Data Engineering LTAT.02.007

Prof. Ahmed Awad, Ass Prof. Riccardo Tommasini
Mohamed Ragab, Kristo

Document-oriented Databases (MongoDB)

Agenda

- Why document-based?
  - Why the RDB is not enough?
- What is Document?
- What is MongoDB?
  - SQL VS. MongoDB Concepts
  - MongoDB Data Model
  - How can we access MongoDB?
  - MongoDB CRUD operations.
    - CREATE, READ, UPDATE, DELETE

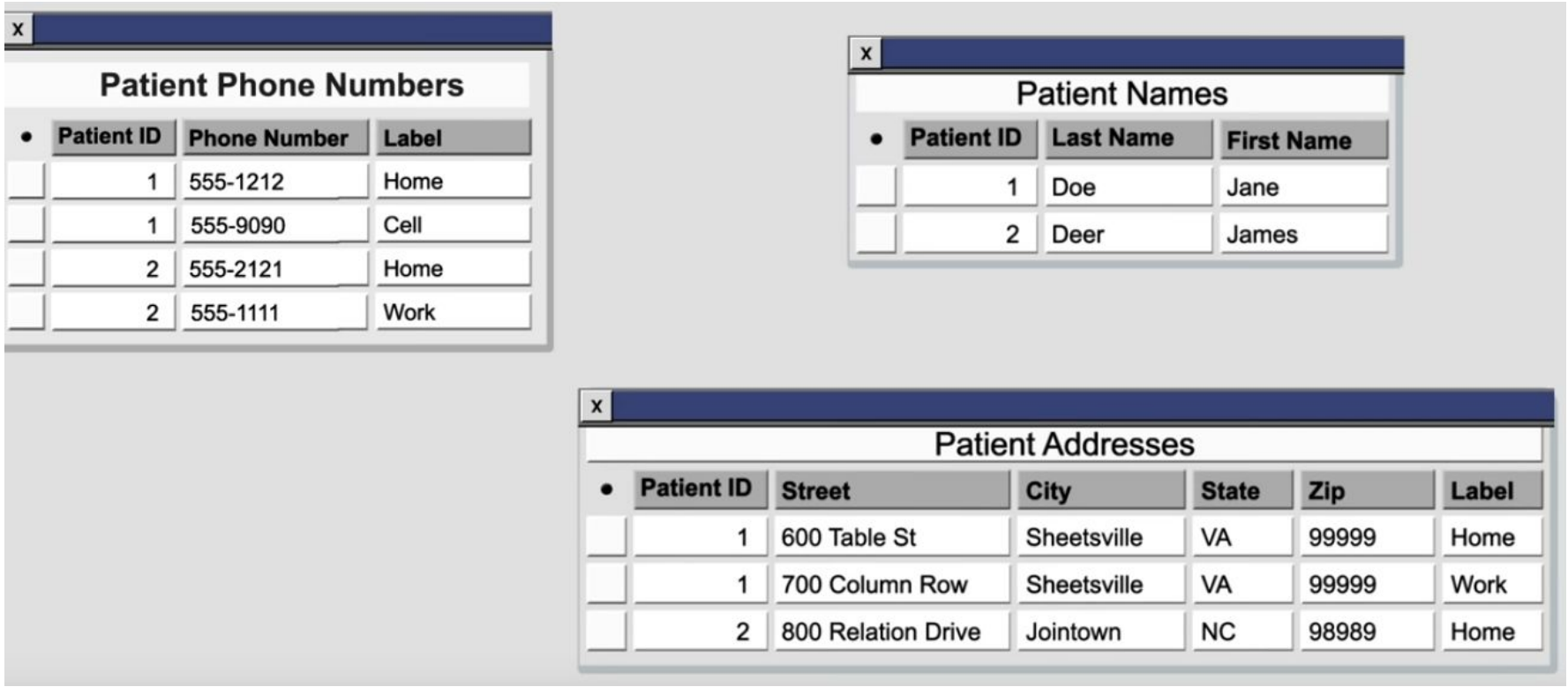
Why the RDB is not enough?

