Computer Science and Application 计算机科学与应用, 2020, 10(10), 1789-1797
Published Online October 2020 in Hans. http://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2020.1010189

Hans汉斯

结合编译原理与数据库的计算机系统能力培养

张志远

中国民航大学计算机科学与技术学院，天津
Email: zyzhangcauc@163.com

收稿日期：2020年10月6日；录用日期：2020年10月21日；发布日期：2020年10月28日

摘　要

目前国内的计算机系统能力培养多以组成原理、操作系统和编译系统为主，鲜有数据库管理系统(DBMS)开发的相关介绍。作为系统软件的重要组成部分，本文设计了一个DBMS开发项目，以此培养学生的计算机系统能力。项目结合数据库原理与编译原理等计算机专业骨干课程，以DBMS原型设计开发为载体，综合锻炼学生的程序设计能力、算法与数据结构应用能力，数据库原理和编译原理技术的理解与实现能力，为培养具有扎实理论基础和极强实践能力的计算机专业本科生提供了一条有效途径。

关键词

编译原理，数据库，系统能力培养，DBMS

System Development Capability Training of Computer Science with Compiler Principles & Database

Zhiyuan Zhang

School of Computer Science & Technology, Civil Aviation University of China, Tianjin
Email: zyzhangcauc@163.com

Received: Oct. 6th, 2020; accepted: Oct. 21st, 2020; published: Oct. 28th, 2020

Abstract

Computer system ability training in China is mainly based on computer organization, operating system and compiling system, while little about the development of database management system (DBMS). As DBMS is an important issue of the system software, this paper designs a DBMS

文章引用: 张志远. 结合编译原理与数据库的计算机系统能力培养[J]. 计算机科学与应用, 2020, 10(10): 1789-1797.
DOI: 10.12677/csa.2020.1010189

development project to cultivate students' computer system ability. The project mainly combines the courses of database principle and compiler principle, takes DBMS prototype design and development as the carrier, comprehensively exercises the students' programming ability, algorithm and data structure application ability, and the understanding and implementation ability of database principle and compiler principle technology, so as to provide an effective way for cultivating computer major undergraduates with solid theoretical foundation and strong practical ability.

Keywords

Compiler Principle, Database, System Capability Training, DBMS

1. 引言

计算机系统能力培养近年来一直受到教育部计算机专业教指委和各大高校的重视。在硬件系统能力培养上，北京航空航天大学等高校以计算机组成原理课程为核心，以 MIPS 指令集为基础，通过让学生自主设计 CPU，完成一台功能相对完整的计算机[1] [2]。辅以裁剪操作系统内核和 GCC 编译器使该计算机可编译和执行简单的 C 语言程序，极大提高了学生的硬件系统设计能力。华中科技大学结合数字逻辑和计算机组成原理两门课程，从门电路开始在 FPGA 上设计 CPU，其上配合接口、操作系统、编译器等相应课程的实验内容，有效解决课程实验目标离散、实验内容无序割裂不能融合构成完整计算机系统的问题[3]。桂林电子科技大学在计算机专业系统能力培养方面构建了基础能力培养、专业能力培养和综合能力培养三个层次，硬件系列课程实践环节和软件系列课程实践环节的“三横两纵”体系[4]，使学生初步具备计算机系统综合开发能力。

