

Unit 12: MongoDB.

2021/2022

Free courses!!

- [M001: MongoDB Basics](#)
- [M042: New Features and Tools in MongoDB 4.2](#)
- [M103: Basic Cluster Administration](#)
- [M121: The MongoDB Aggregation Framework](#)
- [M201: MongoDB Performance](#)
- [M220J: MongoDB for Java Developers](#)
- [M220JS: MongoDB for Javascript Developers](#)
- And many more:

<https://university.mongodb.com/courses/catalog>

Contents

12.1. Introduction.

12.2. Document and CRUD basics.

12.3. Data schemas and data modelling.

12.4. More about creating and importing documents.

12.5. More about reading data.

12.6. More about updating data.

12.7. More about deleting data.

12.8. Indexes.

12.9. Aggregation framework.

12.10. Working with numeric data.

12.11. Security and user authentication.

12.12. Performance, fault tolerance and deployment.

12.13. Transactions.

12.14. Geospatial data.

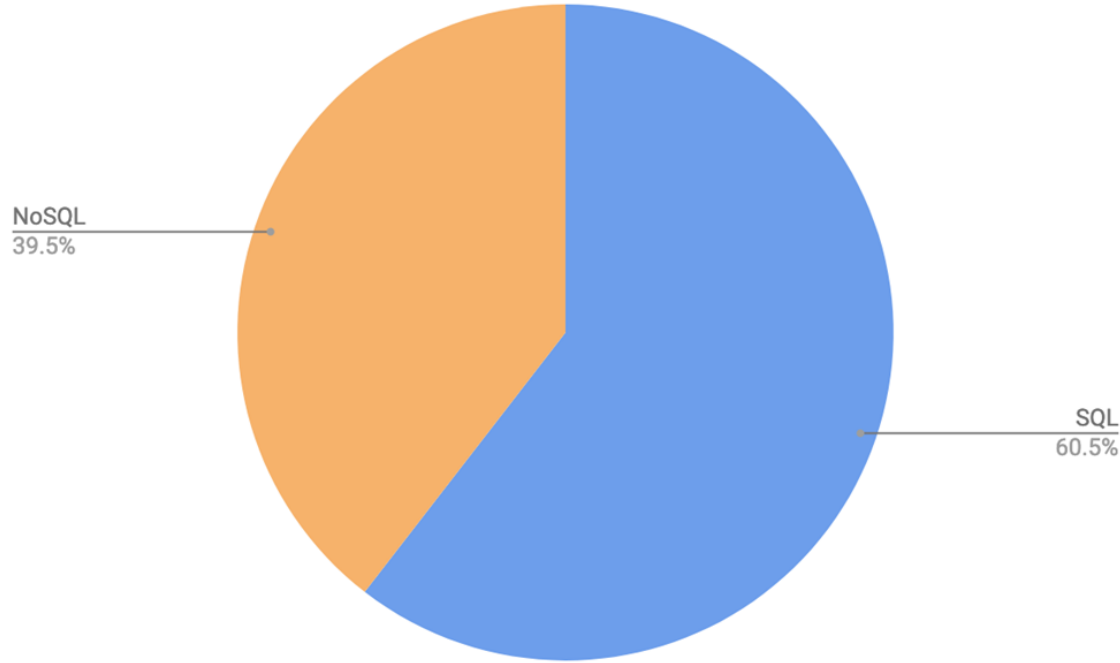
12.1. Introduction.

12.1.1. What is MongoDB?

Source: Wikipedia

- **MongoDB is a** cross-platform **document-oriented database management system**. In a nutshell: **it is a database solution**.
- English word: [huMONGOus](#).
- **Classified as a NoSQL DBMS, MongoDB uses JSON-like documents.**
- MongoDB is developed by MongoDB Inc. and licensed under the Server Side Public License (SSPL).
- Website: <https://www.mongodb.com/>

12.1.1. What is MongoDB?



2019 Database Trends – SQL vs. NoSQL, Top Databases, Single vs. Multiple Database Use

12.1.1. What is MongoDB?

Document

A way to organize
and store data as a set
of field-value pairs

```
{  
  <field> : <value>,  
  <field> : <value>,  
  "name"  : "Lakshmi",  
  "title" : "Team Lead",  
  "age"   : 26  
}
```