Patient Contact Information

Patient ID	Last Name	First Name	Personal Email	Work Email	Home Phone	Work Phone	Cell Phone	Emergency Contact Phone	Home Street	Home City	Home State	Home Zip	Work Street	Work City	Work State	Work Zip
1	Doe	Jane	jdoe@aol.com		555-1212		555-9090		600 Table St	Sheetsville	VA	99999	700 Column Row	Sheetsville	VA	99999
2	Deer	James	jdeer@mindspring.net		555-2121	555-1111			800 Relation Drive	Jointown	NC	98989				
3	Apryl	Lenser	jorgb@sbcglobal.net		202-555-0116		202-555-0193		676 George Street	Sheboygan	WI	53081				
4	Delmy	Trammel	johndo@icloud.com		202-555-0107	202-555-0107	202-555-0195		69 Coffee Dr.	Ronkonkoma	NY	11779				
5	Sharolyn	Spence	carmena@comcast.net	jorgb@sbcglobal.net	202-555-0165			202-555-0183	8778 Spruce Ave.	Stratford	CT	6614				
6	Cassondra	Yeats	helger@mac.com		202-555-0109		202-555-0124		463 SmokyHollow St	Carmel	NY	10512				
7	Margo	Varney	amichalo@mac.com		202-555-0192		202-555-0188		42 S. Highland Lane	Hartselle	AL	35640				
8	Mira	Pfaff	rasca@yahoo.com		202-555-0198		202-555-0168		492 Pearl Street	Saint Charles	IL	60174				
9	Raymon	Ryer	ghost@gmail.com		202-555-0150	202-555-0150			21 St Louis Street	Clarksville	TN	37040	3 SE. Logan St.	Arvada	CO	80003
10	Kimbra	Gravel	parrt@aol.com		202-555-0142				7576 Homewood St	Davison	MI	48423				
11	Sumiko	Cullinan	padme@yahoo.ca		202-555-0154		202-555-0170		31 Nichols Court	Nanuet	NY	10954				
12	Boris	Ellzey	schwaang@verizon.net	johndo@icloud.com	202-555-0182				8224 Eagle Drive	Rome	NY	13440				
13	Dagmar	Morano	dougjj@yahoo.com		202-555-0126				3 Grove Dr.	Saint Cloud	MN	56301				
14	Trista	Knuckles	anicolao@verizon.net		614-555-0184		614-555-0188		370 Big Rock Cove	Olive Branch	MS	38654				
15	Lenny	Walcott	harpes@optionline.net		614-555-0119				2 Philmont Avenue	Lapeer	MI	48446	560 W Hudson St.	Olympia	WA	98512
16	Jodie	Manion	mpiotr@comcast.net		614-555-0125			614-555-0519	960 Sycamore St.	Enfield	CT	6082				
17	Erlinda	Eisenmenger	twoflower@msn.com		614-555-0118	614-555-0118			77 Hillcrest Court	Eden Prairie	MN	55347				
18	Jaquelyn	Daffron	mwilson@msn.com		614-555-0196		614-555-0128		94 Charles Rd.	Williamstown	NJ	8094				
19	Barbie	Brandis	carreras@outlook.com		614-555-0187				11 Grove Drive	Sioux Falls	SD	57103				
20	Apryl	Lenser	jorgb@sbcglobal.net	apryl@comcast.net	202-555-0116				676 George Street	Sheboygan	WI	53081				

https://www.youtube.com/watch?v=EE8ZTQxa0AM

Why the RDB is not enough?



Patient Phone Numbers

Patient ID	Phone Number	Label
1	555-1212	Home
1	555-9090	Cell
2	555-2121	Home
2	555-1111	Work

Patient Names

Patient ID	Last Name	First Name
1	Doe	Jane
2	Deer	James

Patient Addresses

Patient ID	Street	City	State	Zip	Label
1	600 Table St	Sheetsville	VA	99999	Home
1	700 Column Row	Sheetsville	VA	99999	Work
2	800 Relation Drive	Jointown	NC	98989	Home

Why the RDB is not enough?

Patient Phone Numbers

Patient Names

Patient Meds

Patient ID	Date Prescribed	Valid Through	Medication Name	Dosage	Notes	Physician ID
1	1/15/2017	2/15/2017	Hyberdoze	15mg 2x daily		245-23415-13
1	9/1/2015	11/15/2015	Comanapracil	6mg 1x daily	take with food	245-23415-13
1	8/23/2016	9/14/2016	Vaxadrin	250mg 4x		832-82356-89

Patient Visits

Patient ID	Date	Physician ID	Reason for Visit	Notes
1	4/19/2014	245-23415-13	mysterious tingling	patient had a battery on their tongue
2	4/2/16	832-82356-89	numbness in third arm	no third arm actually present
3	11/9	456-83223-11	double vision	sounds fun!

