SIM GLOBAL EDUCATION

UNIVERSITY OF WOLLONGONG AUSTRALIA

CSCI235 – Database Systems

MongoDB Databases, Collections, Documents

Sionggo Japit

sjapit@uow.edu.au

3 October 2021

Acknowledgements

The following presentation were adapted from the lecture slides of:

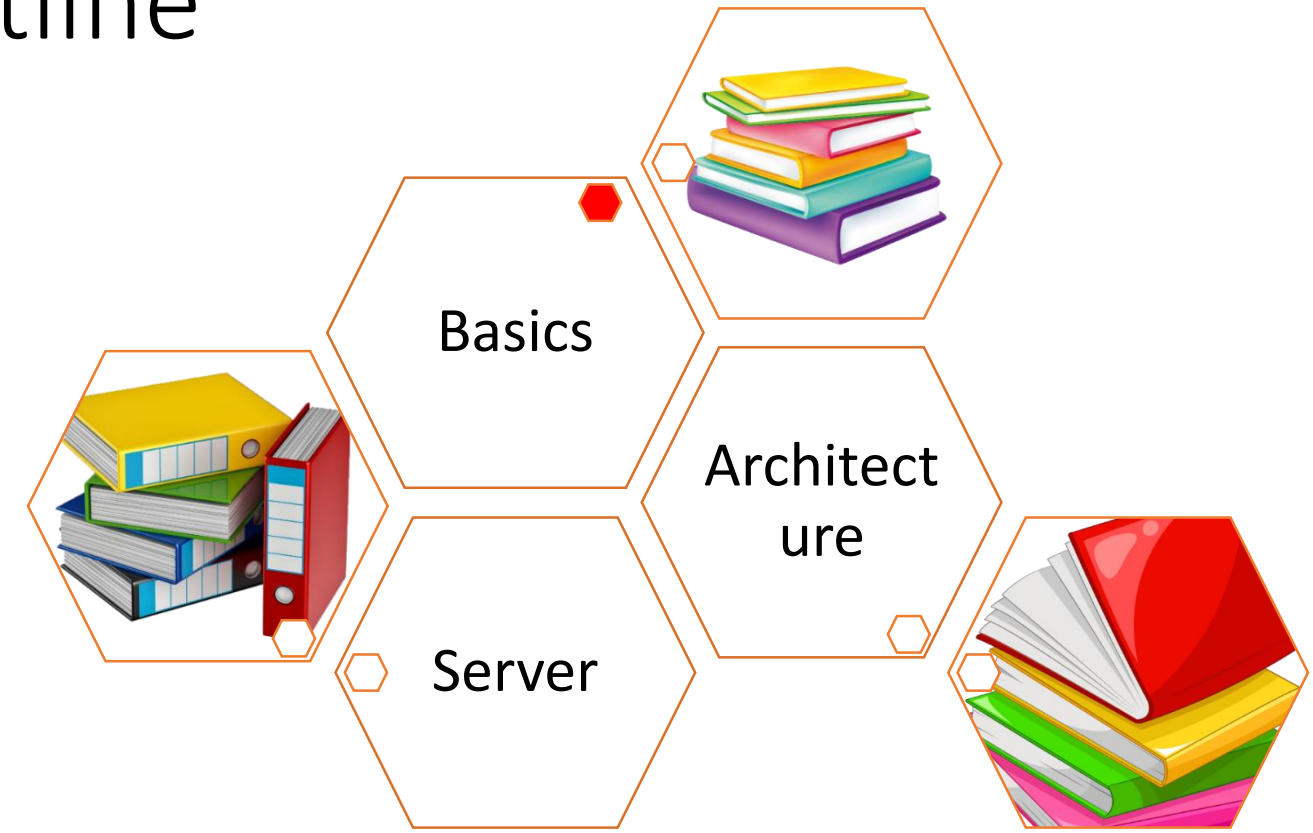
CSCI235 – Database Systems, 15MongoDBDatabasesCollectionsDocuments

By Dr Janusz R. Getta,

School of Computing and Information Technology

University of Wollongong, Australia

Outline



MongoDB basics

- MongoDB is a database system that belong to a class of so called NoSQL database systems based on a data model different from the relational model and data definition, manipulation, retrieval, and administration languages different from SQL.
- MongoDB data model (BSON) is based on a concept of key:value pairs grouped into documents and arrays.
- MongoDB database system operates on a number of databases.
- A MongoDB database is a set of collections
- A MongoDB collection is a set of documents
- A MongoDB document is a set of key:value pairs

