

## Banco de Dados II

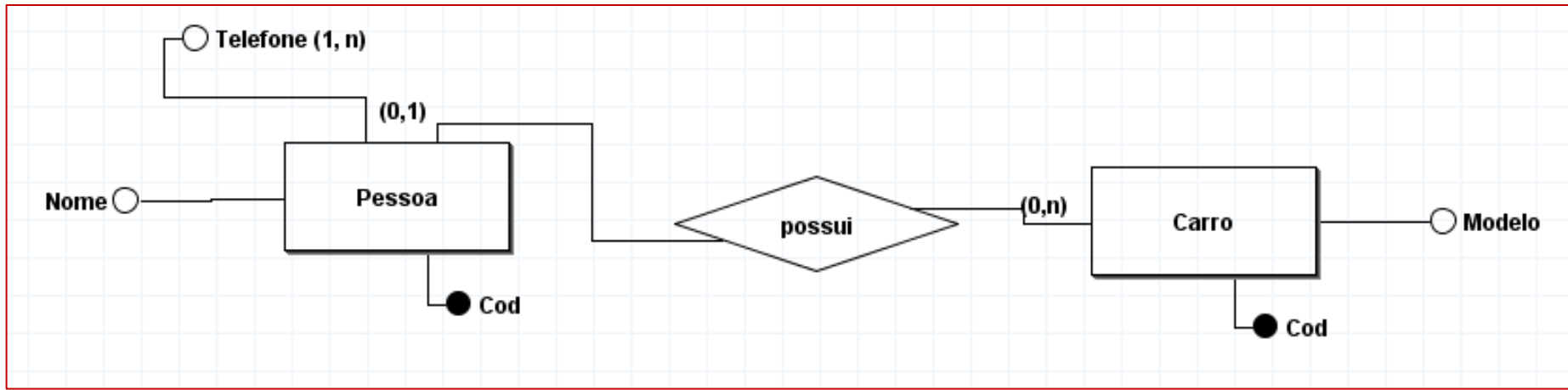
### Projeto de BD baseado em documentos e relacionamentos

Profa. Damires Souza

[damires@ifpb.edu.br](mailto:damires@ifpb.edu.br)



# Como implementar relacionamentos no MongoDB?



# Relacionando documentos

| Implementação                            | Documentos   |
|--|--|
| <b>Referência</b>                        | <pre>{   _id: "88456707830",   nome: "Webber",   telefone: "999999999",   carro: objectId("9BWHE21JX24060960") } {   _id: "9BWHE21JX24060960",   modelo: "gol 2019",   preço: 38.000,00 }</pre>                                      |
| <b>Aninhamento<br/>(<i>embedded</i>)</b> | <pre>{   _id: "76052657278",   nome: "Sidartha",   telefones: [     {telefone: "999957211"},     {telefone: "988081046"}   ],   carro: {     NVI: "C021E21JX240LP925",     modelo: "fiesta 2019",     preço: "25.000,00"   } }</pre> |

# Princípio no Mongodb

---

- Dados estão normalmente **desnormalizados** ou armazenados com **dados embutidos** em documentos para **remover a necessidade de junções**
- Pode-se, em alguns casos, referenciar documentos de diferentes coleções
  - \*\* Apesar das referências, o mongoDB **não** garante integridade referencial

<https://docs.mongodb.com/manual/reference/database-references/>



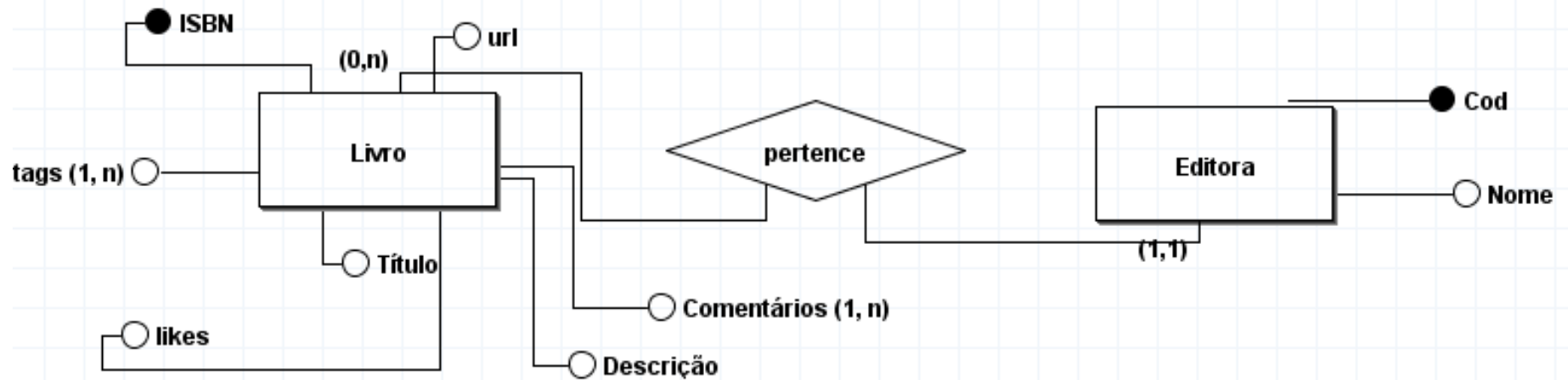
# Referências

---

- Pode ser feita salvando o campo **\_id** de um documento em outro documento como uma **referência**
- A aplicação pode executar **uma segunda consulta** para retornar os dados relacionados
  - **Ou pode-se usar o método \$lookup**
- Referências fornecem mais **flexibilidade** do que a incorporação ou aninhamento de documentos
  - ▶ **No entanto, algumas vezes, as aplicações devem emitir mais consultas** para resolver as referências



