

11. Métodos da classe Driver usados na biblioteca Neo4j em Python

Para entender melhor como funciona a conexão entre uma aplicação Python e o banco de dados Neo4j, foi consultada uma ferramenta de Inteligência Artificial sobre os principais métodos disponíveis na classe Driver do pacote neo4j-driver.

11.1 driver.session()

Cria uma sessão usada para executar comandos Cypher no banco de dados.

Exemplo:

```
from neo4j import GraphDatabase

driver = GraphDatabase.driver("neo4j://localhost:7687", auth=("neo4j", "senha"))

with driver.session() as session:

    result = session.run("MATCH (n) RETURN n LIMIT 3")

    for record in result:

        print(record)
```

11.2 driver.verify_connectivity()

Verifica se o driver consegue se conectar ao servidor Neo4j.

Exemplo:

```
driver.verify_connectivity()

print("Conexão com o Neo4j verificada com sucesso.")
```

11.3 driver.close()

Fechá o driver e encerra a conexão com o banco de dados.

Exemplo:

```
driver.close()
```

11.4 driver.executable_query()

Permite criar consultas parametrizadas e reutilizáveis.

Exemplo:

```
query = driver.executable_query("MATCH (p:Pessoa {nome: $nome}) RETURN p")

with driver.session() as session:

    result = session.run(query, nome="Maria")

for record in result:

    print(record)
```

11.5 Resumo dos Métodos

session() - Abre uma sessão para executar consultas Cypher

verify_connectivity() - Testa a conectividade com o Neo4j

close() - Encerra a conexão do driver

executable_query() - Cria consultas parametrizadas e reaproveitáveis

12. Referência ao uso de IA na elaboração do conteúdo

Parte do conteúdo técnico desta documentação foi elaborada com o apoio da ferramenta de Inteligência Artificial ChatGPT, utilizada para pesquisa sobre a classe Driver da biblioteca Neo4j em Python. A referência completa está listada na seção de Referências.

Referências:

CHATGPT (OpenAI). Métodos da classe Driver usados na biblioteca Neo4j em Python. Conteúdo gerado em 30 nov. 2025 via plataforma ChatGPT.