MongoDB basics

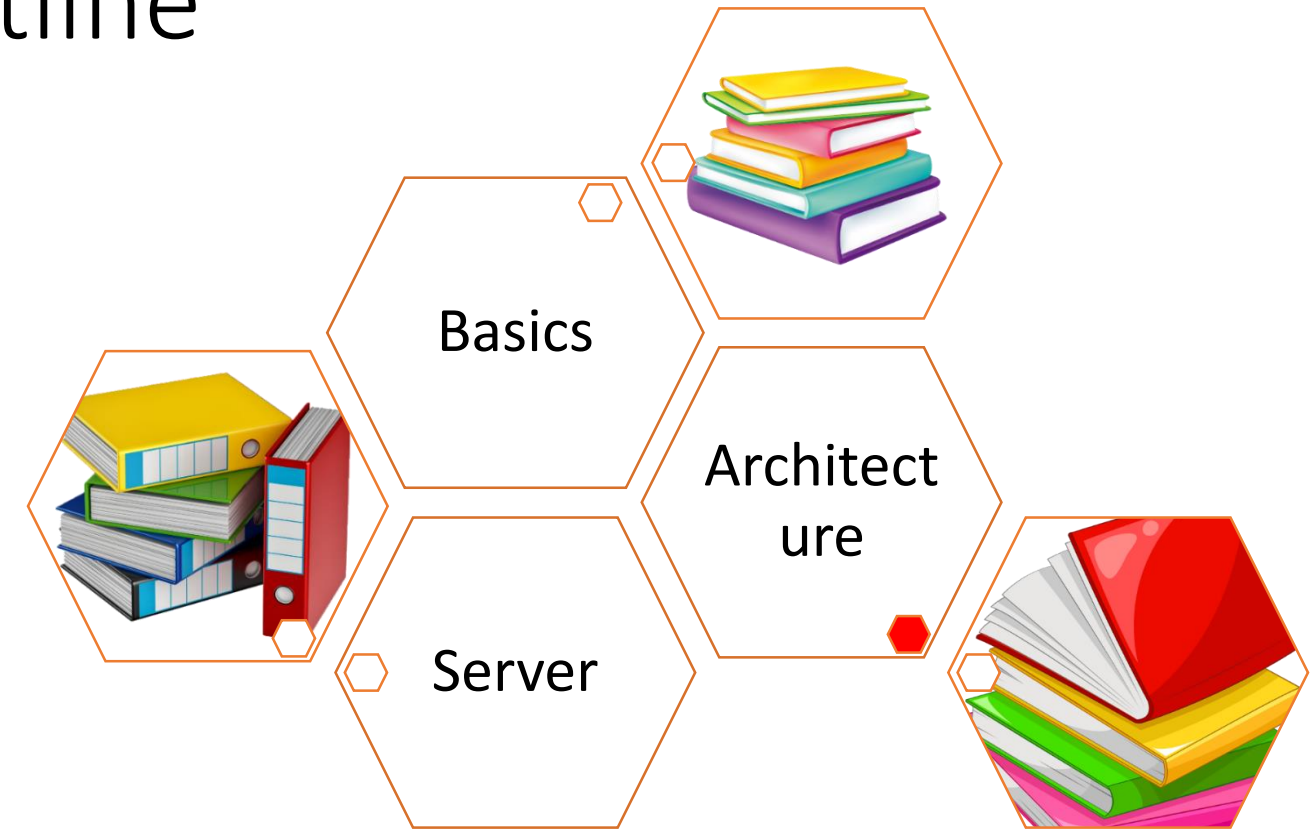
- A MongoDB value is either an atomic value or a document or an array.
- A MongoDB atomic value is of one of the types included BSON specification like number, string, date, etc.
- A MongoDB array is a sequence of values.
- Each MongoDB key:value pair must have a unique key within a document.
- Each MongoDB document must have a unique identifier within a collection.
- Each MongoDB collection must have a unique name within a database.

