

MongoDB,

uma introdução

Live de Python #174



1. SQL e Documentos?

Estruturado ou não estruturado?

2. Bancos de Documentos

Uma visão geral sobre
semi-estruturados

3. MongoDB

Orientado a documentos

4. Mão na massa

Vimos aqui pra isso, não é mesmo?



picpay.me/dunossauro



apoia.se/livedepython



PIX



Ajude o projeto



Ademar Peixoto, Alex Lima, Alex Lopes, Alexandre Harano, Alexandre Santos, Alexandre Tsuno, Alexandre Villares, Alynne Ferreira, Alysson Oliveira, Amaziles Carvalho, André Rocha, Arnaldo Turque, Artur Zalewska, Bruno Batista, Bruno Oliveira, Caio Nascimento, Carlos Chiarelli, Cleber Santos, César Almeida, Davi Ramos, David Kwast, Diego Guimarães, Diego Ubirajara, Dilenon Delfino, Donivaldo Sarzi, Elias Soares, Eric Niens, Eugenio Mazzini, Everton Alves, Fabiano Gomes, Fabio Barros, Fabio Castro, Fabrícia Diniz, Flavkaze Flavkaze, Francisco Alencar, Franklin Silva, Fábio Serrão, Gabriel Simonetto, Gabriel Soares, Gabriela Santiago, Geandreson Costa, Guilherme Castro, Guilherme Felitti, Guilherme Ostrock, Henrique Machado, Israel Fabiano, Italo Silva, Johnny Tardin, Jonatas Leon, Jonatas Oliveira, Jorge Plautz, Jose Mazolini, José Prado, João Lugão, João Schiavon, Juan Gutierrez, Jônatas Silva, Júlia Kastrup, Kaio Peixoto, Kaneson Alves, Leonardo Cruz, Leonardo Galani, Leonardo Mello, Lidiane Monteiro, Lorena Ribeiro, Lucas Barros, Lucas Mello, Lucas Mendes, Lucas Teixeira, Lucas Valino, Luciano Ratamero, Maiquel Leonel, Marcela Campos, Marcelo Rodrigues, Maria Clara, Marina Passos, Matheus Vian, Natan Cervinski, Nicolas Teodosio, Osvaldo Neto, Patric Lacouth, Patricia Minamizawa, Patrick Gomes, Paulo Tadei, Pedro Pereira, Peterson Santos, Rafael Lino, Reinaldo Silva, Renan Moura, Revton Silva, Rodrigo Ferreira, Rodrigo Mende, Rodrigo Vaccari, Ronaldo Silva, Sandro Mio, Silvio Xm, Thiago Araujo, Thiago Borges, Thiago Bueno, Tyrone Damasceno, Vinícius Bastos, Vítor Gomes, Wendel Rios, Wesley Mendes, Willian Lopes, Willian Rosa, Wilson Duarte, Yury Barros, Érico Andrei



Obrigado você



RECEBEMOS DE Empresa Teste LTDA. OS PRODUTOS CONTANTES DA NOTA FISCAL INDICADA AO LADO		NFe Nº 000175 Série 1
Data de recebimento	Identificação e assinatura do receptor	

 Bling www.bling.com.br	DANFE Documento Auxiliar da Nota Fiscal Eletrônica		Controle do Fisco 
	1-Entrada	2	
	2-Saída		
	Nº 000175 SÉRIE: 1 Página: 1 de 1		

Natureza da operação Venda de mercadorias			Número de protocolo de autorização de uso da NFe DOCUMENTO SEM VALOR FISCAL
Inscrição Estadual	Inscr. est. do subst. trib.	CNPJ	Chave de acesso da NFe - consulta no site: www.nfe.fazenda.gov.br 43.0908.90.627.936/0001-30-55-001-000.000.175-000.896.536-

Destinatário/Remetente

Nome / Razão Social Dionísio de Baco	CNPJ 111.111.111-11	Inscrição Estadual 010/000000	Data emissão 07/08/2009
Endereço Rua dos Vinhedos, 386	Bairro Vinhedos	CEP 95.700-000	Data saída 07/08/2009
Município Bento Gonçalves	Fone/Fax 3454-6877	UF RS	Hora saída 16:01