# Exemplo – Livro e Editora



# Coleção Livro

```
db.getCollection('livro').find({})
```

livro 0.002 sec.

| Key  | Value                                | Type     |
|--|--------------------------------------|----------|
| ▼ (1) ObjectId("637b73b0c255db481fdce783") | { 8 fields }                         | Object   |
| _id  | ObjectId("637b73b0c255db481fdce783") | ObjectId |
| titulo                                     | MongoDB2 – Como usar?                | String   |
| descricao                                  | MongoDB – Como                       | String   |
| by   | MongoDBExpert                        | String   |
| url  | http://www.mongodbexpert.com         | String   |
| ▼ tags                                     | [ 4 elements ]                       | Array    |
| [0]  | mongodb                              | String   |
| [1]  | database                             | String   |
| [2]  | NoSQL                                | String   |
| [3]  | Document                             | String   |
| likes                                      | 100.0                                | Double   |
| ▼ comentarios                              | [ 2 elements ]                       | Array    |
| > [0]                                      | { 4 fields }                         | Object   |
| > [1]                                      | { 4 fields }                         | Object   |
| ▼ (2) ObjectId("63971d040321ca93a7999775") | { 4 fields }                         | Object   |
| _id  | ObjectId("63971d040321ca93a7999775") | ObjectId |
| title                                      | Testando Refs                        | String   |
| url  | http://www.mongodbexpert.com         | String   |
| editora                                    | ObjectId("63971cc10321ca93a7999774") | ObjectId |

# Exemplo

```
db.editora.insertOne(  
  {nome: "Mágica",  
   cidade: "João Pessoa"} )
```

```
db.getCollection('editora').find({})
```

editora 0.008 sec.

```
/* 1 */
```

```
{  
  "_id" : ObjectId("62b9ab4d13bb1ee1abd95bf8"),  
  "nome" : "Mágica",  
  "cidade" : "João Pessoa"  
}
```

```
db.livro.insertOne(  
  {title:"Testando Refs entre coleções",  
   url:"http://www.mongodbexpert.com",  
   editora:ObjectId("60a2781a0044ab8859767c74")})
```

ID da  
Editora



# Exemplo

```
db.getCollection('editora').find({})
```

editora 0.008 sec.

```
/* 1 */
```

```
{
  "_id" : ObjectId("62b9ab4d13bb1ee1abd95bf8"),
  "nome" : "Mágica",
  "cidade" : "João Pessoa"
}
```

```
/* 2 */
```

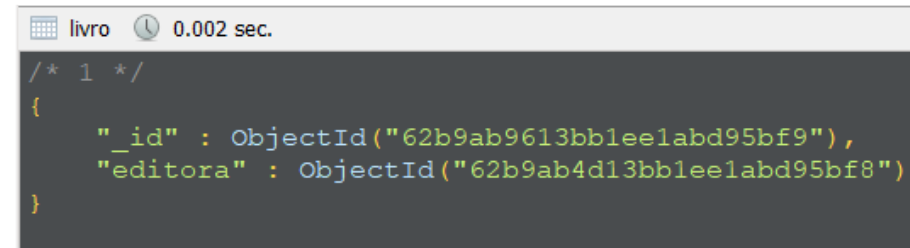
```
{
  "_id" : ObjectId("62b9ab9613bb1ee1abd95bf9"),
  "title" : "Testando Refs",
  "url" : "http://www.mongodbexpert.com",
  "editora" : ObjectId("62b9ab4d13bb1ee1abd95bf8")
}
```

# Para consultar

---

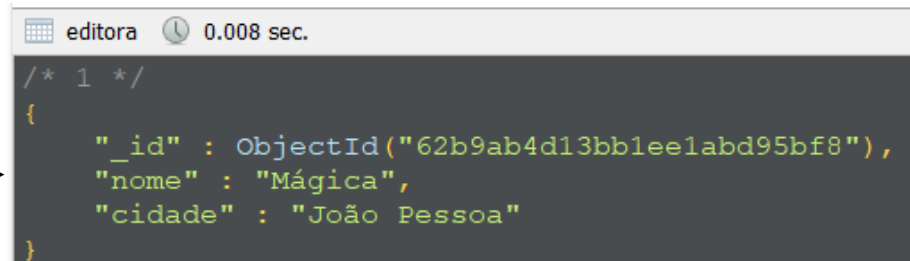
- Usando o **ObjectId**, pode-se consultar os documentos

`db.livro.findOne({"title":"Testando Refs entre Coleções"},"editora":1})`



```
livro 0.002 sec.
/* 1 */
{
  "_id" : ObjectId("62b9ab9613bb1ee1abd95bf9"),
  "editora" : ObjectId("62b9ab4d13bb1ee1abd95bf8")
}
```