MongoDB basics

- A sample MongoDB document describing a person:

```
{"_id": ObjectId(),
 "full name": {"first name":"James",
               "initials":null,
               "last name":"Bond"},
 "employee number":"007",
 "skills":["cooking", "painting", "gardening"],
 "cars owned":[ {"rego":"007-1",
                 "made":"Porsche"},
                {"rego":"007-2",
                 "made":"Ferrari"} ],
 "secret codes":[ [1,2,3,4],
                  [9,8,7,5] ],
 "date of birth":new Date("1960-01-01")
}
```

Outline



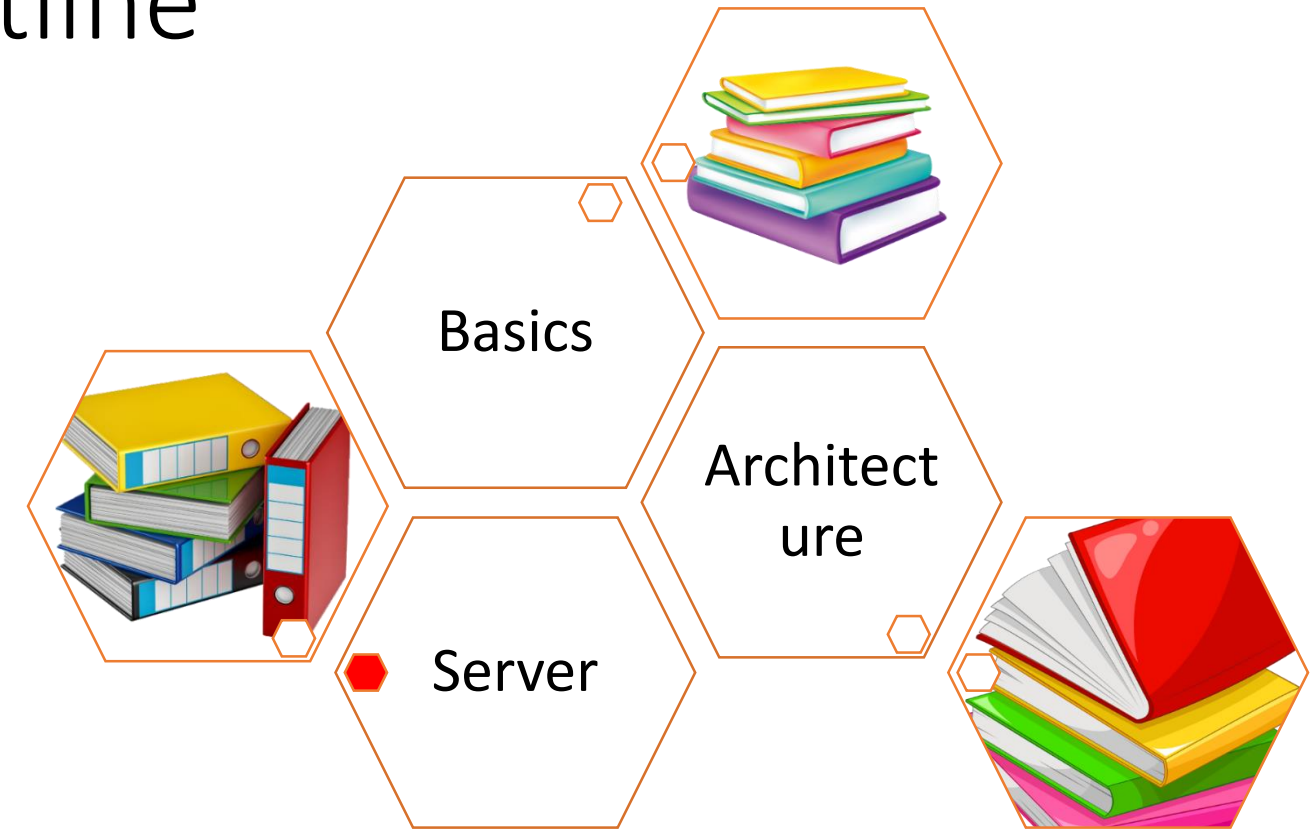
MongoDB architecture

- MongoDB flexible storage architecture automatically manages the movement of data between storage engine technologies using native replication.
- MongoDB stores data as documents in a binary representation called BSON (Binary JSON).
- MongoDB query model is implemented as methods or functions within the API of a specific programming language, as opposed to a completely separate language like SQL.
- MongoDB provides horizontal scale-out for database on low cost, commodity hardware or cloud infrastructure using a technique called sharding, which is transparent to applications.

