

NoSQL adatbázisok



Automatizálási és
Alkalmazott
Informatikai Tanszék

NoSQL jelentése

- Gyűjtőnév a relációstól eltérő adatmodellt használó adatbázisokra
- „Not Only SQL” – félrevezető
- Nem az SQL-t váltja le, hanem a relációs adatmodellt
- Mik a nehézségek a relációs adatmodellel?

Kötött séma



- A tábla definíciója rögzíti a sémát, eltérni nem lehet
- Változtatás bonyolult
- Bővítés bonyolult
- Sokszínű adatokra ráhúzni nehéz
 - > Vagy sok tábla lesz
 - > Vagy sok NULL érték
- A kötött séma ugyanakkor kikényszeríti az adatok helyes formátumát

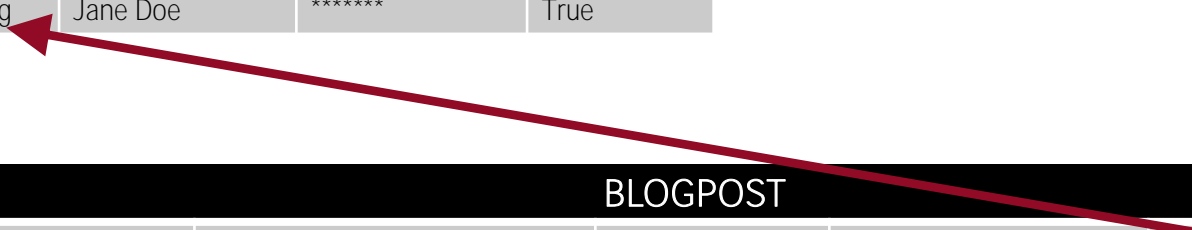
Kapcsolattípusok

- A kapcsolat nem elemi fogalom, helyette kulcshivatkozás
- Egy-egy: mindegy melyik oldalon van a kulcshivatkozás
- Egy-több: a több oldalon hivatkozunk az egy oldalra
- Több-több: mindkét oldalon több elemre kellene hivatkozni
 - > Kapcsolótábla szükséges

Blog alkalmazás relációs sémája

AUTHOR			
Email 	Name	Password	Active
john@example.com	John Doe	*****	True
jane@something.org	Jane Doe	*****	True

BLOGPOST				
Id 	Title	Content	CreationDate	Author 
1	Moving to our new house	...	2017-02-03	john@example.com
2	My new chocolate pie recipe	...	2017-02-15	jane@something.org



Join

- Kulcshivatkozással szimulálunk kapcsolatot
- Kapcsolat bejárása join művelettel
 - > Descartes-szorzat, majd szűrés
 - > n és m rekordot tartalmazó tábláknál $O(n*m)$
 - > Több táblánál tovább szorzódik...
 - > Több-több kapcsolat különösen költséges
- Séma normalizálása
 - > Megszünteti a redundanciát
 - > De szétszórja az adatokat több táblába

NoSQL célkitűzések

- Szolgálja ki a BigData alkalmazásokat
 - > Volume: sok adatot tudjon tárolni
 - > Velocity: képes legyen elég gyorsan írni
 - > Variety: ne legyen túl kötött sémája (vö. relációs)
- Tipikusan nincs kötött séma, vagy opcionális
- Igyekeznek a join jelenséget kiküszöbölni

CAP elmélet

- Konzisztencia: minden olvasás a legfrissebb adatot adja, vagy hibát jelez (Consistency)
- Rendelkezésre állás: minden kérésre hibamentes válasz érkezik (Availability)
- Partícionálhatóság: a rendszer akkor is működik, ha a csomópontok közt elvesznek (késnek) üzenetek (Partitioning)
- Legfeljebb kettő teljesülhet
- Elosztott működés (=skálázhatóság) elengedhetetlen
- Vagy konzisztenciát, vagy rendelkezésre állást kell választanunk...