`db.editora.findOne({"_id":{"$in":[ObjectId("62b9ab4d13bb1ee1abd95bf8")]}})`



```
editora 0.008 sec.
/* 1 */
{
  "_id" : ObjectId("62b9ab4d13bb1ee1abd95bf8"),
  "nome" : "Mágica",
  "cidade" : "João Pessoa"
}
```

# \$Lookup

---

## **\$lookup => JOIN**

- Realiza a busca em duas coleções diferentes e **agrega** os documentos que possuem **o mesmo identificador**
  - As coleções devem residir no mesmo database

```
{ $lookup: {  
  from: <collection to join>,  
  localField: <field from the input documents>,  
  foreignField: <field from the documents of the "from" collection>,  
  as: <output array field> } }
```



# Exemplo

```
db.livro.aggregate([
  {$lookup:
    {
      from: "editora",
      localField: "editora",
      foreignField: "_id",
      as: "Editora do livro"
    }
  },
  {$match:
    {"Editora do livro.cidade": "João Pessoa"}
  }
])
```

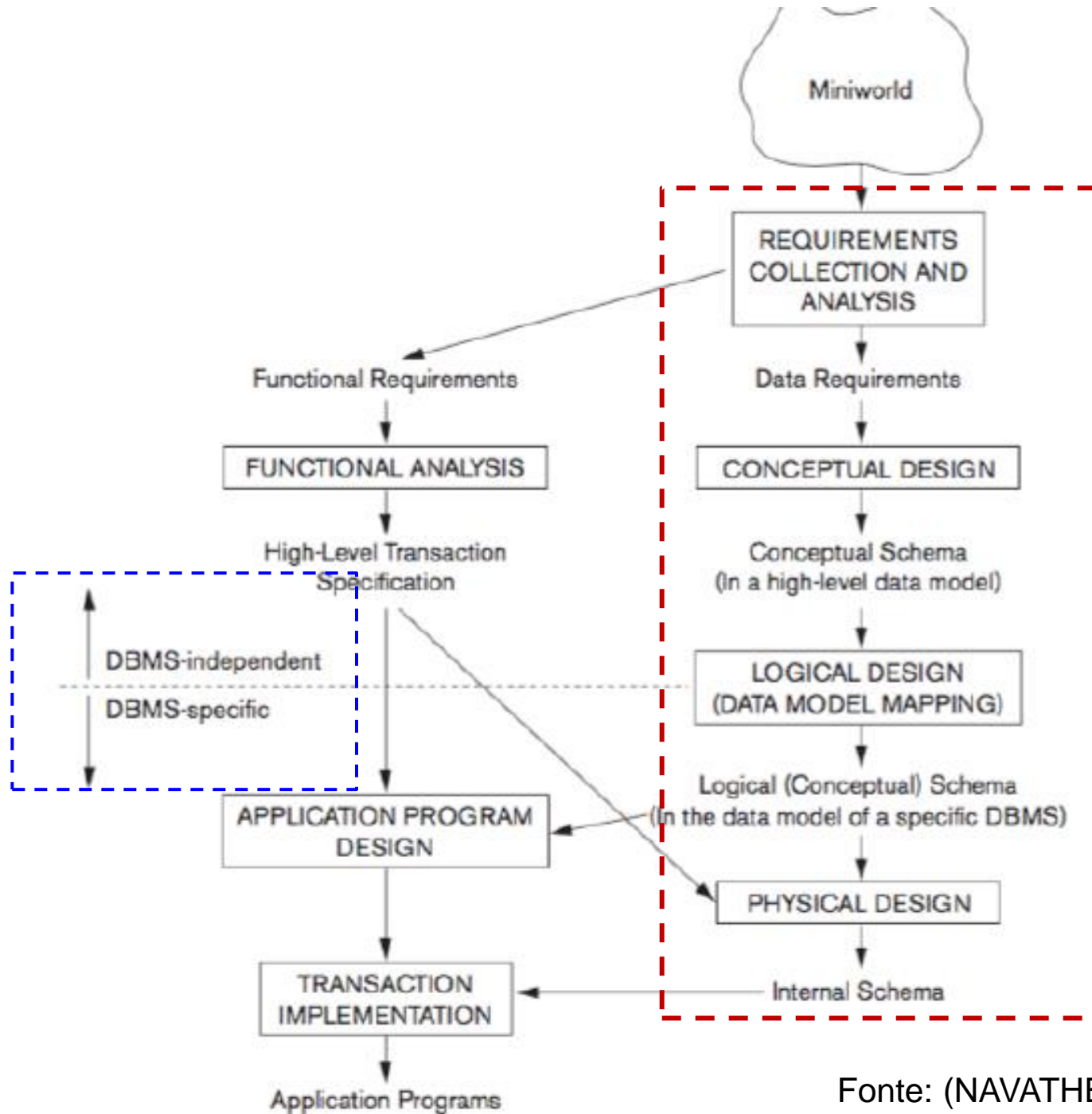
```
/* 1 */
{
  "_id" : ObjectId("62b9ab9613bb1ee1abd95bf9"),
  "title" : "Testando Refs",
  "url" : "http://www.mongodbexpert.com",
  "editora" : ObjectId("62b9ab4d13bb1ee1abd95bf8"),
  "Editora do livro" : [
    {
      "_id" : ObjectId("62b9ab4d13bb1ee1abd95bf8"),
      "nome" : "Mágica",
      "cidade" : "João Pessoa"
    }
  ]
}
```



---

Como  
projetar?



