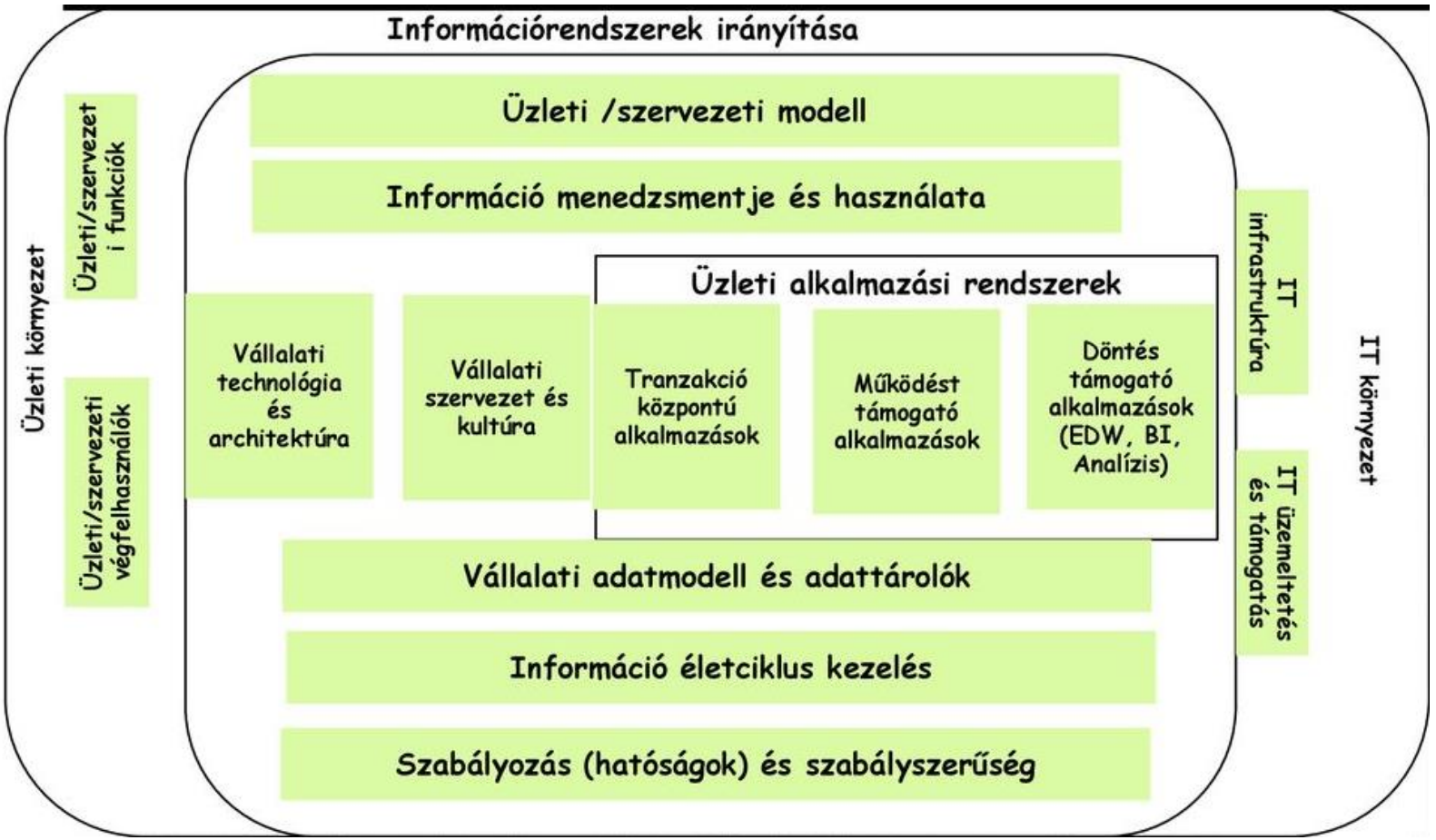
A nagy mennyiségű adatok kezelésénél alkalmazható adatbázis technológiák

Big data

- Mik a nagyméretű adatok (Big data)?
- 3V modell (volumen, viharsebesség, változatosság)
- Adatmennyiség = data volume
- Nagysebességgel keletkezik = high velocity
- Nagy változatosság = variety
- Amikor a hagyományos adatkezelési eljárások akadályokba ütköznek
- Inkább egy koncepció, mint egy pontos definíció, fogalom meghatározás