在软件系统能力培养上，中国科学技术大学[5]和同济大学[6]等高校以编译原理课程为核心，通过逐层深入的方式让学生开发一个 Java 语言子集或 C 语言子集的编译程序，使高级语言能够被正确地编译和执行，提高了学生的软件系统设计能力。苏州大学编译原理实验则注重正规式到 NFA 的自动转换等编译核心算法的实现，并将 SQL 解析和 LaTex 源文件解析加入语法分析实验中[7]。中国矿业大学配合工程教育改革，以培养学生解决复杂工程问题能力为目标，基于开源的 Clang + LLVM 设计和组织编译原理课程教学[8]。中国人民大学[9]等高校则以数据库原理课程为核心，开发了具有自主知识产权的数据库系统。斯坦福大学[10]在数据库方面开设了 CS145、CS245、CS345、CS346、CS347 等多门课程，其中 CS145 只介绍数据库系统的使用，CS345 和 CS347 介绍数据库理论知识，CS245 介绍数据库系统的实现，CS346 是 DBMS 实现的实验课。目前国内数据库原理课程的实验教学主要还是以 SQL 语句、存储过程等如何使用某种数据库产品的使用为主[11]，鲜有关于 DBMS 系统开发的实验教学。对于计算机科学与技术专业的学生来说，不应只局限于应用软件的开发，系统软件的设计和开发更能体现和提高计算机专业学生的软件开发能力。DBMS 是一种常用的系统软件，而开发一个 DBMS 更是综合了程序设计、数据结构、操作系统、数据库原理、编译原理的众多知识点，能极大提高学生的系统软件设计和开发能力。本文尝试设计一个实践教学项目，要求实现一个简单的 DBMS 系统，包括数据库的创建和删除、表的创建和删除，以及记录的增删查改等功能，借此培养学生的计算机系统软件开发能力。

2. 教学目的

本项目通过设计和开发一个功能相对完整的数据库管理系统(DBMS)，融会贯通程序设计、算法与数据结构、编译原理、数据库原理等多门课程内容，专业理论素养和实践能力相结合，综合培养学生的系统软件开发能力。实现的 DBMS 系统主要具备以下功能：数据库定义语言(DDL)和数据库操纵语言(DML)的编译以及查询计划的生成；实现创建数据库和数据表、实现单表增删查改功能；实现创建索引(聚集索引和非聚集索引)，实现多表连接查询功能。

3. 教学要求

3.1. 编译器设计

编译器实现将 SQL 语言解释翻译成相应的语义动作或查询计划，其依据为所支持的标准 SQL 语法。首先定义 SQL 语言的上下文无关文法，保证其符合自上而下或自下而上语法分析的要求；然后定义其语法制导翻译规则，保证 SQL 语言的语义能够得到正确的解释和执行。整个翻译过程分为词法分析、语法分析、语义分析和中间代码生成等环节。其中词法分析将 SQL 语句分解为独立的词法单位记号，如保留字，标识符，运算符及分界符等；语法分析检查 SQL 语句是否符合相应的语法规范，即是否能从上下文无关文法正确推导出 SQL 语句或者 SQL 语句是否可以正确归约到文法的开始符号；语义分析在语法分析的基础上检查 SQL 语句是否合理，如表名或字段名是否存在，运算符和字段类型是否匹配等；中间代码生成根据语法制导翻译规则生成相应的语义动作或生成查询计划，然后遍历查询计划并执行相应代码返回检索数据。

3.2. 数据库设计

数据库中需存储元数据信息和用户数据信息，为简单起见，不考虑存储日志数据。元数据是指数据库、表及索引等的定义，包括字段名称、类型、大小以及约束条件等。元数据在 SQL 语句翻译中起到类似于编译编程序符号表的作用，用于进行语义检查及数据存储空间的分配。由于要经常检索元数据，要求其查询速度要快，可考虑以适当的形式常驻内存，其持久化也要考虑日后读取的方便快捷性。

用户数据是指表中实际存储的数据。最简单的方式是为每一张表建立单独的存储文件，文件名即为表的名称。也可通过在元数据中设置表的存储位置偏移量的方法将所有数据存储在一个大的数据文件中。表中数据通常都是有结构的，每条记录的长度一般都相同，可根据记录长度实现随机访问。若记录长度不同，可考虑增加索引列表记录每条记录的起始位置和长度信息。由于索引列表中每条记录的长度是相同的，因此也可通过变址方式实现记录的随机访问。

