

TinyDB

BANCO DE DADOS ORIENTADO A DOCUMENTOS.

BY: DAVI LUCCIOLA

01 INTRODUÇÃO

02 INSERINDO REGISTROS

03 CONSULTAS

04 ATUALIZANDO/REMOVENDO REGISTROS



CONTEUDOS

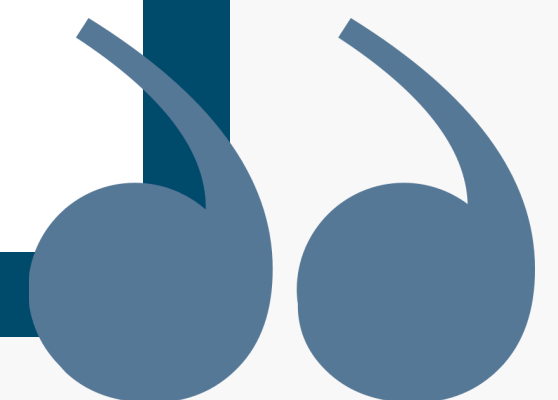


O QUE É O TINYDB?

O TinyDB é um banco de dados **orientado a documentos** utilizado para sistemas menores que utiliza um JSON como banco de dados. Pode ser utilizado como alternativa ao SQLite3.

DOCUMENTOS:

Documentos são estruturas de dados onde cada documento pode ter sua própria estrutura. No TinyDB será um quase **um dicionário do Python**.



CRIANDO O BANCO DE DADOS



```
1  from tinydb import TinyDB
2
3  db = TinyDB('./database.json', indent=4)
4  document = {'name': 'Davi', 'profissao': 'Developer'}
5
6  db.insert(document)
```

No código acima estamos criando o nosso banco com a classe “TinyDB”, e passamos como parametro o path do JSON que será o nosso banco de dados.

CRIANDO O BANCO DE DADOS

Ao lado temos um exemplo do JSON criado como banco de dados.

Cada dado será armazenado em uma “tabela”, que no JSON será representada por uma chave, sendo a tabela padrão “_default”

```
1  {
2    "_default": {
3      "1": {
4        "name": "Davi",
5        "profissao": "Developer"
6      }
7    }
8  }
```

INSERINDO DADOS



```
1  from tinydb import TinyDB
2
3  db = TinyDB('./database.json', indent=2)
4  table_produtos = db.table('produtos') # Podemos atribuir a tabela a uma variavel
5
6  produto1 = {'nome': 'Notebook', 'preco': 3999.90, 'processador': 'Intel Core i5', 'ram_gb': 8}
7  produto2 = {'nome': 'Geladeira', 'preco': 4500.90, 'volume_litros': 300, 'dispenser_agua': False}
8
9  db.table('produtos').insert(produto1) # Podemos inserir acessando a tabela diretamente
10 table_produtos.insert(produto2) # Ou podemos inserir através da variavel da tabela
```



```
1  {
2    "produtos": {
3      "1": {
4        "nome": "Notebook",
5        "preco": 3999.9,
6        "processador": "Intel Core i5",
7        "ram_gb": 8
8      },
9      "2": {
10       "nome": "Geladeira",
11       "preco": 4500.9,
12       "volume_litros": 300,
13       "dispenser_agua": false
14     }
15   }
16 }
```

CONSULTANDO DADOS



```
1  from tinydb import TinyDB
2
3  db = TinyDB('./database.json', indent=2)
4
5  # Todos os Produtos
6  db.table('produtos').all()
```


BUSCANDO DADOS



```
1  from tinydb import Query
2
3  Produto = Query()
4
5  # Filtrando por atributo
6  db.table('produtos').search(Produto.preco < 4000)
7
8  # Buscando um dado especifico por chaves
9  db.table('produtos').get(Produto['name'] == 'Geladeira')
```

BUSCANDO DADOS



```
1  from tinydb import Query
2
3  Produto = Query()
4
5  # Operador 'E' - 'and' do python
6  db.table('produtos').search(Produto.nome.search('Note') & Produto.preco < 4000)
7
8  # Operador 'OU' - 'or' do python
9  db.table('produtos').search(Produto.nome == 'Geladeira' | Produto.nome == 'Fogão')
```

DOC ID

Quando buscamos uma informação com os métodos “search” ou “get”, o TinyDB nos trará um Documento, esse documento terá um “doc_id”, que será o identificador unico daquele documento em nosso banco de dados.



The diagram illustrates a document structure. A blue arrow points from the 'DOC ID' title to the 'doc_id' field in the nested document. The structure is as follows:

```
1 {  
2   "default": {  
3     "1": {  
4       "name": "Davi",  
5       "profissao": "Developer"  
6     }  
7   }  
8 }
```

ATUALIZANDO DADOS



```
1  from tinydb import TinyDB, Query
2
3  db = TinyDB('./database.json', indent=2)
4  table_produtos = db.table('produtos')
5
6  Produto = Query()
7  notebook = table_produtos.get(Produto.nome == 'Notebook')
8  table_produtos.update({'preco': 2000}, doc_ids=[notebook.doc_id])
```

REMOVENDO DADOS



```
1  from tinydb import TinyDB, Query
2
3  db = TinyDB('./database.json', indent=2)
4  table_produtos = db.table('produtos')
5
6  Produto = Query()
7  geladeira = table_produtos.get(Produto.nome == 'Geladeira')
8  table_produtos.remove(doc_ids=[geladeira.doc_id])
```



OBRIGADO!