浙 江 大 學

本 科 生 毕 业 论 文（设计）

题目 DartGrid-OWL 查询推理机的设计与实现

姓名与学号 郑耀文 3053027048

指导教师 陈华钧 博士

年级与专业 05 级计算机科学与技术

所在学院 计算机科学与技术学院

A Dissertation Submitted to Zhejiang University for the Degree of Bachelor of eEngineering

TITLE: The Design and Implementation of DartGrid-OWL Inference Engine

Author: Yaowen Zheng

Supervisor: Professor Zhaohui Wu

Doctor Huajun Chen

Subject: Computer System Architecture

College: College of Computer Science

Submitted Date: June 1st, 2009

浙江大学本科生毕业论文（设计）诚信承诺书

1.本人郑重地承诺所呈交的毕业论文（设计），是在指导教师的指导下严格按照学校和学院有关规定完成的。

2.本人在毕业论文（设计）中引用他人的观点和参考资料均加以注释和说明。

3. 本人承诺在毕业论文（设计）选题和研究内容过程中没有抄袭他人研究成果和伪造相关数据等行为。

4. 在毕业论文（设计）中对侵犯任何方面知识产权的行为，由本人承担相应的法律责任。

毕业论文（设计）作者签名：

_____年_____月_____日

摘要

在互联网飞速发展的背景下，大规模数据共享对社会进步，尤其是科学领域有着非凡的意义，但是由于系统的定制性和历史原因等种种原因所形成的数据封闭性、数据孤岛等难题严重阻碍了科学数据的有效共享。

为了解决对异质异构数据的集成与共享，浙江大学网格实验室致力于通过语义与网格技术来解决这个难题，并借此开发出 DartGrid 语义数据库网格系统。

尽管有效地缓解了数据库异构查询的问题，但是 DartGrid 在 SPARQL 重写过程中存在不足，当前版本的 DartGrid 在语义层的解析薄弱，这使得本体技术在 DartGrid 中得不到充分的应用，并且不具备推理功能。

新一版的 DartGrid 会对 SPARQL 重写过程进行重新设计和实现，会更符合 SPARQL 分析模型并且更好的利用现有的语义网技术，同时在翻译的同时会增添推理功能。

本文中，首先将分析当前 DartGrid 的不足，同时对 OWL 语言特性进行介绍并给出推理的基本概念，之后，会提出新一版 DartGrid 的设计目标，同时会对新一版 DartGrid 的体系结构、设计思路和实现进行介绍和说明。

关键词 SPARQL，DartGrid，OWL，推理

Abstract

In presence of the rapid development of the internet, people consider sharing with each other large amount of data become very meaningful, especially sharing the data of science area. It really blocks the pace of people to share common science data since customization of certain system, or some historical reason, which lead to some serious problem such as the closure of data.

In order to integrate the data come from heterogeneous relational databases. The CCNT laboratory of Zhejiang University aim to solving the problem with semantic and grid technology, and then DartGrid is developed.

Although the current version of DartGrid(DartGridV3) has relieve the problem caused by heterogeneous relational database. But there is a huge defect on the SPARQL query rewriting process, which makes it vail on ontology concept dealing.

The new version of DartGrid will take large effort to design and implement the SPARQL rewriting process, it will become more reasonable on semantic view and additional capability of inference will be realized.

In this paper, it will come first that the analyse of the inefficient design of DartGridV3. Also, the concept of OWL and Inference will be simply introduced. And the design goals and structure of the new version of DartGrid will be presented, as well as some real case which show you the power of the new version of DartGrid.

Keywords SPARQL，DartGrid，OWL，Inference

目录

第1章 绪论

1.1 课题背景

在当今这个知识爆炸飞速发展的信息时代，各个科学领域的科学数据也呈现指数级增长，这海量的科学信息对于我国乃至人类世界都是宝贵的财富。而来自不同组织的大规模数据开放式共享将无疑会对科学进步提供很大的主力。

与此同时，世界范围内很多研究人员已经意识到数据网络共享化的重要性，但是存放这些科学信息的数据存储库往往都是在相对孤立的情况下由世界各地的实验室开发出来的，它们往往使用不同的标识制度，不兼容的术语和不同的数据格式。当然，数据共享还存在着安全与隐私的问题。

DartGrid 旨在设计一个平台能够将来自不同数据库的数据进行整合和封装[1]，使得上层开发应用模块能够无需了解数据库的数据格式以及数据库结构等信息就能对所需要的信息进行搜索，并且能够借助当前最新的语义网知识进行本体层次设计，从而平台具备基本的推理能力。

1.1.1 DartGrid 简介

DartGrid 是一个网络框架应用平台，其包含一组实用的本体工具，通过语义网技术来帮助开发者整合异构数据库。其建立目的为

- 通过富语义将分布式传统数据库进行内部关联。
- 以一个巨大的分布式数据库提供基于本体的查询，查询和导航工具。
- 在顶层增加推倒能力来增加数据的利用度和重用度。

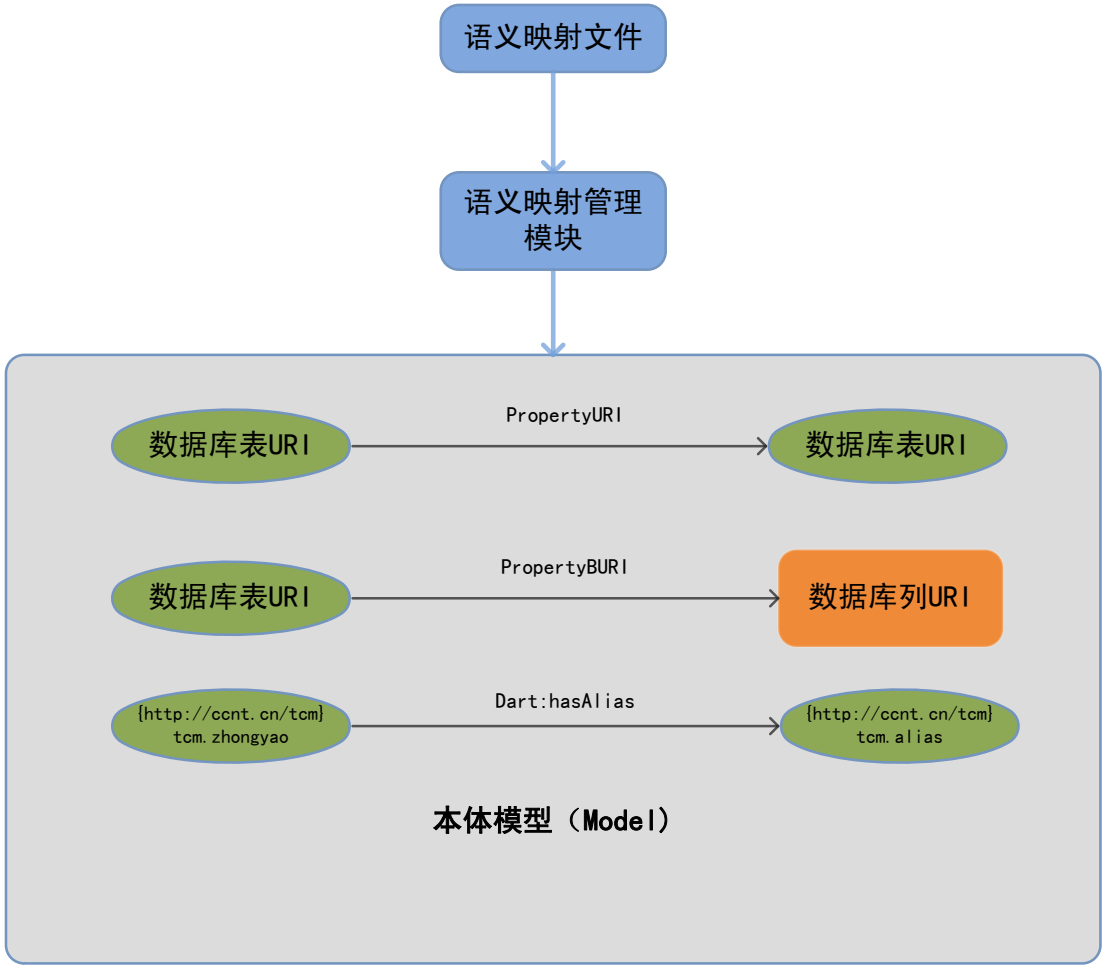
DartGrid 发展至今，已经开发出了一组使用的语义网工具[2]：

- DartMapping：可视化映射工具来帮助 DBA 定义分布式数据模式到 RDF/OWL 本体的语义映射
- DartQuery：一个基于本体的查询接口，将用户传入的 SPARQL 语义查询重写成一组 SQL 查询
- DartSearch：一个基于本体的搜索引擎对数据库进行全文索引并且根据索引返回语义信息。

这些语义网应用为中国中药学院（CATCM）进行开发和部署。它通过超过 70 个类和 800 个属性的标准本体对 70 个传统中药数据库进行内部连接。用户或

机器仅仅需要和语义层进行交互，而数据库间的语义连接使得用户可以从一个数据库起点查询衍生到其他数据库找到相关数据。语义层同时使得系统能够回答这样的跨越多个数据库的语义查询“这种药治什么病”或者“什么样的要能够治疗这种疾病”，而不是像以前搜索引擎那样基于关键词的搜索。

1.1.2 当前 DartGrid 内核的不足



DartGridV3 在语义层映射的方法如下：

语义映射管理模块将语义映射文件生成一个 Jena 本体模型，该模型中存放的是形如（数据库表 URI，PropertyURI，数据库表 URI），或者是（数据库表 URI，PropertyURI，数据库列 URI）的 Resource，其中，Resource 的 URI 是数据库表唯一标识，即数据库表被看作单独的数据节点，Resource 的关联属性是另外一个 Resource，即两个表的联接被看作本体模型中数据节点的关联，Resource

的值属性是一个数据库列的 URI，即数据库列被看作是数据库表数据节点自身的属性，将数据库结构（而非数据库本身数据）做成资源模型，这样的实现思路是非常值得商榷的，尽管一定程度上能够支持 SPARQL 查询，但由于结构问题导致 SPARQL 支持拓展性比较差，当前模块中比较明显的缺陷有以下 2 个：

- 语义层缺失：DartGridV3 只能读入映射文件和数据库注册文件，映射文件本该作为本体对象和数据库对象的桥梁，而在 DartGridV3 的实现中，映射文件同时扮演了本体文件和映射文件的角色，因此导致映射文件中的本体类 URI 完全失效，只有本体属性保留。
- SPARQL 语言格式制约：基于这样的 Model 生成方法，在 SPARQL 的 WHERE 从句中的三元组不能出现值，因为三元组只能是数据库实例 URI，这对查询理解将会造成致命的歧义，所以为了缓解这个矛盾，DartGridV3 所支持的 SPARQL 查询都形如

```
PREFIX  ns:     <http://ccnt.cn/mt#>
 SELECT DISTINCT  ?alias
 WHERE{
   ?x  ns:hasAlias  ?a .
   ?a  ns:name      ?alias .
   ?x  ns:concept   ?concept .
   FILTER ( ?concept = "大黄")
 }
```

1.1.3 DartGridV4 的设计目的

DartGridV3 在 SPARQL 翻译过程中变量映射中采取的方法是非常值得商榷的。因此 DartGridV4 将推翻 DartGridV3 在 SPARQL 查询重写过程中变量映射部分的实现，增加本体信息管理模块，支持 OWL 本体文件的读入，并且利用当前语义网的开源推理包为 DartGrid 加入知识库从而使得 DartGrid 查询过程中能够进行面向 RDF 和 OWL 的推理。

