Professor Fausto Cintra
professor@faustocintra.com.br

MongoDB

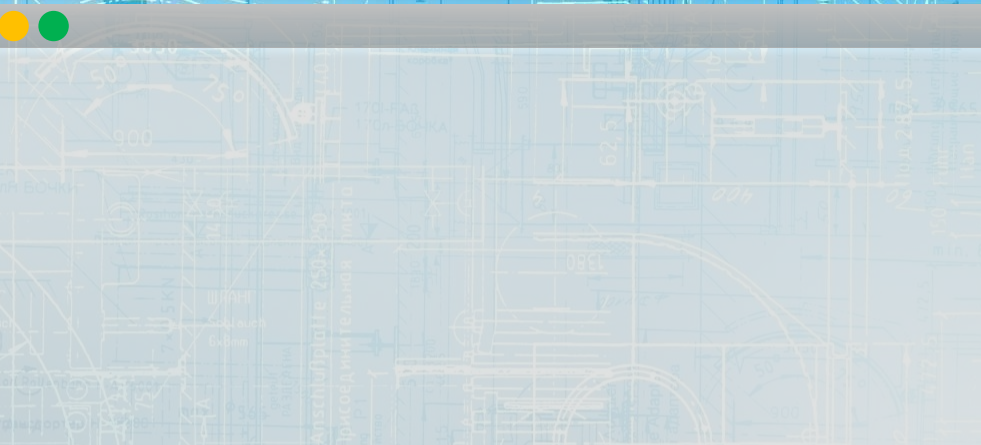
Uma introdução

VISÃO GERAL

O que é o MongoDB?

- **Banco de dados de código aberto orientado a documentos**
  - documentos representam ou codificam dados (ou informação) em formatos ou codificações padronizados
  - Banco de dados NoSQL
  - Usa os formatos de dados JSON e BSON (JSON binário)
  - Sem esquema
  - Campos não têm tipo predefinido
  - Suporte a transações ACID multidocumentos foi introduzido na versão 4.0 (2018)
  - Sem suporte a JOINs (não há necessidade)
  - Alta disponibilidade: *replica set*
  - Escalabilidade horizontal por *sharding*