Label
Home
Work
Home

Why this has Drawbacks?

- Hard to Understand
- Hard to add features
- Inefficient
  - Pulling Data From so many Places (more joins)

Why document-based?

Patient ID: 1

Last Name: Doe

First Name: Jane

Phone Numbers:

Label: Home
Number: 555-1212

Label: Cell
Number: 555-9090

label: Cell
number: 555-1111

Addresses:

label: Home
street: 600 Table St.
city: Sheetsville
state: VA
zip: 99999

label: Work
street: 300 Column Row
city: Sheetsville
state: VA
zip: 99999

Patient ID: 2

Last Name: Deer

First Name: John

Phone Numbers:

label: Home
number: 555-2121

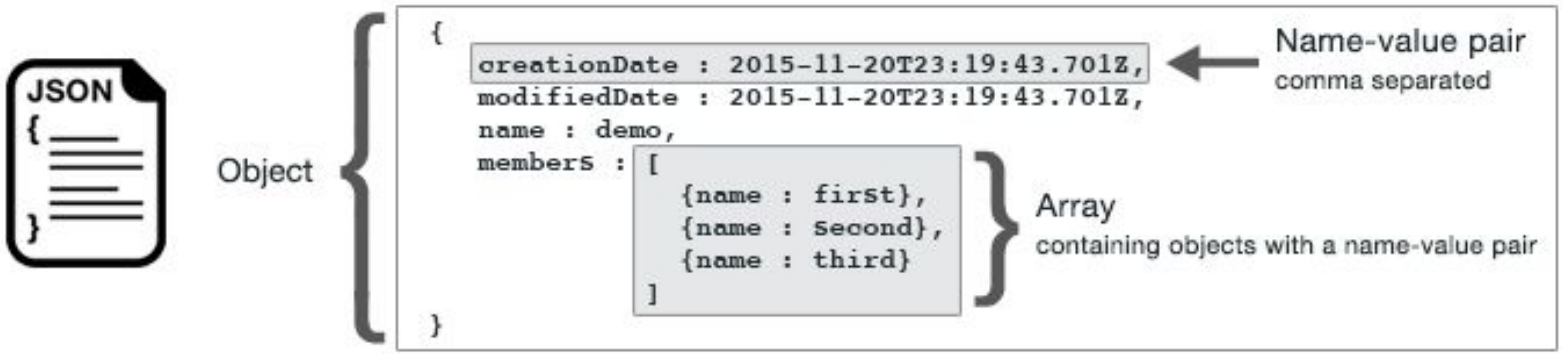
Addresses:

label: Home
street: 800 Relation Drive
city: Jointown
state: NC
zip: 99999

Prescriptions:

What is a document?

- JSON documents consist of fields, which are name-value pair objects.
- The fields can be in any order, and be nested or arranged in arrays.



https://aws.amazon.com/documentdb/what-is-json/

What is MongoDB?

Humongous

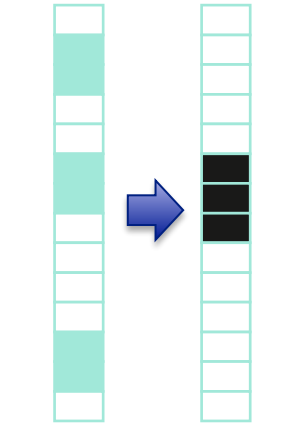
Because it can store lots and lots of data!

Overview – MongoDB

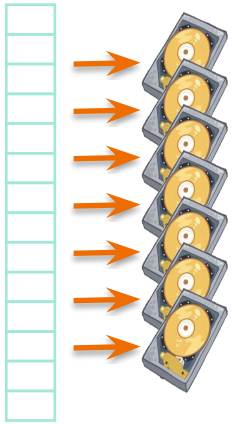
- **Open-source**
- **Document-oriented** database.
- Data is stored in **JSON-like** documents.
- Designed with both **scalability** and developer agility.
- Dynamic schemas.
- Automatic data sharding.

