NoSQL Databases

Dr Charaka Palansuriya

Outline

- What is a NoSQL database?
- NoSQL databases
- Data Models
- MongoDB
- Accessing via applications
- Performance optimisation
- Archiving and backup
- Summary

What is a NoSQL database?

- "Not Only SQL"
- The term is used to refer to non-relational databases
- This include many other types of databases
  - XML or JSON document based databases
  - Graph databases
- Designed for distributed storage with high horizontal scalability
  - Suitable for large structured, semi-structured or unstructured data
  - Used for structured data since applications can store objects without using things like Object Relational Mapping (ORM)
- No schemas are required (a.k.a, schema-less)
  - Gives flexibility in storing documents with different content
- No join operations
- No transactions

Data Models

- Key-Value
- Document
- Column
- Graph

Key-Value Data Model

- Data is stored as key-value pairs
  - i.e.. Hashtable that is persistent in disks
- Key
  - Typically a simple number or string
  - Used to retrieve the value associated with it
- Value
  - Could be anything. E.g.,
    - Simple numbers or strings
    - Large structured/unstructured document
    - Pictures or Videos
- Amazon Dynamo data store is the pioneer in this category
  - Now Amazon DynomoDB

Document Data Model

- Collections of documents are stored
- Each document have a unique ID
- No schema to describe what the structure of the document should be
  - Gives flexibility is storing documents with different structure and content in a single collection
  - Often a document is retrieved as a single entity
    - Similar to the key-value data model
  - Or **search** documents containing certain data
- Popular choice is to store data as JSON documents
  - Each JSON document store set of key-values (can be nested)
  - XML is another alternative but not that popular

Document Data Model

- Example document from MongoDB:

```
{
        _id: "53f72bd87cf6efbdac437613”,
        name: “John Smith”,
        contact: {
                phone: “1234-4567”,
                email: “me@abc.com”
        },
        department:”Finance”,
        employeeID: 987
}
```

- MongoDB and CouchDB are document based NoSQL databases

Column Data Model

- Columns are treated individually
  - Allows processing as array
  - Column values are stored contiguously
    - i.e., in column specific data files
  - Faster processing of aggregated functions
    - SUM, AVG, COUNT, etc.
- Rows (i.e., tables) could be constructed from column values
- Examples
  - Google's BigTable
  - Amazon SimpleDB
  - Apache Cassandra

Graph data model

- Nodes represent entities
  - Have properties: E.g., ID, Name, Age, etc.
- Edges represent relationship
  - Have properties: E.g., ID, Knows, Member
- Each node and edge has a unique ID
- Each nodes knows adjacent nodes
- Example: Neo4J

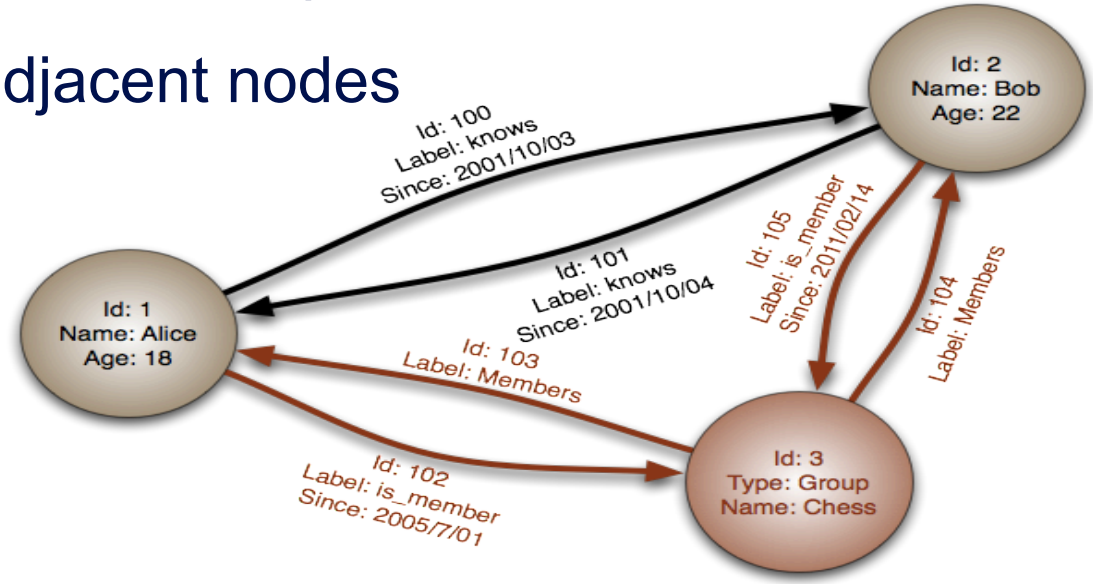


Image source: http://upload.wikimedia.org/wikipedia/commons/3/3a/GraphDatabase_PropertyGraph.png