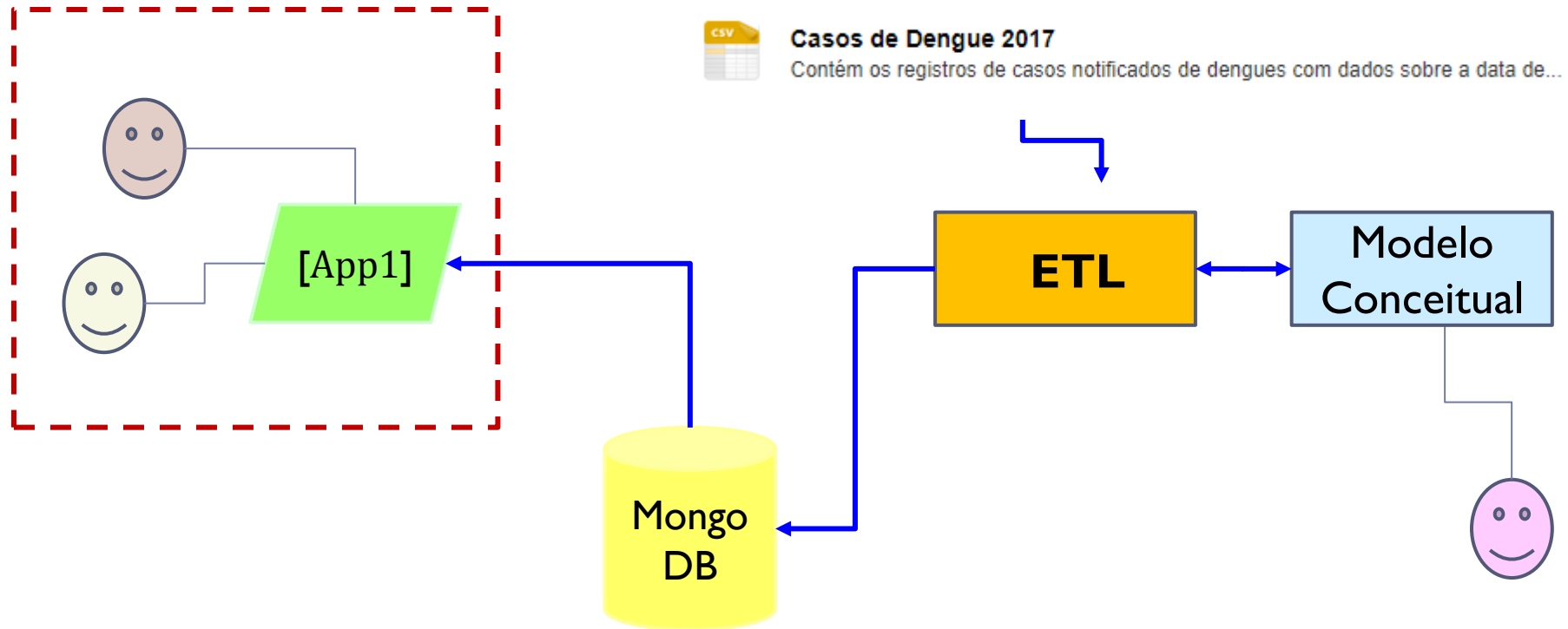


# Um exemplo de aplicação real

## Casos de Dengue, Zika e Chikungunya

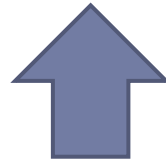
Registro dos casos de Dengue, Zika e Chikungunya com registros nas unidades de saúde, públicas ou particulares.

CSV JSON



<http://dados.recife.pe.gov.br/dataset/casos-de-dengue-zika-e-chikungunya>

Modelar uma estrutura/classe/entidade para o MongoDB  
é **mais simples** do que uma modelagem para BDR,  
porém esse trabalho pode ficar **complicado**  
**se pensarmos de forma relacional**



Trabalhar em **nível**  
**conceitual** de modelagem de  
BD **ajuda muito**





# Como faremos???

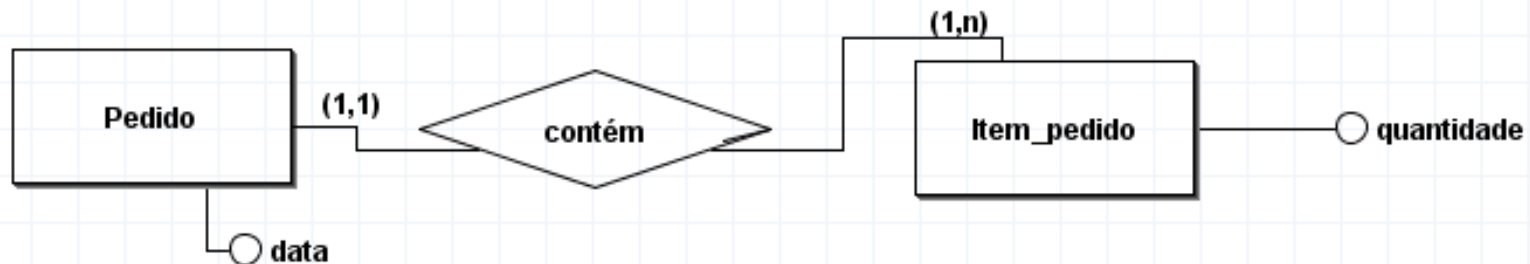
---



- I. Como a aplicação precisa/vai usar os dados?
- II. A aplicação precisa de CRUD?
- III. Quais as características dos dados?
- IV. Os dados devem estar agrupados ou normalizados?
- V. Quais as principais entidades/classes?