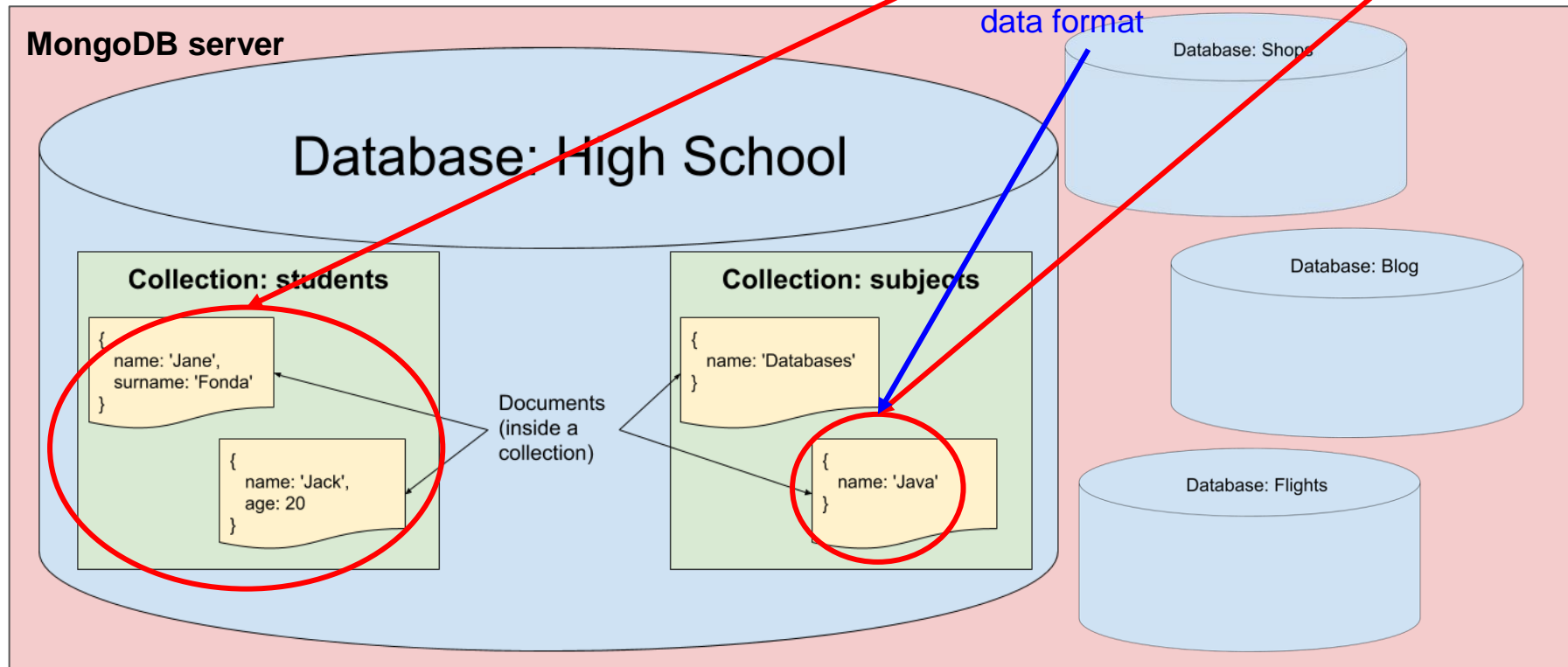
12.1.1. What is MongoDB?

Collection

An organized store of documents in MongoDB, usually with common fields between documents

```
{  
  "name" : "Lakshmi".  
  {  
    "name" : "Pavi".  
    {  
      "name" : "Lenny",  
      "title" : "Engineer",  
      "age" : 25  
    }  
  }  
}
```


12.1.1. What is MongoDB?



12.1.1. What is MongoDB?

JavaScript Object Notation

- JSON (BSON) data format:

JSON is converted internally to a Binary version of JSON

Nested data (way to create relationships between data)

Embedded documents inside an array

Field = key + value

```
{
  Key: "name": Value: "Sergi",
  "surname": "González",
  "address": {
    "street": "Caracas",
    "city": "Palma"
  },
  "interests": [
    { "name": "Databases" },
    { "name": "Javascript" }
  ]
}
```

The diagram illustrates a MongoDB document structure. A blue box highlights the first field, "name": "Sergi", with labels "Key:" and "Value:". A red box highlights the "address" field, which is a nested document containing "street": "Caracas" and "city": "Palma". A teal box highlights the "interests" field, which is an array containing two embedded documents: {"name": "Databases"} and {"name": "Javascript"}. Arrows point from descriptive text blocks to these specific parts of the document: a pink arrow from "JSON is converted internally to a Binary version of JSON" points to the document as a whole; a red arrow from "Nested data (way to create relationships between data)" points to the "address" field; and a teal arrow from "Embedded documents inside an array" points to the first document in the "interests" array.

