Μη σχεσιακές βάσεις δεδομένων : Η περίπτωση της MongoDB

Σαπουντζή Ιμπραήμ

Ανάγκη ανάπτυξης μη σχεσιακών βάσεων δεδομένων

Big Data

- **3Vs**(Gartner): **Volume**(TB/PB), **Velocity**(online συστήματα, κοινωνικά δίκτυα, αισθητήρες), **Variety**(ημι δομημένα και αδόμητα δεδομένα, - emails, documents, text, 3d, ήχος και εικόνα)

- Δύσκολο να αποθηκευτούν, επεξεργαστούν και αναλυθούν με τις παραδοσιακές εφαρμογές και τεχνικές (**RDBMS, data werehouses, data marts**)

- **Τεχνολογίες** Big Data:
  - Ανάγκη αποθήκευσης και διαχείρισης δεδομένων big data, **noSQL συστήματα**
  - Ανάγκη ανάλυσης δεδομένων με κατανεμημένους μηχανισμούς, εξόρυξη γνώσης, λήψη αποφάσεων, στατιστική, **MapReduce, Hadoop**

Χαρακτηριστικά μη σχεσιακών βάσεων δεδομένων nosql

- **schema-less/free αποθήκευση**
  - αποθήκευση χωρίς προκαθορισμένο σχήμα(!=RDBMS)
  - αποθήκευση με σχήματα τα οποία αλλάζουν δυναμικά(ALTER)

- **μοντέλο αποθήκευσης δεδομένων**
  - ποικιλία δομών δεδομένων και δυνατότητα αποθήκευσης ημι δομημένων και αδόμητων δεδομένων
  - δομές δεδομένων που ταιριάζουν καλύτερα στους προγραμματιστές και σε διαδικτυακές εφαρμογές (Web 2.0-
- **κατανεμημένα συστήματα (distributed)**
  - οριζόντια κλιμάκωση (προσθήκη κόμβων σε ένα cluster) αυτόματα
  - εκμετάλλευση των δυνατοτήτων που προσφέρει το cloud computing
  - RDBMS κάθετη κλιμάκωση
    - προσθήκη CPU,RAM, DISK σε έναν server = μονοτονική αρχιτεκτονική

- **Διαθεσιμότητα και αυτόματο failover**
  - αυτόματο replication χωρίς την προσθήκη ιδιαίτερου λογισμικού

- **Διαχείριση των δεδομένων μέσω αντικειμενοστραφών APIS**
  - **!SQL**

Είδη μη σχεσιακών βάσεων

- **Key - Value Stores**
  - key-value pairs - K1 : Value1

- **Document-oriented databases**
  - αποθήκευση documents βάση ενός key
  - υποστήριξη λιστών, πινάκων, δεικτών και εμφωλευμένων αντικειμένων
  - αντιπροσωπεύονται μέσω ενός μοναδικού κλειδιού
  - indexes, secondary indexes
  - JSON / BSON/ XML …
  -

- **Columnar Databases**
  - τα δεδομένα στους πίνακες αποθηκεύονται κατά στήλες, σε αντίθεση

    με τις σχεσιακές βάσεις δεδομένων όπου τα δεδομένα αποθηκεύονται κατά γραμμές
  - κατάλληλες σε συστήματα OLAP

- **Graph Databases**
  - χρησιμοποιούν την θεωρία των γράφων και αναπαριστούν τα δεδομένα τους υπό μορφή γράφων
  - σχέσεις μεταξύ των γράφων
  - SPARQL query language
  - κατάλληλα για κοινωνικά δίκτυα, μεταφορές, χάρτες πολύπλοκες σχέσεις μεταξύ χρηστών

ACID vs BASE

ACID

- **Atomicity** : commit or rallback
- **Consistency** : transactions never observe or cause inconsistent data - εξασφάλιση συνέπειας βάσης
- **Isolation** : transactions are not aware of concurrent transactions - δεν μπορούν να έχουν πρόσβαση σε δεδομένα που τροποποιούνται
- **Durability :** οι αλλαγές που πραγματοποίησε ένα transacation επιτυχημένο θα παραμείνουν και μετά την κατάρρευση του συστήματος