ACID típusú adatbázisok

- A konzisztenciát választják
- Konzervatívabb megközelítés, pl. relációs adatbázisok
- A konkurenciát tranzakciókkal kezelik
 - > **A**tomicity: vagy az egész lefut, vagy semmi
 - > **C**onsistency: konzisztens állapotba visz át
 - > **I**solation: hatása azzal ekvivalens, mintha egyedül futna
 - > **D**urability: hatása tartós
- Ha ütközés van, akkor rollback

BASE típusú adatbázisok

- A rendelkezésre állást választják
 - > **Basically Available**: általánosan rendelkezésre álló
 - > **Soft State**: az információ lejárhat, érvényét vesztheti
 - > **Eventual Consistency**: a csomópontokon az adatok nem minden pillanatban konzisztensek, de később azok lesznek
- Az erős konzisztencia hiánya problémás...
 - > Lehetnek konkurens módosítások, ezeket fel kell oldani
 - > Nem tudunk semmit az adat frissességéről
- NoSQL adatbázisok döntően BASE elvűek

Sharding

- Elosztott adatbázisoknál használatos technika
- Az adatok egy kulcs szerint szét vannak válogatva szerverek közt
 - > Jól elosztható a terhelés
 - > Tudjuk, hol kell az adatokat keresni
 - > MapReduce is hatékonyan működik vele

NoSQL típusok

- Négy alaptípus
 - > Key-value stores (kulcs-érték tárolók)
 - > Document stores (dokumentumtárak)
 - > Column families (oszlopcsaládok)
 - > Graph databases (Gráfadatbázisok)

Kulcs-érték tárolók

- Az egész egy nagy map (dictionary, asszociatív tömb)
- Lehet memóriabeli vagy perzisztens
- Az érték általában összetett adattípus is lehet
 - > Map
 - > Lista
 - > Halmaz
 - > Rendezett halmaz
 - > Stb.

Dokumentumtárak

- Dokumentum ~ XML, JSON, YAML, BSON
 - > Tulajdonképpen kulcs-érték párok sorozata
 - > Érték is lehet újabb dokumentum
 - > Nincs kötött dokumentumséma, a kulcsok tetszőlegesek lehetnek
 - > Ezért az objektomorientált adatok jól leképezhetők
- Tekinthető specializált kulcs-érték tárnak
 - > Gazdagabb API
 - > Itt a dokumentum belső felépítése bejárható, lekérdezhető

Oszlopcsaládok

- Hasonló a relációshoz, de soronként eltérhetnek az oszlopok
- Sor tulajdonképpen kulcshoz rendel sorrendezett kulcs-érték párokat
 - > Vö. dokumentumtár, kulcs-érték tároló
 - > A határvonal nem éles, inkább nézőpont kérdése
- „Super column”: maga az érték is kulcs-érték párok sorozata

Gráfadatbázisok

- Relációs modellben a kapcsolat nem elemi fogalom
- Gráfokban elemi fogalom az él, konstans lépéssel bejárható
- Általában az élekre is tehetünk property-ket
- Hipergráfokban **egy él kettőnél több** csomópontot is összeköthet

Polyglot perzisztencia

- Minden részfeladathoz a megfelelő eszköz
 - > „Hagyományos” pénzügyi adatok → Relációs
 - > Termékkatalógus → Dokumentum
 - > Ajánlórendszer → Gráf
 - > Bevásárlókocsi → Kulcs-érték
- Kihívások
 - > Komplexitás: sok library, sok interfész
 - > Integritás megőrzése kézileg az adatbázisok közt
- Multi-model adatbázisok, pl. ArangoDB
- <https://martinfowler.com/bliki/PolyglotPersistence.html>

Néhány népszerű NoSQL adatbázis

1. MongoDB (dokumentum)
2. Cassandra (oszlopcsalád)
3. Neo4j (gráf)
4. Redis (kulcs-érték)

MongoDB

- Dokumentumtár
 - > Lehet benne tömb vagy beágyazott dokumentum
- Teljesítmény
 - > Beágyazott dokumentumok gyorsan kiolvashatók
 - > Széleskörű indexelési lehetőségek
- Gazdag JavaScript alapú lekérdezőnyelv
 - > Adat pedig bináris JSON, azaz BSON
- Magas rendelkezésre állás, replikáció
- Skálázhatóság, sharding
- Bosch, Cisco, Forbes, Expedia, The Guardian

(De)normalizálás

- Hivatkozások a kliensben oldhatók fel
- Jobb teljesítményt ad a denormalizálás
- Dokumentum írása atomi művelet
- Gazdag indexelési lehetőségek