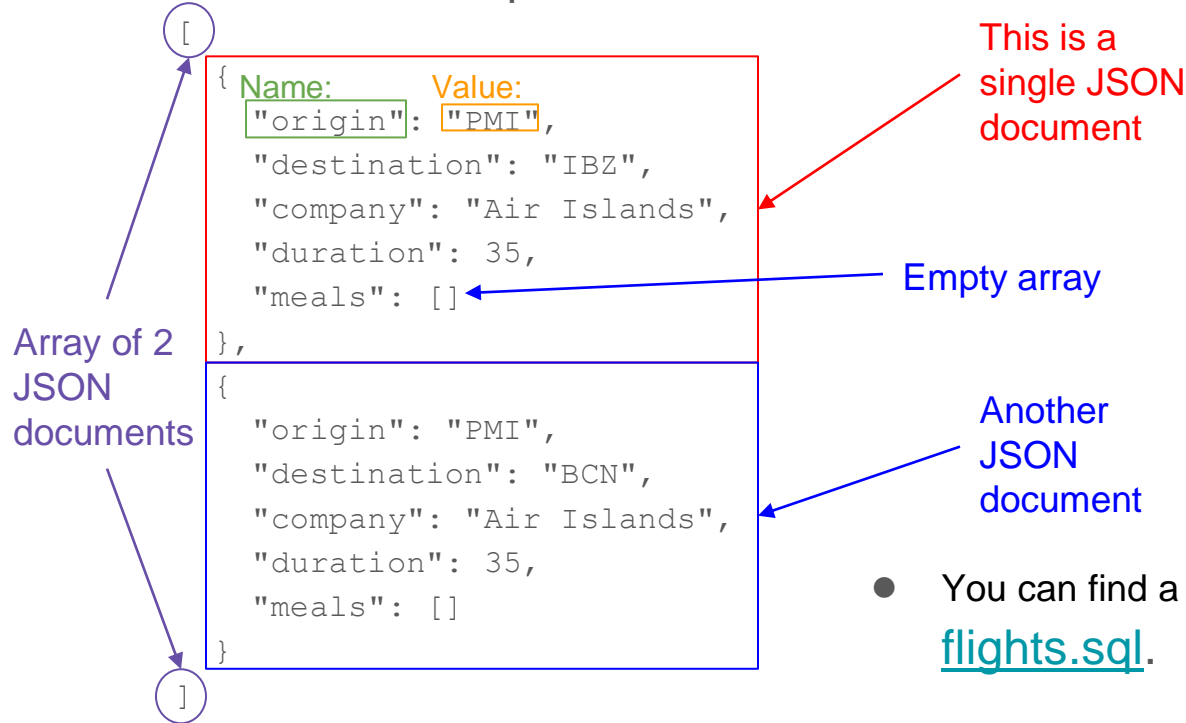
Document
inside a
collection of
a database

12.1.1. What is MongoDB?

[Mapping Terms and Concepts from SQL to MongoDB](#)

12.1.2. BSON Data Structure.

- JSON file example:

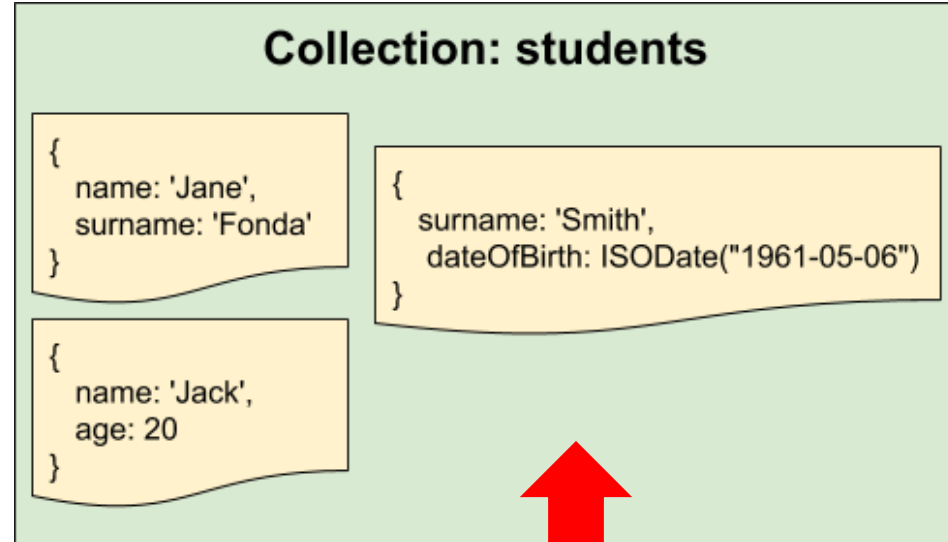


- Watch: <https://www.youtube.com/watch?v=iiADhChRriM>

- You can find a more complex example here [flights.sql](#).

12.1.2. BSON Data Structure.

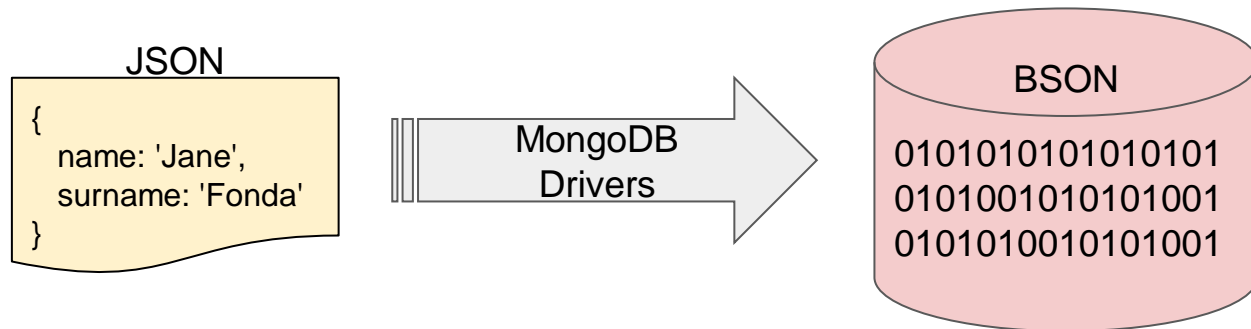
- MongoDB is NoSQL, that is:
 - We do not normalize data (sometimes we store data together in a JSON document).
 - We may not have a schema in a MongoDB database (documents may have different structure inside the same collection).
- Structured data (e.g. PostgreSQL) VS Flexibility (MongoDB).
- Relational model needs to JOIN relations to access related data. With MongoDB you may have related data inside the same document (or maybe not!!).