1.2 论文结构

第二章，介绍了语义网和相关技术，以及当前已经有的实现语义网技术的工具。

第三章，简单介绍了推理的概念，并给出了 DartGridV4 语义网格在语义层上的设计目标，并介绍了 DartGridV4 的运行模式并描述了当前体系结构。

第四章，介绍了 SPARQL 基本知识，并提出查询重写的设计思路

第五章，介绍 DartGridV4 对于 SPARQL 的实现，并且给出案例描述了 DartGridV4 当前的实现情况。

第六章，总结了当前 DartGridV4 的实现情况，分析了当前 DartgGridV4 和 Jena 的优劣，并且对 DartGrid 的推理能力的未来应用价值进行了展望。

第2章 技术背景及相关工作

2.1 语义 Web

语义 Web 是万维网的进化版本，在语义网中，所有服务和资源都会被赋予语义，这将会彻底改变现有互联网仅仅依靠单纯的文字信息来共享资源的模式，语义网将能够真正理解并满足人或自动推理机对网络内容的使用请求。因此语义网取代当前万维网是时代的必然趋势。

2.1.1 语义网简介

上世纪 80 年代，Tim Berners-Lee 发明了互联网上的超文本系统，促成了万维网（World Wide Web）的诞生，使网络互连技术用于人们的信息交流与共享。但是，目前大多数页面的设计仅仅针对人类自身的，不便于机器自动处理——现在网页信息的表现方式，多为自然语言、图片、声音等方式，这适应于人们的阅读和收听需求。但是，这些媒介固有的不确定性引起数据格式的多样性，而无法被计算机理解。到目前为止，Web 大多数开发成为人们阅读文档的媒体，而在提供可自动处理的数据和信息方面，则发展较慢。语义 Web 就是想弥补这方面的不足。

在 2000 年的世界 XML 大会上，Tim Berners-Lee 做了题为 Semantic Web 的演讲，对语义 Web 的概念进行了解释，并提出了语义 Web 的体系结构。它通过扩展现有互联网，在信息中加入表示其含义的内容，使计算机可以自动与人协同工作[4]。语义 Web 力图将“理解信息的含义是人类的专利”这一局面成为历史，使得计算机在一定程度上也同样可以做到，从而有助于信息共享、再利用，并使网络能够提供动态的、个性化的、主动的服务[5]。在 Tim Berners-Lee 看来，语义 Web 是对互联网本质的变革。

目前，围绕语义 Web 的研究与开发正如火如荼的展开。在国外方面，典型的参与大学与研究机构有 MIT，Maryland，Manchester，Stanford，CMU 等；在国内方面，浙江大学在 Web 信息的知识表示方面进行了研究。浙江大学 CCNT 实验室针对 Web 上信息的不同性质分别在基于认知逻辑的语义表达模型，基于时态逻辑的流程语义表达模型和媒体流数据（非文本型数据）的语义表达或描

述方法方面展开了研究。在基于本体的应用研究中，浙江大学 CCNT 实验室建立了国内第一个基于语义的数据库网格 DartGrid[6]，支持网格环境下数据库资源的动态化的语义注册[7]、分布式的语义查询和知识级的语义浏览[8]。初步建立中医药本体库，为一体化语言系统提供基础[9]；并且基于语义 Web，开发成功具有本体论推理服务的语义浏览器。

在工业应用方面，有惠普公司支持开发的 Jena[10]语义 Web 工具包，有由欧盟资助，W3C 所领导的 SWAD 系列项目；在标准化方面，W3C 国际标准化组织已陆续完成了 RDF[11]、OWL[12]等的标准化规范工作。

2.1.2 语义 Web 的层次结构

为了实现语义信息服务智能化与自动化的目标，语义研究者们开发了许多新技术并提出了一系列的技术标准。Tim Berners-Lee 在综合了语义 Web 研究领域的最新成果的基础上，提出了语义 Web 模型。这一模型得到了语义 Web 研究者的认同。

Tim Berners-Lee 在 XML2000 大会上提出的语义 Web 体系结构如图[13]：

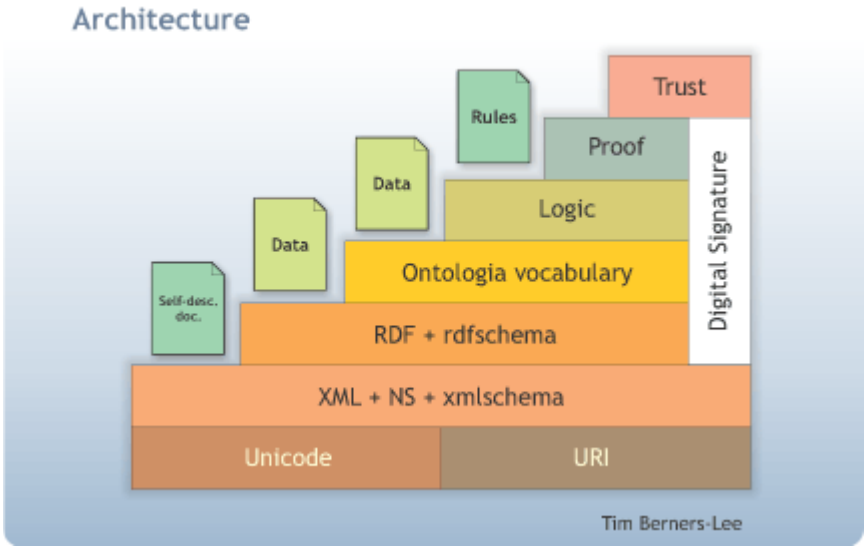


图 2.1 语义 Web 层次结构

自底向上来看：

第一层，**作为资源的标识机制**，提供对资源标准化的名字描述，规定了文档中字符的编码方式和资源的统一标识，即 UNICODE 和 URI；

第二层，**XML、命名空间和 XML Schema**，定义了结构化的数据描述方式，

使用自定义的标签对文档的结构进行标注，规范文档的语法格式，就可以方便的利用计算机处理文档，在统一模式的定义下交换文档.只有标注名并不能提供语义，所以在语义 Web 结构中 XML 只是作为语法层，来为语义 Web 的建立提供语法基础。

第三层，**明确文档中标引对象之间的关系**，即资源描述层 RDF+RDF Schema。RDF 是对结构化的元数据编码、交换和重用的一个基础。RDF 定义了元素之间的关系，表现为三元组的集合（类似于句子的主语、谓语、宾语）。在该语义 Web 模型中，信息以 RDF 句子的形式存储，即以统一的方式来存储数据，便于机器理解。XML 加上 RDF(S)就相当于建立了人工智能中的语义网络（SEMENTIC NETWORK），可以进行一定的推理。使用 XML+RDFS(S)，人们可以建立各自的语义 Web，只要有一套自成体系的术语就可以了。

第四层，**明确标引项的精确含义**，要让计算机相互理解对方的内容，需要有一套共同的标准的概念体系，这就是 Ontology（本体）。在人工智能领域和互联网研究领域，一个本体描述了一个特定研究领域的一个形式化的、共享的概念化模型。本题非常适合于描述互联网上各种不同的、分散的、半结构化的信息资源。通过定义共享的、通用的领域理论，本体帮助人和机器明确的交流，支持语义级的交换，而不是语法级的。本题在语义网技术中是最为关键的部分。

第五层，**逻辑层**，提供了规则与推导方法，从而便于在本体层上进行推理，得到有用的语义信息。从一定的程度上讲，本体层定义的是否合理直接关系到推理的难易和结果的有效性。

第六层，**证据层**，其在逻辑层基础上使代理可以交换推理的结果。为了检查这些结果，需要将各代理的内部推理机制转化为一种通用的证据表示语言。

2.1.3 本体与描述语言

Ontology 的概念起源于哲学领域，即“对世界上客观存在物的系统地描述”。而在计算机科学领域，Ontology 的核心含义是一个描述由一组类型，属性和关系属性组成的世界的模型。

之后本体概念变得愈发流行，在信息系统、知识管理等领域中，越来越多的人研究和使用本体，并给出了许多不同的定义。其中最著名并被引用得最为广泛的定义是由 Gruber 提出的，“本体是概念模型的明确的规范说明”[14]。Fensel 对这个定义进行分析后认为本体的概念包括四个主要方面[15]：

- 概念化(conceptualization)：客观世界中现象的抽象模型；
- 明确(explicit)：概念及它们之间联系都被精确定义；
- 形式化(formal)：精确的数学描述；
- 共享(share)：本体中反映的知识是其使用者共同认可的。

虽然不同研究者对本体有不同的描述，但是从内涵上来看，他们对本体的认识是一致的，都是把本体当作某个领域内（可以是特定领域的，也可以是更广的范围）不同主体（人、代理、机器等）之间进行交流（对话、互操作、共享等）的一种语义基础，即由本体提供明确定义的词汇表，描述概念和概念之间的关系，作为使用者之间达成的共识。因此，本体的用途包括交流、共享、互操作、重用等等。

在计算机领域讨论本体，首先就面临着本体究竟是如何描述的，也就是概念的形式化问题。对应的研究内容就是本体的描述语言。随着 Web 的发展，又出现了一系列基于 Web 的本体语言，也叫做本体标记语言，如 SHOE、XOL、RDF、RDFS、OIL、DAML、DAML＋OIL、OWL。 本文将主要介绍 OWL（Web 本体语言，Web Ontology Language），因为它是这次推理机设计的主要语言框架标准。

图 2.2 RDF 的三元组图

OWL（Web本体语言，Web Ontology Language）[16]是W3C推荐的本体描述

语言的标准，位于W3C绘制的本体语言栈的栈顶。它是为了在WWW上发布和共享本体而提供的语义标记语言。OWL是在DAML+OIL[17]的基础上发展起来的，作为RDF(S)的扩展，目的是提供更多的原语以支持更加丰富的语义表达，并更好的支持推理。

2.1.4 中医药中的本体

在上述的诸多本体定义中，对中医药一体化语言系统最有启发的是如下理解，本体是某种概念化体系的规范说明，某种概念化体系就是自成体系的某一科学领域的概念化体系，而科学领域内的科学数据属于该科学领域的某一个概念，因此，中医药一体化语言系统可以纳入到语义网建设的范畴中，中医药本体就是中医药科学数据的元信息。

目前，本体技术在工程中的具体应用还不是很成熟，在实际运用中存在着各种各样的问题，但这并不影响人们对本体技术发展前景的肯定，语义网的发展需要的实践，在实际应用中本体理论将会被进一步完善。而同时应注意到，图书情报科学中的分类法，主题法等文献组织技术显然可以作为开发本体的基础，因此，本体的理论发展和实际应用，需要计算机学科提供处理技术、图书情报学科继承和发展建模方法，以及每个学科领域的专家提供该领域的概念系统。在众多科研人员的努力之下，本体必将为人机对话架起便捷的桥梁。目前对于本体在中医药领域的应用，浙江大学CCNT实验室是走在全国最前列的。

2.2 语义网工具

随着语义网的蓬勃发展，很多公司和学院开始认识到语义本体的重要性，相应的许多开源的工具包也相继诞生。

2.2.1 Jena

Jena是一个开源的JAVA语义网框架，提供了能够对于RDF图中读取数据或写入数据的API，在Jena框架中，图本身被认为是Model，而文件、数据库、URL或者这三者间任意的组合都可以成为Model的数据源，一个Model可以通过SPARQL查询。Jena和Sesame相似，却提供了对OWL的支持。框架内置了一些推理机并通过DIG Interface提供了对外部推理机的支持。

2.2.2 Pellet 推理机

Pellet是一个对于OWL 2 DL的Java开源推理机。它提供了标准且顶尖的对于本体的推理服务。

Pellet同时也包括了对于包括OWL 2EL的OWL 2数据图表的支持。它同时也吸收了各类优化技术，包括命名的新型优化，连续查询回复和增量型推理等。是非常强大的基于Jena框架的推理包。

第3章 DartGridV4 体系结构

3.1 推理简介

如果当前存在如下本体类层次结构：(限于篇幅，不会给出具体类定义)