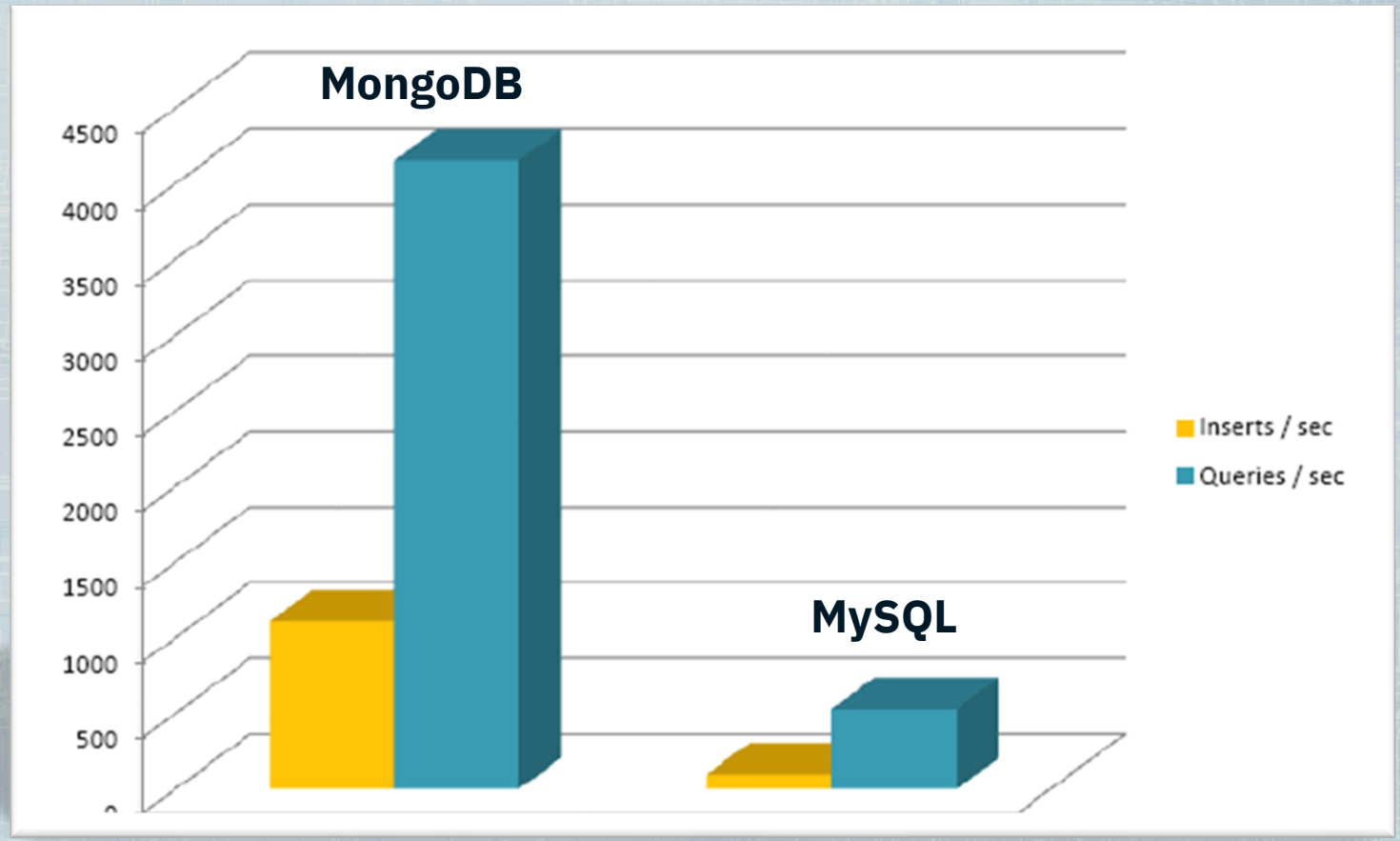
Origem



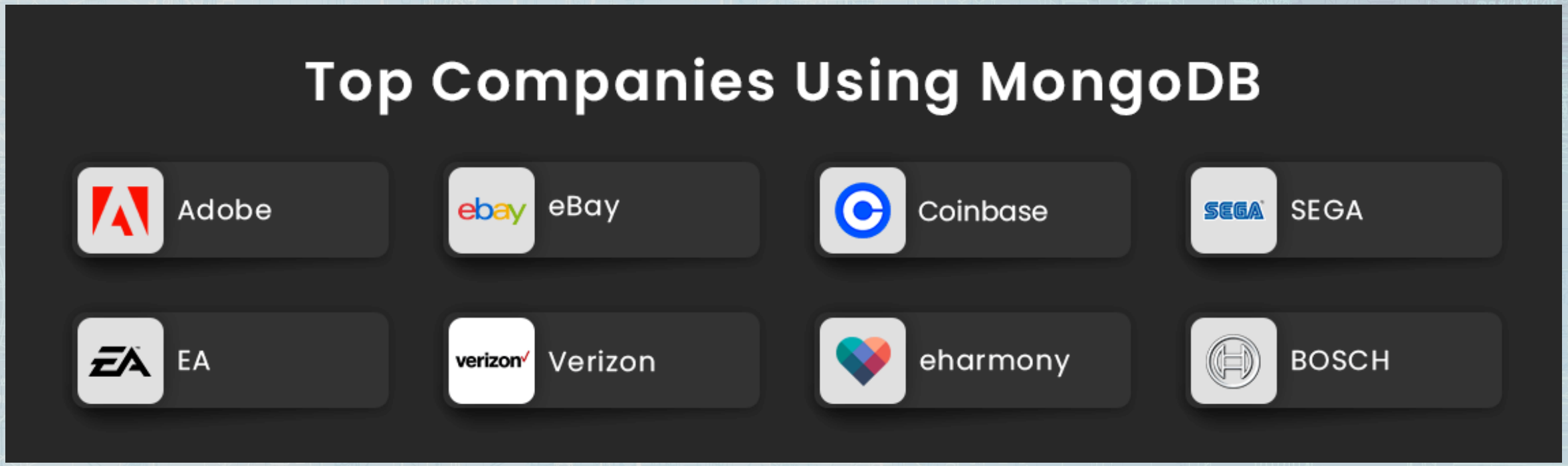
- Criado em 2007 com o nome de 10gen pela empresa de anúncios Doubleclick, mais tarde adquirida pela Google
- Ganhou o nome atual em 2013
- MongoDB deriva seu nome do termo **humongous**, que significa **gigantesco**, em inglês

