图数据库Neo4j
空间数据存储组织

彦朝环

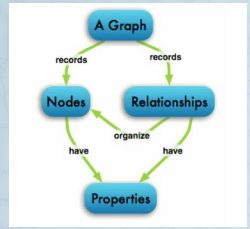
目录

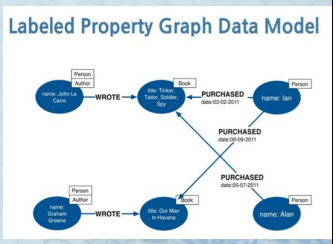
一、图数据库简介

- 设计理念
- Neo4j的设计动机是为了更好地同时也更高效地描述实体之间的关系。在现实生活中，每一个实体都于周围的其他实体有着千丝万缕的关系，这些关系里面所存储的信息甚至要大于身体本身的属性。
- 图数据库作为重点描述数据之间关系的数据库应运而生，成为了NoSQL中非常重要的一部分。而Neo4j正是图数据库中最为优秀的之一

两种类型的数据：

- 一、节点Node：
  - 节点类似于E-R图中的实体（entity），每个实体可以有0到多个属性，这些属性以key-value对的形式存在，并且对属性没有类别要求，也无需提前定义。另外，还允许给每个节点打上标签，以区别不同类型的节点。
- 二、关系Relationship：
  - 关系类似于E-R图种的关系（relationship），一个关系有一个起始节点和一个终止节点构成。另外和node一样，关系也可以有多个属性以及标签

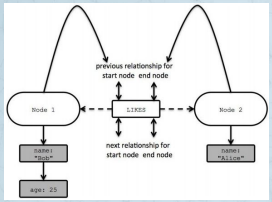




特性：
一、关系在创建的时候就已经实现了，因而在查询关系的时候是一个O(1)的操作
二、所有的关系在Neo4j中都是同等重要的
三、提供了图的深度优先搜索、广度优先搜索、最短路径、简单路径以及Dijkstra等等算法

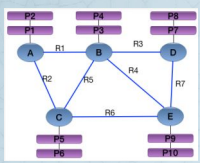
二、neo4j的存储结构

- 1、neo4j 中节点和关系的物理存储模型

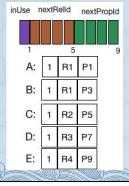


Node和Relationship 的 Property 是用一个 Key-Value 的双向列表来保存的；Node 的 Relatsionship 是用一个双向列表来保存的，通过关系，可以方便的找到关系的 from-to Node. Node 节点保存第1个属性和第1个关系ID。

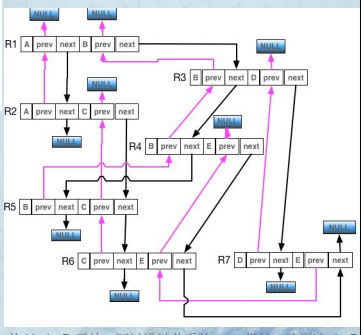
示例1



Node 的存储示例图如下
每个Node 保存了第1个Property
和 第1个Relationship



•关系的存储示意图如下：

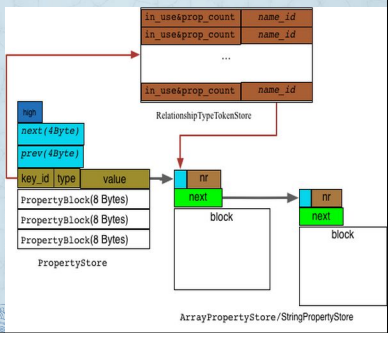


从 Node-B 开始，可以通过关系的 next 指针，遍历Node-B 的所有关系，然后可以到达与其有关系的第1层Nodes,在通过遍历第1层Nodes的关系，可以达到第2层Nodes

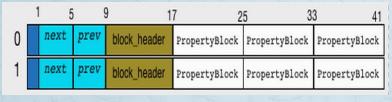
2、Property 的存储

neo4j 中, Property 的存储是由 PropertyStore, ArrayPropertyStore, StringPropertyStore 和PropertyKeyTokenStore 4种类型的Store配合来完成的.