Példa – vállalati környezet

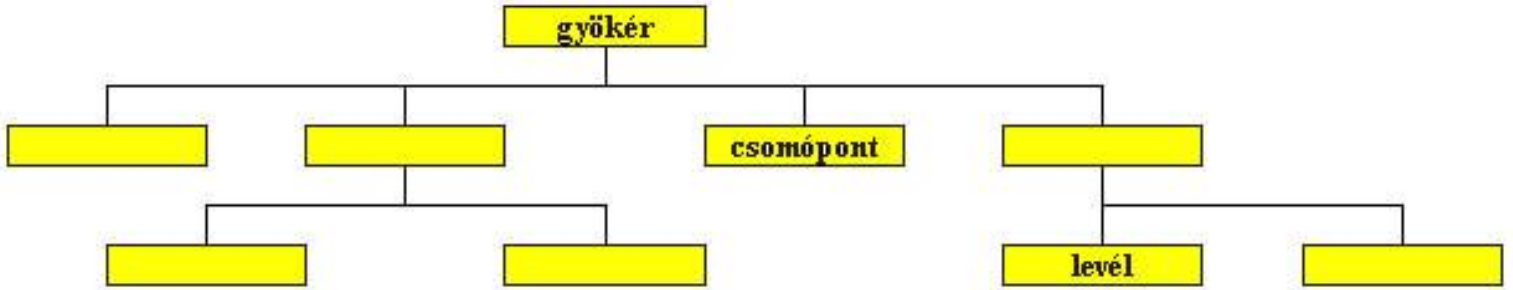


Hagyományos adatmodellek

- Hagyományos (vállalati) információrendszerek információkezelése
- Strukturált adatok, tárolási lehetőségek
- Relációs adatbázis-kezelő
- Objektum-orientált adatbázis-kezelő
- Hálós adatbázis
- Hierarchikus adatbázis

Hierarchikus adatbázis modell

- A hierarchikus modell volt a legelső az adatbáziskezelőkben és egyben a leginkább korlátozott.



- Az adatbázis több egymástól független fából állhat. A fa csomópontjaiban és leveleiben helyezkednek el az adatok. A közöttük levő kapcsolat, szülő-gyermek kapcsolatnak felel meg.

Hierarchikus adatbázis modell

- Így csak 1:n típusú kapcsolatok képezhetők le segítségével. Az 1:n kapcsolat azt jelenti, hogy az adatszerkezet egyik típusú adata a hierarchiában alatta elhelyezkedő egy vagy több más adattal áll kapcsolatban.
- A hierarchikus modell természetéből adódóan nem ábrázolhatunk benne n:m típusú kapcsolatokat (lásd a háló modellt). Emellett további hátránya, hogy az adatok elérése csak egyféle sorrendben lehetséges, a tárolt hierarchiának megfelelő sorrendben.

Hierarchikus adatbázis modell

- A hierarchikus adatmodell alkalmazására a legkézenfekvőbb példa a családfa.
- A főnök-beosztott viszonyok vagy egy iskola szerkezete is leírható ebben a modellben.
- Az iskola esetén többféle hierarchia is felépíthető. Egyrészt az iskola több osztályra bomlik és az osztályok tanulókból állnak. Másrészt az iskolát az igazgató vezeti, a többi tanár az ő beosztottja és a tanárok egy vagy több tantárgyat tanítanak. (redundancia)

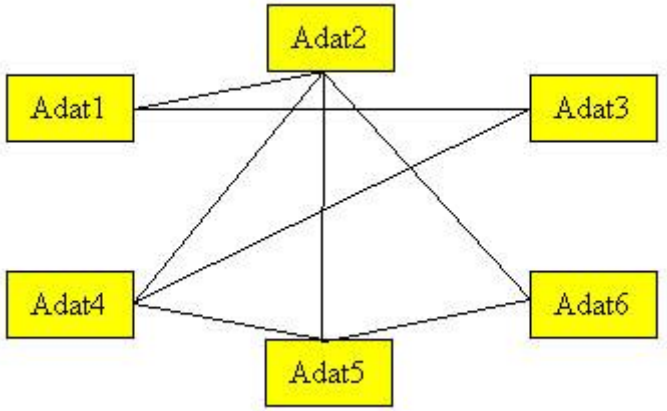
Hierarchikus adatbázis modell

Hálós adatbázis modell

- A hálós adatmodell esetén az egyes azonos vagy különböző összetételű adategységek (rekordok) között a kapcsolat egy gráffal írható le.
- A gráf csomópontok és ezeket összekötő élek rendszere, melyben tetszőleges két csomópont között akkor van adatkapcsolat, ha őket él köti össze.
- Egy csomópontból tetszőleges számú él indulhat ki, de egy él csak két csomópontot köthet össze, így minden adategység tetszőleges más adategységekkel lehet kapcsolatban. Ebben a modellben n:m típusú adatkapcsolatok is leírhatók az 1:n típusúak mellett.