Rápido

- Entre 2 e 10 vezes mais rápido que o MySQL



Quem usa o MongoDB



https://mongodb.com/who-uses-mongodb

Usando o MongoDB

- MongoDB pode ser baixado e instalado gratuitamente
  - Possui versões para as plataformas Windows, Linux e MacOS
- Pode ser usado também pré-instalado em nuvem (DBaaS – *Database as a Service*) por meio do serviço **MongoDB Atlas**
  - https://atlas.mongodb.com
  - Oferece um plano gratuito para iniciar
- Possui uma ferramenta gratuita de administração e consulta chamada **Compass**
  - https://www.mongodb.com/pt-br/products/compass
  - Versões para Windows, Linux e MacOS

Diferenças conceituais

RDBMS	MongoDB
banco de dados (*database*)	banco de dados (*database*)
tabela (*table*)	**coleção (*collection*)**
linha/registro (*row*)	**documento (*document*)**
coluna (*column*)	**campo (*field*)**
índice (*index*)	índice (*index*)
junção (*join*)	**documento incorporado ou vinculado (*embedded/linked document*)**
chave primária (*primary key*)	chave primária (primary key)
Você escolhe nome e tipo da chave primária	**A chave primária é um campo chamado *_id*, adicionado automaticamente**
agregação (*aggregation*, "*group by*")	**sequência de agregação (*aggregation pipeline*)**
Orientado a esquema	**Sem esquema**

Estrutura

- Uma instância do MongoDB pode ter um ou mais *databases*
- Uma *database* pode conter uma ou mais ***collections***
  - Podem ser pensadas como tabelas em um RDBMS, embora com muitas diferenças
- Uma *collection* pode ter zero ou mais **documentos**
  - Na mesma coleção, não precisam nem mesmo ter os mesmos campos
  - São o equivalente a registros/linhas em RDBMS
  - Podem incorporar outros documentos (subdocumentos)
  - Pode ter um ou mais campos
- Índices em MongoDB funcionam de forma semelhante aos dos RDBMS

CRUD

Create, Retrieve, Update, Delete

Comandos para CRUD

Create

- `db.collection.insertOne({name: “Max”, age: 19})`
- `db.collection.insert([ {name: “Max”, age: 19}, {name: “Sue”, age: 19} ])`

Read

- `db.collection.findOne({name: “Max”}) // Um documento`
- `db.collection.find({age: 19}) // Vários documentos`

Update

- `db.collection.findOneAndUpdate({name: “Sue”}, {$set: {age: 20}})`
- `db.collection.updateMany({}, {$set: age: 20}})`

Delete

- `db.collection.findOneAndDelete({name: “Sue”})`
- `db.collection.remove({age: 20})`

Mais exemplos de CRUD

```
db.user.insertOne ({
   first: "John",
   last: "Doe",
   age: 39
});
```