若查询列无索引条目，则需进行全表扫描，因此在常用查询列上建立索引可加快查询速度。索引分为聚簇索引和非聚簇索引，每张表上只能建立一个聚簇索引，表中记录按照聚簇索引物理排列记录顺序。聚簇索引对范围查询非常有效，此时只需找出初始记录和终止记录的位置，返回两者之间的所有记录即可。可建立多个非聚簇索引，为优化查询性能，一般采用 B + 树，其实现需要较好地掌握算法和数据结构中的相关内容，是项目实现的难点。当以表的索引列为查询条件时，SQL 编译器应能生成带有索引的查询计划，而不是采用全表扫描的方式。多表连接查询时，也是先从索引开始，将相关记录读入内存后再按照规则进行连接。若不能全部读入内存，可采用存储为临时文件的方式。

3.3. 人机交互接口

DBMS 提供人机交互接口连接用户和数据库。接受用户输入的 SQL 语句，利用编译器对 SQL 语句进行翻译，对 SQL 语句中的错误给出正确提示，生成并执行查询计划，最后返回查询结果。

4. 教学过程设计

4.1. 框架设计

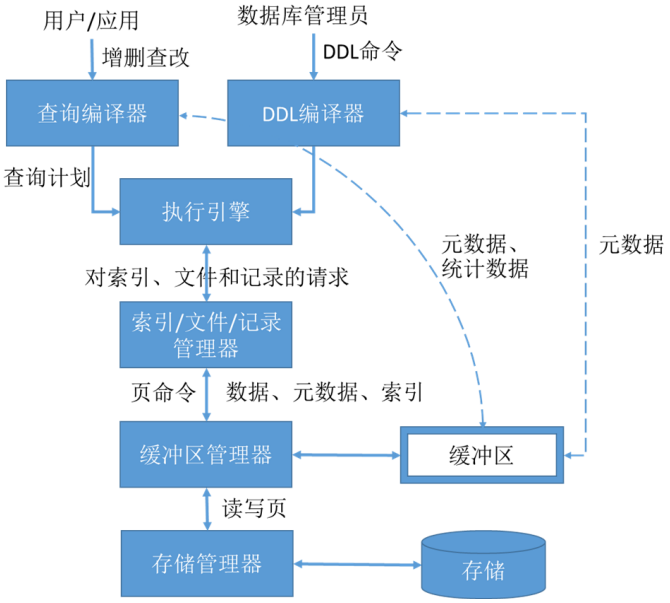

Figure 1. DBMS system construction
图 1. DBMS 系统框架结构图

DBMS 框架结构如图 1 所示。实线是控制流和数据流，虚线仅表示数据流。DDL 编译器负责数据库创建、删除，模式的创建、删除及修改，索引维护等数据定义语言的编译工作。查询编译器负责数据增删查改等 DML 语言的编译工作，根据缓冲区中的元数据信息进行语义分析，生成查询计划。同时根据缓冲区的数据统计信息选择一个较优的查询计划并提交给执行引擎。执行引擎向资源管理器发出一系列小数据单元的请求(如记录)，资源管理器负责控制数据文件、数据格式以及索引文件。查询数据的请求传递给缓冲区管理器，缓冲区管理器的任务是从持久化的辅存中将一部分数据以页的形式存取到主存中，以加快查询速度。

4.2. 编译器设计

4.2.1. 文法设计

采用自上而下的递归下降语法分析方法，因此将文法设计为 LL(1)文法。实现了创建数据库(create table)，创建表(create table)，删除数据库(drop database)，删除表(drop table)，插入数据(insert into tablename)，删除数据(delete from tablename)，查询数据(select from tablename)功能。

SQL -> DDLSQL | DMLSQL
DDLSQL -> create CREASQL | drop DROPSQL
CREASQL -> database *Identifier*; |
table *Identifier* (ATTR_DEF_LIST);
ATTR_DEF_LIST -> ATTR_DEF OPT_ATTR_DEF_LIST
ATTR_DEF -> *Identifier* TYPE
OPT_ATTR_DEF_LIST -> , ATTR_DEF OPT_ATTR_DEF_LIST | ε
TYPE -> bit | int | smallint | tinyint | float | numeric(*intNum* , *intNum*) |
datetime | smalldatetime | char(*intNum*) | varchar(*intNum*)
DROPSQL -> database *Identifier*; | table *Identifier*;
DMLSQL -> INSSQL | SELSQL | DELSQL