# Conceito de Agregado

- ▶ **Unidade de acesso e de distribuição**, tornando-se possível trabalhar com um **objeto que contém outros dados/objetos relacionados**.
- ▶ Autossuficiente
- ▶ **Contém toda a informação que se possa precisar**
  - ▶ Evitando a necessidade de JOINS/equivalentes na hora de fazer queries.



# Um exemplo

SUFPC045.15 Consulta: Pedidos de Compra / Item 12/05/2021 18:53

---

**Pesquisa**

Cia 11351 STG - LOJA TESTE

Filial

Item 1796797 Smartphone Samsung Galaxy A30s 64GB Dual Chip Android 9.0 CUN Todos

Emissão a Entrega de a Situação Aberto

Pesquisar

---

**Ped. Compra**

| Pedido | # | Pr. Unitário | Qt. Pedida | Qt. Entregue | Qt. Canc. | Qt. Liq. | Qt. Saldo | Dt. Entrega | Situação   |
|--------|---|--------------|------------|--------------|-----------|----------|-----------|-------------|------------|
| 29     | 1 | 800.0000     | 1          | 0            | 0         | 0        | 1         | 12/05/2021  | Aberto     |
| 30     | 1 | 2,100.0000   | 1          | 0            | 0         | 0        | 1         | 12/05/2021  | Aberto     |
| 31     | 1 | 2,980.0000   | 1          | 0            | 0         | 0        | 1         | 12/05/2021  | Aberto     |
| 34     | 1 | 2,450.0000   | 1          | 0            | 0         | 0        | 1         | 31/05/2021  | Aberto     |
| 35     | 1 | 2,450.0000   | 2          | 0            | 0         | 0        | 2         | 12/05/2021  | Aberto     |
| 37     | 1 | 2,450.0000   | 1          | 0            | 0         | 0        | 1         | 12/05/2021  | Aberto     |
|        |   |              |            |              |           |          |           |             |            |
|        |   |              |            |              |           |          |           |             |            |
| Total  |   |              | 7          |              |           |          | 7         | Dt. Sit.    | 12/05/2021 |

Fornecedor SINGER SINGER DO BRASIL INDUSTRIA E COM LTDA

Pr. Unit. Orig. 800.0000 Qt. Excesso 0 Fat. Futura 0 Cotação

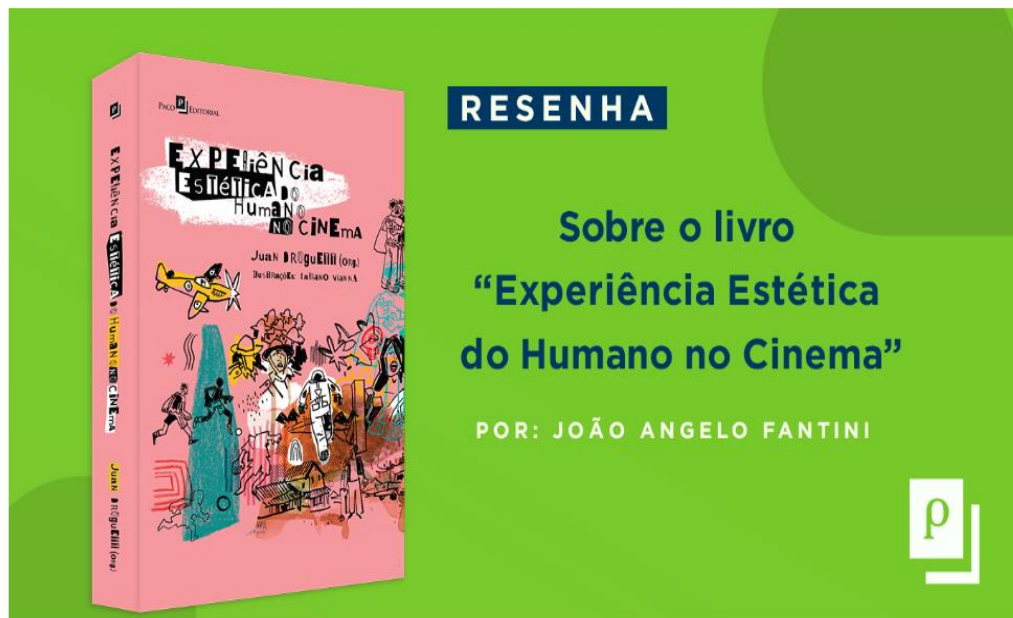
Agenda Horário Doca

Observação Gerado Manual

Descrição Smartphone Samsung Galaxy A30s 64GB Dual Chip Android 9.0 T Aprovação Aprovado

Pedido Requisição Recebimento

# Voltando a Livros: Aplicação exemplo



CONTEÚDOS PACO, LANÇAMENTOS, RESENHAS DE LIVROS

## Resenha sobre o livro “Experiência Estética do Humano no Cinema”

30 de março de 2021 / o comentários

**C**onfira a resenha escrita por João Angelo Fantini sobre o livro "Experiência Estética do Humano no Cinema" organizado pelo autor Juan Droguett e publicado pela Paco Editorial em 2021.