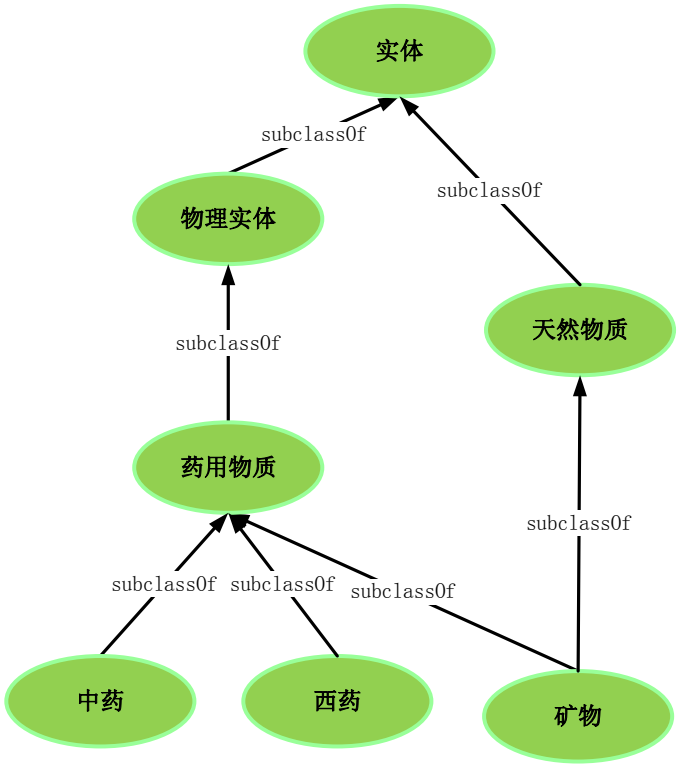


图 3.1 类继承层次

通过图3.1可以看出，类继承有别于一般树状结构，而是一种无环连通图，某一个类可以同时是两个类的子类（即某一个节点同时可以有两个父节点），在本例中，矿物同时是药用物质和天然物质的子类。

现在某人要查所有的物理实体，那么根据上图，他查的实际上是所有物理实体类实例以及其子类的实例（类的实例就是现实注明类型为该类的一个本体实例），且物理实体类的子类不仅包括显示声明的药用物质，还包括药用物质的所有子类（非显示声明），因此在这里给出一个推理机的简单定义，能够理解非显

示声明的需求（也可以称为潜在需求）的智能代理程序。

能够从显示信息拓展到非显示信息[18]，需要的是知识库，即公理的集合。公理是推理规则，比如A是B的子类，B是C的子类=>A是C的子类，或者A是B的子属性=>A的Domain是B的Domain或者其子类，这些知识都是RDFS和OWL本身包含的一些基本知识库，由于OWL_DL比RDFS在语言表述能力要强很多，包括类定义（Restriction，Enumerate）以及属性类型（TransitiveProperty，SymmetricProperty），所以OWL_DL的基本知识库要比RDFS庞大且复杂的多。

作为在某个实际领域运用语义技术的开发，我们可以添加自定制的规则，比如药A能够治疗疾病B，实体C是药A的成分=>实体C是具有治疗疾病B功效的药用成分。通过这些规则，我们能够对现有资源进行更好的知识发现。

3.2 DartGrid 设计目标

前文提到，DartGrid 的发展历程中，尽管在异构数据库异构的查询上做了很多工作并且取得了不错的成绩，在 SPARQL 的解析上却是令人失望的，所以这个版本中将会把 SPARQL 查询重写，尤其是变量结果集实例映射部分进行彻底的重构，并且对相关文件注册模块进行全面的接口升级，因此 DartGrid 的代码结构将会焕然一新，以下是笔者拟定的对 DartGridV4 的设计目标：

- 支持资源文件的动态安装
- 支持语义映射文件中包含自定制的 View 信息
- 支持包含非显示映射本体资源的 SPARQL 查询
- 支持包含 RDF:TYPE 的 SPARQL 查询（其中宾语为非变量）
- 支持包含 Literal 宾语三元组的 SPARQL 查询
- 支持 SELECT 从句中包含实例对象查询的 SPARQL 查询

DartGridV4 目标在查询重写过程中引入语义层，SPARQL 查询重写过程中能够进行的只有基于本体层次的推理（数据还没有就位），进一步的推理实现必须在数据封装阶段完成。

3.2.1 资源文件的动态安装

DartGridV3 将资源文件集成在模块中，需要添加的文件必须在程序运行前拷

贝指定目录下才有效，这样的做法无法满足调用模块希望能够在程序启动后另行指定资源文件的需求，因此，DartGridV4 为资源管理模块定制了文件读入接口，可以在代码中通过具体文件路径将指定文件读入，并且 DartGridV4 内核无需重启。

3.2.2 支持语义映射文件中包含自定制的 View 信息

对于不同地域不同部门服务器中存放科学数据的分布式异构数据库，他们都会极力避免来自不明对象的针对数据库的修改，即便是视图的添加，因此，作为数据封装的进一步体现，DartGrid 将会提供在本体映射文件中提供定义 View 信息的功能模块。这将进一步将封装。

3.2.3 支持包含非显示映射本体资源的 SPARQL 查询

这是体现推理的重要环节之一，如果仅仅能够查询显示映射本体资源，那么 DartGrid 不外乎是一种将某一种标准查询语句翻译成另外一种形式的查询语句，这样完全无法体现出语义网的巨大潜力。

比如我们有本体类中药，同时我们知道中药有两个子类方剂和草药，而其中方剂和草药都直接和数据库表映射，我们希望的是通过构造 SPARQL 查询所有的中药。这是一个很基础的基于继承关系的推导，在 SPARQL 中我们要查询的是中药，其实际语义是我们要查的是所有中药类实例和中药类子类所对应的实例。

之前的例子是基于类继承关系的推导，同时 DartGrid 也支持基于属性继承关系的推导，比如我们知道有一个属性名为 Label，之后它有两个子属性 AltLabel 和 PrefLabel，并且子属性和具体数据库列映射，我们希望通过构造 SPARQL 查询中草药的 Label。和前者相似，该 SPARQL 的实际语义是查询中草药的 Label 和所有的 Label 子属性所对应的值。

3.2.4 支持包含 RDF:TYPE 的 SPARQL 查询

SPARQL 中 WHERE 从句中出现的变量的具体含义是由他们所在的三元组所决定的，那么如果我们希望表示希望构造的是查询中药的 Label 的 SPARQL，就需要用 RDF:TYPE 来限定某一个查询变量的语义含义范围。其三元组形式应该形如（为了篇幅，这里省略了声明部分：

SELECT ?label

```
{?tcm      skos:label      ?label.
 ?tcm      RDF:TYPE     dart:中药.
}
```

3.2.5 支持包含 Literal 宾语三元组的 SPARQL 查询

在之前的 DartGrid 版本中，包含约束条件的 SPARQL 必须被写成：

```
PREFIX   ns:      <http://ccnt.cn/mt#>
SELECT DISTINCT   ?alias
WHERE{
    ?x   ns:hasAlias   ?a .
    ?a   ns:name         ?alias .
    ?x   ns:concept    ?concept .
    FILTER ( ?concept = '大黄')
 }
```

在上述 SPARQL 查询中，为了表述查询概念词为大黄，我们必须定义一个中间变量?concept 来做约束，在上一个版本的实现中，这是一种无奈的选择。

我们希望的 SPARQL 查询实际上是：

```
PREFIX   ns:      <http://ccnt.cn/mt#>
SELECT DISTINCT   ?alias
WHERE{
    ?x   ns:hasAlias   ?a .
    ?a   ns:name         ?alias .
    ?x   ns:concept   '大黄'.
 }
```

在这一版本中，包含 Literal 宾语的三元组的 SPARQL 将会同样被 DartGridV4 所理解。

3.3 DartGrid 运行模式

DartGrid 的主要目的是封装数据，然后通过 SPARQL 通过语义映射文件将 SPARQL 重写成多条 SQL 查询语句，获取事先注册的异构数据库的数据，并且进行语义包装成语义数据返回，所以可以将整个模块的运作分为三个阶段：(1).

SPARQL 翻译阶段；(2).数据库数据获取阶段；(3).数据封装阶段

3.3.1 SPARQL 翻译阶段

SPARQL 翻译阶段的主要目的是将 SPARQL 进行解析并封装成 SPARQL 信息类，结合语义文件、数据库语义映射文件和数据库配置文件，将 SPARQL 转换生成一个数据库 SQL 查询计划类。其中数据库 SQL 查询计划类应该是一个包含生成一至多个 SQL 查询所需要的完全信息。

这个阶段还可以粗分成三个阶段：

- 变量语义对象映射：SPARQL 最初会先经过语义层，被 SPARQL 解析类提取出每一个模块的信息（SELECT、WHERE、CONSTRAINT 等），封装成一个 SPARQL 查询信息类，封装了所有的查询信息，并且根据属性的 domain 和 range 以及 rdf:type 等作为限制，将部分资源变量映射到一个具体的语义类。这个阶段仅仅用到语义文件注册信息类。
- 变量数据库对象映射阶段：对于每一个 SPARQL 的三元组，根据属性在语义映射类中找到对应的 JOIN 关系，并且判断 JOIN 的主语表和宾语表所对应的语义类是否是该属性主语变量和宾语变量所允许的范围。生成模糊 SPARQL 解决方案，每一个 SPARQL 变量和多个数据库实例表映射，然后根据数据库实例之间本身的约束信息生成精确 SPARQL 解决方案。
- 最终 SQL 生成计划：此时 SPARQL 精确解决方案已经具备，然后将属性为 ObjectProperty 的三元组翻译成 Join，如果主语、宾语满足一定条件，可以将两个 Join 合并成一个，将属性为 DataTypeProperty 且宾语为变量的三元组翻译成 Select，将属性为 DataTypeProperty 且宾语为值得三元组翻译成 Constraint 信息

3.3.2 数据库数据获取阶段：

这个阶段中，传入的对象是一个 SQL 查询计划类，与其说是计划类，其实本质上只是包含所有数据库查询所需要的信息，但是 SQL 查询计划类未必仅仅是生成一个简单的 SQL，而可能生成一个 SQL 批处理操作，比如可能要从多个不同的表中获取数据后，然后根据特殊的方式进行连接。或者因为从不同的数据库获取数据，所以可能要从 A 数据库获取数据并进行投射（Project），然后在

B 数据库创建对应的临时表，将 A 提取的数据放置在 B 数据库的临时表中，然后进行 Join 操作，获取完数据后，将 B 数据库的临时表删除。这样的 SQL 查询计划也是可能的。

这个阶段也可以粗分成两个阶段：

- SQL 查询批处理语句的生成：现阶段已经支持的 SQL 批处理模式是 Optional 查询模式和跨库查询模式。
- SQL 查询结果获取：这一部分将已经查到的数据库数据简单的封装成一个自定制数据库结果集，因为可能涉及到从多个 ResultSet 进行结果合并，同时数据获取为了交互性能的考虑是以和 SQL 数据获取相异步的线程的形式存在的，也就是在查询开始的时候，数据获取线程就开始运作，当然如果数据没有就位则线程会被锁。

3.3.3 数据封装阶段

至今为止数据都是存放在 SQL 中，没有办法进行语义处理，在这个阶段中，数据库中的数据将会通过被封装成 Model 数据，每一个从数据库中查询到的记录都会被封装成数据库实例。

根据某种既定的非时序性方式生成 URI，作为封装后的本体实例的唯一标识，其列会根据本体映射文件的映射信息将列映射到本体属性。

3.4 体系结构

DartGridV4 语义网格内核采用分层结构，如图 4.1 所示，自顶向下依次是调用层，语义层，服务层，资源层。层次结构很好的简化了部件间的耦合，使得模块间结构清晰，也方便了流程的错误控制，对维护和代码重用都有着非常可观的用处。比如服务层的本体信息服务模块还可以为 DartSearch 所应用。