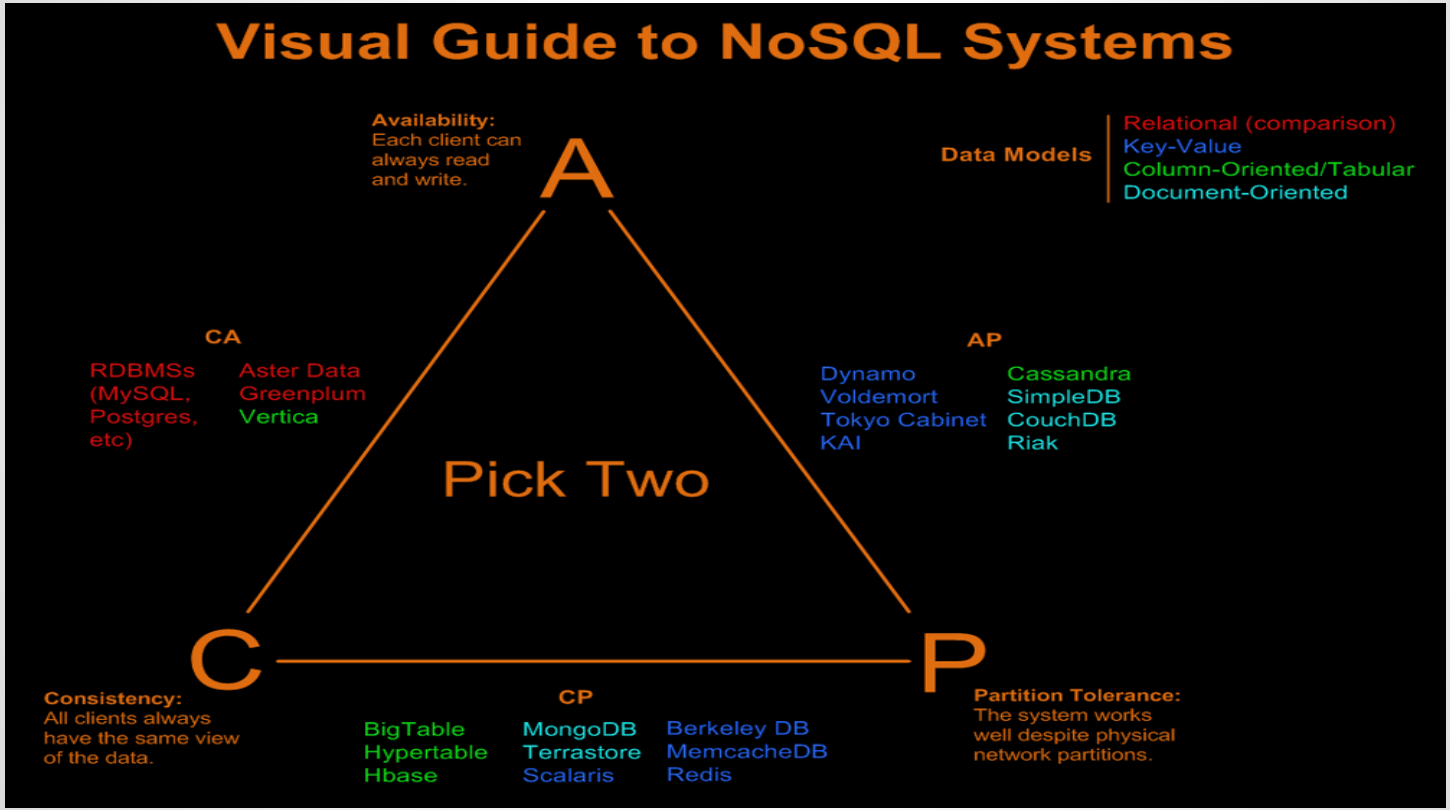
-Strong Consistency-

BASE

- **Basically Available :** η βάση είναι συνεχώς προσβάσιμη ακόμα και κάποιοι κόμβοι είναι μη διαθέσιμοι [replication]
- **Soft State :** μπορεί να υπάρξει ένα χρονικό διάστημα ασυνέπειας της βάσης, κρατώντας δεδομένα ενδεχόμενης συνέπειας [stale data]
- **Eventually Consistent** : ύστερα από ένα χρονικό διάστημα η βάση επανέρχεται σε γνωστή συνεπή μορφή

-Eventual Consistency-

Θεώρημα CAP


Visual Guide to NoSQL Systems
Availability:
Each client can always read and write.
A
Data Models
Relational (comparison)
Key-Value
Column-Oriented/Tabular
Document-Oriented
CA
RDBMSs (MySQL, Postgres, etc)
Aster Data
Greenplum
Vertica
Pick Two
AP
Dynamo
Voldemort
Tokyo Cabinet
KAI
Cassandra
SimpleDB
CouchDB
Riak
C
P
Consistency:
All clients always have the same view of the data.
CP
BigTable
Hypertable
Hbase
MongoDB
Terrastore
Scalaris
Berkeley DB
MemcacheDB
Redis
Partition Tolerance:
The system works well despite physical network partitions.

- **JSON-style documents (*BSON*)**
  - unique id(ObjectID)
  - embedded documents and objects [Array]
  - array of sub-documents
  - modeling data
    - embedded documents [maximun document size 16MB σε αυτήν την έκδοση (2.8)]
    - references
- **performance**
  - C++
  - full indexes, B-trees
  - memory mapped files για την διαχείριση των δεδομένων
  - no joins
  - no transactions
  - not relational
- **scalability**
  - replication + auto-sharding