MongoDB is fast and scalable

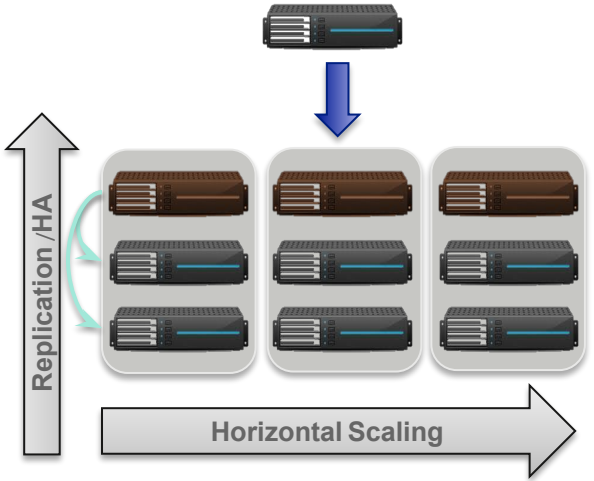
Better data locality



In-Memory Caching



Distributed Architecture



MongoDB- Facts #1

- No Schemas
- No transactions
- No joins
- Max document size of 16MB
  - Larger documents handled with **GridFS**

GoodtoKnow

https://docs.mongodb.com/manual/core/gridfs

MongoDB- Facts #2

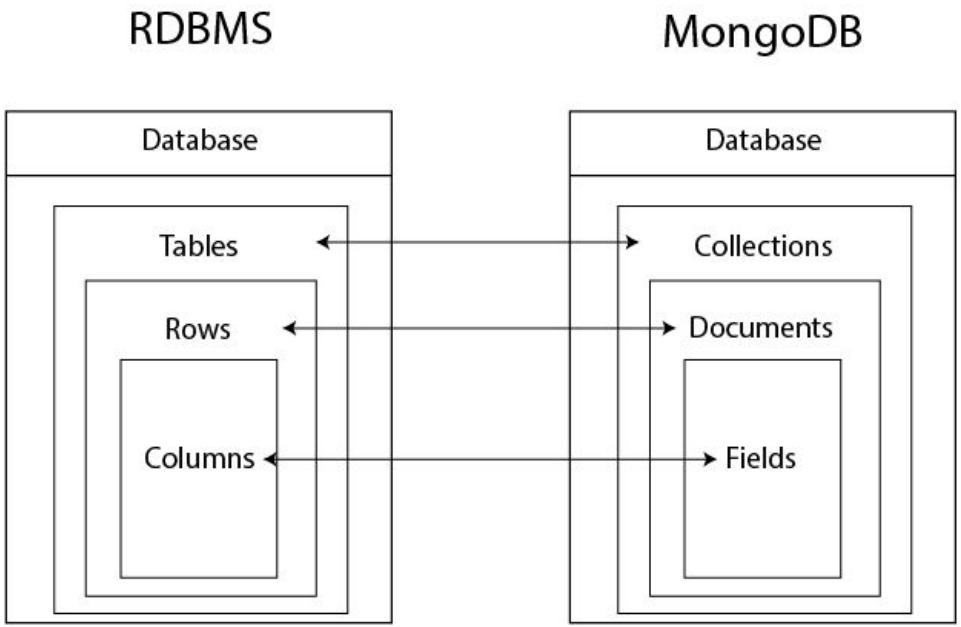
- Runs on most common OSs
  - Windows
  - Linux
  - Mac
  - Solaris
- Data stored as BSON (Binary JSON)
- used for speed
- translation handled by language drivers

GoodtoKnow

More Facts about MongoDB

- No Just NoSQL, very flexible document model.
- Shell is a full **JavaScript interpreter**.
- Support **many indices**
  - But **only one** can be used for **sharding**.
  - More than **2-3** are still discouraged
  - **Full-text indices** for text searches, **spatial indices**.
- A **SQL** connector is available
  - But bare in mind that **it is not relational**, not designed for joins and normalized data.