MongoDB architecture

- In-Memory storage engine enables performance advantages of in-memory computing for operational and real-time analytics workloads.
- MongoDB Enterprise Advanced provides extensive capabilities to defend, detect, and control access to data (data security)
- MongoDB Ops Manager makes easy for operations teams to deploy, monitor, backup and scale the system (system management).
- MongoDB Atlas provides all of the features of Database as a Service cloud computing model.

Outline



MongoDB server

- Starting MongoDB server with default options.

```
sudo service mongod start
```

- Starting MongoDB server with options

```
mongod - -dbpath data - -port 4000
```

- Starting MongoDB command based shell

```
Mongo - -port 4000
```

MongoDB server

- Getting the first help from MongoDB shell:

```
help
```

db.help()	Help on db methods
show dbs	Show database names
show collections	Show collections in current database
use db_name	Set current database
…	…

MongoDB server

- Setting a current database

```
use database-name
```

For example, using a database local

```
use local
```

- Creating and switching to a new database mydb

```
use mydb
```

MongoDB server

- Listing the database

```
show dbs
```

```
local 0.000GB
mydb 0.000GB
```

- Creating a new collection with an empty document:

```
db.mycol.insert({ })
```

MongoDB server

- Listing the contents of a collection:

```
db.mycol.find()
```

```
{ "_id" : ObjectId("5eb0fb63e4a99be6e549561c") }
```

- Listing the collections:

```
show collections
```

```
mycol
```

MongoDB server

- Creating a new non empty document:

```
db.mycol.insert({"one":"1", "many ones":[1, 1, 1, 1]})
```

- Listing the contents of a collection:

```
db.mycol.find()
```

```
{ "_id" : ObjectId("5eb0fb63e4a99be6e549561c") }
{ "_id" : ObjectId("5eb0fd83e4a99be6e549561d"),
"one" : "1", "many ones" : [ 1, 1, 1, 1 ] }
```

MongoDB server

- Listing the nicely formatted contents of a collection

```
db.mycol.find().pretty()
```

```
{ "_id" : ObjectId("5eb0fb63e4a99be6e549561c") }
{ "_id" : ObjectId("5eb0fd83e4a99be6e549561d"),
        "one" : "1",
        "many ones" : [ 1,
                        1,
                        1,
                        1 ]
}
```

MongoDB server

- Removing all documents from a collection

```
db.mycol.remove({})
```

- Removing a collection

```
db.mycol.drop()
```

- Removing a database

```
db.dropDatabase()
```

MongoDB server

Let a file data.js contains the following insert methods

```
db.mycol.insert({"CITY": {"name":"Wollongong",
                          "population":"80K",
                          "country":"Australia",
                           "state":"New South Wales"} });
db.mycol.insert({"EMPLOYEE": {      "enum":1234567,
                                    "full-name":"Janusz R.Getta",
                                    "salary":"200K",
                                    "hobbies": [ "cooking",
                                                 "painting",
                                                 "gardening"]} })
```

MongoDB server

- Processing a script inserts two documents into a collection mycol

```
load("data.js")
```

- Listing a collection mycol

```
db.mycol.find().pretty()
```

```
{
        "_id" : ObjectId("5eb10547e4a99be6e5495622"),
        "CITY" : {
                "name" : "Wollongong",
                "population" : "80K",
                "country" : "Australia",
                "state" : "New South Wales"
                } }
{
        "_id" : ObjectId("5eb10547e4a99be6e5495623"),
        "EMPLOYEE" : {
                "enum" : 1234567,
                "full-name" : "Janusz R.Getta",
                "salary" : "200K",
                "hobbies" : [
                        "cooking",
                        "painting",
                        "gardening"
                ]
        } }
```

MongoDB query language

- MongoDB query language is based on a concept of pattern matching.
- A query is expressed as a BSON pattern and all document from a collection that match the pattern are included in an answer.
- A method find() is used to match a pattern with the documents in a collection

