Neo4j安全测试

概念

背景

1.图数据库

在计算机科学中，图数据库（英语：graph database，GDB[1]）是一个使用图结构进行语义查询的数据库，它使用节点、边和属性来表示和存储数据。该系统的关键概念是图，它直接将存储中的数据项，与数据节点和节点间表示关系的边的集合相关联。这些关系允许直接将存储区中的数据链接在一起，并且在许多情况下，可以通过一个操作进行检索。图数据库将数据之间的关系作为优先级。查询图数据库中的关系很快，因为它们永久存储在数据库本身中。可以使用图数据库直观地显示关系，使其对于高度互连的数据非常有用。

2.neo4j简介

Neo4j是一个高性能的,NOSQL图形数据库，它将结构化数据存储在网络上而不是表中。它是一个嵌入式的、基于磁盘的、具备完全的事务特性的Java持久化引擎，但是它将结构化数据存储在网络(从数学角度叫做图)上而不是表中。Neo4j也可以被看作是一个高性能的图引擎，该引擎具有成熟数据库的所有特性。程序员工作在一个面向对象的、灵活的网络结构下而不是严格、静态的表中——但是他们可以享受到具备完全的事务特性、企业级的数据库的所有好处。

Neo4j因其嵌入式、高性能、轻量级等优势，越来越受到关注.

3.Neo4j特点

(1)完整的ACID支持

(2)高可用性

(3)轻易扩展到上亿级别的节点和关系

(4)通过遍历工具高速检索数据

4.Neo4j的常见应用场景

(1)金融行业的反洗钱模型

(2)社交网络图谱

(3)企业关系图谱

4.Neo4j使用

|  |
| --- |
| from neo4j import GraphDatabase  # 连接配置  URI = "neo4j://localhost:7687"  USER = "neo4j"  PASSWORD = "xxxxxx"  class Neo4jCRUD:  def \_\_init\_\_(self, uri, user, password):  self.driver = GraphDatabase.driver(uri, auth=(user, password), encrypted=True)  def close(self):  self.driver.close()  # 数据插入示例  def create\_person(self, name, age):  with self.driver.session() as session:  result = session.execute\_write(  self.\_create\_and\_return\_person, name, age  )  print(f"Created person: {result['name']}, ID: {result['id']}")  @staticmethod  def \_create\_and\_return\_person(tx, name, age):  query = (  "CREATE (p:Person {name: $name, age: $age}) "  "RETURN id(p) AS id, p.name AS name"  )  result = tx.run(query, name=name, age=age)  record = result.single()  return {"name": record["name"], "id": record["id"]}  # 数据查询示例  def find\_person(self, name=None):  with self.driver.session() as session:  if name:  query = ("MATCH (p:Person) WHERE p.name = $name RETURN p")  result = session.run(query, name=name)  else:  query = "MATCH (p:Person) RETURN p ORDER BY p.age"  result = session.run(query)  print("\n查询结果:")  for record in result:  node = record["p"]  print(f"- Name: {node['name']}, Age: {node['age']}")  if \_\_name\_\_ == "\_\_main\_\_":  neo4j = Neo4jCRUD(URI, USER, PASSWORD)  # 插入数据  neo4j.create\_person("Alice", 30)  neo4j.create\_person("Bob", 25)  neo4j.create\_person("Charlie", 35)  # 查询所有人员  neo4j.find\_person() |

白盒测试

|  |
| --- |
| import/s+neo4j|from/s+neo4j |

|  |
| --- |
| .run( |

反例

直接拼接

|  |
| --- |
| driver = GraphDatabase.driver(uri, auth=(user, password), encrypted=False)  with driver.session() as session:  if name:  # 参数化查询  query = (f"MATCH (p:Person) WHERE p.name = '{name}' RETURN p")  result = session.run(query) # ✅ 安全  # 结果处理（保持不变）  print("\n查询结果:")  for record in result:  node = record["p"]  print(f"- Name: {node['name']}, Age: {node['age']}") |

正例

参数化查询

|  |
| --- |
| driver = GraphDatabase.driver(uri, auth=(user, password), encrypted=False)  with driver.session() as session:  if name:  query = ("MATCH (p:Person) WHERE p.name = $name RETURN p")  result = session.run(query, name=name)  else:  query = "MATCH (p:Person) RETURN p ORDER BY p.age"  result = session.run(query)  print("\n查询结果:")  for record in result:  node = record["p"]  print(f"- Name: {node['name']}, Age: {node['age']}") |

反例

neo4j.conf无认证配置，可免密访问neo4j服务器

|  |
| --- |
| dbms.security.auth\_enabled=false |

正例

neo4j.conf开启认证配置，访问neo4j服务器前需要认证

|  |
| --- |
| dbms.security.auth\_enabled=true |

安全配置

https://neo4j.com/docs/operations-manual/current/security/checklist/

[欢迎访问neo4j文档! — neo4j 0.1.0a documentation](http://neo4j.com.cn/public/docs/index.html)