CATEGORIAS

Artigos

Conteúdos Paco

Cultura

Ebook

Imprensa

Institucional

Lançamentos

Notas

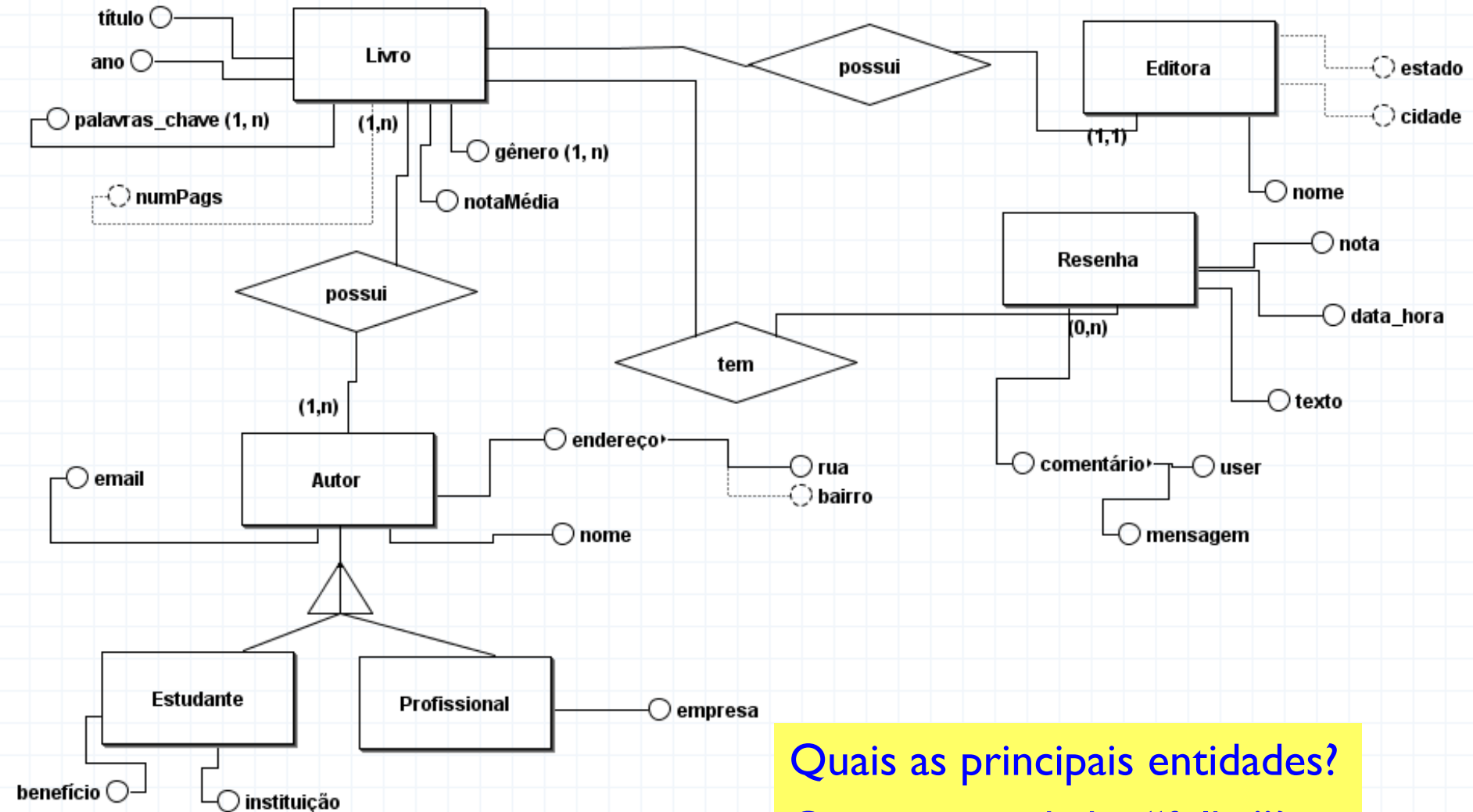
Notícias Acadêmicas

Paco na Imprensa

Publique obra acadêmica

Publique obra de literatura

# Aplicação Exemplo: Modelo Conceitual



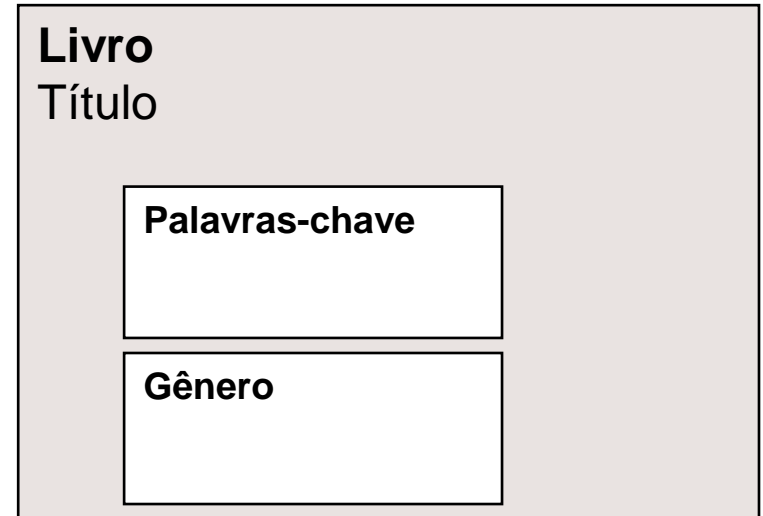
Quais as principais entidades?