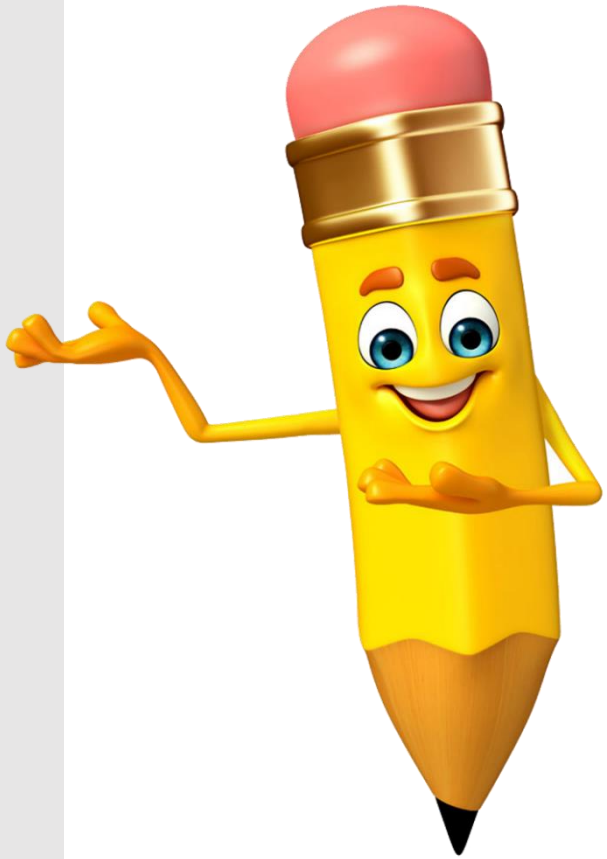
```
db.department.find({“age”: 25})
```

- Matching of an empty pattern{} with a collection returns an entire collection.

```
db.department.find({})
```

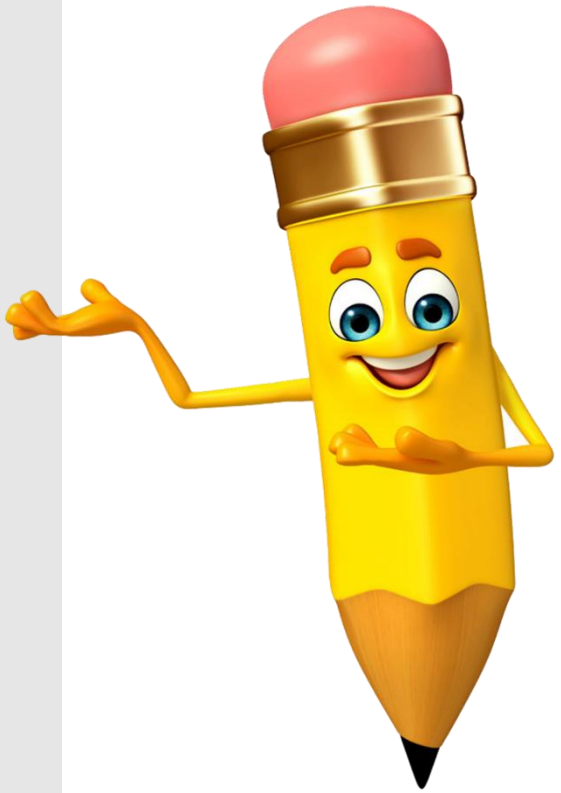
```
db.department.insert({
      "name":"School of Computing and Information Technology",
      "code":"SCIT",
      "totalNumOfStaff":30,
      "budget":1000000,
      "address": {"street":"Northfields Ave",
                  "bldg":3,
                  "city":"Wollongong",
                  "country":"Australia"},
      "courses": [{"code":"CSCI235",
                   "title":"Database Systems",
                   "credits":6},
                  {"code":"CSIT115",
                   "title":"Data Management and Security",
                   "credits":6},
                  {"code":"CSIT317",
                   "title":"Database Performance Tuning",
                   "credits":6},
                  {"code":"CSIT321",
                   "title":"Software Project",
                   "credits":12}
                 ]
});
```

The first document



```
db.department.insert({
        "name":"School of Astronomu",
        "code":"SOA",
        "totalNumOfStaff":5,
        "budget":10000,
        "address": {"street":"Franz Josef Str",
                    "bldg":4,
                    "city":"Vienna",
                    "country":"Austria"},
        "courses": [{"code":"SOA101",
                     "title":"Astronomy for Kids",
                     "credits":3},
                    {"code":"SOA201",
                     "title":"Black Holes",
                     "credits":6},
                    {"code":"SOA301",
                     "title":"Dark Matter",
                     "credits":12}
                    ]
});
```