Faturas

Número	Vencimento	Valor	Número	Vencimento	Valor	Número	Vencimento	Valor
000175/1	06/09/2009	76,66	000175/2	06/10/2009	76,67	000175/3	05/11/2009	76,67

Cálculo do imposto

Base de cálculo do ICMS	Valor do ICMS	Base de cálculo do ICMS Subst.	Valor do ICMS Subst.	Valor total dos produtos
230,00	27,60	0,00	0,00	230,00
Valor do frete	Valor do seguro	Desconto	Outras despesas acessórias	Valor do IPI
0,00	0,00	0,00	0,00	0,00
Valor total da nota				230,00

SQL

VS

Doc

Como pensamos
em bancos de
dados?

Como pensamos em bancos de dados?



Geralmente, quando pensamos em bancos de dados, pensamos em estruturas como tabelas:

Nome	Email	Telefone
Eduardo	eu@dunossauro.live	999999999999
Fausto	fausto@dunossauro.live	998888888888

Existe algum problema em pensar assim?



Literalmente, **NÃO**. Porém, esse modelo, em alguns momentos, não representa os dados como gostaríamos ou não oferece a flexibilidade necessária.

Um problema comum!



Se em um determinado dia, tivermos que armazenar dois telefones, como faríamos?

Nome	Email	Telefone
Eduardo	eu@dunossauro.live	999999999999
Fausto	fausto@dunossauro.live	998888888888

Um problema comum!




Se em um determinado dia, tivermos que armazenar dois telefones, como faríamos?

Nome	Email	Telefone	Telefone 2
Eduardo	eu@dunossauro.live	999999999999	NULL
Fausto	fausto@dunossauro.live	998888888888	997777777777

Um problema comum!



Uma solução tradicional



ID	UID	Telefone	Telefone 2
1	1	999999999999	NULL
2	1	998888888888	997777777777

ID	Nome	Email
1	Eduardo	eu@dunossauro.live
2	Fausto	fausto@dunossauro.live

Um problema comum!

ID	Nome	Email
1	Eduardo	eu@dunossauro.live
2	Fausto	fausto@dunossauro.live

ID	UID	Rua	Número	CEP
1	1	Avenida ..	42	33333333
2	2	Rua ...	982	22222222

ID	UID	Cargo	Departamento
1	1	1	1
2	2	2	5

ID	UID	Telefone	Telefone 2
1	1	999999999999	NULL
2	2	998888888888	997777777777

ID	Cargo
1	QA
2	Dev

O que nós queríamos
representar mesmo?



Uma pessoa que usa o sistema!



```
1  @dataclass
2  class Pessoa:
3      nome: str
4      telefone: str
5      telefone_comercial: str
6      departamento: str
7      cargo: str
8      logradouro: str
9      número: int
10     cep: str
11
```

```
1  eduardo = Pessoa(
2      'Eduardo',
3      '999999999999',
4      '888888888888',
5      'P&D',
6      'QA',
7      'Rua dos bobos',
8      0,
9      '33333333'
10 )
```

A complexidade para construir



Em um mundo baseado em web, onde esses dados vão ser requeridos via API, essa é a complexidade média de uma requisição.

```
1  SELECT
2  pessoa.nome,
3  pessoa.email,
4  telefone.pessoal,
5  telefone.comercial,
6  enderecos.logradouro,
7  enderecos.numero,
8  enderecos.cep,
9  ...
10 FROM pessoas
11 JOIN telefones
12 ON telefones.uid == pessoa.id
13 JOIN enderecos
14 ON enderecos.uid == pessoa.id
15 ...
```

O retorno que gostaríamos



Quando no final a representação esperada para transicionar na web é um JSON ou um ***DOCUMENTO***

```
1  # eduardo.__dict__
2  {'nome': 'Eduardo',
3   'telefone': '999999999999',
4   'telefone_comercial': '888888888888',
5   'departamento': 'P&D',
6   'cargo': 'QA',
7   'logradouro': 'Rua dos bobos',
8   'número': 0,
9   'cep': '33333333'}
```