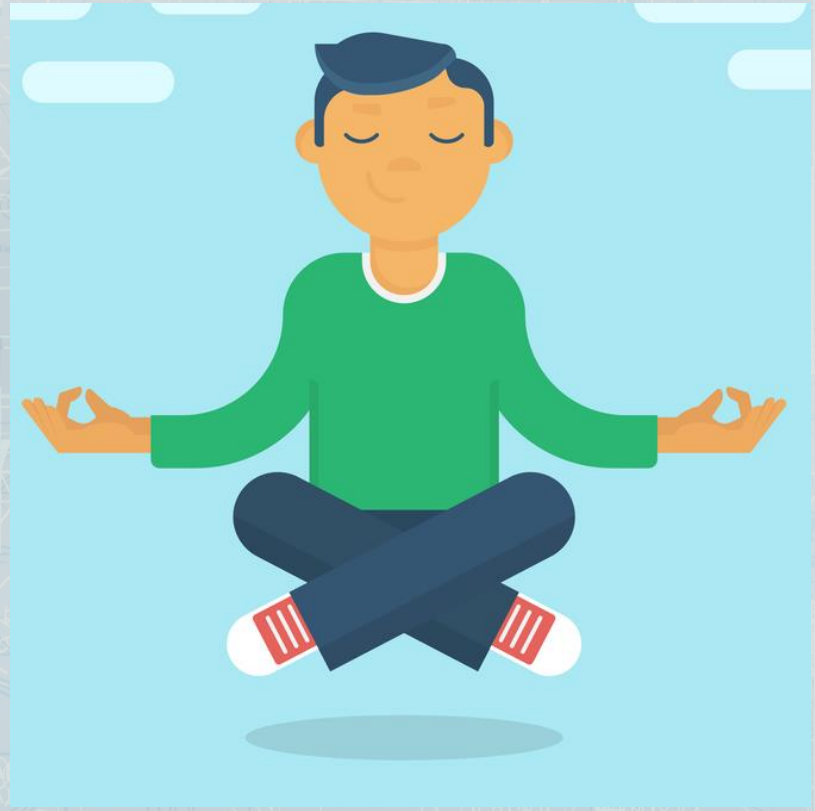
```
db.user.find ({
   "first": "John",
   "last": "Doe",
});
```

```
db.user.updateOne(
   {"_id" : ObjectId("51…")},
   {
      $set: {
         age: 40,
         salary: 7000
      }
   }
)
```

```
db.user.remove({
   "first": /^J/
})
```

(remove todos os documentos em que o campo "first" comece com "J")

Calma!



- **Não é necessário decorar os comandos do MongoDB**
  - Os *slides* seguintes mostrarão **exemplos**

- Embora iremos **praticar** alguns desses comandos durante algumas aulas, **não precisaremos usá-los diretamente** no *back-end* do Projeto Interdisciplinar que iremos implementar
  - No *back-end*, a interface com o MongoDB ficará a cargo da biblioteca **Prisma**, que abstrai a maioria dos comandos

INSERINDO INFORMAÇÕES

Inserção de informações

RDBMS: criação de tabela e inserção de registro	MongoDB: criação de coleção e inserção de documento
CREATE TABLE Teacher_Info(Teacher_id Varchar(10), Teacher_Name Varchar(10), Dept_Name Varchar(10), Salary Number(10), Status char(1), PRIMARY KEY (id));	db.createCollection(“Teacher_Info”);
INSERT INTO Teacher_Info (Teacher_id, Teacher_Name, Dept_Name, Salary, Status) VALUES (“Pic001", “Ravi”, “IT”, 30000, “A“);	db.Teacher_Info.insertOne({ Teacher_id: “Pic001", Teacher_Name: “Ravi", Dept_Name: “IT”, Sal:30000, status: "A" });

Exemplo de coleção no MongoDB

```
{
    "_id": ObjectId("4efa8d2b7d284dad101e4bc9"),
    "Last Name": "DUMONT",
    "First Name": "Jean",
    "Date of Birth": "01-22-1963"
},

{
    “_id": ObjectId("4efa8d2b7d284dad101e4bc7"),
    "Last Name": "PELLERIN",
    "First Name": "Franck",
    "Date of Birth": "09-19-1983",
    "Address": "1 Chemin des Loges",
    "City": "VERSAILLES"
}
```

