迭代二测试文档

版本	作者	完成时间	内容
v1.0	蒋祚竑	2021.4.18	持久层集成测试

概述:

迭代二后端的主要工作内容为对图谱、节点和关系进行持久化。由于迭代二对持久化的内容提出了新的需求,因此笔者重构了持久层代码,对重构后的代码重新进行了测试。

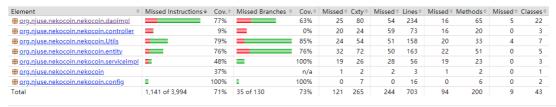
####

测试度量:

测试类数	测试方法	Daolmpl覆盖	NodeDaolmpl覆盖	RelationShipDaoImpl			
量	数	率	率				
3	4 + 1 + 1 = 6	77%	86%	67%			

使用Jacoco生成的测试覆盖率报告如下:

nekocoin



<u> </u>			-	-	-	-	-	-		
<u> </u>	67%	n/a	3	9	16	48	3	9	0	1
	86%	71%	7	23	18	106	3	16	0	1
			-	~	~	~	~	~		

测试用例概述:

1.testInsertGraph, testInsertCategory, testInsertEntities:测试方法代码如下







测试目的:

测试Graph节点类,Category节点类,Entity节点类的插入、删除、查询方法是否正确。由于Graph, Category, Entity三个类均继承了Node节点类,并且对他们进行插入、查询、删除的逻辑类似,因此在测试方面采用了几乎相同的测试步骤。

测试步骤:

- 1.使用自己实现的generateGraph方法随机生成一个Graph图谱对象,并插入数据库中(Category, Entity对象的插入都依赖数据库中已存在的graph节点)
- 2.调用nodeDao.insert(Node node)方法,插入一个使用TestCaseAutoGenerate类随机生成的对应类型的节点,并对该节点的两种查询方法返回值进行测试(一种返回boolean即表示是否存在,另一种则根据查询结果序列化一个节点对象)
 - 3.调用对应节点类型的删除方法删除节点,并调用对应的查询方法检查删除是否成功
 - 4.删除插入的Graph图谱对象

2.testUpdateEntity:测试方法代码如下



测试目的:

由于在迭代二中前端涉及大量对图元的修改操作(例如修改坐标、修改颜色形状等),因此需要测试 前端图元对应的Entity实体类的修改操作逻辑是否能正确持久化。

测试步骤:

- 1.插入图谱
- 2.生成一个用于测试的Entity类型对象
- 3.再生成一个新的Entity类型对象newEntity(该对象作为参数传入update方法)
- 4.检查原节点是否存在
- 5.通过get方法获取修改后的节点,并与newEntity进行对比
- 6.删除图谱

####

3.testRelation:代码如下





测试目的:

测试节点之间的关系能否正确持久化。

测试步骤:

- 1.插入一个随机生成的图谱以及两个随机生成的Entity节点
- 2.为这两个节点建立一个关系,并使用查询方法查找该关系
- 3.调用更新关系的update方法,对调原关系的源节点和目标节点,并重设关系名
- 4.使用查询方法查找更新后的关系

4.testUploadGraph:测试代码如下



测试目的:

使用自己实现的generateTestGraphWithEntities方法用户传入预定义的json格式图谱文件,测试该随机生成的图谱能否在后端正常持久化。

测试步骤:

- 1.生成一个包含100个Entity节点,10个Category节点,500条关系的图谱graph
- 2.插入该图谱,并通过retrieveGraphByld方法从数据库中获得一个图谱对象getRes,并比较该图谱对象与graph的节点数量。