Banco de Docu mentos

Uma visão geral
sobre dados
semi-estruturados

Orientado a documentos

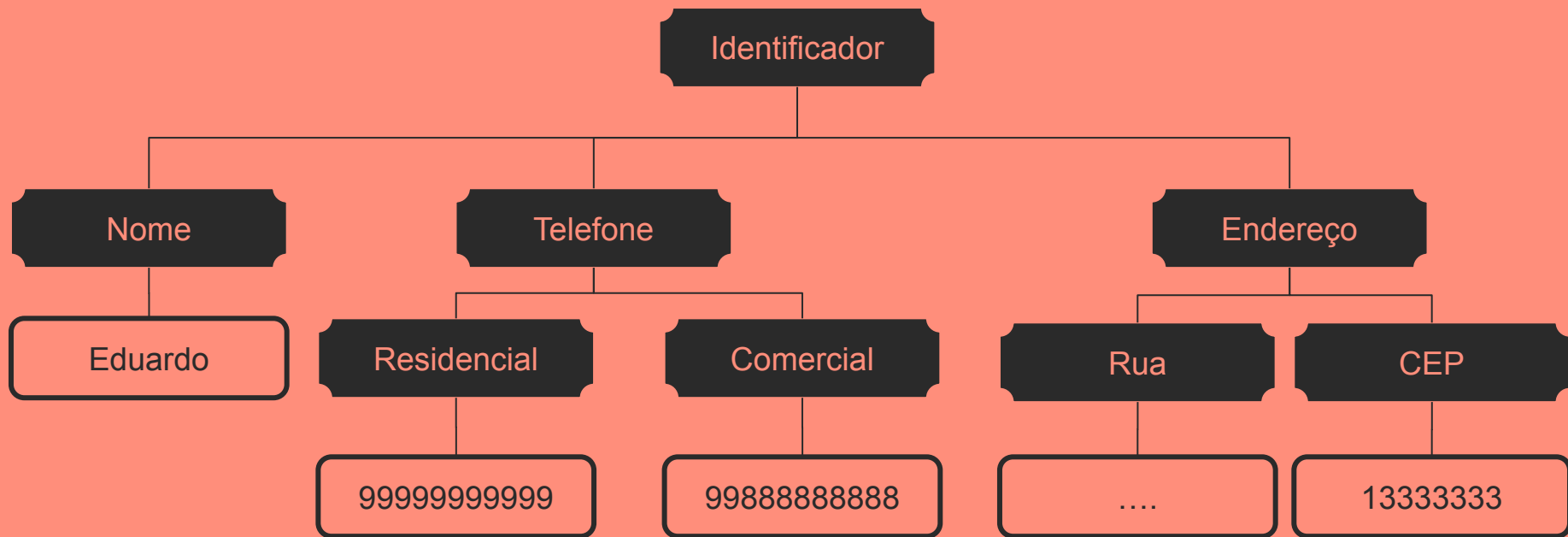


Normalmente pensamos em tabelas e esquemas de banco.

Tabelas são "inflexíveis", pois a estrutura tem que ser sempre mantidas, o que mantém uma "estrutura" padrão.

A ideia dos documentos é ser mais flexível.

Dados semi-estruturados



Tipos semiestrutturados



- Json
- Bson
- XML
- YAML
- ...

Json



JavaScript Object Notation é talvez a forma mais popular que temos como exemplos de dados semiestruturados.

```
{  
  "field": "value",  
  "chave": "valor",  
  "nome": "Eduardo"  
}
```

```
{  
  "nome": "Eduardo",  
  "telefones": ["9999999999", "888888888"]  
  "endereço": {  
    "logradouro": "Rua dos bobos",  
    "numero" : 0  
  }  
}
```

```
{  
  "nome": "Eduardo",  
  "telefones": ["9999999999", "888888888"] # array  
  "endereço": { # sub documento  
    "logradouro": "Rua dos bobos",  
    "numero" : 0  
  }  
}
```

MongoDB

Um banco de
dados orientado a
documentos

Uma visita ao site oficial



O banco de dados para aplicativos modernos

MongoDB é um banco de dados de propósito geral, baseado em documentos, distribuído criado para desenvolvedores de aplicativos modernos e para a nuvem desta era.