Quais as entidades “folha”?

# Regras de Mapeamento

---

- **Analisar:**
  - Entidades principais
  - Entidades folhas
  - Tipos de atributos
  - Hierarquias
  - Relacionamentos e nível de agregação
  - Verificar autonomia das entidades



# Regras de Mapeamento

---

- **Classe/entidade => Coleção**
  - ▶ Livro -> coleção de livros
- Instância => **documento** da coleção
  - ▶ Livro de Harry Potter é documento da coleção de livros
- **Atributo simples => campo simples**
  - ▶ Titulo -> titulo
- ▶ Atributo opcional => **campo opcional**



# Regras de Mapeamento

---

- Atributo multivalorado (lista) => **array** de elementos
  - ▶ Palavra-chave -> array de palavras chaves
- Atributo composto => **documento/objeto embutido**
- Relacionamento com agregação/composição => [array de] **documento/objeto embutido**
  - ▶ Livro “possui” editora; editora fica agregada a livro





# Regras de Mapeamento

---

- Relacionamentos **1:1** ou **1:N** => [array de] subdocumento/objeto embutido
- Relacionamentos **N:N** =>
  - array de subdocumentos/objetos embutidos dos dois lados
  - ou
  - referências

Livro possui autor???



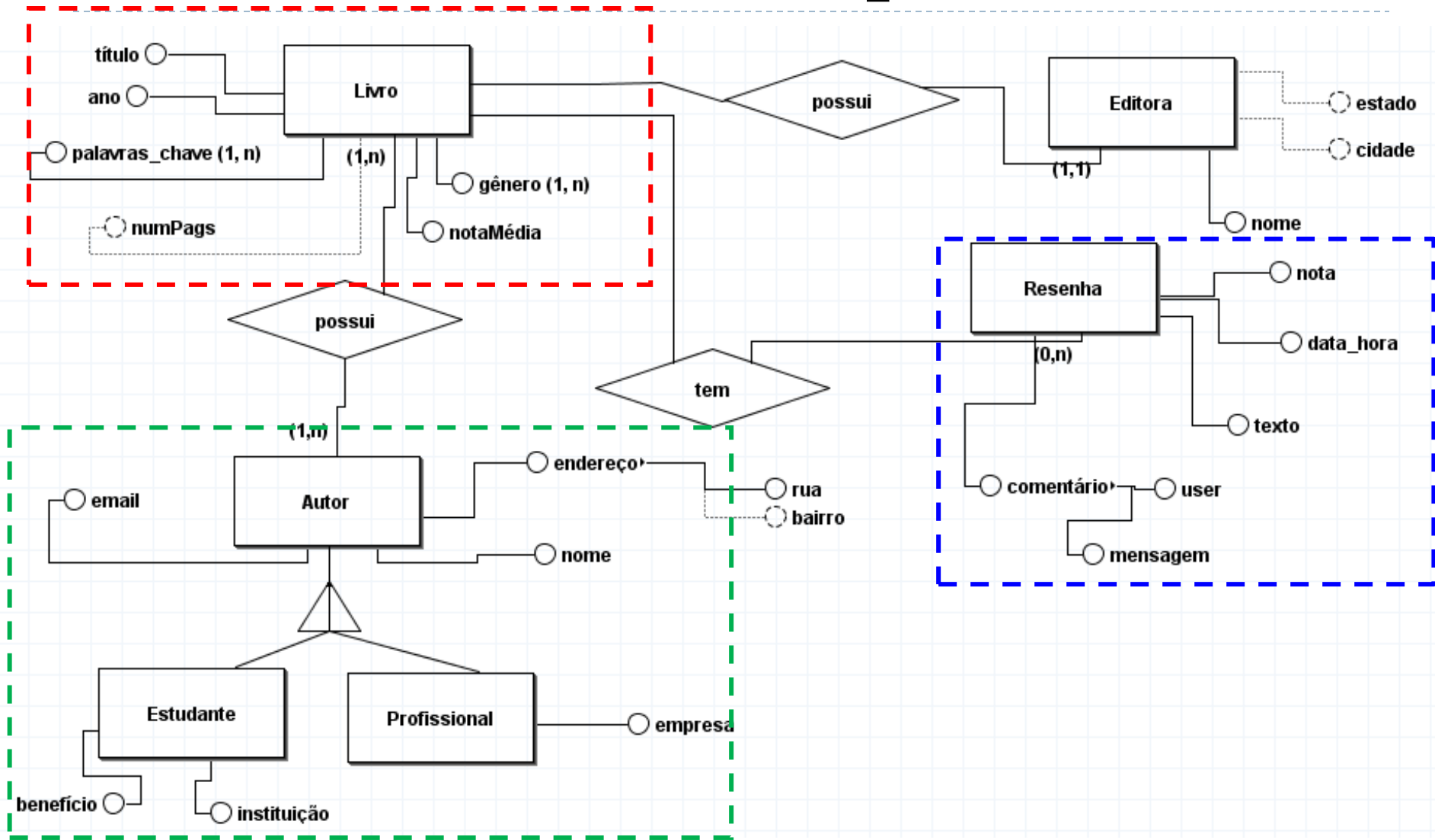
# Regras de Mapeamento

---

- **Generalização** com ênfase na **superclasse** => **uma coleção de documentos**
  - ▶ Cada documento pode ser referente a uma instância de subclasse
  - ▶ Os atributos específicos das subclasses são mapeados como atributos opcionais
- **Generalização** com ênfase nas **subclasses** => **pode separar em coleções de documentos**
  - ▶ Cada documento terá atributos específicos das subclasses
  - ▶ **Opção menos desejada...**



