Graph Database Specification

# 1. 开发目的

1). 设计出可以供桌面应用开发使用的轻量级数据库；

2). 在设计过程中完善图数据库理论和实践经验；

3). 拓展图数据库的应用领域。

# 2. 项目划分

## 2.1 内核开发

目标：参照Neo4j数据库的基本架构，利用C#平台实现图数据库。

功能（任务）列表：

1、图为节点和连边的宿主元素，承担主调任务，节点类和连边类提供底层辅助功能；

2、数据库手工建立：

1）、图的建立与成员初始化；

2）、节点的插入、删除，与属性的插入、删除和修改；

3）、连边的插入、删除，与取值修改；

3、数据批量导入：

1）、XML文件的读取；

2）、XML文件的保存；

产出：最后完成一个可以存储数据的图数据库类库，能够实现数据逐条或批量的增、删、改操作。

## 2.2 可应用最小系统

目标：在前一阶段基础上，实现图数据库的查找功能，设计高效的查找引擎。

功能（任务）列表：

1、分析Neo4j查询语言的基本语法，并分解查找工作流程；

2、在GraphDB中实现查找算法；

3、能够通过类似Neo4j的语言（简化后）对图数据库进行查找。

产出：在类库中增加查找功能接口。

## 2.3 数据库管理系统

目标：在之前完成的数据库类库的基础上，实现一个数据库管理系统，提供可视化的操作界面。

功能（任务）列表：

1、单一操作，能够建立数据库，并逐条插入、修改和删除数据；

2、批量操作，对已建立的数据库，实现批量插入、修改和删除操作；

3、提供文字化的查询功能，并通过文字或列表返回查询结果；

4、实现可视化效果，图数据库的结构显示和图形化查询功能。

产出：一个可运行软件，可以图形化或文字化地对图数据库进行逐条或批量数据的增删改查操作。

# 3. 模块设计

## 3.1 内核结构

内核结构说明，如图3.1所示：

1、图包含一个节点列表和一个连边列表；

2、节点包含一个名称，一个类型和若干个（0-n）个属性三个主体部分，另外包含一个出边列表和一个入边列表。

3、连边包含一个类型、一个取值构成的主体部分、一个起始节点指针和一个终止节点指针。

4、节点的属性列表是由若干个key和value的键值对组成，可以由用户自主添加。

5、节点入边和出边列表中内容是对应边的指针数组或引用数组。

6、连边的起始和终止节点指针式对应节点的指针或引用。



图3.1 内核元素结构示意图

## 3.2 数据库创建与维护

### 3.2.1 数据库创建与数据插入

数据库创建流程说明：

1、主调函数调用Database类提供的操作接口CreateDatabase（）创建数据库对象；

2、Database类调用Graph类的构造函数，创建一个图对象；

3、创建成功则返回数据库对象给调用者，否则返回错误原因。

数据节点插入流程：

1、主调函数调用Database类接口AddNodeData()；

2、Database类调用Graph类接口AddNode()；

3、Graph类调用Node构造函数创建一个新数据节点；

4、将Node对象加入Graoh中节点列表，并逐层返回OK；否则逐层返回失败原因。

关系连边插入流程：

1、主调函数调用Database类接口AddEdgeData()；

2、Database类调用Graph类接口AddEdge()；

3、Graph类调用Edge构造函数创建一个新数据节点，创建成功则返回该Edge对象；

4、Graph类调用起始节点Node对象的AddEdge函数和终止节点Node对象的RegisterInbound函数完成连边在节点间的注册；

5、将Edge对象加入Graoh中连边列表，并逐层返回OK；否则逐层返回失败原因。



图3.2 数据库创建、数据插入、关系插入操作序列图

### 3.2.2 节点和连边内部数据修改



图3.2节点和连边内部信息修改操作序列图

如图3.2所示，节点和连边的内部数据相互独立，不存在依赖关系，所以依次调用接口函数进行修改即可，详细流程在此不做赘述。

*注意：数据修改是指修改节点和连边内部的附加信息，但节点内部的核心信息不能改动。如果需要修改节点和连边的和信息，则需要对节点或连边进行删除，随后重新创建后加入。*

### 3.2.3 节点和连边的删除

关系连边删除流程，如图3.3：

1、主调函数传入待删除连边起始节点和终止节点的名称、类型以及连边类型数据；

2、Graph类首先查找连边的起始节点和终止节点，以及节点之间的同类型边；

3、随后从起始节点的outbound列表和终止节点的inbound列表中分别注销目标连边；

4、最后从Graph类的EdgeList中将目标连边移除，完成后返回OK；否则返回失败原因。



图3.3连边删除操作序列图

数据节点删除流程，如图3.4：

1、主调函数传入待删除节点的名称、类型；

2、Graph类在节点列表中查找匹配的节点；

3、Graph类调用目标节点的成员函数ClearEdge，清除该节点有关的所有连边；

4、节点内部将inbound中所有连边从其起始节点的outbound中注销，将outbound中所有连边从其目标节点的inbound中注销，并将这些连边都加入待清除列表，返回给Graph类。

5、Graph类从EdgeList中删除列表所有记录下来的连边，从NodeList中删除目标节点；

6、Graph类删除完成后返回OK；否则返回出错原因。



图3.4节点删除操作序列图

## 3.3 文件读取与存储

### 3.3.1读取XML文件

XML数据导入流程，如图3.5所示：

1、主调函数调用InitData接口启动导入流程；

2、DataBase对象清除内部数据，并创建新的Graph对象；

3、Graph对象从文件中取出xml数据并调用Node构造函数，生成节点加入NodeList；

4、Graph对象从文件中取出xml数据并调用Edge构造函数；

5、对于每条连边，Graph根据文件中编号找到其起始和终止节点；

6、将起始节点和终止节点地址写入连边内部的两个指针，最后在两个节点中注册该条边；

7、将当前边加入EdgeList，并继续，直到完成所有节点和连边的生成。



图3.5 读取XML文件导入数据序列图

### 3.3.2 输出XML文件

数据导出到XML文件流程，如图3.6所示：

1、由上层主调函数提供待导出文件路径；

2、Database调用Graph的ToXML函数构造XML文档；

3、Graph类调用AdjustNodeIndex重新给节点依次赋予编号（之前如果有删除操作，会导致原有编号之间存在空值）；

4、Graph类分别遍历节点列表和连边列表，并调用各个节点和连边的成员函数ToXML生成各自的XML节点；

5、Graph类根据顺序将返回的XML节点组织成XML文档，并返回DataBase；

6、Database类将XML文档保存到指定路径，并返回OK；否则返回出错原因。



图3.6 保存数据到XML文件序列图

## 3.3 数据查询

## 3.4 程序接口