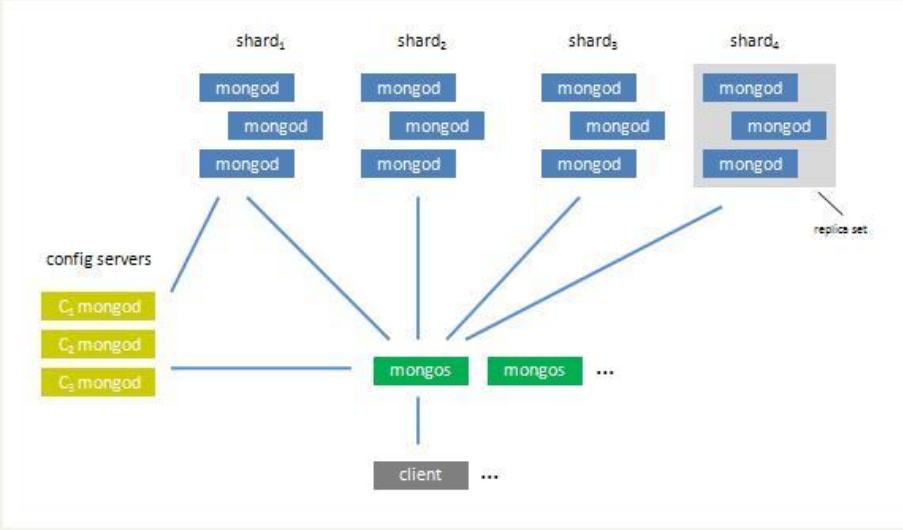
- Από προεπιλογή **CP**, μπορεί να ρυθμιστεί και **AP**[availability, stale data ok]
- **drivers** : C, C++, Java, JavaScript, perl, PHP,Python, Ruby, C#, Erlang, κ.α
- **open source**

Storage

- ***pre-allocation and padding* :** όταν ένα αρχείο φτάσει ένα ορισμένο μέγεθος, δεσμεύεται χώρος για το επόμενο χωρίς να ζητήσει η εφαρμογή την δημιουργία του. 1ο αρχείο 64MB, 128 ως 2g. Προσθήκη επιπλέον χώρου στο τέλος του document (**padding**) ώστε σε φόρτο εργασίας με πολλά **updates** να μην χρειαστεί η μεταφορά του σε άλλο σημείο στον δίσκο (!=Filesystem fragmantation)
  - *-Power of 2size allocation*
  - *-exact fit allocation*
- ***memmory mapped files*** *: map() τα αρχεία από τον δίσκο στην RAM, μέσω της virtual memory, κίνδυνος για page faults!*

- ***Journal Files*** : write-ahead-logging σε ένα αρχείο journal (durability)
  - commit στο journal κάθε 100millisec
  - γράφει τα δεδομένα στα αρχεία κάθε 60 sec
  - group updates τα writes στον δίσκο
  - σε περίπτωση κατάρρευσης του συστήματος χρησιμοποιείται το journal για να επανέλθουν οι τελευταίες αλλαγές στη βάση

- ***shard nodes*** : υπεύθυνοι για την αποθήκευση των δεδομένων, διαμερισμός των δεδομένων
- ***config servers*** : metadata και πληροφορίες δρομολόγησης που δείχνουν στους routers σε ποιο κόμβο είναι αποθηκευμένα τα δεδομένα, *two-phase-commit*
- ***query routers*** **:** επικοινωνία ενός η περισσότερων client με την βάση δεδομένων, στέλνουν τα requests στους αντίστοιχους κόμβους shard



Replication

- **μηχανισμός διατήρησης αντιγράφων των δεδομένων σε άλλου κόμβους**
  - εξασφαλίζεται η διαθεσιμότητα του συστήματος και το αυτόματο failover σε περίπτωση κατάρρευσης κόμβου

- ***replica sets* - 12 κόμβοι , 1 Primary - 11 Secondary**
  - Ο primary καταγράφει τις αλλαγές σε ένα oplog, και στην συνέχεια ο κάθε secondary ενημερώνεται ασύγχρονα
- **όλα τα writes και reads στέλνονται στον primary, strong consistency**
  - ρύθμιση να στέλνονται τα reads στους δευτερεύοντες, αύξηση απόδοσης αλλά eventually conistent data

- **αυτόματο failover**
  - κάθε κόμβος στέλνει hearbeats (pings) στους άλλους κόμβους
  - σε περίπτωση κατάρρευσης του primary, αυτόματα εκλέγεται ένας secondary
  - χρήσης atribers, κόμβους ψηφοφόρος χωρίς δεδομένα

Αυτο-Sharding

- **διαδικασία κατανομής των δεδομένων σε πολλούς κόμβους μέσα στο cluster**
  - επεκτασιμότητα της βάσης (scale-out)
  - όλοι οι κόμβοι shards μια λογική βάση δεδομένων
  - shard collection
  - αυτόματο sharding, σε περίπτωση νέων κόμβων ισοκατανομή του όγκου των δεδομένων και του φόρτου εργασίας

- **διαμερισμός των documents σύμφωνα με ένα shard key ορισμένο από τον χρήστη**
  - index ή compound index που πρέπει να υπάρχει σε κάθε document
  - διαχωρισμός των τιμών των documents σε chunks
    - range based partitioning π.χ order_id = 30;
    - has based partitioning π.χ timestamp

Ευχαριστώ πολύ,

συνέχεια στο blog