The second document



```
db.department.insert({
                "name":"School of Physics",
                "code":"SOPH",
                "totalNumOfStaff":25,
                "budget":100000,
                "address": {"street":"Victoria St",
                                "bldg":25,
                                "city":"Cambridge",
                                "country":"UK"},
                "courses": [{"code":"SOPH101",
                                 "title":"Special Relativity",
                                 "credits":6},
                                {"code":"SOPH102",
                                 "title":"General Relativity",
                                 "credits":12},
                                {"code":"SOPH103",
                                 "title":"Quantum Mechanics",
                                 "credits":18}
                               ]
});
```

The third document

Simple queries

- Populate documents to the department collection using load():

```
> load("/home/csci235/Script/department.js")
true
```

- Find total number of documents in a collection

```
> db.department.count()
3
```

Simple queries

- Find all departments whose code is SCIT

```
db.department.find({"code":"SCIT"})

{ "_id" : ObjectId("5eb11785e4a99be6e5495624"), "name" :
"School of Computing and Information Technology", "code" :
"SCIT", "totalNumOfStaff" : 30, "budget" : 1000000,
"address" : { "street" : "Northfields Ave", "bldg" : 3, "city" :
"Wollongong", "country" : "Australia" }, "courses" : [ { "code"
: "CSCI235", "title" : "Database Systems", "credits" : 6 }, {
"code" : "CSIT115", "title" : "Data Management and
Security", "credits" : 6 }, { "code" : "CSIT317", "title" :
"Database Performance Tuning", "credits" : 6 }, { "code" :
"CSIT321", "title" : "Software Project", "credits" : 12 } ] }
```

Simple queries

- Find total number of departments whose code is SOPH

```
> db.department.find({"code":"SOPH"}).count()
1
> db.department.count({"code":"SOPH"})
1
```

Simple queries

- Find all departments whose name is School of Physics and whose code is SOPH.

```
> db.department.find({"name":"School of
Physics","code":"SOPH"})
{ "_id" : ObjectId("5eb12169e4a99be6e5495629"),
"name" : "School of Physics", "code" : "SOPH",
"totalNumOfStaff" : 25, "budget" : 100000, "address" :
{ "street" : "Victoria St", "bldg" : 25, "city" :
"Cambridge", "country" : "UK" }, "courses" : [ { "code" :
"SOPH101", "title" : "Special Relativity", "credits" : 6 },
{ "code" : "SOPH102", "title" : "General Relativity",
"credits" : 12 }, { "code" : "SOPH103", "title" :
"Quantum Mechanics", "credits" : 18 } ] }
```

Boolean expressions

- Comparison “key”=“value”

```
{“key”:”value”}

{“key”: {$eq: “value”} }
```

- Comparison “key” > “value”

```
{“key”: {$gt: “value”} }
```

Boolean expressions

- Disjunction (“key1” = “value1”) or (“key2” = “value2”)

```
{$or: [{“key”:”value1”}, {“key2”:”value2”}]}
```

- Conjuction (“key1” = “value1”) and (“key2” = “value2”)

```
{$and: [{“key1”:”value1”}, {“key2”:”value2”}]}
```

Boolean expressions

- Boolean expression ((“key1” = “value1”) or (“key2” = “value2”)) and (“key3” = “value3”)

```
{$and: [ {$or: [ {“key1”:”value1”},
{“key2”:”value2”} ] }, {“key3” = “value2”} ] }
```

- Negation of a comparison “key” not =“value”

```
{“key”: {$not: {$eq: “value”}}}
```

Boolean expressions

- Negation of an expression not ((“key1” = “value1”) or (“key2” = “value2”))

```
{$nor: [ {“key1”:”value1”}, {“key2”:”value2”} ] }
```

- Negation not (“key1” = “value1”)

```
{$nor: [ {“key1”:”value1”} ] }
```

MongoDB server

- Stopping MongoDB server with default options

```
sudo service mongod stop
```

References

- MongoDB Architecture, https://www.mongodb.com/mongodb-architecture

- Chodorow K. MongoDB The Definitive Guide, O'Reilly, 2013, Chapter 2