No-schema (or schemaless)

12.1.2. BSON Data Structure.

- **JSON is stored internally as BSON for efficiency.**
 - BSON: Binary data.
 - BSON: extends JSON Types (e.g. more detailed data types for numbers).
 - BSON: Efficient storage
 - BSON specification: <https://www.json.org/> and <http://bsonspec.org/>
 - Cool tool: <https://next.json-generator.com/>

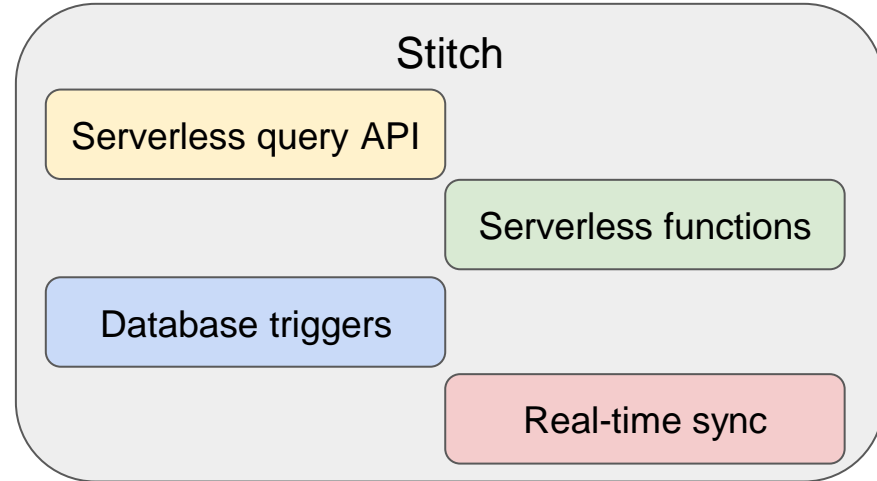
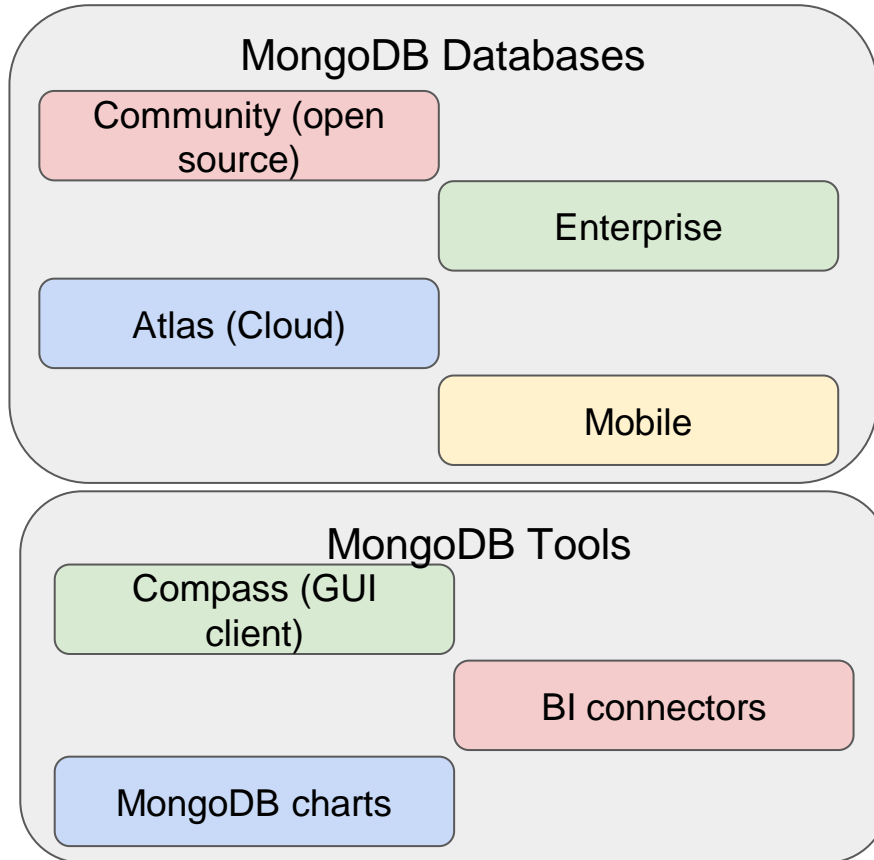


<https://www.mongodb.com/json-and-bson>

12.1.3. MongoDB ecosystem.

- MongoDB is developed by MongoDB Inc. and licensed under the Server Side Public License (SSPL).
- Server Side Public License FAQ: <https://www.mongodb.com/licensing/server-side-public-license/faq>

