboldalak (2011.)

Weboldal	Terhelés
Google	24500 tablet szerver, 1.2 millió adatbázis lekérdezés másodpercenként, 16Gb/s lekérdezés
Facebook	Inbox: 100 TB, 150 gépes kaszter Adattárház: 15 Petabyte adat, 1400 gép, 11200 CPU
Youtube	
Microsoft Live, Bing	
Yahoo!	92 Petabyte adat, a legnagyobb lekérdezés 10ezer gépen fut párhuzamosan 73 óráig
Twitter	Adatbázis növekedés 7 TB naponta, 2+ PB évente
Wikipedia	
BBC	
Myspace	
Amazon	

A legnagyobb weboldalak adatbázisai

Weboldal	Főbb adatbázismotor	Adatbázis típusa
Google	GFS, Googe BigTable	Columnar NoSQL
Facebook	Cassandra, Hadoop/HIVE	Columnar NoSQL
Youtube	Memcached	Key-Value
Microsoft Live, Bing	Azure	Tuple store, RDBMS
Yahoo!	Hadoop, PNUTS	Columnar NoSQL
Twitter	FlockDB, Cassandra, Hadoop/Hbase	Graph, Columnar NoSQL
Wikipedia	Memcached, Flatfile, MySQL	Key-Value, Flat file, RDBMS
BBC	CouchDB	Document
Myspace	Aster Data nCuster	MPP RDBMS + MapReduce
Amazon	Amazon Dynamo	Columnar NoSQL

Mi van a relációson túl?

- Hierarchikus DB
- Multidimenziós DB
- Document store
- Gráf DB
- Key/value store (on disk, in RAM)
- Object DB
- ...

Összefoglaló nevük: NoSQL adatbáziskezelők

NoSQL

- Relációs rendszerek gyengéi
  - Sok dokumentum indexelése
  - Nagyforgalmú weboldalak kiszolgálása
  - Adatstream-ek szolgáltatása
- Általános jellemzők
  - Gyenge konzisztenciagaranciák
  - Elosztott architektúra

Hierarchikus adatbázisok

- Az adatok faszerű struktúrákban
- Rekordorientált szemlélet
- 1:N kapcsolatok natív leképezése
- Imperatív lekérdezhetőség
- IMS (IBM), Windows registry (Microsoft)

Multidimenziós adatbázisok

- N-dimenziós tömbök
- Cellák: tényadatok, melyeket a dimenziók koordinátáival címezhetünk
- Segédstruktúrák (indexek)
- Nem tévesztendő össze a relációs alapú dimenzióssal
- Hyperion Assbase, Cognos, Oracle Express, Microsoft,...

Dokumentum adatbázisok

- Natív OODB (vagy lehet réteg egy relációs felett)
- Minden rekord egy dokumentum, aminek tetszőleges számú, nevű és méretű mezője lehet (szemistrukturált adat)
- Nincs üres mező, a mezők többszörös adatelemeket is tartalmazhatnak
- hasonló „eredmény” érhető el pl. XML-lel -> minden XML adatbázis dokumentum DB is
- Egyszerű használat, könnyű programozhatóság
- CouchDB, IBM Lotus Notes, MongoDB, RavenDB, XML databases: MarkLogic Server, eXist

Gráf adatbázisok (ld. holnapután részletesen)

- Az adattárolást gráfstruktúrák (csomópontok, élek + attribútumok) valósítják meg
- A gráf lehet tetszőleges vagy spec. (ld. hálós adatbázisok)
- OO alkalmazások esetén (is) hatékonyabb
- Gyakran változó séma esetén
- IDMS (hálós), Neo4j, AllegroGraph, Core Data, FlockDB

Key/value adatbázisok

- A legegyszerűbb NoSQL DB
- Kulcs+adat (akár blob, a tartalma közömbös)
- Az alkalmazás felelőssége az adat értelmezése
- Nincs hagyományos séma
- Hozzáférés gyakran csak a kulcsértéken kereszül, igen jól optimalizálható
- Természetesen az adat is indexálható
- A tárolás lehet diszken, memóriában, rendezve vagy anélkül, módosítást kizárva (ld. CDB),...
- BigTable, CDB, Memcachedb, Redis, SimpleDB, Tokyo Cabinet, Tuple space, NMDB, Memcachedb, Berkeley DB,...

Mit lehet feladni a sebességért?

- Elvileg mindent, ami egy DBMS-t jellemez:
  - Atomicity
  - Consistency
  - Isolation
  - Durability
- A relációs modell előnyeit is:
  - Kényelem
  - Változatos lekérdezések hatékonyan