这些层次提供了语义网格内核两项基本功能：一是本体语义查询到数据库 SQL 查询的等价转换，二是执行数据库查询并对返回的结果进行语义包装。

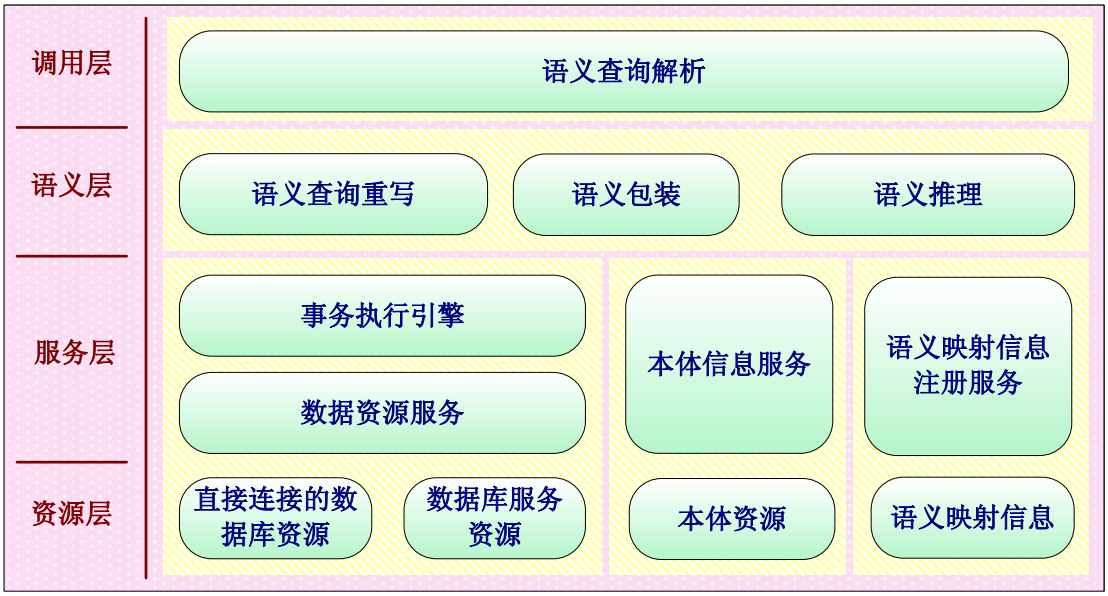


图 3.2 DartGridV4 内核体系结构

DartGridV4 语义网格内核各层的基本功能分述如下：

◆ **调用层**

调用层提供了 DartGrid 语义查询的调用接口，它接收符合 SPARQL 语法的本体查询串，对查询串进行解析，对查询字符串中的关键信息进行提取和包装。 封装的自定制 SPARQL 查询信息类传递给下层进行处理，待下层操作全部完成，调用层将把包装好的语义结果返回给终端用户。

◆ **语义层**

语义层的主要功能可以分为三类：

(1)、查询重写算法：将 SPARQL 翻译成 SQL 查询计划，这个过程会用到语义映射注册文件以及本体资源文件。

(2)、语义包装：将数据库获取到的元数据包装成语义资源，我们可以初步认为一条获取到的记录会被包装成一个本体实例。

(3)、语义推理：原本在查询重写中基于使用的语义框架包可能内置了一些基本的推理机，但是由于查询重写阶段中存放在数据库中具体数据并没有获取到，所以能够进行的推理只能是基于本体类和本体属性的继承特性的推理，而在获取数据后的这个阶段将要进行的推理就是针对实例数据本身的。

◆ **服务层**

服务层的主要功能可以分为两种类型：一类是提供元数据服务，包括维护本体模型，语义映射信息以及数据库资源信息。另一类是提供数据库事务

执行引擎，负责执行查询计划，并协调控制子计划之间的执行顺序和参数传递。

- **资源层**

资源层描述了三种资源元数据的存储格式，分别是数据库服务资源文件、语义映射信息文件和本体资源文件，并且针对每个模块提供了资源解析器，这些模块本身独立于 DartGrid，可以在其他应用中重用相关资源文件格式和资源解析器。

第4章 SPARQL 重写设计

4.1 SPARQL 简介

SPARQL 是由 W3C 制定和推荐的 RDF 查询语言[19]，它通过图形模式匹配实现对多个 RDF 图的查询。RDF 图本质上一个三元组的集合，每个三元组表示 RDF 图中两个数据节点之间的关联关系或者某个数据节点自身的属性。SPARQL 查询本身也是一个三元组表示的图，不过 SPARQL 的三元组中可以包含变量。

SPARQL 查询的基本模式就是三元组匹配，通过匹配得到查询变量和数据值的对应关系，这种对应关系在 SPARQL 中成为“绑定”（binding）。

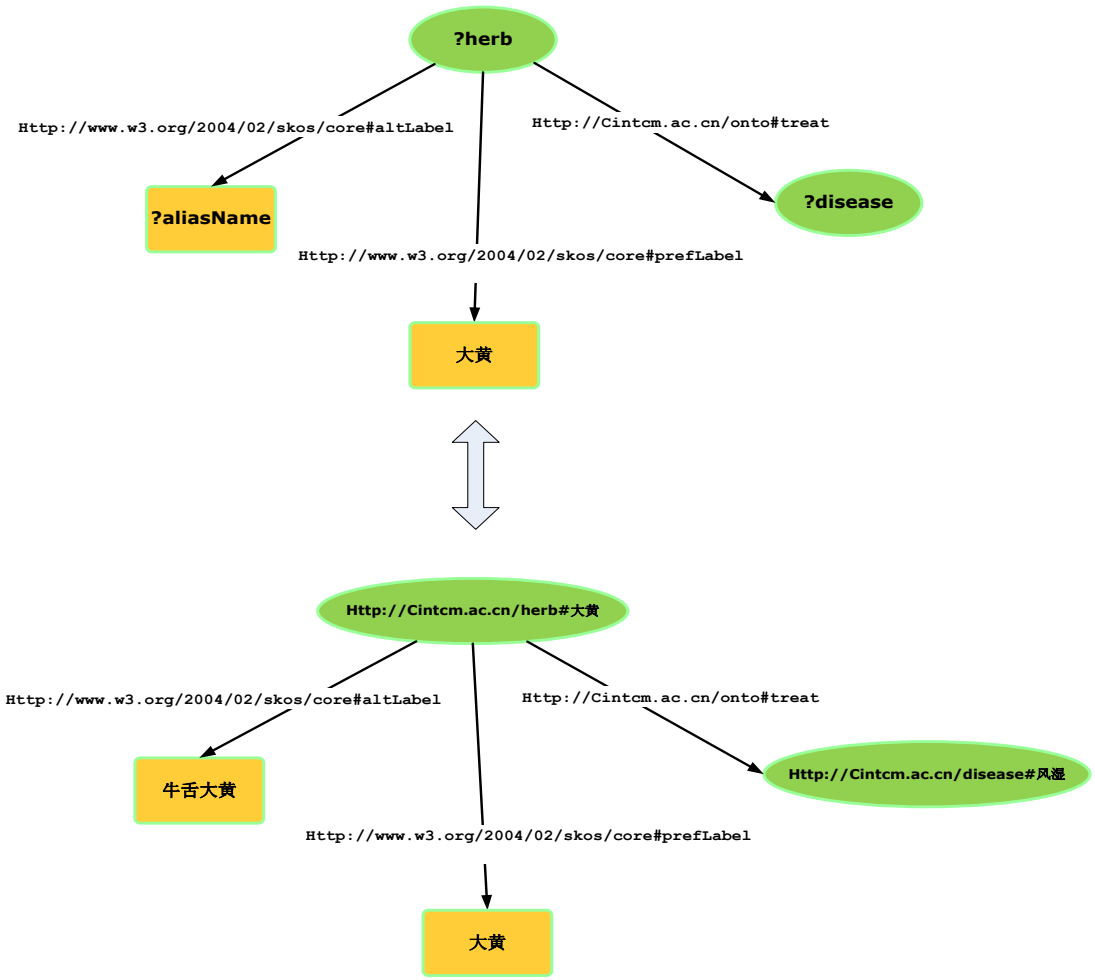


图 4.1 SPARQL 查询示意图

图 4.1 是 SPARQL 查询的一个示意图，查询结果是：

变量	结果	结果类型
?herb	http://cintcm.ac.cn/herb#大黄	实例
?disease	http://cintcm.ac.cn/disease#风湿	实例
?aliasName	牛舌大黄	值（Literal）

表 4.1 查询结果

4.1.1 基本语法

SPARQL 的语法形式与关系数据库中的结构化查询语言 SQL 比较相似，但仅仅是语法形式上的相似，两者是有本质区别的：SQL 是基于关系代数模型来构造查询的，而 SPARQL 是基于图的模型来构造查询。整体上来说，SPARQL 语句可以分成四个部分：声明部分，结果集，数据源，查询模式。为方便说明，先给出一个 SPARQL 查询的示例。