<https://docs.mongodb.com/manual/core/data-modeling-introduction/>

MongoDB kritika

- <http://www.sarahmei.com/blog/2013/11/11/why-you-should-never-use-mongodb/>

MongoDB fejlesztői támogatás

- Sok nyelvhez van driver
 - > C, C++, C#, Java, Node.js, Perl, PHP, Python, Ruby, Scala
- <https://docs.mongodb.com/getting-started/shell/drivers/>
- JDBC driver is létezik
 - > <https://www.progress.com/jdbc/mongodb>
- Több REST API implementáció

Cassandra

- Oszlopcsalád, super column támogatással
- FaceBook fejlesztése, Google BigTable alapjain
- Magas rendelkezésre állás, nincs SPOF
- Flexibilis séma
- Elasztikusan skálázható, lineárisan skálázódik
- Gyors írás
- Tranzakciókat is támogat (de rendelkezésre állás az elsődleges)
- FaceBook, Twitter, Cisco, Netflix

Cassandra skálázhatóság

- Egyenrangú csomópontok klaszterbe szervezhetők
- Replikáció: bármelyik node írható/olvasható
 - > Konfliktusnál kliens időbélyege dönt
 - > Rendelkezésre állás, hibatűrés

Cassandra adatmodell

- Keyspace = adatbázis
- Oszlopcsalád = tábla
- **Elsődleges kulcs**
 - > Partíció kulcs: melyik csomóponton lett tárolva az adat
 - Általában UUID
 - > Klaszter kulcs: a partíción belüli sorrend
- Replikációs stratégia
- Replikációs faktor

Cassandra Query Language (CQL)

- Szintaxisa nagyon hasonlít az SQL-re, de sok a megszorítás:
 1. WHERE: feltétel csak kulcsoszlopra adható meg
 2. Nincs JOIN: oszlopcsaládok közt nem lehet sorokat összekapcsolni
 3. Nincs GROUP BY
 4. ORDER BY csak klaszterkulcson

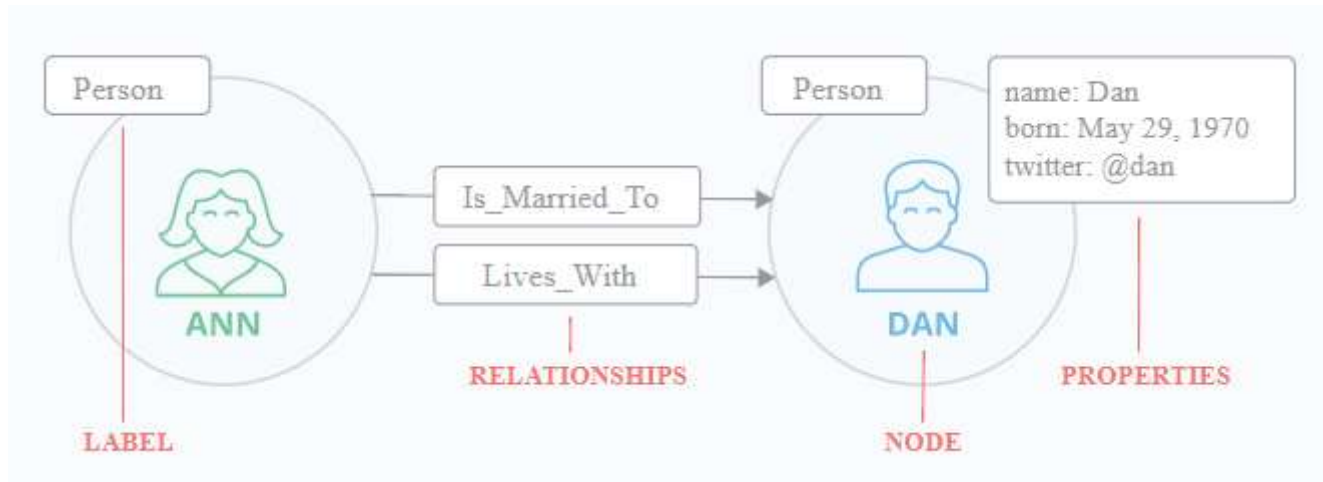
Cassandra fejlesztői támogatás

- Számos nyelvhez driver: Java, Python, Ruby, .NET, Node.js, PHP, C++, Scala, Clojure, Erlang, Go, Haskell, Rust, Perl