INSSQL -> insert into *Identifier* INSSQL2
INSSQL2 -> (IDLIST) INSSQL3
IDLIST -> *Identifier* OPT_IDLIST
OPT_IDLIST -> , *Identifier* |ε
INSSQL3 -> values (EXPLIST) ;
EXPLIST -> EXPRESSION OPT_EXPLIST
OPT_EXPLIST -> , EXPRESSION OPT_EXPLIST |ε
EXPRESSION -> TERM EXPRESSION_1
EXPRESSION_1 -> ADDOP TERM EXPRESSION_1 |ε
ADDOP -> + | -
TERM -> FACTOR TERM_1
TERM_1 -> MULOP FACTOR TERM_1 |ε
MULOP -> * | / | %
FACTOR -> (EXPRESSION) | - EXPRESSION |
Identifier | *intNum* | *floatNum* | *stringNum* | *null* |
SELSQL -> select SELSQL1 FROM *Identifier* SQLSQL2
SELSQL1 -> * | IDLIST
SELSQL2 -> where BOOLEXPRESSION |ε
BOOLEXPRESSION -> BOOLTERM BOOLEXPRESSION_1
BOOLEXPRESSION_1 -> or BOOLTERM BOOLEXPRESSION |ε
BOOLTERM -> BOOLFACTOR BOOLTERM_1
BOOLTERM_1 -> and BOOLFACTOR BOOLTERM_1 |ε
BOOLFACTOR -> NOT BOOLEXPRESSION | (BOOLEXPRESSION) |
REL_EXPRESSION
REL_EXPRESSION -> EXPRESSION RELOP EXPRESSION
RELOP -> = | <> | > | >= | < | <=
DELSQL -> delete from *Identifier* DELSQL1
DELSQL1 -> where BOOLEXPRESSION ; | ;

4.2.2. 编译器类图设计

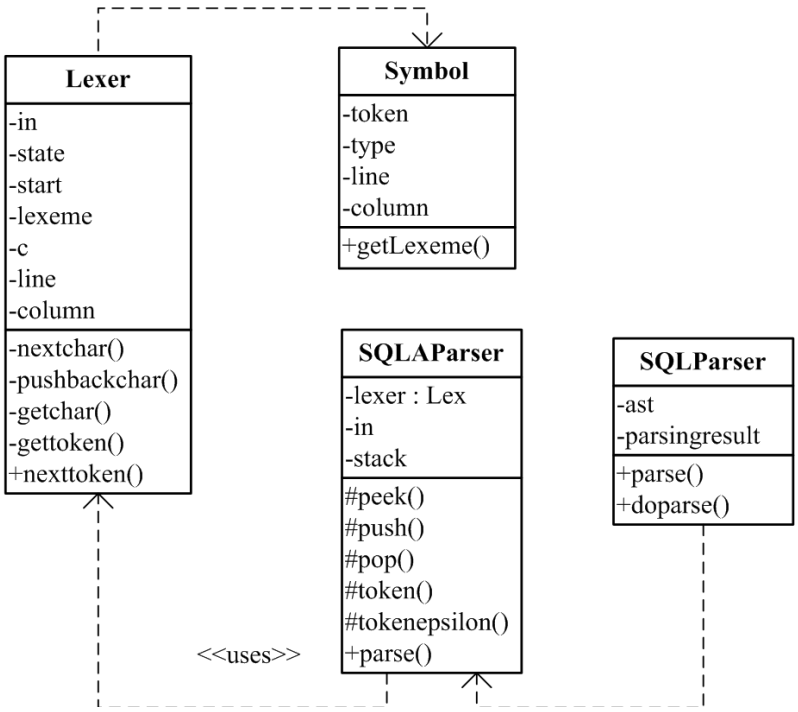

Figure 2. UML class diagram of compiler
图 2. 编译器主要类图设计

图 2 给出了编译器涉及到的主要类，其中每个类只标出了比较重要的一些成员变量和方法。Symbol 类和 Lexer 类负责词法分析，其中 Symbol 类存放词法单位记号，如常量、变量、保留字、运算符等，Lexer 类是词法分析的主类，负责识别单词符号，其中的 nexttoken()方法返回语法分析所需的下一个词法记号。SQLAParser 是一个抽象类，包括语法分析中涉及到的一些栈操作的定义，其中的 token 方法就从 Lexer 中读取下一个词法记号。SQLParser 是 SQLAParser 类的扩展，接收用户输入的 SQL 语句并进行语法分析。