12.1.3. MongoDB ecosystem.



12.1.3. MongoDB ecosystem.

- **“MongoDB Database:** This is obvious, the database you want to work on it. It has community edition and has all the features you need for now.
- **Atlas (Cloud):** You can focus your data and all the system and ops related tasks done for you.
- **Mobile:** MongoDB Mobile lets you store data where you need it, from iOS, Android, and IoT devices to your backend in the cloud – all using a single database. MongoDB Mobile lets you build the fastest, most reactive, always-on apps.
- **Compass:** This is the user interface to manage your MongoDB. Visually explore your data. Run ad hoc queries in seconds. Interact with your data with full CRUD functionality. View and optimize your query performance.
- **BI Connectors:** The MongoDB BI Connector lets you use MongoDB as a data source for your BI and analytics platforms. Seamlessly create the visualizations and dashboards that will help you extract the insights and hidden value in your multi-structured data.
- **MongoDB Charts:** [...] MongoDB Charts is the fastest and easiest way to create visualizations of MongoDB data. Connect to any MongoDB instance as a data source, create charts and graphs, build dashboards, and share them with other users for collaboration.”

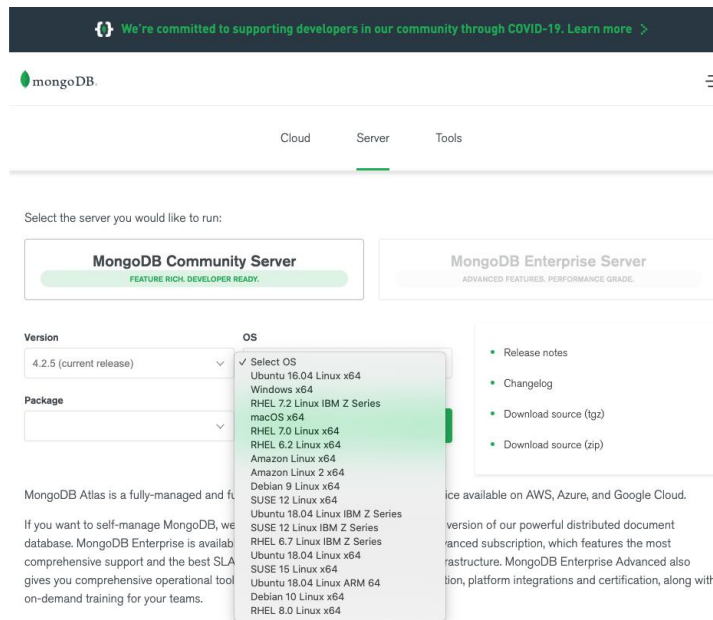
Source: mrtgr.com

- **“Stitch:**
 - Exposes the full power of working with documents in MongoDB and the MongoDB query language, directly from your web and mobile application front-end code. A powerful rules engine lets developers declare fine-grained security policies.
 - Allows developers to run simple JavaScript functions in the Stitch serverless platform, making it easy to implement application logic, securely integrate with cloud services and microservices, and build APIs.
 - Executes functions in real time in response to changes in the database or user authentication events.
 - Automatically synchronizes data between documents held locally in MongoDB Mobile and the backend database.
 - This is the snapshot of the ecosystem. You can find much more detail on <http://www.mongodb.com> . I’m very excited about Stitch and its abilities. See you in the next article.”

12.1.4. Installing MongoDB.

- Available for MacOS, Windows and Linux.
- Community Server is the free edition, download it from here:

<https://www.mongodb.com/download-center/community>



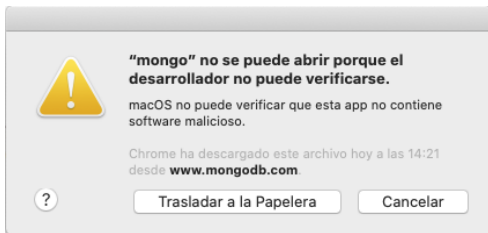
12.1.4. Installing MongoDB.

Source: MongoDB

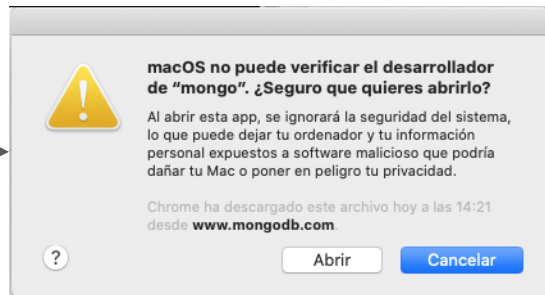
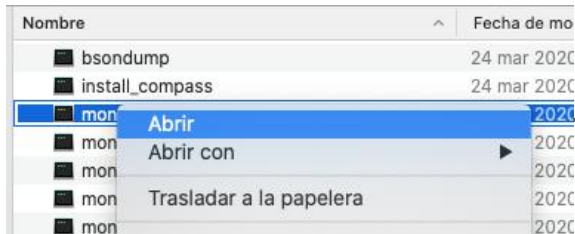
- Official installation guides:
 - Windows: <https://docs.mongodb.com/manual/tutorial/install-mongodb-on-windows/>
 - macOS: <https://docs.mongodb.com/manual/tutorial/install-mongodb-on-os-x/>
 - Linux: <https://docs.mongodb.com/manual/administration/install-on-linux/>

