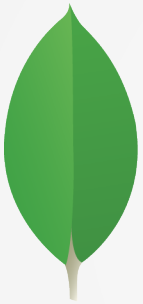


mongoDB®

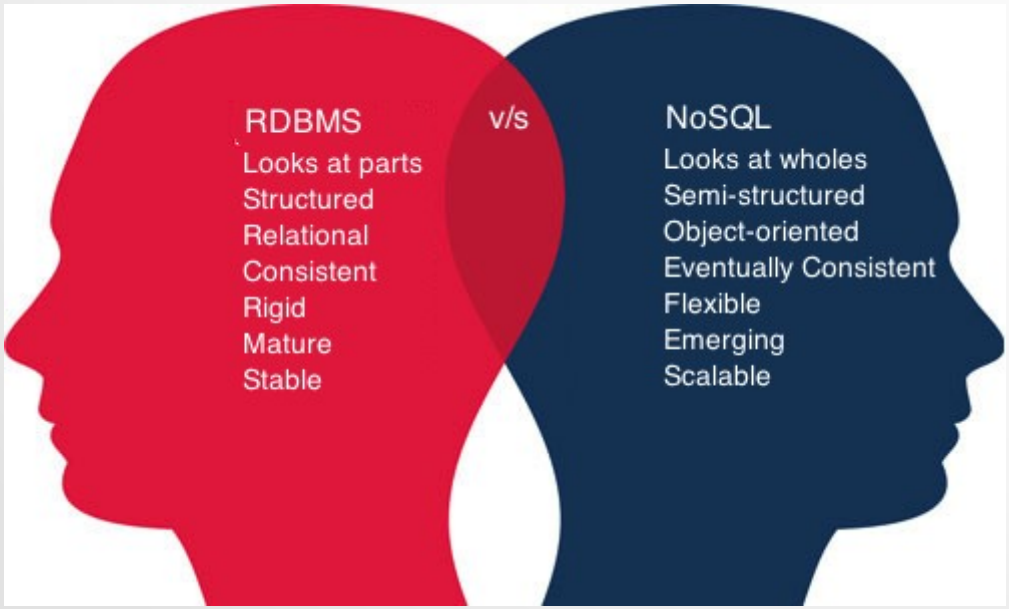
Bhakti Atul Pradhan

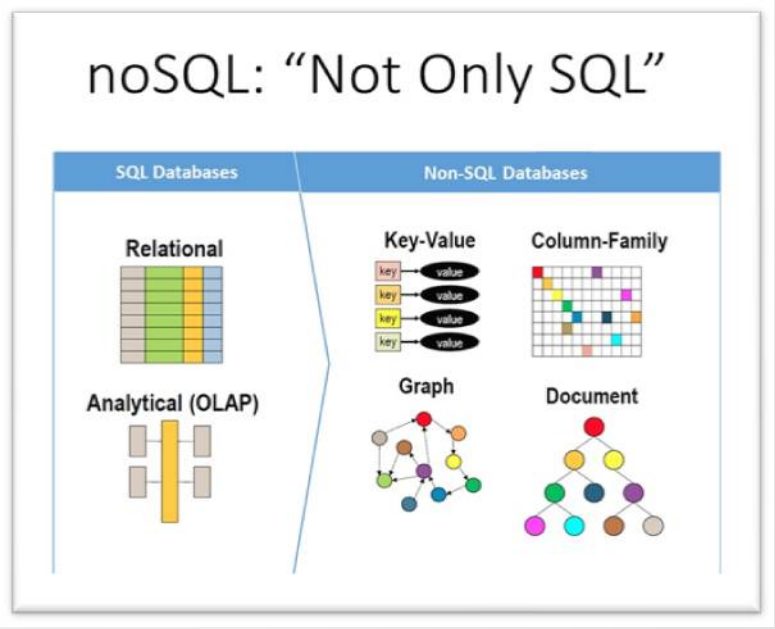
a drop of KNOWLEDGE

Agenda

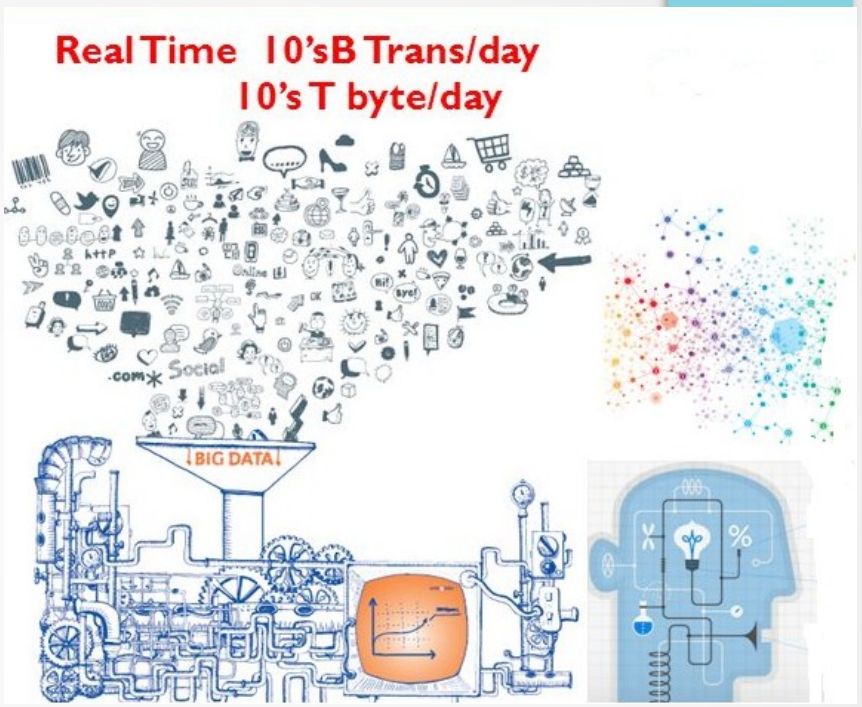
NO SQL (Not Only SQL)

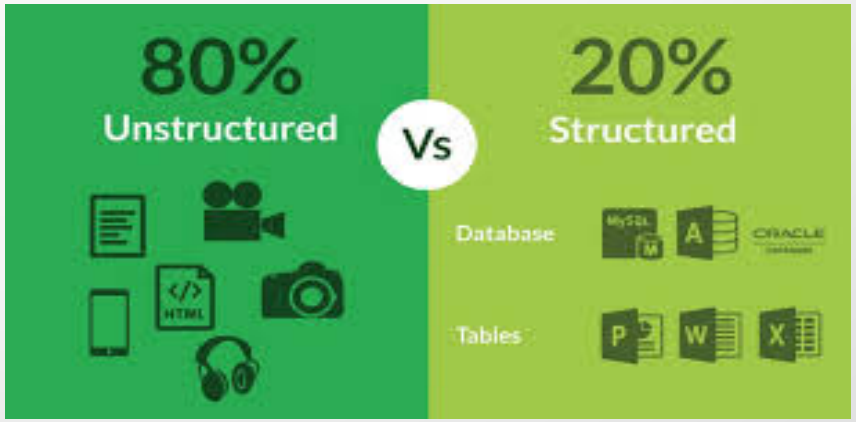
A NoSQL database provides a mechanism for storage and retrieval of data that is modeled in means other than the tabular relations used in relational databases.





Why NoSQL / MongoDB





What / Who MongoDB

- MongoDB – hu**mongo**us DB
- MongoDB is a free and open-source cross-platform document-oriented database that provides high performance, high availability, and automatic scaling.
- MongoDB uses JSON like documents with flexible schema.
- Developed by 10 gen in 2007, later renamed to MongoDB Inc in 2013.
- It is published under a combination of the GNU (General Public License) and the Apache License.

MongoDB Features

- Document Based
- Distributed
- High Performance
- Rich Query Language
- High Availability — Replication
- High Scalability – Sharding
- Support for Multiple Storage Engines
- Dynamic — No rigid schema.
- Flexible – field addition/deletion have less or no impact on the application
- Heterogeneous Data
- Data Representation in JSON or BSON
- Geospatial support
- Document-based query language that's nearly as powerful as SQL
- Easy Integration with common languages java, node etc and BigData Hadoop too.
- Cloud distributions such as AWS, Microsoft, RedHat,dotCloud and SoftLayer etc

When MongoDB?