4.3. 数据存储组织

在底层，一个数据库对应一个后缀名为 mdf 的文件。数据按块组织，每个块的大小设置为 4096 个字节，即 4 k 大小。数据库初始大小为 16 k 个数据块，即 64 M。当存储容量不够时，每次增加 4 M，也就是 1024 个数据块。

4.3.1. 元数据组织

元数据占一个数据块。包含数据块的使用情况，有多少张表，每张表模式对应的数据块列表，如图 3 所示。简单起见，数据块连续分配，其使用情况包含两部分内容：已经使用的最大块号 blockUsed、删除不用的块号列表 removedBlockList。一个数据块被删除后，就将其加入 removedBlockList 列表，再次申请新块时，优先从其中选取。若没有，则新分配一个最大块号。

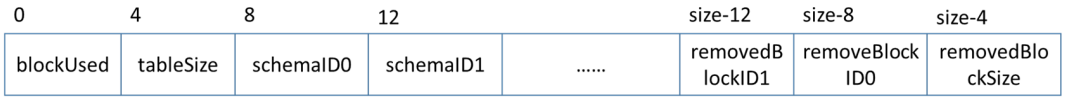

Figure 3. Arrangement of meta block
图 3. 元数据块的组织

每个表模式占另外一个单独的块，包含第一个数据块的编号，模式名称以及多个属性字段信息，如图 4 所示。对于长度不定的字符串，例如模式名称 schemaName，采用两部分存储：第一部分用 4 个字节存储字符串的长度信息，第二部分存储字符串本身的信息。由于 java 中存储一个字符需要 2 个字节，因此其占用的字节数为字符串本身长度的 2 倍。

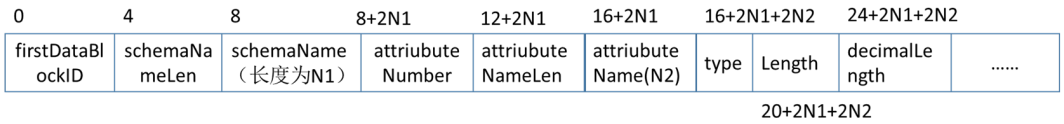

Figure 4. Arrangement of schema block
图 4. 模式块的组织

4.3.2. 数据存储组织

数据块的按照链接方式组织在一起，每一个数据块均在开始位置表明下一个数据块的编号，若为−1，则表示是最后一个数据块。每条记录按照模式指定的长度定长存储，如图 5 所示。插入记录时，根据 recordNumber 将指针移动到新纪录的位置，写入后将 recordNumber 加 1。读取和修改时，只需将指针移动到相应的位量后，按照相应的数据类型进行读取和写入。为简单起见，删除某条记录时，后面的记录相应往前移动。