Neo4j

- „Labeled property graph”
 - > Csomópontok és élek
 - > Mindkettőn kulcs-érték párok
 - > Élek irányítottak
 - > Opcionális séma
- Cypher: intuitív lekérdezőnyelv
- Neo4j Browser: adatvizualizáció
- ACID működés
- Közvetlen, indexek nélkül tárolt kapcsolatok
- Elosztott, magas rendelkezésre állású
- NASA, Microsoft, IBM, Cisco, Ebay, Airbnb

Neo4j adatmodell



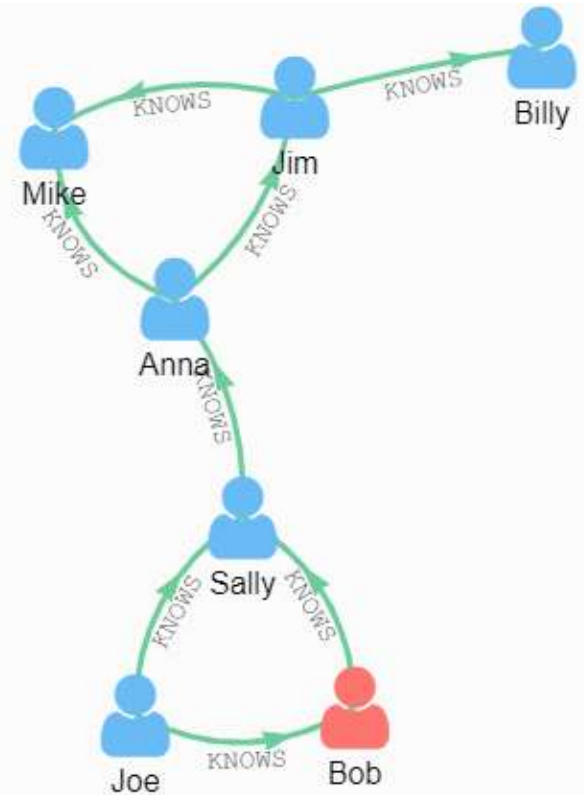
Cypher

Common Friends

Find friends in common between Joe and Sally

```
MATCH
  (user:Person)-[:KNOWS]-(friend)-
  [:KNOWS]-(foaf:Person)
WHERE
  user.name = "Joe" AND foaf.name = "Sally"
RETURN
  friend.name AS friend
```

Joe and Sally both know Bob.



<https://neo4j.com/developer/>

Neo4j fejlesztői támogatás

- Több nyelvhez van driver
 - > Java
 - > JavaScript
 - > Python
 - > .NET
- <https://neo4j.com/download/other-releases/#drivers>
- De van HTTP (REST) API is a többi platformhoz
 - > <http://neo4j.com/docs/developer-manual/current/http-api/>

Redis

- Memóriabeli kulcs-érték tár, opcionális durabilitással
- Replikáció
- Számos adattípus: string, list, map, set, sorted set, bitmap, térbeli koordináta
 - > Hyperloglog: halmaz elemszámát becsüli alacsony hibával
- Felhasználás: session, cache, üzenetsor, leaderboard
- Amazon, Microsoft

Redis fejlesztői támogatás

- Számos driver: Bash, C, C++, C#, Clojure, Lisp, Erlang, Go, Haskell, Java, Lua, Matlab, Node.js, Objective-C, Perl, PHP, PL/SQL, Python, Ruby, Scala, Swift (nem teljes a lista)
- <https://redis.io/clients>
- REST API: <http://webd.is/>

MongoDB demó

- Sample dataset:
<https://github.com/OpenKitten/Mongo-Assets/blob/master/primer-dataset.json>
- Mongo Shell (mongo.exe)
- <https://docs.mongodb.com/manual/mongo/index.html>

Cassandra demó

- <https://academy.datastax.com/planet-cassandra/cassandra>
- http://docs.datastax.com/en/archived/cql/3.0/cql/cql_using/use_set_t.html
- Most akkor flexibilis a séma vagy nem?
 - > <https://www.datastax.com/dev/blog/schema-in-cassandra-1-1>

Neo4j demó

- <https://neo4j.com/download/community-edition/>
- <http://neo4j.com/docs/cypher-refcard/current/>
- <http://localhost:7474/>
- Sample Datasets
 - > :play movie-graph
 - > :play northwind-graph