# Voltando ao exemplo



# Entidade Livro

---

- **Principal Entidade/classe**
- Resolver 1:N => Livro + Palavras-Chave + Gênero

```
db.livro.insertOne(  
  {titulo:"Sistemas de Bancos de Dados",  
    palavras_chave:["SGBD","relacional","NoSQL"],  
    ano: 2017,  
    gênero: ["Computação","Projeto"],  
    notaMédia: 4.0  
  })
```

# Livro + Editora + Autor

---

```
db.livro.insertOne(  
  {titulo:"Projeto de Banco de Dados",  
    palavras_chave:["SGBD","relacional"],  
    ano: 2015,  
    gênero: ["Computação","Projeto"],  
    notaMédia: 4.0,  
    editora: "Pearson",  
    autor: [  
      {nome:"Carlos Heuser", email: "ch@example.com"},  
      {nome:"Carlos Heuser Jr", email: "chj@example.com"}  
    ]  
  })
```

---



# Autor + Livro

---

```
db.autor.insert (  
  {nome: "Carlos Heuser ",  
   email: "ch@example.com",  
   livro: [  
     {num:1, titulo:"Projeto de Banco de Dados"},  
     {num:2, titulo:"Projeto de NoSQL"}  
   ]})
```



# Com referência

---

```
db.livro.insertOne(
```

```
{título:"Conde de Monte Cristo2",
```

```
  numpags: 1663,
```

```
  ano: 2012,
```

```
  palavraschave:["revolução","vingança"],
```

```
  gênero: ["drama", "literatura"],
```

```
  notamedia: 4,
```

```
  editora: "Zahar",
```

```
  autor: [
```

```
    {"01":
```

```
      ObjectId("62b9ab4d13bb1ee1abd95bf8")    }]])
```

# Resenha

---

```
db.resenha.insertOne(  
  {  
    livro: ObjectId("62b9ab9613bb1ee1abd95bf9"),  
    data: new Date (2022,11,21),  
    texto: "Adorei, sensacional",  
    comentário: [ { user:'Mario', mensagem:'Poderia  
ser melhor'}, { user:'Rebeca',  
mensagem:'Excelente'}],  
    nota: 5  
  }  
)
```





# Classe/entidade Autor

```
db.autor.insert (  
  {nome: "Joana Brandão"  
  ,  
    email:  
    "jb@example.com",  
    livro: [  
      {num:1, titulo:"Projeto  
de Banco NoSQL"}],  
    beneficio: 1.000,  
    instituição: "IFPB"  
  })
```

Estudante

```
db.autor.insert (  
  {nome: "Mariana Chaves"  
  ,  
    email:  
    "mc@example.com",  
    livro: [  
      {num:1, titulo:"Projeto  
de Banco NoSQL"}],  
    empresa: "XXXX"  
  })
```

Profissional

# Dicas

---

- i. Projete seu esquema de acordo com as necessidades dos usuários e da aplicação.
- ii. Combine os objetos em um documento, se você vai usá-los juntos.
  - i. Caso contrário separe-os, mas veja como serão acessados.
- iii. Duplique os dados (com limites), se for interessante, porque melhora a performance de acesso.

**A estrutura obtida e usada fica bem mais próxima da aplicação e da realidade ???**



# Referências

---

- ▶ BANERJEE, S. et al. Towards Logical Level Design of Big Data. In: IEEE 13<sup>th</sup> International Conference on Industrial Informatics (INDIN). [S.l.: s.n.], 2015. p. 1665 –1671.
- ▶ ELMASRI, R.; NAVATHE, S. B. Sistemas de Banco de Dados. Pearson Education, 2017.
- ▶ BUGIOTTI, F. et al. Database Design for NoSQL Systems. In: International Conference on Conceptual Modeling. [S.l.: s.n.], 2014. p. 223 – 231.
- ▶ SILVA, I. 2017. Um Modelo Conceitual de Dados e uma Ferramenta CASE para Aplicações de Persistência Poliglota. Dissertação de Mestrado, UFPE.



# Referências

---

- ▶ Freitas, M.C., Souza, D.Y., Salgado, A.C.: Conceptual Mappings to Convert Relational into NoSQL Databases. In: ICEIS (I). pp. 174–181 (2016)Lima, C. (2016).
- ▶ Projeto Lógico de Bancos de Dados NoSQL Documento a partir de Esquemas Conceituais Entidade-Relacionamento Estendido (EER). Dissertação de Mestrado, UFSC.
- ▶ Jihane Mali, Faten Atigui, Ahmed Azough, Nicolas Travers. ModelDrivenGuide: An Approach for Implementing NoSQL Schemas. International Conference, DEXA 2020, Sep 2020, Bratislava, Slovakia. pp.141-151.