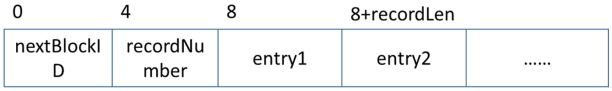

Figure 5. Arrangement of data block
图 5. 数据块的组织

4.3.3. 数据库类图设计

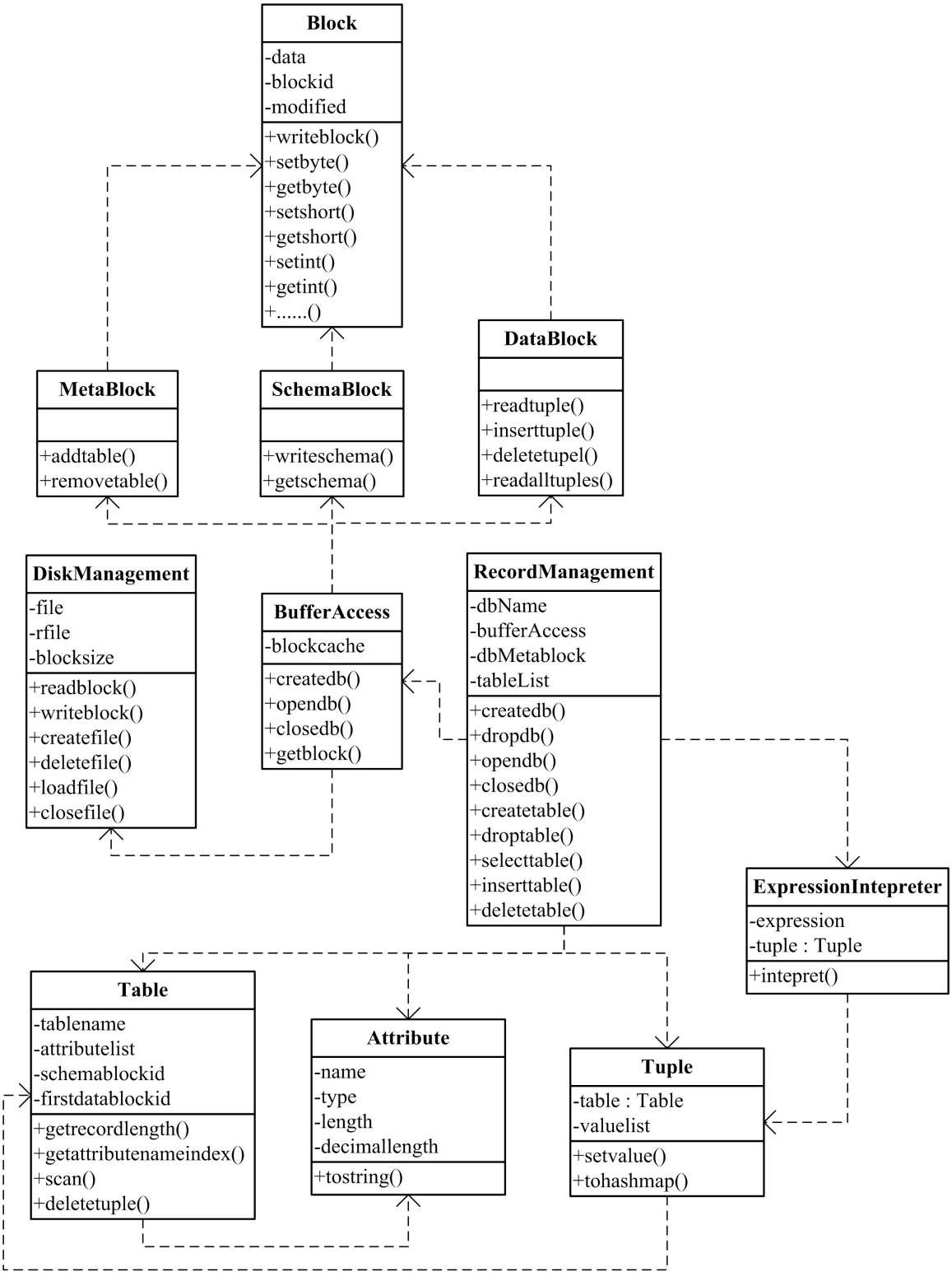

Figure 6. UML class diagram of database management
图 6. 数据库组织管理主要类图设计

数据库类图设计如图 6 所示。Block 类是所有数据块的父类，定义了基本的数据块操作，包括读写各种类型的数据。MetaBlock 负责元数据块管理，SchemaBlock 负责模式数据块管理，DataBlock 负责存放实际数据的块的管理。Diskmanagement 类负责和底层的数据文件打交道，程序运行过程中，数据块都先存放在 BufferAccess 中，在缓冲区内进行管理。RecordManagement 类是编译器和数据处理的联系通道，编译器识别的各种命令如 create table，insert table 等都是直接调用该类相应的方法实现的。Table 管理具体某张表，Attribute 类存储该表中具体的某个属性列的定义，Tuple 类则具体存储表中的某条记录。ExpressionIntepreter 负责解释 SQL 语句中的表达式，并将其转换为实际的值。