Terminology: SQL vs MongoDB

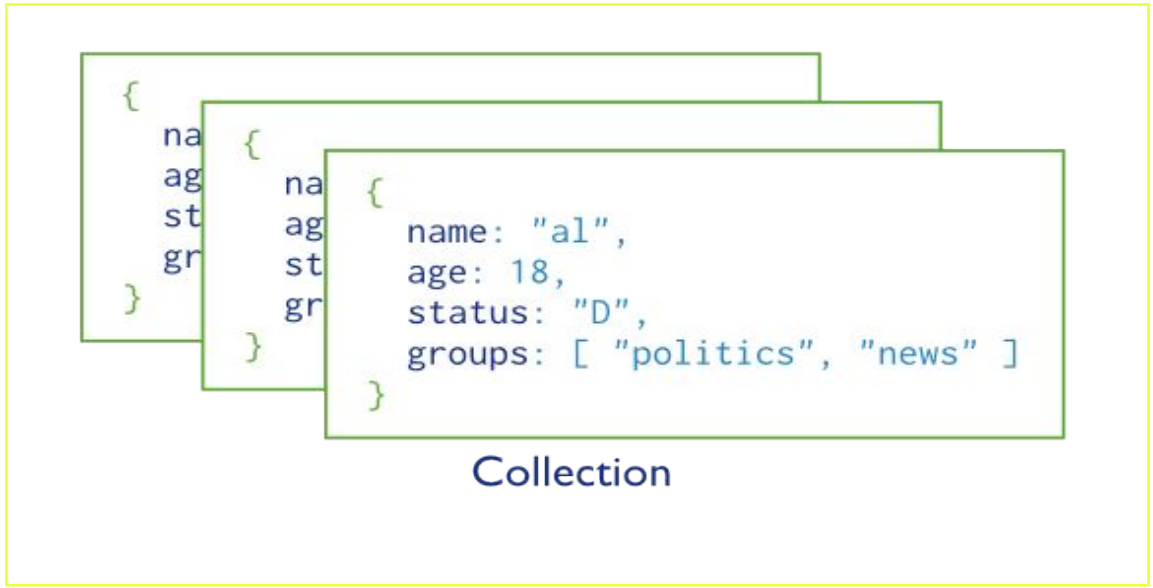


BSON Format

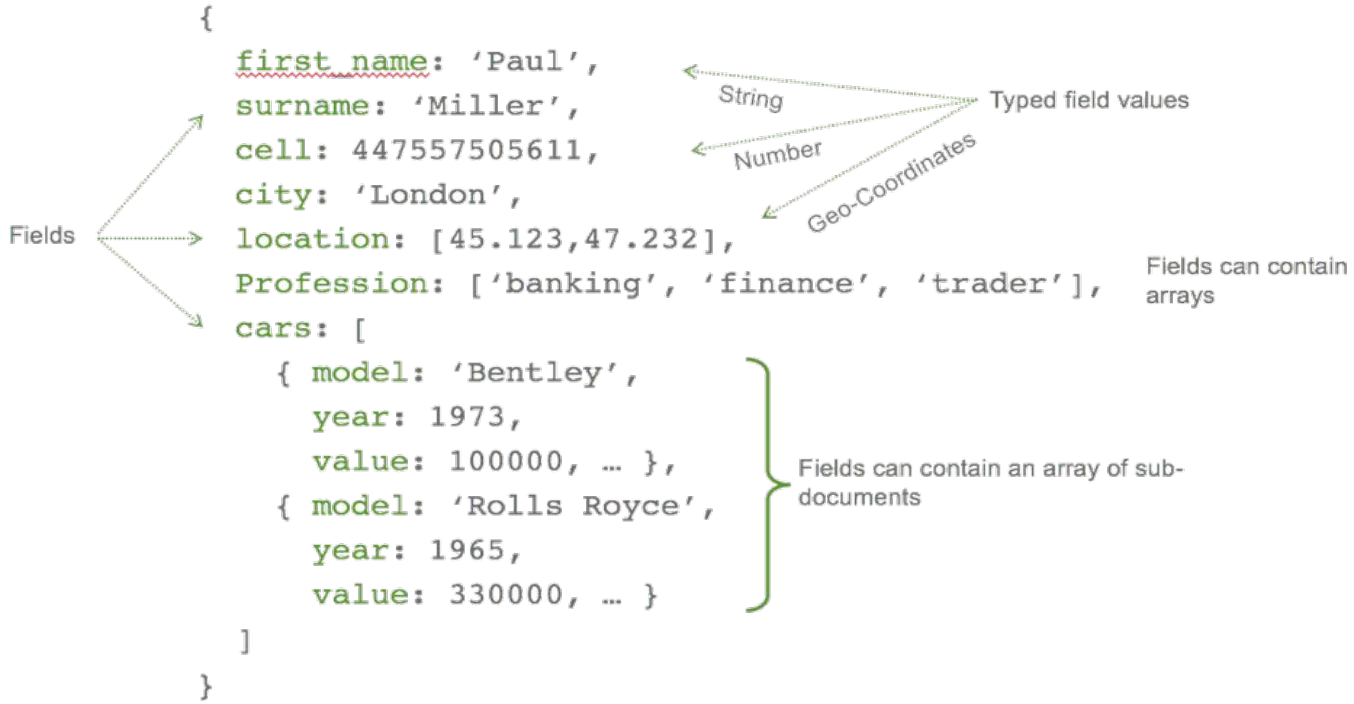
	JSON	BSON
Encoding	UTF-8 String	Binary
Data Support	String, Boolean, Number, Array	String, Boolean, Number (Integer, Float, Long, Decimal128...), Array, Date, Raw Binary
Readability	Human and Machine	Machine Only