SPARQL 查询

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
SELECT ?who ?g ?mbox
FROM <http://example.org/dft.rdf>
WHERE
{
    ?g dc:publisher ?who .
    GRAPH ?g { ?x foaf:mbox ?mbox }
}
```

4.1.1.1 声明部分

在 RDF 数据图中包含两种基本的数据类型：一种是 Literal（文字型），它主要用来表示一些文字类型的值，比如某一个人的名字，家庭住址等。另一种是 IRI（Internationalized Resource Identifiers，国际化资源标识符），IRI 是在 URI（Uniform Resource Identifier，统一资源标识符）的基础上发展而来的，用于表示 Internet 网络上的各种资源实体。与 Literal 不同，IRI 只是一个资源代号，它本身并没有值。但是人们可以通过 IRI 找到对应的资源，并从 IRI 子节点中获取资源的具体信息。图 4.1 中椭圆型表示 IRI 类型，矩形表示 Literal 类型。

IRI 的语法结构可以分为前缀（prefix）和局部名称(fragment identifier)两部分，通常 IRI 的前缀比较长书写起来不方便，人们就用一个简单的词作为前缀的缩写。声明部分主要声明这些前缀的缩写。例 5.1 中 PREFIX foaf: <http://xmlns.com/foaf/0.1/> 和 PREFIX dc: <http://purl.org/dc/elements/1.1/>就是前缀声明。

4.1.1.2 结果集

SPARQL 查询语法中规定了四种结果集形式：SELECT, CONSTRUCT, DESCRIBE 和 ASK。其中，SELECT 结果集以一张表的形式返回所有匹配的结果，表的列是变量名，每一条记录表示一个的变量绑定。CONSTRUCT 结果则是每一条结果记录表示成为 CONSTRUCT 子句描述的图的形式 DESCRIBE 结果集由 SPARQL 查询处理器对某一个变量进行描述，具体描述的内容取决于查询处理器的实现，在 SPARQL 语法标准中没有规定。ASK 结果集询问当前 SPARQL 查询是否匹配成功，若成功则返回 YES，失败则返回 NO。在这四种结果集中，最常用的是 SELECT 结果集。

4.1.1.3 数据源

数据源部分相当于 SQL 中的 FROM 子句，它规定了本次查询的 RDF 数据集。数据源中一般会给出几个 RDF 文件的 IRI 引用，SPARQL 查询处理器可以从相应位置获取这些 RDF 数据。数据源在 SPARQL 中通常是可以省略的。

4.1.1.4 查询模式

查询模式描述了一个查询子图，这个图中包含 SPARQL 变量和一些约束条件。查询处理器依据这个子图去搜索 RDF 数据集，并返回匹配的数据结果。与 RDF 图的表现形式一样，SPARQL 查询模式也是由三元组组成，另外为了实现一些基本的查询约束，SPARQL 语法标准还规定了一组条件谓词：比如像 ORDER BY, DISTINCT,LIMIT，OFFSET 等等。

4.2 系统设计分析

DartGridV4 的关键改进，在于 DartQuery 模块中对 SPARQL 查询重写部分对于语义映射进行了更符合实际语义环境的诠释。模块对本体概念和数据库概念的映射给出了清晰且逻辑性更强的映射关系图。

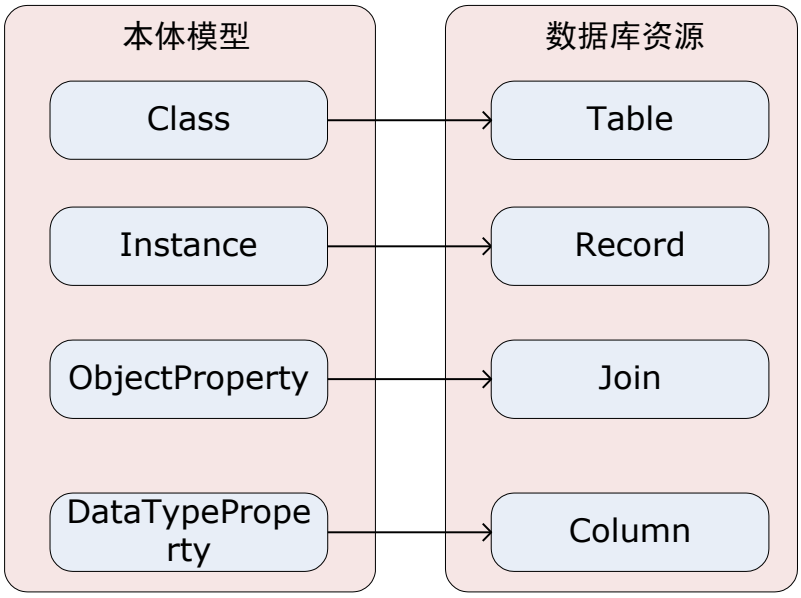


图 4.2 本体模型数据库资源映射图

如图 4.1 所示，本体类将会被映射到数据库表，对象属性（描述本体实例关系）将会被映射到数据库资源的联结信息，值属性（描述本体实例本身性质）将会被映射到数据库列对象。

当然这里必须要重点强调的是本体实例会被映射到一个数据库记录，这个记录所代表的不是某一张表中存放的某一行记录，而是某一次具体 SPARQL 查询中所获取到的可能来自多张表连接后形成的某一张临时表中所对应的记录，将记录和本体实例映射是前文提到的数据库数据封装阶段的工作，完成这项任务不仅需要记录提供完整的属性数据，而且需要建立一种机制使得记录能够生成一个唯一资源标识符 URI，作为本体实例对象不可或缺的标识。

4.2.1 SPARQL 和 SQL

创建、读取、更新和删除（Create/Read/Update/Delete，CRUD）操作是数据库操作中最基本也是重要的，CRUD 操作通常是使用关系数据库系统中的结构化查询语言（Structured Query Language，SQL）完成的。随着 Web 变得更加具有面向数据特性，因此需要从基于 SQL 的 CRUD 操作转移到基于语义 Web 的 CRUD 操作。

我们都知道在实际操作中 SQL 可能非常复杂，但这里只是基于 SQL 的基础结构 select * from*where *进行分析。

用一个很简单的实例来说明，查名为 John Smith 的人的住址、生日、和真实姓名

SQL 部分：

```
SELECT realname, dob, location
FROM UserTable
WHERE realname = "John Smith";
```

SPARQL 部分：