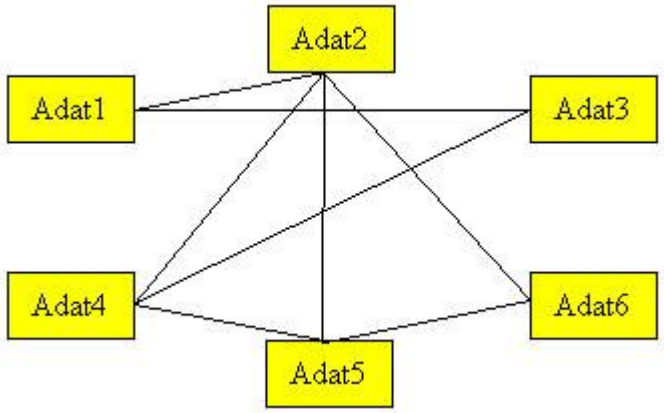
Hálós adatbázis modell

- A hierarchikus és a hálós modell esetén az adatbázisba fixen beépített kapcsolatok következtében csak a tárolt kapcsolatok segítségével bejárható adat-visszakeresések oldhatók meg hatékonyan (sok esetben egyébként hatékonyabban, mint más modellekben).
- További hátrányuk, hogy szerkezetük merev, módosításuk nehézkes.



Hálós adatbázis modell

- Az iskolai példánál maradva az egyes diákok, illetve tanárok közötti kapcsolat hálós modellben írható le. Minden diákot több tanár tanít és minden tanár több diákot tanít. (páros gráf, két diszjunkt halmaz)



Relációs adatbázis modell

- A relációs az egyik legáttekinhetőbb és a 80-as évektől kezdve a legelterjedtebb adatmodell.
- A relációs modellben az adatokat táblázatok soraiban képezzük le.
- A legfontosabb eltérés az előzőekben bemutatott két modellhez képest az, hogy itt nincsenek előre definiált kapcsolatok az egyes adategységek között, hanem a kapcsolatok létrehozásához szükséges adatokat tároljuk többszörösen. Ezzel egy sokkal rugalmasabb és általánosabb szerkezetet kapunk.

Objektum-relációs adatbázis modell

- Az objektum relációs adatmodell a relációs adatmodell bővítésével állt elő.
- Egyrészt az objektum orientált megközelítésben használt osztály, objektum, öröklődés fogalmakat alkalmazza a relációs adatbázis táblákra és a lekérdező nyelvet is ez irányba bővíti.
- Másrészt pedig támogatja az adatmodell bővítését saját adattípusokkal és azokat kezelő beépített függvényekkel.

A nagy méretű adatok kezelése

- Az adatok nagy része (85-90%) ma strukturálatlan formában keletkezik
- Ezeket az adatokat egyelőre nem aknázzák ki:
- Nehéz elemezni (parsing), modellezni, értelmezni
- A nagy méretű adatok esetében:
- A strukturált, félig-strukturált, strukturálatlan és a polistrukturált adatokat is kezelni kell.
- Pl.: e-levél, weblap tartalom, videó, audió stb.
- Adatbázis tartalom, naplóállományok, XML állomá-nyok, strukturálatlan szöveges dokumentumok, weblapok, grafikák.

Vállalati információrendszerek

- Az információrendszerek dokumentum központúvá válnak
- Nem csak a nagy mennyiség és méret, hanem az adatszerkezet típusok magas fokú heterogenitása is probléma.
- Hagyományos rendszerszervezési, elemzési és tervezési módszerek, információkezelési eljárások nem kielégítőek.

Adatbázisrendszerek evolúciója

- RDBMS (Relational Database Management System)
- SQL, relációs adatbázis-kezelés
- Strukturált adat:
- Minden adatelemre létezik meta-adat,
- SQL tárolás, elérés pontosan definiált.
- Strukturálatlan adat:
- Az adatbázis séma nem írja le pontosan.