Nenhum banco de dados deixa você mais produtivo.



Try MongoDB free in the cloud!

Start free

<https://www.mongodb.com/pt-br>

MongoDB



- SSPL - Server Side public License
- 2009
- Baseado em documentos
- Tem uma grande gama de plugins disponíveis

Relações com SQL



SQL	Mongo
Database	Database
Tabela	Coleção
Linha	Documento
ID	_id

Iniciando o mongo

Vamos usar docker, mas você pode instalar em sua máquina se quiser ou usar na nuvem com atlas

```
version: '3.1'

services:

  mongo:
    image: mongo
    restart: always
    environment:
      MONGO_INITDB_ROOT_USERNAME: root
      MONGO_INITDB_ROOT_PASSWORD: example
    ports:
      - 27017:27017
```

Pincelando o express

O mongo-express é uma interface web amigável para o gerenciamento do mongo, pode servir para exemplificar e visualizar os dados.

```
mongo-express:
  image: mongo-express
  restart: always
  ports:
    - 8081:8081
  environment:
    ME_CONFIG_MONGODB_ADMINUSERNAME: root
    ME_CONFIG_MONGODB_ADMINPASSWORD: example
    ME_CONFIG_MONGODB_URL: mongodb://root:example@mongo:27017/
```

Bson



Binary JSON, além de ocupar menos espaço por ser binário, permite tipos diferentes de dados como:

- byte
- int [32, 64]
- uint (somente positivos)
- double
- decimal
- date
- objectId
- array

Códi
go

Vimos aqui pra
isso, não é
mesmo?

Bibliotecas Python



- pymongo - Oficial sync
- motor - Oficial Async
- MongoEngine - ODM
- ODMantic - ODM async

Bibliotecas Python



- **pymongo - Oficial sync**
- motor - Oficial Async
- MongoEngine - ODM
- ODMantic - ODM async



```
1  from pymongo import MongoClient
2
3  # "mongodb://user:password@host:port"
4  uri = "mongodb://root:example@localhost:27017"
5
6  client = MongoClient(uri)
```

Acessando o banco e as coleções



```
from pymongo import MongoClient

uri = 'mongodb://root:example@localhost:27017'
client = MongoClient(uri)

database = client['live_de_python']
collection = database['pessoas']
# ou
database = client.live_de_python
collection = database.pessoas
```

Bora fazer um CRUD? - Create



```
collection.insert(  
    {'nome': 'Eduardo', 'idade': 17}  
)
```

Bora fazer um CRUD? – Read



```
collection.find(  
    {'nome': 'Eduardo'}  
)
```

Operadores de queries



```
collection.find(  
    {'idade': {'$operador': 'valor'}}  
)
```

Operadores de queries – comparação



operador	O que faz?
\$eq	Igual a (==)
\$ne	Diferente de (!=)
\$gt	Maior que (>)
\$gte	Maior ou igual (>=)
\$lt	Menor que (<)
\$lte	Menor ou igual (<=)
\$in / \$nin	Igual (ou diferente) a um do array

Bora fazer um CRUD? - Update



```
collection.update(  
    {'idade': 18},  
    {'$set': {'idade': 'dezoito'}}  
)
```


Operadores de update



operador	O que faz?
\$set	Faz o update
\$unset	Remove o campo
\$rename	Renomeia o campo

Bora fazer um CRUD? - Update



```
collection.update_many(  
    {'idade': 18},  
    {'$set': {'idade': 'dezoito'}}  
)
```

Bora fazer um CRUD? - Delete



```
collection.remove(  
    {'idade': {'$op': 'value'}}  
)
```



picpay.me/dunossauro



apoia.se/livedepython



PIX



Ajude o projeto