RECUPERANDO INFORMAÇÕES

find() × SELECT

MongoDB	`db.users.find(` `  { age: { $gt: 18 } },` ← query criteria `  { name: 1, address: 1 }` ← projection `).limit(5)` ← cursor modifier (← collection)
SQL	`SELECT _id, name, address` ← projection `FROM users` ← table `WHERE age > 18` ← select criteria `LIMIT 5` ← cursor modifier

Exemplos (1)

Declarações SELECT SQL	Declarações find() do MongoDB
`SELECT * FROM Teacher_info;`	`db.Teacher_info.find()`
`SELECT * FROM Teacher_info WHERE sal = 25000;`	`db.Teacher_info.find( {sal: 25000})`
`SELECT Teacher_id FROM Teacher_info WHERE Teacher_id = 1;`	`db.Teacher_info.find( {Teacher_id: "pic001"})`

Exemplos (2)

Declarações SELECT SQL	Declarações find() do MongoDB
SELECT * FROM Teacher_info WHERE status != "A“;	db.Teacher_info.find({status:{$ne:"A"}})
SELECT * FROM Teacher_info WHERE status = "A" AND sal = 20000;	db.Teacher_info.find({ status:"A", sal:20000 })
SELECT * FROM Teacher_info WHERE status = "A" OR sal = 50000;	db.Teacher_info.find({ $or: [{ status: "A" } , { sal:50000 }] })
SELECT * FROM Teacher_info WHERE sal > 40000	db.Teacher_info.find({ sal: { $gt: 40000 } })
SELECT * FROM Teacher_info WHERE sal < 30000	db.Teacher_info.find({ sal: { $lt: 30000 } })

Exemplos (3)

Declarações SELECT SQL	Declarações find() do MongoDB
SELECT * FROM Teacher_info WHERE status = "A" ORDER BY SAL ASC	db.Teacher_info.find({ status: "A" }).sort({ sal: 1 })
SELECT * FROM users WHERE status = "A" ORDER BY SAL DESC	db.Teacher_info.find({ status: "A" }).sort({sal: -1 })
SELECT COUNT(*) FROM Teacher_info;	db.Teacher_info.count() **OU** db.Teacher_info.find().count()
SELECT DISTINCT(Dept_name) FROM Teacher_info;	db. Teacher_info.distinct(“Dept_name")

ATUALIZANDO INFORMAÇÕES

Exemplos

Declarações UPDATE SQL	Declarações update() do MongoDB
`UPDATE Teacher_info SET Dept_name = “ETC" WHERE sal > 250000`	`db.Teacher_info.update( { sal: { $gt: 25000 } }, { $set: { Dept_name: “ETC" } }, { multi: true } )`
`UPDATE Teacher_info SET sal = sal + 10000 WHERE status = "A"`	`db.Teacher_info.update( { status: "A" } , { $inc: { sal: 10000 } }, { multi: true } )`

EXCLUINDO INFORMAÇÕES

Exemplos

Declarações DELETE SQL	Declarações remove() do MongoDB
DELETE FROM Teacher_info WHERE Teacher_id = “pic001"	db.Teacher_info.remove({ Teacher_id: "pic001“ });
DELETE FROM Teacher_info;	db.Teacher_info.remove({})

Conclusão

- Em suma, o MongoDB
  - é um banco de dados NoSQL
  - é rápido, muito rápido
  - é simples de utilizar, especialmente em sua versão em nuvem
  - é baseado em JSON
  - possui uma API clara e poderosa
  - é facilmente escalável
  - encaixa-se no Teorema CAP como um sistema CP (consistência + tolerância a partição)
    - mesmo assim, devido ao seu esquema de partição, oferece boa disponibilidade

Para saber mais

- BOAGLIO, Fernando. **MongoDB**: Construa novas aplicações com novas tecnologias. São Paulo: Casa do Código, 2015
- PANIZ, D. **NoSQL**: Como armazenar os dados de uma aplicação moderna. Casa do Código, 2016, p. 9-42