Új kihívások és megoldások

- Skálázhatóság
- Infrastruktúra menedzsment
- Végfelhasználó kielégítő kiszolgálása
- Adatmodellezés
- NoSQL adatbázisok kifejlesztése
- Map-reduce technológia a párhuzamos feldolgozásra (funkcionális programozás)

Adatbázisrendszerek evolúciója

- Not Only SQL , NoSQL
- DBMS
- Teljesítmény:
- Elosztott hálózati architektúra
- Masszív párhuzamos, konkurens adatfeldolgozás (high concurrency)
- Particionálással szembeni tolerancia
- Kiterjeszthetően skálázható architektúra

NoSQL

- Kulcs-érték pár (key-value pair)
- Gráf adatbázis
- Dokumentum adatbázis
- XML által definiált adatszerkezet

  Az XML biztosítja, hogy méretében, tartalmában a legkülönbözőbb információkat méretükhöz és tartalmukhoz legjobban illeszkedő szerkezetek-ben hozzuk létre digitális, szoftverek által értelmezhető és feldolgozható formában.
- JSON (JavaScript object notation)

Big Data feldolgozásának kérdései

Adattárolás

Hagyományos relációs adatbázis-kezelő rendszerek (RDBMS) előnyei:
komplex tranzakciók,
másodpercenként akár több ezer lekérdezés kezelése,
hatékony lekérdező nyelv.
Hátránya: RDBMS-ekre gyakran nagyon nehéz leképezni a Big Data problémákat.

Big Data feldolgozásának kérdései

Adattárolás

Ennek okai:

- az alkalmazott séma már az adat beérkezése előtt rögzített,
- nagy mennyiségű félig strukturált és strukturálatlan adat kezelése nehézkes,
- az ACID tulajdonságok sok esetben nem praktikusak,
- az architektúra nem mindig teszi lehetővé a könnyű skálázhatóságot (pl. nehezen partícionálható, inhomogén adatok esetében).

Hardver architektúra

A hagyományos RDBMS-ekhez használt hardverek leggyakrabban nagy teljesítményű, párhuzamos adatfeldolgozásra optimalizált számítógépek, amelyekhez külső tárolók kapcsolódnak.
Ez az architektúra hatékony véletlenszerű adatelérést és párhuzamos feldolgozást biztosít, de magas költséggel jár.
A számítóteljesítmény nehezebben skálázható, mint a tárolókapacitás.
A kommunikáció a tároló és a feldolgozást végző gép között overhead.

Hardver architektúra

Nagy adatmennyiség feldolgozásához kis költséggel bővíthető, jól skálázható hardverre van szükség
→**klaszter architektúra**
Kisebb teljesítményű általános célú számítógépek összekapcsolása hálózaton keresztül.
Jellemzően minden csomópont egy elosztott fájlrendszeren (pl. NFS) tárolt adatokon dolgozik.
Számítás-intenzív feladatokhoz jól illeszkedik.
De: mivel az adatok egy elosztott fájlrendszeren érhetők el, ezért jelentős a hálózati adatforgalom→ ez szűk keresztmetszet lehet nagy adathalmazok esetén.

Big Data feldolgozásának kérdései

Kihívások

Adatmodell: hogyan osztható szét az adat hatékonyan a klaszter csomópontjaira?

Programozási modell: hogyan készíthető könnyen olyan hatékony alkalmazás, amely képes a klaszteren futva az adatokat feldolgozni?

Hogyan kezeljük a gépek meghibásodásait (csomópont kiesése, diszkek hibái stb.)?

Big Data feldolgozásának kérdései

Lehetséges megoldás: Apache Hadoop

A Hadoop egy nyílt forráskódú, hibatűrő, elosztott rendszer nagy adathalmazok tárolásához és feldolgozásához.

A Hadoop környezet a következő megoldásokat kínálja:

Adatmodell: *HDFS* elosztott fájlrendszer
Programozási modell: *MapReduce*
Erőforrás kezelés és ütemezés: *YARN*

A Hadoop és RDBMS-ek összehasonlítása

RDBMS: schema-on-write

A sémát az adatok bevitele előtt létre kell hozni.

A betöltés során az adatot az adatbázis belső struktúrájának megfelelő formátumba kell transzformálni.

A séma bővítése vagy módosítása szükséges minden új típusú adat bevitele előtt egy rekordba.

Gyors olvasás, jó kezelhetőség

Hadoop: schema-on-read

Az adatot egyszerűen fel kell másolni a tárolóra, nincs szükség transzformációra.

Új típusú adatok bármikor bekerülhetnek.

Az olvasás/feldolgozás során kell egy megfelelően implementált ETL (kinyerési, átalakítási, betöltési) eszközt alkalmazni.

Gyors írás, flexibilitás

Dr. Hajdu András, Tóth János: Nagy adathalmazok elosztott feldolgozása című tananyaga alapján készült (részben)