Let's practice

- Install mongodb in your computer following the video that we watched before and reading the official installation guide for your operating system.
- With newer versions of MacOS you may get this problem running mongo and mongod:



- To solve it just run one time the program from Finder:



Let's practice

linux shell → \$
mongo shell → >

- After all the installation process you will be able to access mongodb from mongo shell:

default port 27017

MongoDB server running:

```
$mongo --dbpath /Users/sergio/mongodb/data/db
```

```
$mongo --dbpath /Users/sergio/mongodb/data/db --port 27018
```

Accessing MongoDB server from mongo shell:

```
$mongo --port 27018
```

```
sergio — mongod --dbpath ~/mongodb/data/db — 134x36
2020-04-04T16:40:28.592+0200 I FTDC [initandlisten] Initializing full-time diagnostic data capture with directory '/Users/sergio/
mongodb/data/db/diagnostic.data
2020-04-04T16:40:28.925+0200 I <unsharded> SHARDING [LogicalSessionCacheRefresh] Marking collection config.system.sessions as collection version:
2020-04-04T16:40:28.925+0200 I NETWORK [listener] Listening on /tmp/mongodb-27017.sock
2020-04-04T16:40:28.925+0200 I NETWORK [listener] Listening on 127.0.0.1
2020-04-04T16:40:28.925+0200 I CONTROL [LogicalSessionCacheReap] Sessions collection is not set up; waiting until next sessions reap
interval: config.system.sessions does not exist
2020-04-04T16:40:28.925+0200 I NETWORK [listener] waiting for connections on port 27017
2020-04-04T16:40:28.925+0200 I STORAGE [LogicalSessionCacheRefresh] createCollection: config.system.sessions with provided UUID: 5b2
81be2-be22-4102-b237-8818077b78be and options: { uuid: UUID("5b281be2-be22-4102-b237-8818077b78be") }
2020-04-04T16:40:29.002+0200 I SHARDING [ftdc] Marking collection local.oplog.rs as collection version: <unsharded>
2020-04-04T16:40:29.526+0200 I INDEX [LogicalSessionCacheRefresh] index build: done building index _id_ on ns config.system.ses
2020-04-04T16:40:30.049+0200 I INDEX [LogicalSessionCacheRefresh] index build: starting on config.system.sessions properties: { v:
2, key: { lastUse: 1 }, name: "lsidTTLIndex", ns: "config.system.sessions", expireAfterSeconds: 1800 } using method: Hybrid
2020-04-04T16:40:30.049+0200 I INDEX [LogicalSessionCacheRefresh] build may temporarily use up to 200 megabytes of RAM
2020-04-04T16:40:30.049+0200 I INDEX [LogicalSessionCacheRefresh] index build: collection scan done. scanned 0 total records in 0
seconds
2020-04-04T16:40:30.049+0200 I INDEX [LogicalSessionCacheRefresh] index build: inserted 0 keys from external sorter into index in
0 seconds
2020-04-04T16:40:30.171+0200 I INDEX [LogicalSessionCacheRefresh] index build: done building index lsidTTLIndex on ns config.syste
m.sessions
2020-04-04T16:40:30.315+0200 I COMMAND [LogicalSessionCacheRefresh] command config.system.sessions command: createIndexes { createIn
dexes: "system.sessions", indexes: [ { key: { lastUse: 1 }, name: "lsidTTLIndex", expireAfterSeconds: 1800 } ], $db: "config" } numYie
ls: 0 reslen:14 locks: { ParallelBatchWriterMode: { acquireCount: { r: 2 } }, ReplicationStateTransition: { acquireCount: { w: 3 } },
Global: { acquireCount: { r: 1, w: 2 } }, Databases: { acquireCount: { r: 1, w: 2, w: 1 } }, Collection: { acquireCount: { r: 4, w: 1,
R: 1, W: 2 } }, Mutex: { acquireCount: { r: 3 } } } flowControl: { acquireCount: 1, timeAcquiringMicros: 1 } storage: {} protocol:op_m
sg 1390ms
2020-04-04T16:41:38.351+0200 I NETWORK [listener] connection accepted from 127.0.0.1:52385 #1 (1 connection now open)
2020-04-04T16:41:38.352+0200 I NETWORK [conn1] received client metadata from 127.0.0.1:52385 conn1: { application: { name: "MongoDB
Shell" }, driver: { name: "MongoDB Internal Client", version: "4.2.5" }, os: { type: "Darwin", name: "Mac OS X", architecture: "x86_64
", version: "19.4.0" } }
2020-04-04T16:45:28.926+0200 I SHARDING [LogicalSessionCacheReap] Marking collection config.transactions as collection version: <unsh
arded>
```

```
sergio — mongo — 134x36
MongoDB shell version v4.2.5
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id": UUID("ce852464-082e-4b58-9b84-2f9404eb61f5") }
MongoDB server version: 4.2.5
Welcome to the MongoDB shell.
For interactive help, type "help".
For more comprehensive documentation, see
http://docs.mongodb.org/
Questions? Try the support group
http://groups.google.com/group/mongodb-user

Server has startup warnings:
2020-04-04T16:40:27.493+0200 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
2020-04-04T16:40:27.493+0200 I CONTROL [initandlisten] ** Read and write access to data and configuration is unrestricted.
2020-04-04T16:40:27.493+0200 I CONTROL [initandlisten]
2020-04-04T16:40:27.493+0200 I CONTROL [initandlisten] ** WARNING: This server is bound to localhost.
2020-04-04T16:40:27.493+0200 I CONTROL [initandlisten] ** Remote systems will be unable to connect to this server.
2020-04-04T16:40:27.493+0200 I CONTROL [initandlisten] ** Start the server with --bind_ip <address> to specify which IP
2020-04-04T16:40:27.493+0200 I CONTROL [initandlisten] ** addresses it should serve responses from, or with --bind_ip_all t
o bind to all interfaces. If this behavior is desired, start the
2020-04-04T16:40:27.493+0200 I CONTROL [initandlisten] ** server with --bind_ip 127.0.0.1 to disable this warning.
2020-04-04T16:40:27.493+0200 I CONTROL [initandlisten]

---
Enable MongoDB's free cloud-based monitoring service, which will then receive and display
metrics about your deployment (disk utilization, CPU, operation statistics, etc).

The monitoring data will be available on a MongoDB website with a unique URL accessible to you
and anyone you share the URL with. MongoDB may use this information to make product
improvements and to suggest MongoDB products and deployment options to you.

To enable free monitoring, run the following command: db.enableFreeMonitoring()
To permanently disable this reminder, run the following command: db.disableFreeMonitoring()

>
```