```
PREFIX foaf:
<http://xmlns.com/foaf/0.1/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT ?uri ?name ?dob ?location
FROM
<http://www.example.org/graph>
WHERE
{ ?uri  rdf:type  foaf:Person ;
foaf:name  "John Smith" ;
foaf:birthday  ?dob ;
foaf:location ?location .
} ;
```

在比较两句查询语句时，表面上看，SPARQL 版本明显比 SQL 版本长很多，但是做个如果 birthday、name、location 等数据分别存放在三个数据库表中，SQL 将会变得非常的庞大，其中将会充斥着许多 Join 关系。

比较完长度以后从语句格式上可以很明显的注意到他们在 where 部分的不同，这就是两者最根本的不同之处，sparql 在 where 部分是基于三元组的一种限制方式，其中即可以表现为对属性的限制，或者对某一个变量的声明，或者表示和另外一个实体的关系。

4.3 SPARQL 查询重写计划

我们假定三元组为(S, P, O)，其中 S 为主语，P 为谓词，O 为宾语，由于在实现上还存在着难题，我们假定 S、O 只有两种选择，变量或者 Literal，换言之，在现在的 DartGrid 中，S 和 O 不可能是实例 URI。我们假定 dart:ObjecProperty 为所有对象属性的代称，dart:DataTypeProperty 为所有值属性的代称，由此我们

可以看出 SPARQL 的 WHERE 模块中可能出现的三元组可能会是以下三种形式：

类型 ID	主语类型	谓词类型	宾语类型
A	变量	ObjectProperty	变量
B	变量	DataTypeProperty	变量
C	变量	DataTypeProperty	值

表 4.2 三元组描述表

- **A 类三元组：**

因为谓词是某个对象属性，所以主语和宾语本质上都是实例，当然在查询过程中他们的限定因素是类，SPARQL 查询变量会首先在语义层被翻译成本体类，该本体类的作用是作为对数据库实例表筛选的限制条件，只有当某个数据库实例表所对应的本体类是该限定类或其子类，该数据库实例表才可能被看作该 SPARQL 查询变量的映射对象。整个三元组在语义层的含义是两个实体之间的关联，在数据库视图下会被翻译成一个数据库表联接（Join）信息。

- **B 类三元组**

因为谓词是某个值属性，所以主语是一个本体实例，宾语是一个 Literal，该三元组的价值在于确定了?varB 的身份，如之前所说，在 SPARQL 中任何一个变量的含义都是由 WHERE 从句的三元组集决定的。

- **C 类三元组**

很明显这是一个值限制条件，我们判断某一个类实例的某个属性是否为某个确定的值，将会被翻译成 SQL 中的 WHERE 从句中的一个 FILTER。

根据以上所述，SPARQL 查询重写的关键部分就是把以上三种类型的三元组正确的翻译成 SQL 查询计划中的成员，因为 SPARQL 集合中的三元组集合彼此构成了某种约束关系，每一个变量在一个结果集中应该是唯一的，因此，我们需要做的事确定每一个 SPARQL 查询中的变量，将其无二义地映射到某个唯一的数据库实例中，就可以确定一个唯一的 SQL 计划，当然因为一个本体可以映射到多张数据库实例表，所以一个 SPARQL 可能能够翻译成多个 SQL 计划，但是对于一个 SQL 计划中，一个明确的 SPARQL 结果集中每一个变量应该能够唯一的转换成一个数据库实例。

第5章 SPARQL 重写实现

5.1 SPARQL 查询重写流程

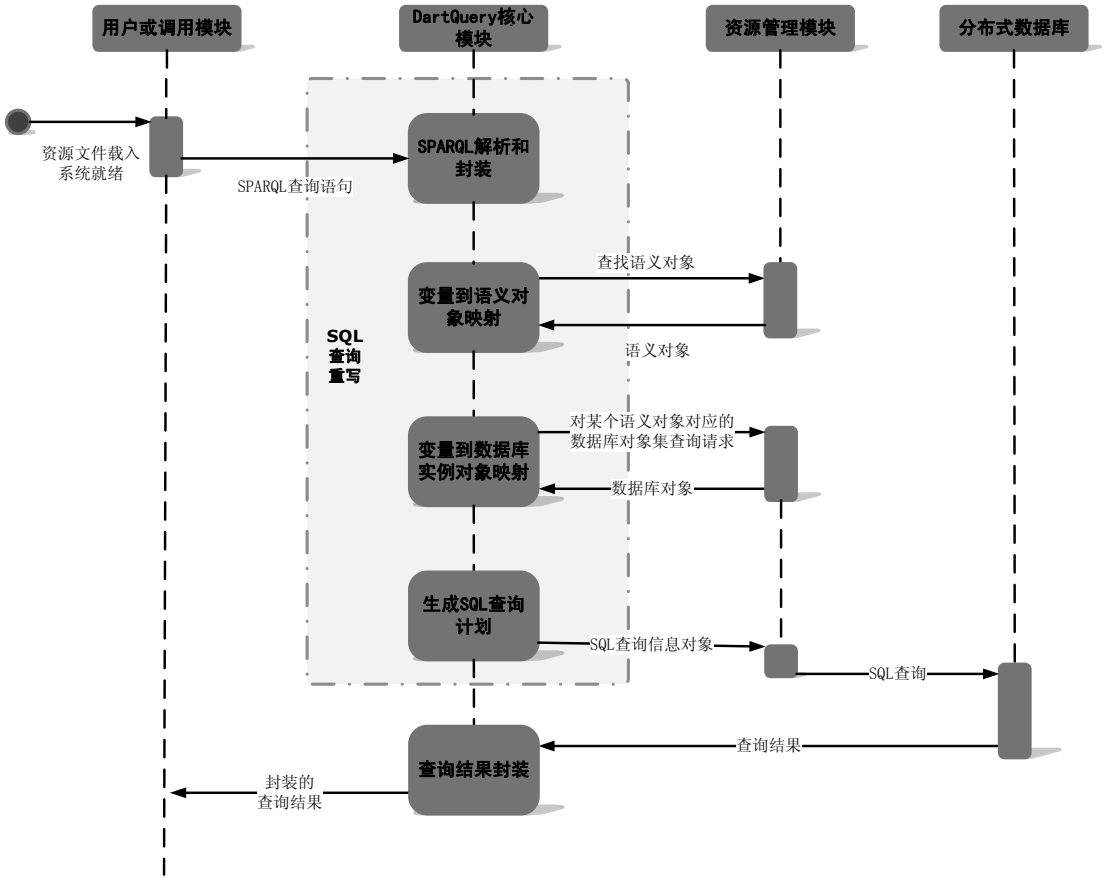


图 5.1 DartGridV4 时序图

图 5.1 展示了 DartQuery 查询重写的整个过程。其中资源管理模块包含本体信息管理模块、本体映射信息管理模块和数据库资源管理模块。

在资源文件装载成功，系统就绪后，用户或上层调用模块传入 SPARQL 查询，发送到 DartQuery 核心模块，DartQuery 通过 SPARQL 解析封装后，将 SPARQL 包装成自定制的 SPARQL 信息类，如果传入的 SPARQL 语句满足 SPARQL 规范并成功封装，DartQuery 将根据 SPARQL 中每个三元组，其中包括 RDF:TYPE，确定 SPARQL 查询中每个变量所映射本体对象。然后根据语义映射文件，将每个已经映射到本体对象的变量进一步映射到唯一的数据库实例中，并生成数据库查询计划对象，SQL 查询计划对象根据数据库信息管理模块查找对应的数据

库连接信息（可能是数据库连接池）并向目标数据库发出查询请求，目标数据库将数据返回给 DartQuery 核心模块。最终，DartQuery 将 SQL 结果集进行语义封装成语义对象结果集返回给上层调用。

DartGrid 运行模式可以粗分成三个阶段：SPARQL 查询重写、数据库数据获取和数据封装。其中本次 DartGrid 升级的重点放在了 SPARQL 查询重写部分。下文中流程说明中若涉及 DartGrid 模块限于篇幅不再进行详细介绍，对模块的描述请参考表 5.1。

模块名称	模块描述
SPARQL 解析模块	负责和用户交互的功能模块，用户通过调用曾模块传入 SPARQL 查询并获取查询结果
模糊 SPARQL 结果集	每个 SPARQL 变量映射到一或多个数据库实例对象，数据库实例对象间存在着约束。可以通过模糊 SPARQL 结果集生成一或多个明确 SPARQL 结果集
明确 SPARQL 结果集	每个 SPARQL 变量映射到唯一一个数据库实例对象。可以通过明确 SPARQL 结果集生成 SQL 查询计划。
SQL 查询计划	包含生成一组 SQL 查询所需要的所有信息，包括查询关键词，分页大小，SELECT 从句对象集，FROM 从句对象集和 WHERE 对象从句集。
本体信息管理模块	管理本体信息的模块，能够根据 URI 返回相应的本对象（类、属性），同时提供一些本体类操作接口（比如返回一个类为传入两个类的交集）
语义映射管理模块	管理映射信息的模块，能够根据本体实例 URI（类、属性）返回数据库实例 URI 集合（表、列、联接关系），同样也能够通过数据库实例 URI（表、列），返回本体实例 URI 集合（类、属性）
资源信息管理模块	管理数据库资源信息的模块，能够根据数据库实例 URI（类、属性）返回数据库实例，在注册数据库后自动生成相应的数据库连接池，并且可以根据数据库 URI 获得相应的连接池。

表 5.1 DartGridV4 重要模块描述表

5.1.1 SPARQL 查询分析

图 5.2 SPARQL 查询分析流程图

这一阶段将完成对 SPARQL 查询字符串进行信息提取和封装。

SPARQL 中最重要的信息提取 WHERE 从句的 BLOCK 信息，每一个 BLOCK 都是一组必须满足的三元组信息，其中 BLOCK 数组一般包含 Optional 的 BLOCK 数组。SPARQL 查询的基本模式是三元组匹配，并通过这种匹配获得变量和数据值的对应关系，术语称为“绑定”。

SPARQL 信息提取过程实现利用了 Jena 包的 ElementVistor 类和 QuerySerializer 类，将相关的信息解析函数重载完成自定制解析的工作，其中，ElementVisitor 主要提供了访问 SPARQL 的 WHERE 从句的相关函数接口，而 QuerySerializer 主要提供了 SPARQL 的诸如 SELECT、FROM、ASK 等关键从句的访问函数。

5.1.2 变量到语义对象映射

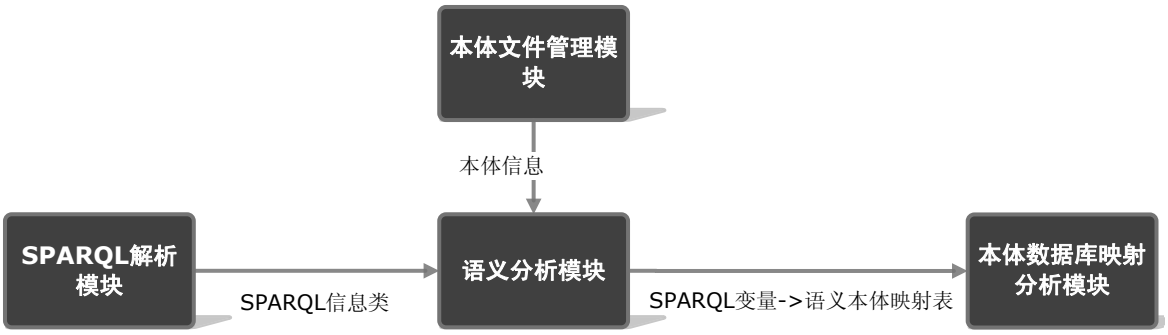


图 5.3 变量到语义对象映射流程图

这一阶段将完成的是对 SPARQL 中变量和语义对象的映射工作。

模块需要完成的是通过本体文件管理模块提供的本体信息，SPARQL 信息类中的变量映射到具体的本体类。

具体的实现算法是，针对每个三元组（S， P， O）

通过谓词的 Domain 和 Range 对 S 和 O 进行语义类限定，并添加到语义本体映射表中，如果谓词是 RDFTYPE，那么直接通过 O 的本体类 URI 对 S 进行

语义类限定。如果语义本体中已经包含需添加的映射变量，那么就对两者的语义信息进行语义类求交。语义类求交利用的是本体文件管理模块的提供的接口，利用 OWL 框架语言的 Restriction 关键词定义一个新的语义本体类，该类为之前传入接口的两个类的交集。

如果属性 Domain 或 Range 为空，模块将会默认 Domain 或 Range 为类继承树中顶端类，即其他所有类都是该类的继承类。变量到数据库实例映射

5.1.3 变量到数据库实例映射

这一阶段将完成的是变量到数据库实例对象的映射工作。

模块需要完成的是通过语义映射管理模块提供的映射信息，把 SPARQL 查询变量映射到具体的数据库实例中。

这一阶段还可以细分成两个阶段：模糊 SPARQL 结果集生成和明确 SPARQL 结果集生成。

5.1.3.1 模糊 SPARQL 结果集生成

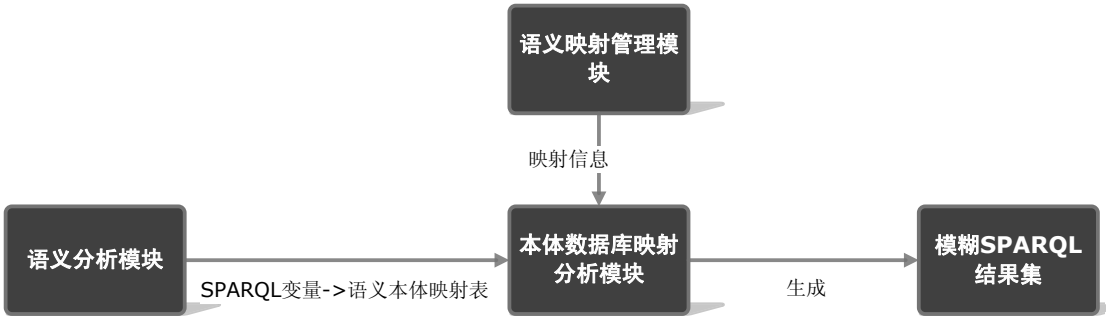


图 5.4 模糊 SPARQL 结果集生成流程图

在这个阶段，模块需要完成的算法是通过语义映射文件提供的映射信息和语义分析模块提供的 SPARQL 变量->语义本体映射表，将 SPARQL 查询中的变量具体映射到一或多个数据库实例对象。

具体的实现算法是，对于每个三元组（S，P，O）：

根据谓词的 URI 通过语义映射管理模块接口找到某一个数据库 Join 描述，判断该 Join 的主语（数据库实例表）所对应的本体类，是否是 S 所对应的本体类或其子类，如果满足，则主语表认为合格，反之不合格，同样的流程对 Join 的宾语进行判断。只有主语表和宾语表都合格的情况下，该 Join 被采纳，则主语表和宾语表分别被添加到 S 和 O 的数据库实例映射集合中。

5.1.3.2 明确 SPARQL 结果集生成

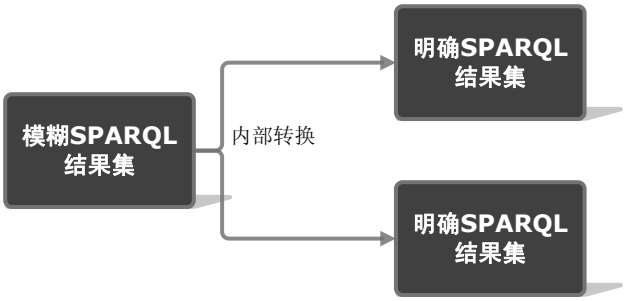


图 5.5 SPARQL 明确结果集生成流程图

这一阶段将完成模糊 SPARQL 结果集到多个明确 SPARQL 结果集的转换。

模糊 SPARQL 结果集的变量数据库实例无法直接转换成 SQL 查询计划，每个变量所对应的数据库是集合间还存在数据库视图下的约束。在这个阶段中，DartQuery 模块会重新遍历每个三元组（S，P，O），如果 P 是对象属性，那么 S 和 O 应该是一个已注册联接（Join）的左联接表和右联接表，如果 P 是值属性，那么 O 应该是 S 的某一列。通过这样的数据库视图下的限制，将每一个 SPARQL 查询和变量和唯一的数据库实例对象映射。生成明确 SPARQL 结果集。

5.1.4 SQL 计划对象生成

这一阶段核心模块将 SPARQL 明确结果集生成 SQL 计划对象。

对于 SPARQL 查询 WHERE 从句中每个三元组（S，P，O），如果该三元组为（描述所用三元组类型，具体请参考表）

- A 类三元组，即（变量，对象属性，变量），通过明确结果集中的语义属性到 Join 映射，根据对象属性获取到联接（Join）映射，并且判断该联接信息能否和其他联接信息合并。经过合并处理的联接集将会被转换为 SQL 的 FROM 信息。
- B 类三元组，即（变量，值属性，变量），主语变量和宾语变量的映射都已经在明确 SPARQL 结果集中生成了，对应的是 SQL 的 SELECT 信息。
- C 类三元组，即（变量，值属性，值），根据明确结果集通过三元组本身获取到列信息，该信息会被转换成 SQL 的 CONSTRAINT 信息（即 WHERE 从句内容）。

SPARQL 的 FILTER 信息也会被转换为 SQL 的 CONSTRAINT 信息。

5.2 实现结果和分析：

为了展现这次 DartGrid 的表现情况，定制了以下资源文件环境：

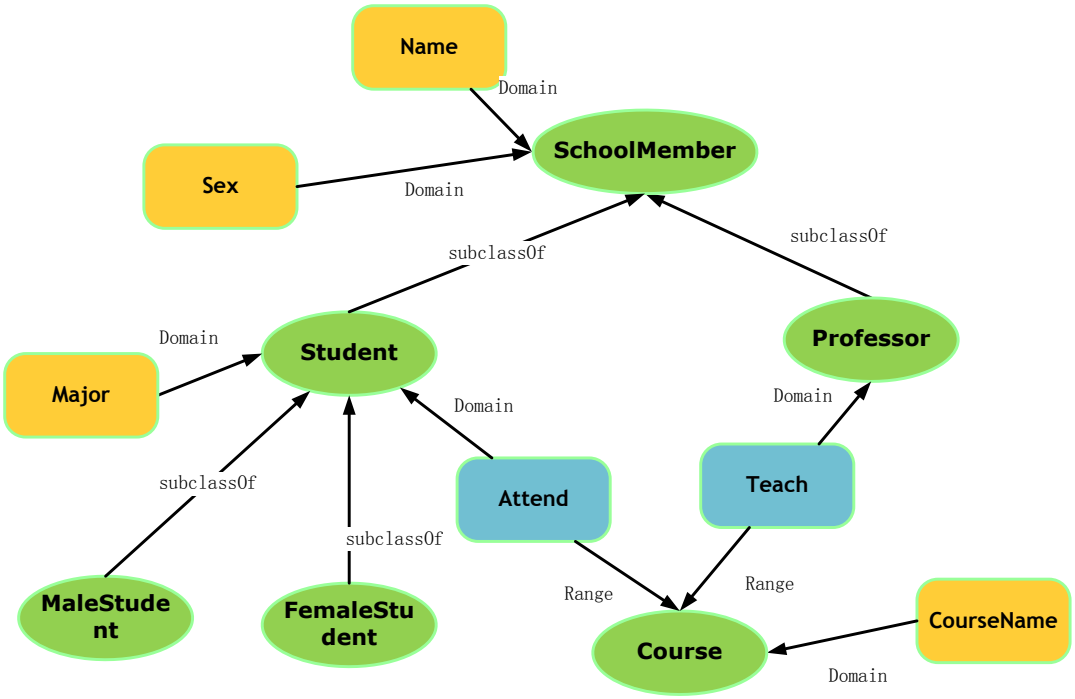


图 5.6 学生案例本体图

本体模型简单介绍：本体类包括：SchoolMember（学校成员），Professor（教授），Student（学生），MaleStudent（男学生），FemaleStudent（女学生），Course（课程）。对象属性包括：Attend（参加），Teach（执教）。值属性包括：Major（主修），Sex（性别），Name（姓名），CourseName（课程名称）。

模型中本体类之间可能存在以下关系，学生能够参加课程，教授能够执教课程。男学生和女学生都是属于特殊的学生，可以认为 Sex 为 Male 的学生即男学生，为 Female 的学生即女学生。

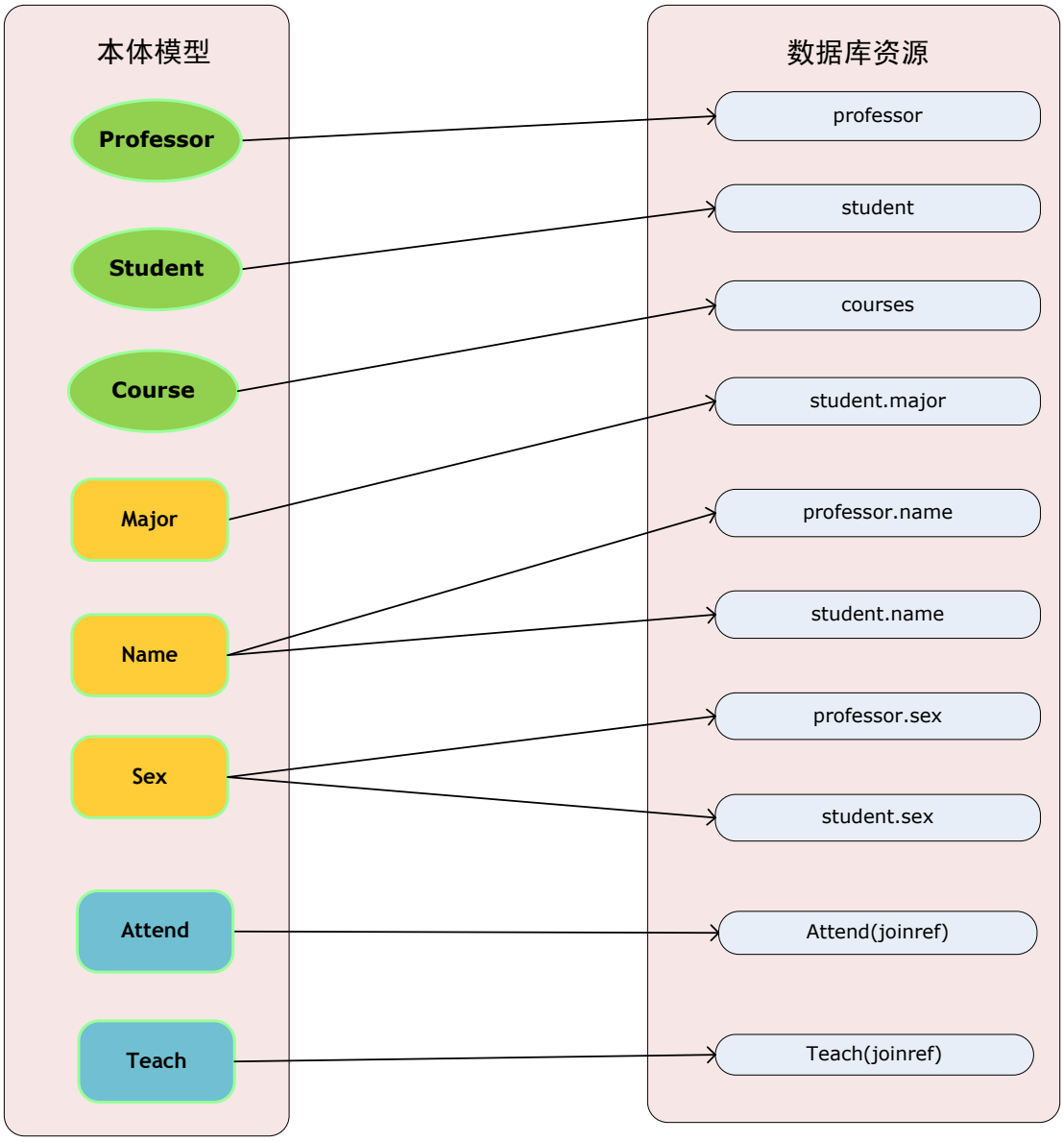


图 5-4 案例本体到数据库实例映射图

本体文件和数据库实例的映射关系如图所示：

映射关系如前文所提到，本体类和数据库表映射，值属性和数据库列映射，对象属性和联接信息映射，其中数据库实例中的联接信息使用联接引用代替，本体映射和数据库实例是一种多对多的映射，一个本体对象可以映射到多个数据库实例中，一个数据库实例也可以被多个本体对象所映射。

前三个案例中将使用上述案例环境

5.2.1 支持包含 RDF:TYPE 的 SPARQL 查询

SPARQL 查询：