- High performance (1000's – millions queries/sec)
- Need flexible schema
- Rich querying with any number of secondary indexes
- Need for replication across multiple data centers globally
- Need to deploy rapidly and scale.
- 99.99999% availability
- Real Time Analysis
- Geospatical Querying
- Processing in real time vs batch
- Agile Project
- Need Strong Data consistency
- Advance Security
- Building Next Gen Solution

Where MongoDB?

- Big Data
- Content Management and Delivery
- Social Infrastructure
- Etc.

- You Expect a High Write Load
- You need High Availability in an Unreliable Environment (Cloud and Real Life)
- You need to Grow Big (and Shard Your Data)
- Your Data is Location Based
- Your Data Set is Going to be Big (starting from 1GB) and Schema is Not Stable

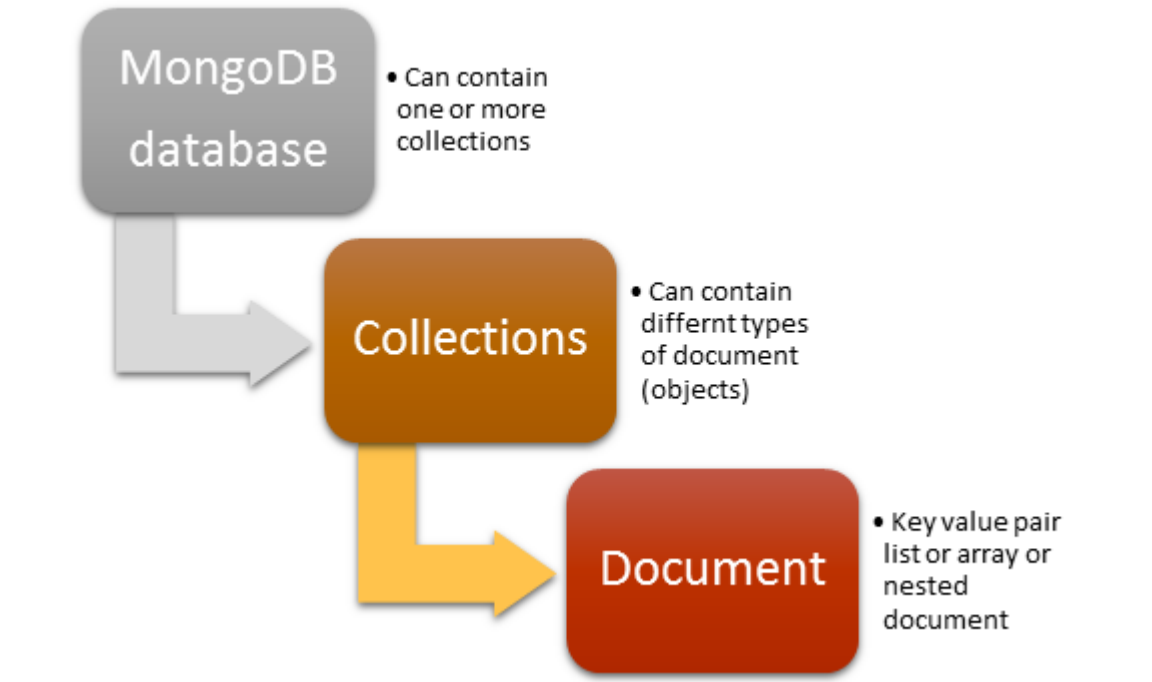
How MongoDB

- Getting Started
- Basics with Mongo Shell
- Mongo Compass
- MongoDB CRUD
- Data Modeling
- Indexes
- Aggregation Framework
- Sharding
- Replication

Getting Started

- Server
  - MongoDB Atlas is a cloud-hosted service for provisioning, running, monitoring, and maintaining MongoDB.
  - Local – Community/Enterprise
- Client
  - Mongo Shell - an interactive JavaScript interface to MongoDB
  - Mongo Compass - GUI for MongoDB

Basics



SQL	MongoDB
Table/View	Collection
Row/Tuple	Document
Column	Field
Primary key	_id field, ObjectId
Index	Index
View	View
Joins	$lookup, Embedded document
Uniformity Schema	Uniformity not Required
Foreign Key	Reference