MongoDB Data Model

- A database is the container for collections.
- A collection in MongoDB is a container for documents.



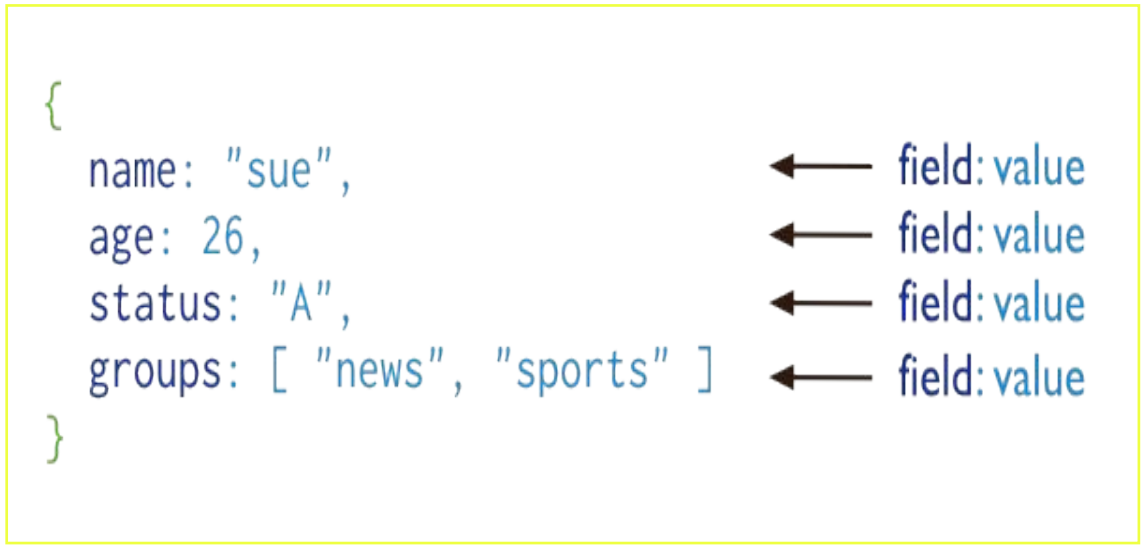
An Example of JSON



https://www.mongodb.com/blog/post/getting-started-with-python-and-mongodb

MongoDB Data Model

Structure of a JSON-document:



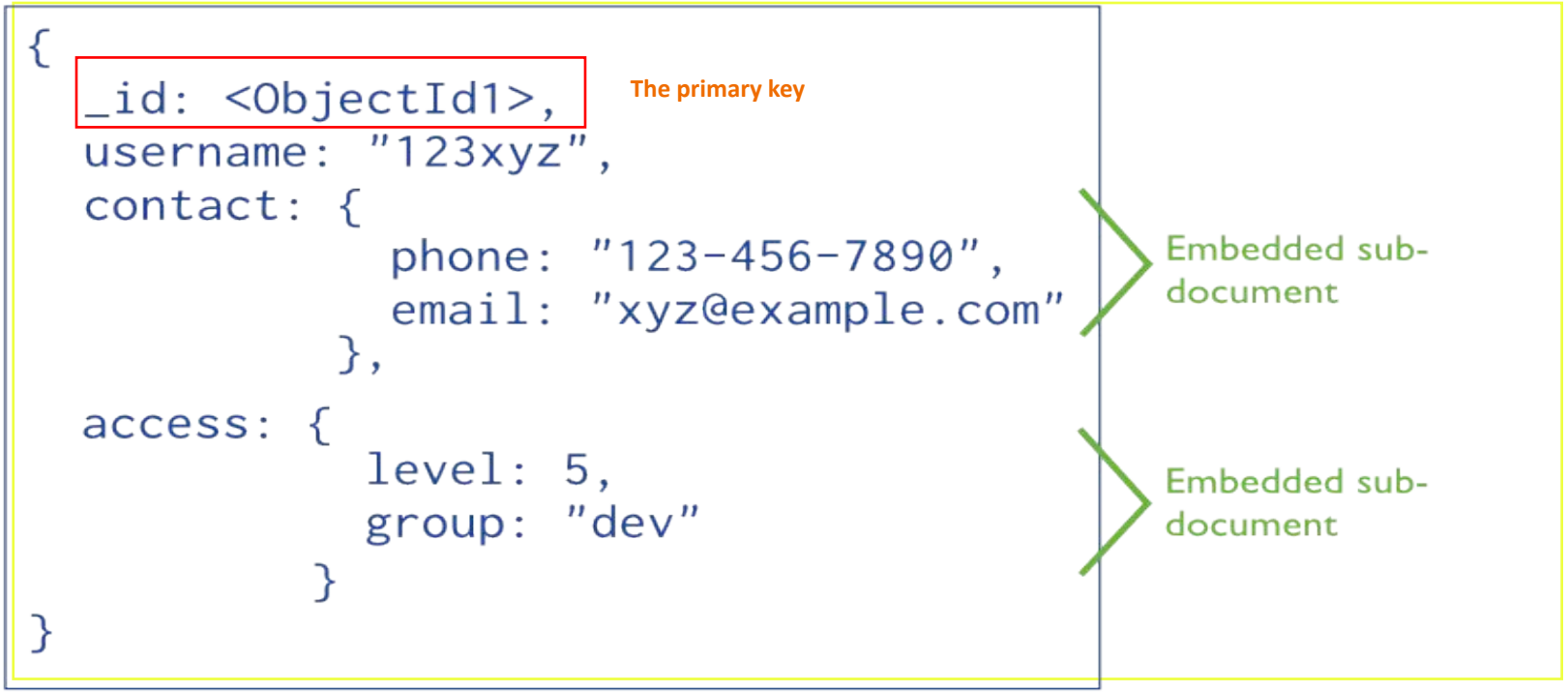
The value of **field**:

- Native data types
- Arrays
- Other documents

Rule: Every document must have an _id.