```
PREFIX slayer: <http://dart.zju.edu.cn/slayer#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT distinct ?stu_name, ?stu_sex
WHERE {
    ?stu    slayer:Name     ?stu_name .
    ?stu    slayer:Sex      ?stu_sex .
    ?stu    rdf:type        slayer:Student .
}
```

查询含义：查询所有学生的姓名和性别

结果集：

```
stu_name        |   stu_sex
---------------------------------
Saber           |   female
Barsarker       |   male
Lancer          |   male
Archer          |   male
Caster          |   female
Rider           |   female
Assassin        |   male
```

结果分析：结果集中只有学生的名称（老师的名称中包含 Mr），证明了对 RDF:TYPE 的基本支持已经实现，这是语义查询中重要的关键词，如前文所言，SPARQL 变量的含义是由查询的 WHERE 从句中形成的 RDF 图所决定，因此通过 RDF:TYPE 来指定所要查询的某一个变量的语义身份是最直观也是最有效的方法。

5.2.2 支持包含非显示映射本体资源的 SPARQL 查询

SPARQL 查询：

```
PREFIX slayer: <http://dart.zju.edu.cn/slayer#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
```

```
SELECT distinct ?person_name    ?person_sex
WHERE {
    ?person      slayer:Name      ?person_name .
    ?person      slayer:Sex       ?person_sex .
    ?person      rdf:type         slayer:SchoolMember .
}
```

查询含义：查询所有学校成员（包括学生和教授）的姓名和性别

结果集：

```
stu_name            |    stu_sex
-----------------------------------------
Saber               |    female
Barsarker           |    male
Lancer              |    male
Archer              |    male
Caster              |    female
Rider               |    female
Assassin            |    male
Mr Neumann          |    male
Mr ShakeSpeare      |    male
Mr Newton           |    male
Mrs CYJJ            |    female
```

结果分析：查询结果中包含了学生信息和老师信息，说明基于非显示关联的查询已经支持，因为根据图的映射关系来看，SchoolMember 并没有显示和数据库中的任何一张表进行映射，能够查询到数据说明 SchoolMember 已经被转译成其子类，并通过子类根据语义映射文件找到了实际对应的数据库表，能够通过知识库推导出隐式需求，说明 DartGrid 已经具备了基本的基于类层次集成的推理功能。

5.2.3 支持包含 Literal 宾语三元组的 SPARQL 查询

SPARQL 查询：

```
PREFIX slayer: <http://dart.zju.edu.cn/slayer#>
```

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT distinct ?person_name  ?person_sex
WHERE {
    ?stu    slayer:Name     ?person_name .
    ?stu    slayer:Sex      ?person_sex .
    ?stu    rdf:type        slayer:Student .
    ?stu    slayer:Attend   ?class .
    ?class  slayer:CourseName  'ComputerScience' .
}
```

查询含义：查询参加 ComputerScience 的所有学生的姓名和性别

结果集：

```
stu_name        |   stu_sex
-------------------------------------------------
Saber           |   female
Barsarker       |   male
Lancer          |   male
Archer          |   male
Rider           |   female
```

结果分析：查询结果中包含了所有参加该课程的学生名称，查询正确。这个 SPARQL 查询的意义有两个，首先，该查询是关联查询，通过某一个本体节点对另外一个节点进行约束，查询含义为参加某个特定课程的学生信息，这说明 DartGrid 可以进行关联搜索，可以进行类似“能够治疗某种疾病的所有药物所包含的所有药物成分的信息”这样的查询。其次，该查询中的基于 Literal 的限定条件是以三元组的方式在 Where 从句中进行添加，这是因为现阶段的模型已经是基于图 6-2 的映射结构进行设计，变量的映射更符合逻辑。

5.2.4 支持语义映射文件中包含自定制的 View 信息

对语义映射文件的文件格式做了调整，允许添加 View 信息，View 自身描述信息的基本结构如图所示。View 在完成定义后和一般的数据库表在使用上没有任何不同，除了在最后的 SQL 查询中 View 会被翻译成一段 SQL 语句（原本这一段工作为在数据库进行）而不是一个 View 名称。

View 信息包括 View 相关表的描述，连接信息描述和约束描述。

View 相关表的描述，包含一个表 URI 和一组列说明，所有 View 相关表中描述的列都会作为 View（可以看作是伪表）的新列，而列定义中使用的别名将作为 View 中该列的列名，如果列定义中没有使用别名则将该列本名作为 View 中该列的列名。View 相关表可以是另外一个 View。

连接信息描述 View 相关表之间的连接信息，其中包括相连的两个表以及相连表条件的描述，多个联接之间形成无环单图，所以会合并成一个联接信息。作为 View 的查询语句中 SELECT 部分。

约束信息描述了对某一个表内容的限定信息，包含了该表指定的类，和约束操作内容（包括正则运算，或者各种比较运算），以及约束值（正则表达式，数字或者字符串）。

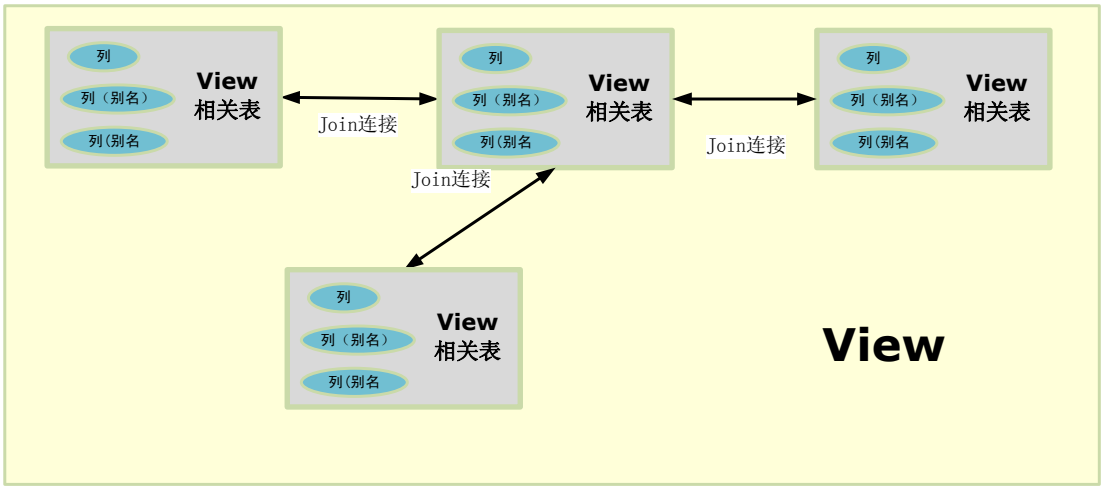


图 5.5 View 格式说明图

对于当前的 View 展现我们对案例资源环境进行了一些补充，首先将学生的映射表进行了视图扩展，将学生表和家庭信息表(family_info)以及个人信息表(personal_info)进行了联接关联(如图 5.6 所示)，同时在本体中添加 MaleStudent，并且与新创建的 MaleStudent 视图进行映射，MaleStudent 视图即所有性别为 male 的学生集合。