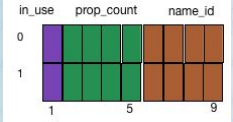
PropertyStore是Property最主要的存储结构，当Property的Key-Value对的Value是字符串或数组类型并且要求的存储空间比较大，在PStore中保存不了，则会存在String/ ArrayPropertyStore这样的DynamicStore 中。如果长度超过一个block ，则分block存储，并将其在String/ ArrayPropertyStore中的第1个block 的block_id 保存到 PropertyStore类型文件相应record 的PropertyBlock字段中。
PropertyKeyTokenStore和StringPropertyStore 配合用来存储Propery的Key部分。Propery的Key是编码的，key 的 id 保存在PropertyKeyTokenStore ，key 的字符串名保存在对应的StringPropertyStore



2.1 PropertyStore类型的存储格式



2.2 PropertyKeyTokenStore的文件存储格式

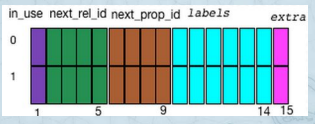


2.3 String 类型属性值的保存过程

2.4 Array 类型属性值的保存过程

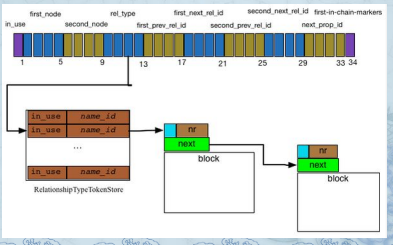
3.Node 数据存储

Node 的存储是由 NodeStore 和 ArrayPropertyStore 2中类型配合来完成的. node 的label 内容是存在ArrayPropertyStore这样的DynamicStore 中，如果长度超过一个block ，则分block存储，并将其在ArrayPropertyStore中的第1个block 的 block_id 保存到 NodeStore类型文件相应record 的labels字段中。

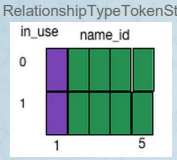


4.Relationship 的存储

Relationship 的存储是由 RelationshipStore , RelationshipGroupStore, RelationshipTypeTokenStore和StringPropertyStore 4种类型的Store配合来完成的. 其中RelationshipStore 是Relationship最主要的存储结构；当一个Node 的关系数达到一定的阀值时，才会对关系分组(group), RelationshipGroupStore 用来保存关系分组数据；RelationshipTypeTokenStore和StringPropertyStore 配合用来存储关系的类型。



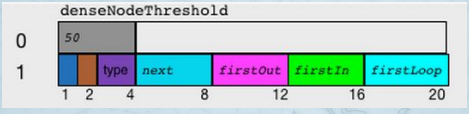
4.1 RelationshipTypeTokenStore的主文件存储格式



4.2 RelationshipStore的文件存储格式

4.3 RelationshipGroupStore类型的存储格式

当Node的Relationship数量超过一个阀值时，neo4j 会对 Relationship 进行分组，以便提供性能。neo4j 中用来实现这一功能的类是RelationshipGroupStore。



下面举一个图的遍历的例子：

•从节点1开始，宽度优先遍历，其存储结构为：01 00000002 ffffffff

•其下一个关系id是2，访问关系2：01 00000001 00000004 00000000 ffffffff 00000001 ffffffff ffffffff ffffffff 得出node 1 -> node 4,同时下个关系是1

•关系1： 01 00000001 00000003 00000000 00000002 00000000 00000003 ffffffff ffffffff node1 -> node 3,node3 有其他关系，所以将node3存入队列，同时访问关系0

•关系0：01 00000001 00000002 00000000 00000001 ffffffff ffffffff ffffffff node1 -> node2，访问完成node1的所有关系，从队列中退出node3

•用于上文相同的方法访问node3

```
(1)-[KNOWS,2]->(4)
(1)-[KNOWS,1]->(3)
(1)-[KNOWS,0]->(2)
(1)-[KNOWS,1]->(3)-[KNOWS,5]->(7)
(1)-[KNOWS,1]->(3)-[KNOWS,4]->(6)
(1)-[KNOWS,1]->(3)-[KNOWS,3]->(5)
```