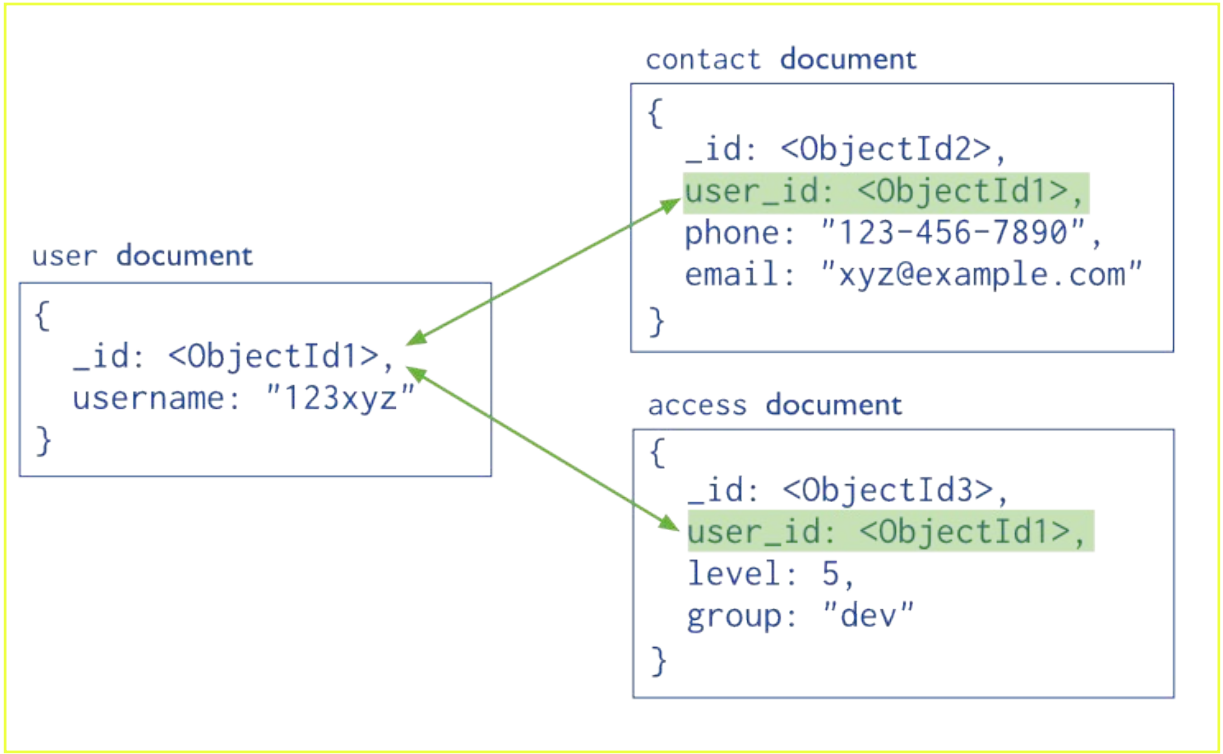
MongoDB Data Model

Embedded documents:



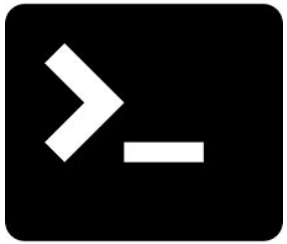
MongoDB Data Model

Reference documents or linking documents



How Can We Access MongoDB

- Shell/ terminal
  - local installation
- MongoDB Compass
  - GUI for MongoDB.
- Using the Mongo Atlas
  - MongoDB in the Cloud.
- Third-Party GUI tools
  - Robo-mongo
- using Applications
  - Python, Scala,...



MongoDB Queries: Create

- CRUD (**Create** – Read – Update – Delete)
  - Create a database: use database_name
  - Create a collection:
    - db.createCollection(name, options)
      - options: specify the number of documents in a collection etc.
  - Insert a document:
    - db.<collection_name>.insert({“name”: “nguyen”, “age”: 24, “gender”: “male”})

MongoDB Queries: Read

- CRUD (Create – **Read** – Update – Delete)
  - Query [e.g. select all]
    - db.<collection_name>.find().pretty()
  - Query with conditions
    - db.<collection_name>.find(
      { “gender”: “female”, “age”: {$lte:20} }).pretty()
    - It’s pattern matching again!

Read – mapping to SQL

SQL Statement	MongoDB commands
SELECT * FROM table	db.collection.find()
SELECT * FROM table WHERE artist = ‘Nirvana’	db.collection.find({Artist:”Nirvana”})
SELECT* FROM table ORDER BY Title	db.collection.find().sort(Title:1)
DISTINCT	.distinct()
GROUP BY	.group()
>=, <	$gte, $lt

Comparison Operators

Name	Description
$eq	Matches value that are equal to a specified value
$gt, $gte	Matches values that are greater than (or equal to a specified value
$lt, $lte	Matches values less than or (equal to) a specified value
$ne	Matches values that are not equal to a specified value
$in	Matches any of the values specified in an array
$nin	Matches none of the values specified in an array
$or	Joins query clauses with a logical OR returns all
$and	Join query clauses with a loginal AND
$not	Inverts the effect of a query expression
$nor	Join query clauses with a logical NOR
$exists	Matches documents that have a specified field

https://docs.mongodb.org/manual/reference/operator/query /

Further Read Features: Aggregates

- SQL-like aggregation functionality
- Pipeline documents from a collection pass through an aggregation pipeline
- Expressions produce output documents based on calculations performed on input documents
- Example:
  - db.parts.aggregate ({$group : {_id: type, totalquantity : { $sum: quanity} } })

MongoDB Queries: Update

- CRUD (Create – Read – **Update** – Delete)
  - db.<collection_name>.update(<select_criteria>,<updated_data>)
  - db.students.update({'name':'nguyen'}, { $set:{'age': 20 } })
  - Replace the existing document with new one: save method:
    - db.students.save({_id:ObjectId('string_id'),

      "name": "ben", "age": 23, "gender": "male"}

MongoDB Queries: Delete

- CRUD (Create – Read – Update – **Delete**)
  - Drop a database
    - Show database: show dbs
    - Use a database: use <db_name>
    - Drop it: db.dropDatabase()
  - Drop a collection:
    - db.<collection_name>.drop()
  - Delete a document:
    - db.<collection_name>.remove({“gender”: “male” })

Now, It's time to say…