5. 系统运行结果

源代码包含 21 个 java 源文件，代码量 3845 行，以下是系统的部分运行结果演示。

```
SQL>create database student;
dbDir/student.mdf created!
SQL>create table student(
sno int,
sname varchar(20),
sage int);
table student created!
SQL>insert into student(1,'zhang',20);
attribute list error at line 0,column 21
SQL>insert into student values(1,'zhang',20);
1 row inserted!
SQL>insert into student values(2,'wang',18);
1 row inserted!
SQL>insert into student(sno, sname, sage) values(3,'li',19);
1 row inserted!
SQL>insert into student(sno, sage, sname) values(4,20,'zhao');
1 row inserted!
SQL>insert into student(sno, sname) values(5,'han');
1 row inserted!
SQL>select * from student;
=============student query result ==============
sno                sname              sage
1                  zhang              20
2                  wang               18
3                  li                 19
4                  zhao               20
5                  han                0
5 rows retrived!
============================================================
SQL>delete from student where sno = 5;
1 rows deleted!
SQL>select * from student;
=============student query result ==============
sno                sname              sage
1                  zhang              20
2                  wang               18
3                  li                 19
4                  zhao               20
4 rows retrived!
============================================================
SQL>create table test ( int a);
attribute Defination error at line 0,column 23
SQL>create table test(a int);
```

```
table test created!
SQL>select * from test;
=============test query result ==============
a
0 rows retrived!
============================================================
SQL>drop table test;
table test dropped!
SQL>select * from test;
no such table:test
SQL>delete from student;
4 rows deleted!
SQL>select * from student;
=============student query result ==============
sno                sname              sage
0 rows retrived!
============================================================
SQL>quit;
meta data reallocated!
exiting dbms....
```

6. 结束语

“DBMS 系统的设计和开发”项目综合运用了计算机科学与技术专业多门课程的知识，能将学生所学理论知识应用于实践，提高其系统软件开发能力。设计开发了其中的部分功能，通过实际运行验证了项目的可行性。一些高层次的内容如索引，多表连接查询以及查询优化等有待进一步实现。

基金项目

中国民航大学教育教学改革项目：CAUC-2017-B1-09。

参考文献

[1] 马殿富, 高小鹏. 基于系统能力培养的计算机专业课程建设报告[R]. 北京: 北京航空航天大学, 2013.

[2] 王雷. 操作系统实验设计[J]. 计算机教育, 2009(17): 54-56.

[3] 谭志虎, 秦磊华, 胡迪青. 面向系统能力培养的计算机专业实践教学模式[J]. 中国大学教学, 2017(9): 80-84.

[4] 陈智勇. 计算机科学与技术专业学生系统能力培养的改革与实践[J]. 计算机教育, 2019(3): 58-61.

[5] 张昱, 陈意云. 编译原理课程实践改革探索[J]. 计算机教育, 2008(8): 24-26.

[6] 王伟, 张军旗, 江建慧, 等. 计算机科学与技术卓越课程行动计划实践——以同济大学计算机系统级课程为例[J]. 计算机教育, 2013(2): 66-69.

[7] 王中卿, 朱培培. 层次化精准编译原理实践教学[J]. 电脑知识与技术, 2020(20): 158-159.

[8] 刘兵, 张辰, 谢红侠, 等. 基于 Clang + LLVM 架构的编译原理课程教学探索[J]. 计算机教育, 2020(1): 42-45, 49.

[9] 王珊, 萨师煊. 数据库系统概论[M]. 第 5 版. 北京: 高等教育出版社, 2015.

[10] Hector Garcia-Monlina, Jeffrey D. Ullman, Jennifer Widom. 数据库系统实现[M]. 第 2 版. 杨冬青, 等, 译. 北京: 机械工业出版社, 2016.

[11] 林颖贤, 浦云明, 等. 基于 OBE 理念的数据库原理课程混合式教学模式[J]. 计算机教育, 2020(7): 62-65.