Mongo DB

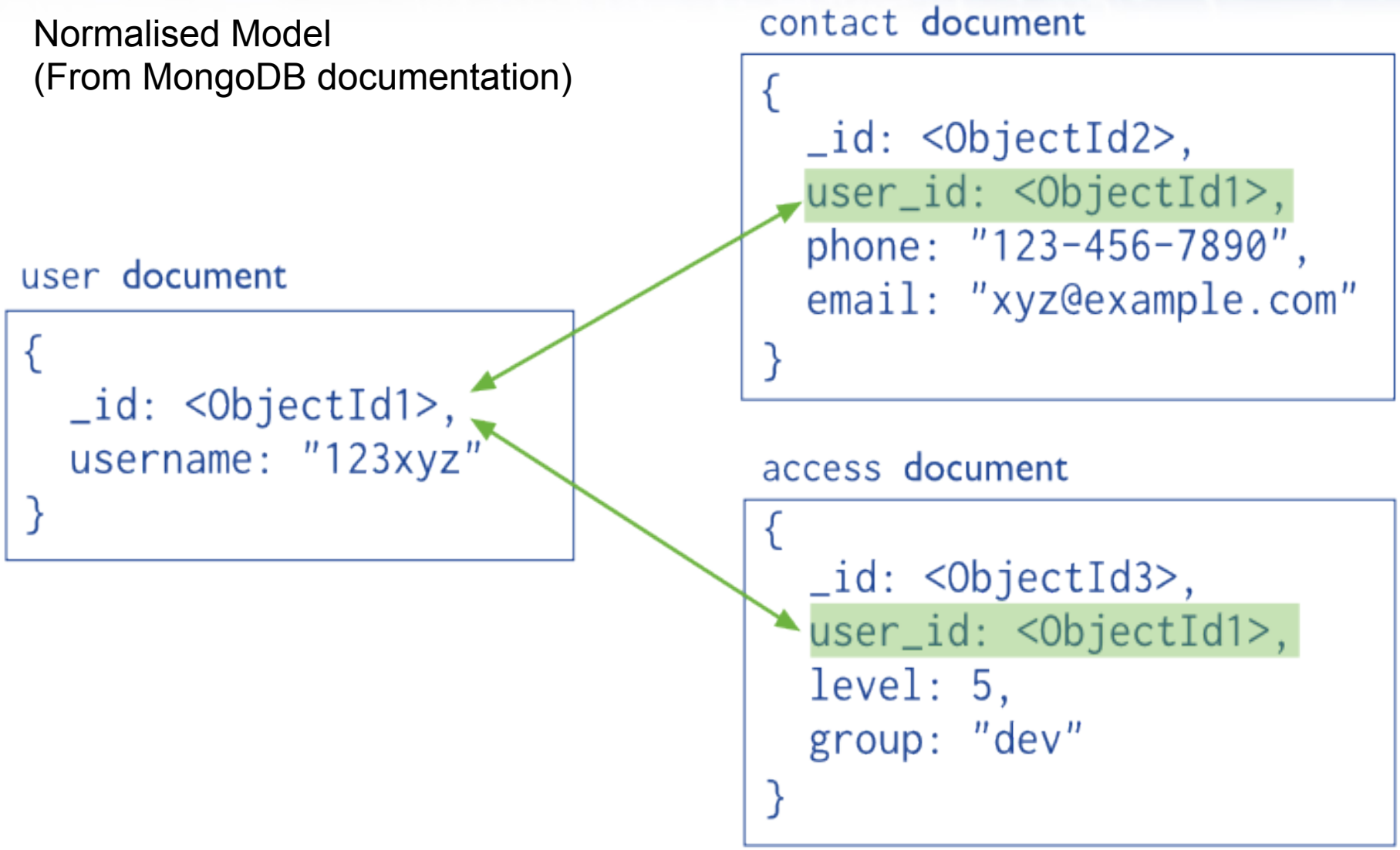
- Open Source JSON-**like** document based database management system
  - Stores document in BSON (Binary JSON) format
    - Binary encoded serialized JSON-like documents
    - Have types like Date that is not part of JSON
  - Officially supports many programming languages
    - C, C++, Java, Python, PHP, Perl and Ruby, C#
  - Written in C++
- High performance
  - Use TCP sockets for fast client server communication
    - Instead of things like REST
- High availability
  - Various data replication strategies can be applied