Help for MongoDB server: \$mongo --help

Help for MongoDB client: \$mongo --help -> inside the client: > help

12.1.4. Installing MongoDB.

Source: MongoDB

- If you want to run MongoDB setting a log file:
 - First create the directory if it doesn't exist:
 - `mkdir /Users/sergi/mongodb/logs`
 - `mongod --dbpath /Users/sergi/mongodb/data/db --logpath /Users/sergi/mongodb/logs/log.log`
- If you want to run MongoDB as a background process `--fork`:
 - `mongod --fork --dbpath /Users/sergi/mongodb/data/db --logpath /Users/sergi/mongodb/logs/log.log`
- If you want to shutdown the server to which you are connected from "mongo" client:
 - `use admin`
 - `db.shutdownServer()`

12.1.4. Installing MongoDB.

Source: MongoDB

- You can save your setting in a file named “mongod.cfg”:
 - `vi /Users/sergio/mongodb/bin/mongod.cfg`
storage:
 `"/Users/sergio/mongodb/data/db"`
systemLog:
 destination: file
 path: `"/Users/sergio/mongodb/logs/log.log"`
 - To run MongoDB using this file:
 ■ `sudo mongod -f /Users/sergio/mongodb/bin/mongod.cfg`
- You have more examples [here](#):

```
systemLog:
  destination: file
  path: "/var/log/mongodb/mongod.log"
  logAppend: true
storage:
  journal:
    enabled: true
processManagement:
  fork: true
net:
  bindIp: 127.0.0.1
  port: 27017
setParameter:
  enableLocalhostAuthBypass: false
...
```

12.1.4. Installing MongoDB.

- Configuration File Options:
<https://docs.mongodb.com/manual/reference/configuration-options/>
- mongo (shell client):
<https://docs.mongodb.com/manual/reference/program/mongo/>
- mongod (daemon process):
<https://docs.mongodb.com/manual/reference/program/mongod/>

12.1.5. Getting Started.

```
> x = 200;
```

```
> x / 5;
```

```
> function factorial (n) {if (n <= 1) return 1; return n * factorial(n - 1);}
```

```
> factorial(5);
```

```
> x = 200;  
200  
> x / 5;  
40
```

```
> function factorial (n) {  
... if (n <= 1) return 1;  
... return n * factorial(n - 1);  
... }  
> factorial(5);  
120
```

12.1.5. Getting started.

- Show existing databases:

- `> show dbs`

- Connect to a database:

- `> use shop`

- Create a collection:

- `> db.products.insertOne({name:
"Little Prince", price: 12.99})
{`

- `"acknowledged" : true,
"insertedId" :`

- MongoDB automatically generates a unique ID → `ObjectId("5e89d4f1dd878b7131f6aa93")`

- Show data in a collection (pretty):

- `> db.products.find().pretty()
{
 "_id" :
 ObjectId("5e89d4f1dd878b7131f6aa93"),
 "name" : "Little Prince",
 "price" : 12.99
}`

- Show data in a collection:

- `> db.products.find()
{ "_id" : ObjectId("5e89d4f1dd878b7131f6aa93"),
 "name" : "Little Prince", "price" : 12.99 }`