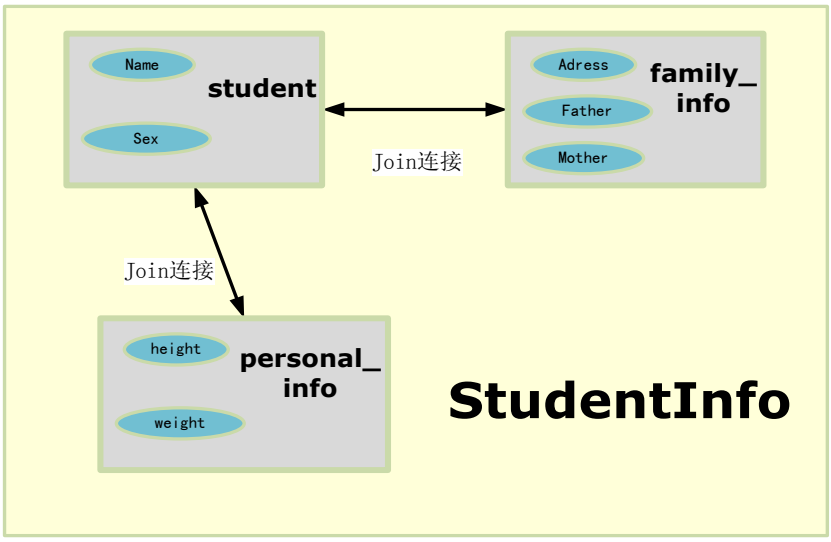


图 5.6 学生信息视图

案例：

SPARQL 查询：

```
PREFIX slayer: <http://dart.zju.edu.cn/slayer#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT distinct ?person_name  ?person_height  ?person_address
WHERE {
    ?stu    slayer:Name     ?person_name .
    ?stu    slayer:Height   ?person_height .
    ?stu    slayer:Address  ?person_address .
    ?stu    rdf:type        slayer:MaleStudent .
    ?stu    slayer:Attend   ?class .
    ?class  slayer:CourseName  'ComputerScience' .
}
```

查询含义：查询参加 ComputerScience 的所有男学生的姓名、身高和住址

结果集：

person_name	personal_height	personal_address
Barsarker	125	ShangHai
Lancer	122	Beijing
Archer	183	Nanjing

结果分析：此次查询对经过拓展的学生信息视图进行了搜索，查询结果根据之前的搜索结果可以看出是正确的。

如图 5.6 所示学生信息由三张表所组成，通过 SID 关联，但是在 SPARQL 查询语句中这些信息却是完全被隐藏的，通过语义文件支持 View 定义的手段，我们可以对某个本体概念进行拓展，即便某个本体（比如学生）的属性信息（诸如身高，家庭信息）分别存放在多张表中，我们可以通过映射文件将这些凌乱的信息通过视图中表联接封装到一个本体中。这样满足了不同用户对同一个数据库进行不同定制的需求。自定制视图还包含了 Join 信息以外的信息，这里查询中我们限定为男学生，因此自定制视图还包含着 Constraint 信息，在这里约束信息如前文所说 sex=male 的学生为男学生。

第6章 总结和展望

本文介绍了语义网的相关知识和 DartGridV4 对语义层的引入和基本推理功能的实现过程和结果。

从现在的运行情况来看，DartGridV4 查询内核首先相比较 DartGridV3 的进步是革命性的，查询重写的再实现使得 DartGrid 真正具备了语义层，在 Sparql 的支持度上有了显著的提高，映射文件也有了很大的提高，同时一定程度上体现本体技术的推理能力，基于本体层次结构查询到非显示映射的本体类实例信息，当然这样的推理能力还远远无法匹配 OWL 的表述能力所内涵的逻辑性，OWL-DL 对本体类的多变的定义方式（基于 Restriction 的本体定义）和本体属性的各种类型（传递属性、对称属性等），这些表述能力都包含着很多隐含的逻辑。而当前的 DartGrid 还不具备处理这些 OWL 特性的能力，但是由于 DartGrid 已经加入了语义层，这使得实现其他 OWL 特性都成为了可能，当然不可否认，DartGrid 要走的路还很长。

当然通过查询重写来实现基于语义查询本身可能就是一种过渡手段，DartGrid 相比较 Jena 的 Sparql 查询现阶段的优势在于数据的获取速度，Jena 尽管对 SPARQL 的支持很强大，但是从没有经过预处理优化的文件中将文件转换成 Model 再进行查询，在数据量很大的情况下，本体文件冗余的表达形式对 XML 解析本身就是一种很大的负担，而且 Jena 在文件数据块的读写组织上也远远无法和数据库媲美，因此，在现阶段，即便没有完全体现出 OWL 的特性，但是 DartGrid 在性能上的优势还是毋庸置疑的。

DartGridV4 从现状来说完成的事语义层的嵌入和轻量级推理的支持，所谓推理，就是能够通过显示条件根据知识库推导出隐式条件的一种逻辑能力，而推导出隐式条件本身就是一种分析能力。如果 DartGrid 拥有的框架结构能够更好的适应推理，而且具备强大的知识库，那么 DartGrid 将可能具备十分可怕的隐式条件推导能力，如果运用在科学领域那么我们可以根据当前现有的科学知识进行知识再发现，或者运用在犯罪学判断一系列话语中是否存在逻辑矛盾，或者运用在搜索引擎中，让自动代理程序能够对现有资源进行更好的组合，为网络用户提供诸如资源推荐或者行程安排这样的活动。这些应用的实现都将对资源尤其是网络资源的利用将会带来革命性的改变。

参考文献

[1] Zhaohui Wu, Huajun Chen, Heng Wang, Yimin Wang, Yuxin Mao, Jinmin Tang, and Cunyin Zhou. Dartgrid: a Semantic Web Toolkit for Integrating Heterogeneous Relational Databases

[2] TCM Application : http://ccnt.zju.edu.cn/projects/dartgrid/tcmgrid.html

[3] Ivan Herman. Introduction to the Semantic web 2003-11

[4] W3C. http://www.w3.org/2001/sw/.

[5] Tim Berners-Lee, James Hendler and Ora Lassila. The Semantic Web, A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities, May 17, 2001.

[6] Huajun Chen, Zhaohui Wu, Chang Huang, Jiefeng Xu. TCM-Grid: Weaving a Medical Grid for Traditional Chinese Medicine. International Conference on Computational Science 2003: 1143-1152

[7] Zhaohui Wu, Huajun Chen, Jiefeng Xu. Knowledge Base Grid: A Generic Grid Architecture for Semantic Web. J. Comput. Sci. & Technol. July 2003, Vol.18, No.4, pp.462-473

[8] Zhaohui Wu, Huajun Chen, ShuiGuang Deng, Yuxing Mao. DartGrid: RDF-Mediated Database Integration and Process Coordination Using Grid as the Platform. APWeb 2005:351-363

[9] Zhaohui Wu, Huajun Chen, Chang Huang, Guozhou Zheng, Jiefeng Xu. DartGrid:Semantic-Based Database Grid. International Conference on Computational Science 2004:59-66

[10] HP Labs. Jena 2 - A Semantic Web Framework.
Available at http://www.hpl.hp.com/semweb/jena2.htm

[11] Dan Brickley and R. V. Guha. Resource Description Framework (RDF) Schema Specication 1.0. W3C Recommendation, 27 March 2000. Available at http://www.w3.org/TR/2000/CR-rdf-schema-20000327/

[12] Deborah L. M., Frank V. H. OWL Web Ontology Language (OWL) Overview. http://www.w3.org/TR/owl-features/.

[13] Grigoris Antoniou, Frank van Harmelen. A Sementic Web Primer. MIT Press, 2004, 10

[14] Neches R ,Fikes R E ,Gruber T R ,et al. Enabling Technology for Knowledge Sharing. AI Magazine ,1991,12(3) :36～56

[15] Gruber T R. A Translation Approach to Portable Ontology Specifications. Knowledge Acquisition ,1993,5 :199～220

[16] Mike Dean, Dan Connolly, Frank van Harmelen, James Hendler, Ian Horrocks,Deborah L. McGuinness, Peter F. Patel-Schneider, and Lynn Andrea Stein. OWL web ontology language reference. W3C Working Draft, 31 March 2003l

[17] Frank van Harmelen, Ian Horrocks, and Peter F. Patel-Schneider. A model-theoretic semantics for DAML+OIL (March 2001). Available at http://www.w3.org/TR/2001/NOTE-daml+oil-model-20011218.

[18] Ian Horrocks, Peter F. Patel-Schneider, and Frank van Harmelen3. From SHIQ and RDF to OWL, The Making of a Web Ontology Language

[19] SPARQL Query Language for RDF. http://www.w3.org/TR/rdf-sparql-query/

致谢

在论文完成之际，谨向我尊敬的导师吴朝晖教授表示由衷的感谢！

在吴老师的带领下，我进入了语义数据库网格的研究，吴老师在学业上予以悉心指导和培养，使我在学术领域和人生哲学方面均有了长足的进步。吴老师所建立的自由宽松、团结合作的学术气氛和良好研究、学习环境，为我学习和工作创造了前提条件；吴老师的广博学识、严谨学风、认真态度、创新精神和对科学事业的不懈追求，是我永远学习的楷模。

衷心感谢实验室的陈华钧老师和姜晓红老师，两位指导老师在研究上的求真务实精神使我获益良多，感谢陈老师和姜老师在研究上给予的指导与帮助。

衷心感谢实验室使用本模块的王军健师兄和宓金华师姐，感谢你们不时为内核模块提出需求并提供测试数据以及错误报告，这些对模块的稳定性和功能升级都有着至关重要的作用。

衷心感谢实验室内核组的其他师兄：杨克特，付志宏，感谢你们关于内核架构的经验总结和传授。

衷心感谢实验室负责本体的师兄：于彤，感谢你在本体概念上进行帮我答疑解惑，让我少走了不少弯路。

衷心感谢我的父母，他们多年来在为人、处世和生活上给予的影响和支持，是我终身奋进的力量源泉。

最后，向所有关心和帮助过我的老师、学友和朋友表示最诚挚的谢意。

郑耀文

公元二零零九年五月 于 求是园

附录

本科生毕业论文（设计）任务书

一、题目：_____

二、指导教师对毕业论文（设计）的进度安排及任务要求：

毕业设计应结合现有“语义网技术”，将在现有 DartGrid 基础上实现基于 OWL 的推理进行融合作为研究重点，明确课题的背景知识、任务和技术路线，充分发挥。

论文需要涵盖以下内容:

1） 研究背景、技术历程、研究意义与应用领域等。

2） 研究目标与研究内容，研究内容应凝聚，可在规定时间内完成。

3） 技术路线与技术思路，提出解决任务的大致技术思路和研究方法。

4） 研究计划，5 月中旬需要完成平台研发，5 月底前完成论文初稿。

在论文编写过程中，要充分参考已有的研究成果，在此基础上进行创新研究，参考文献不少于 10 篇。

起讫日期 **2009** 年 **3** 月 **1** 日 至 **2009** 年 **5** 月 **30** 日

指导教师（签名）_____职称 博士

三、系或研究所审核意见：

负责人（签名）_____

年 月 日

毕 业 论 文（设计） 考 核

一、指导教师对毕业论文（设计）的评语：

论文结合了现有 DartGrid 以及语义网技术进行了分析，指出了当前 DartGrid 的不足，改进了当前 DartGrid 查询流程。分析了推理的基本概念，并对 SPARQL 和 SQL 进行了对比，进而提出了 SPARQL 查询重写的新方法，并利用当前的语义开源包在 SPARQL 重写过程中加入了查询推理，并最终实现了设计目标。论文的内容丰富，举例详细，完成了毕业设计的任务要求，达到了本科毕业论文的要求，建议答辩。

指导教师(签名）_____

年 月 日

二、答辩小组对毕业论文（设计）的答辩评语及总评成绩：

成绩比例	文献综述 占（10%）	开题报告 占（20%）	外文翻译 占（10%）	毕业论文（设计） 质量及答辩 占（60%）	总 评 成绩
分值					

答辩小组负责人（签名）_____

年 月 日