MongoDB: Documents

- Documents contains collections of key-value pairs
- Keys (i.e., field names) are strings
  - _id is reserved for the primary key
    - Uniquely identifies each document
    - If one is not provided then MongoDB creates one automatically
  - Field name cannot contain a dot (**.**) or **null** character
- BSON documents have a maximum size of 16MB
  - So storing string values and nested string values are OK
  - Even binary data like images as long as < 16MB
- Use GridFS for larger documents
  - Breaks a file into chucks and store each chuck as a separate document

MongoDB: Data Modelling

Normalised Model
(From MongoDB documentation)



Normalised Data Model

- Pros:
  - No duplication of data
- Cons:
  - Requires several request for an update
    - First get the references
    - Then get the documents refer to by the references
    - And then update those documents

MongoDB: Data Modelling - continued

De-normalised Model
(From MongoDB documentation)

```
{
  _id: <ObjectId1>,
  username: "123xyz",
  contact: {
             phone: "123-456-7890",
             email: "xyz@example.com"
           },
  access: {
            level: 5,
            group: "dev"
          }
}
```

Embedded sub document

Embedded sub document

De-normalised data model

- Pros:
  - Single document contains all interested data
  - Can be updated with single request
- Cons:
  - Data duplication can occur
  - If necessary, application would need to manage potential data duplication

MogoDB: Mongo Shell

- MongoDB too provide a Command Line Tool
  - Mongo Shell
  - Interactive JavaScript shell
- Allows the 4 basic CRUD operations
  - Create
    - Insert new JSON-like documents to a collection
  - Read
    - Retrieve existing documents in a collection
  - Update
    - Change existing documents in a collection
  - Delete
    - Removes documents from a collection

Examples of using Mongo Shell

- Create a new document in a collection

```
> db.staff.insert({name:"John Smith",email:"me@my.com",department:"Finance"})
```

- Read a document

```
db.staff.find({name:"John Smith"})
```

MongoDB: Applications

- Python is a popular language to write applications for MongoDB
  - Pymongo is the recommended way to interact with MongoDB using python
    - Provides a set of tools and API to interact with MongoDB
- A Java driver for MongoDB exist
  - http://docs.mongodb.org/ecosystem/drivers/java/
- Tools for other languages exists
  - C, C++, C#, Node.js, Perl, etc.
  - For full list see, http://docs.mongodb.org/ecosystem/drivers/

MongoDB indexing

- Works similar to indexing in Relational Database Management Systems (RDBMs)
  - Avoids scanning every document in a collection
  - The unique **_id** field in documents is indexed by default
- Works at the collections level
  - Can index a field or a sub-field of the documents in a collection
  - E.g., Consider “employees” documents collection with field “salary”
    - If the “salary” field is indexed then MongoDB can select which specific documents to search
      - db.employees.find({salary: { “lt”: 20000 } })
        - Narrows the search by scanning only the documents containing values of salary < 20000
- Use indexes to support most common queries
  - This will significantly improve performance of your queries

MongoDB: Data backups and restore

- Provides several backup solutions
  - A suitable solution based on your requirements and size of database
- Mongodump and mongorestore tools
  - Mongodump backs up the database to BSON files
  - Mongorestore re-create a database using the BSON files
  - Suitable for small database systems
- MongoDB Management Service (MMS)
  - A Cloud based backup service provided by MongoDB
    - https://mms.mongodb.com/?pk_campaign=MongoDB-Org&pk_kwd=Backup-Docs
  - Continuous backs up
    - Supports backing up and restoring snapshots
- Backup using OS tools
  - Copy MongoDB's uderlying data files

Summary

- Many types of NoSQL databases
- Suitable for scalable unstructured or semi-structured data
- Also used for structured data where applications can store objects without any transformations
  - Don’t have to use things like Object Relational Mapping (ORM)
- MongoDB is a popular choice for NoSQL document based databases

References

- Introduction to NoSQL, http://www.slideshare.net/dstainer/introduction-to-nosql-databases
- Amazon Dynamo, http://www.allthingsdistributed.com/files/amazon-dynamo-sosp2007.pdf
- MongoDB documentation, http://docs.mongodb.org/manual/