- Show collections:

- `> show collections`

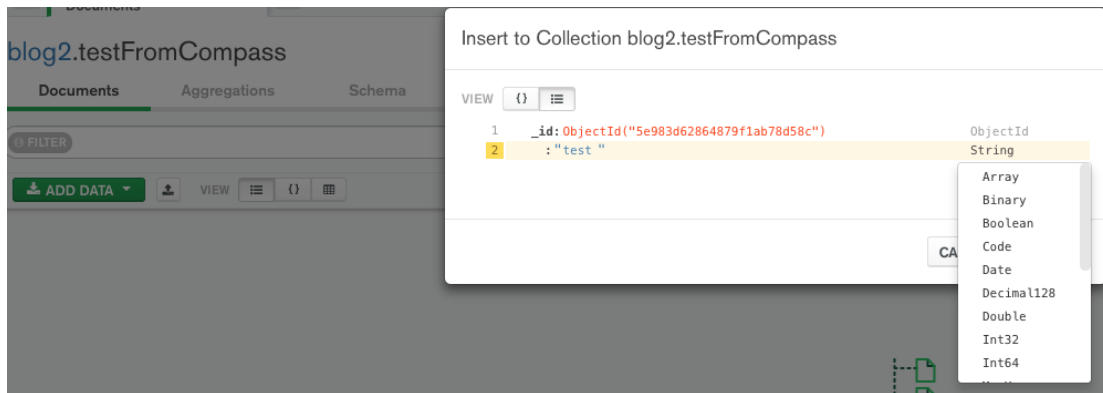
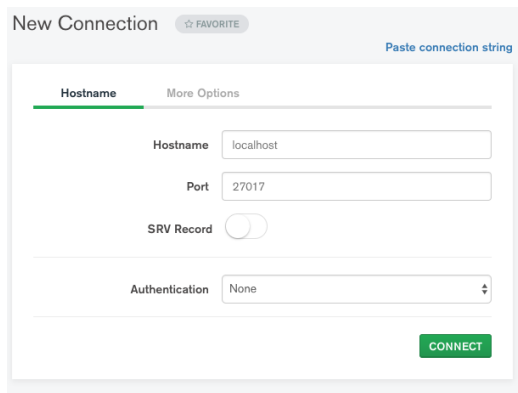
You can set `"_id"` but it must be unique (concept of PK in relational model).

12.1.5. Getting started.

- To remove a database:
 - `use databaseName`
 - `db.dropDatabase()`
- To remove a collection:
 - `db.myCollection.drop()`
- To get statistics about a database:
 - `db.stats()`
- Current DB:
 - `db.getName()` (or simply `"db"`)
- Clean the screen:
 - `cls`

12.1.5. Getting started.

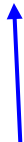
- As you know, I'll use only the shell. But there is a visual tool (now free for all!): [MongoDB Compass](#).
- It's up to you to use it. A good video about Compass (Spanish): <https://youtu.be/dBpRBz5ye-g>



12.1.5. Getting started.

- You can load data with load() function.
 - > load(file.js)

It will search in the path where you executed "mongo" (you can also use an absolute path).



file.js

```
db = db.getSiblingDB("myDB");  
db.myCol.drop();  
db.myCol.insertMany([  
...  
]);
```

12.1.6. Shell vs drivers.

Source: Academind

- Drivers: Drivers are “bridges” between your programming language and MongoDB server.
- Drivers documentation; <https://docs.mongodb.com/drivers/>

Java:

```
import com.mongodb.ConnectionString;
import com.mongodb.client.MongoClients;
import com.mongodb.client.MongoClient;
import com.mongodb.client.MongoDatabase;
import com.mongodb.MongoClientSettings;

// ...
ConnectionString connString = new ConnectionString(
    "mongodb+srv://<username>:<password>@<cluster-address>/test?w=majority"
);
MongoClientSettings settings = MongoClientSettings.builder()
    .applyConnectionString(connString)
    .retryWrites(true)
    .build();
MongoClient mongoClient = MongoClient.create(settings);
MongoDatabase database = mongoClient.getDatabase("test");
```

Python:

```
import pymongo

client = pymongo.MongoClient(
    "mongodb+srv://<username>:<password>@<cluster-url>/test?retryWrites=true&w=majority")
db = client.test
```

Do you understand the difference between working with the shell and using a driver to connect from a programming language?

12.1.6. Shell vs drivers.

```
$ python -m pip install pymongo
$ python -m pip install PyHamcrest
$ python
>>> from pymongo import MongoClient
>>> from pprint import pprint
>>> client = MongoClient("localhost")
>>> db = client.contactsDB
>>> contact = db.contacts.find_one({})
>>> print(contact)
```

Source: <https://dzone.com/articles/getting-started-with-python-and-mongodb>

12.1.6. Shell vs drivers.

```
>>> contacts = db.contacts.find({})  
>>> for doc in contacts:  
>>>     print(doc)
```

Source: <https://dzone.com/articles/getting-started-with-python-and-mongodb>

12.1.7. Basic architecture.