Primary Key

- By default, each document contains an _id field.
- Value serves as primary key for collection.
- Value is unique, immutable, and may be any non-array type.
- Default data type is ObjectId, which is “small, likely unique, fast to generate, and ordered.” Sorting on an ObjectId value is roughly equivalent to sorting on creation time.
- This field is of 12 bytes
  - Date|Mac_Addr|PID|Counter
  - ----|---|--|---

CRUD

- Create
  - db.collection.insert(<document>)
  - db.collection.save(<document>)
  - db.collection.update(<query>, <update>, { upsert: true })
- Read
  - db.collection.find(<query>, <projection>)
  - db.collection.findOne(<query>, <projection>)
- Update
  - db.collection.update(<query>, <update>, <options>)
- Delete
  - db.collection.remove(<query>, <justOne>)

Schema Design

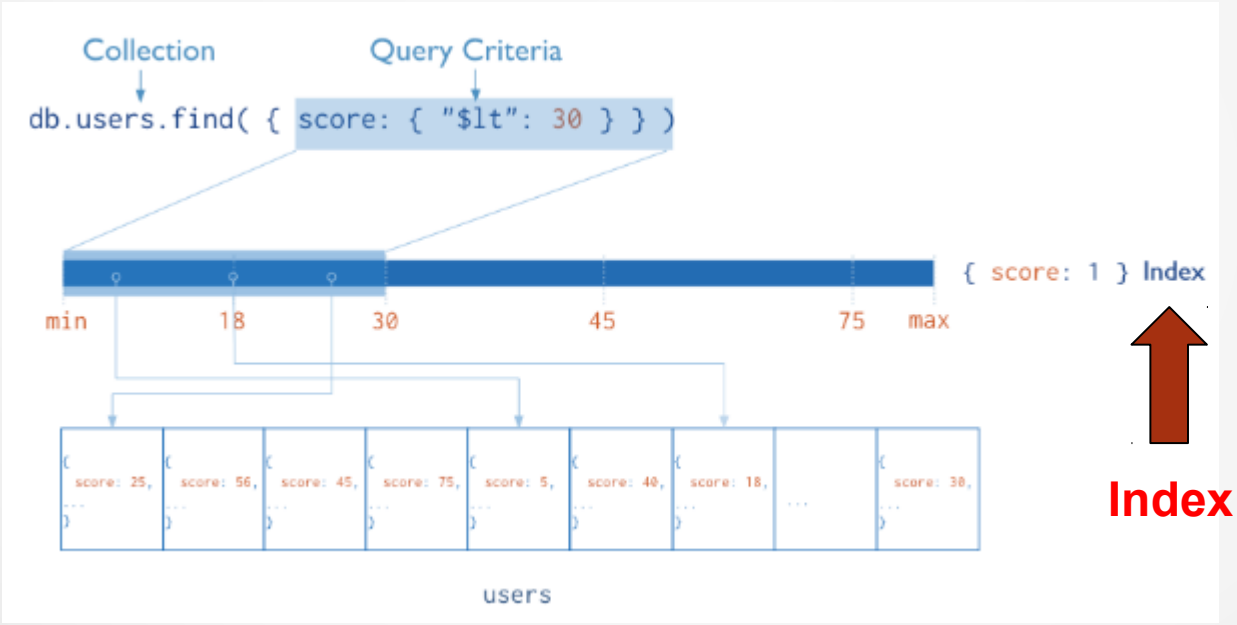
- 3NF vs Application Driven Schema
- Embedded or Not Embed (Linking)
  - Access same time by the application
  - Its existance is dependant on the parent existance
  - MongoDB cannot be more than 16MB, so if document gets greater than 16MB move to different collection

Schema Design

- One to One
  - Person, current address → Embed
- One to Many
  - City, Person → Link/Reference
- One to Few
  - Posts, Comments → Embed
- Many to Many
  - Books : Authors → Link
  - Students : Teachers → Link

Indexes

Indexes are special data structures that store a small portion of the collection's data set in an easy to traverse form.



Index

Indexes

- **Creation index**
  - db.users.ensureIndex({ score: 1 })
- **Show existing indexes**
  - db.users.getIndexes()
- **Drop index**
  - db.users.dropIndex({score: 1})
- **Explain—Explain**
  - db.users.find().explain()
  - Returns a document that describes the process and indexes

Questions

Thank You