# 《DBMS》实验报告四

**报告人：**冷友方；王大伟；罗传文；赵丹；杨婧如.

**学院：**信息学院

**日期：**2016年12月5日

# 1 语法树的形式

以 SELECT 语句为例来说明本实验中语法树的形式，如下图 2 所示：最上面是主体节点；然后根据 SELECT 中的各个域，生成四个链表；然后每个链表中的各个结点链接起来，直到 NULL 为止。最后查询计划生成器会根据这些链表生成相应的关系代数树。

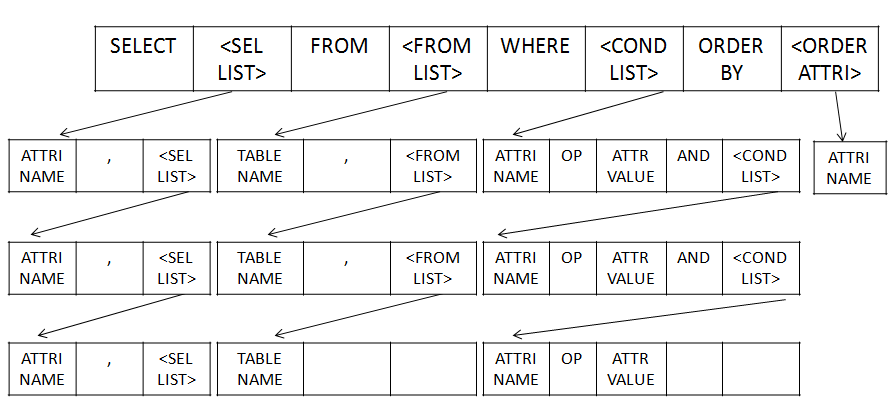


图2

# 2词法与语法分析的实现

词法与语法分析使用 lex 与 yacc 实现，利用 flex 与 bison 进行编译。

## 2.1 词法语法分析实现

sql.l 文件进行词法分析，从用户输入的语句中解析出保留关键词。

sql.y 文件进行语法分析，输入为 lex 的词法分析结果，输出为语法分析结果。

### 2.1.1实现的SQL语句

创建table命令: CREATE table;

删除table命令: DROP TABLE table;

创建索引命令: CREATE INDEX table (column);

删除索引命令: DROP INDEX table (column) ;

select语句:

select\_clause FROM fromlist where\_clause order\_clause ;

SELECT unique sellist;

SELECT unique '\*';

insert语句:INSERT INTO table VALUES (insert\_list );

delete语句: DELETE FROM table where\_clause ;

显示所有创建的表: SHOW TABLES ;

### 2.1.2识别出的关键字

|  |
| --- |
| SELECT FROM WHERE ORDER BY ASC DESC  ALL UNIQUE DISTINCT  CREATE TABLE DROP INDEX  INSERT INTO VALUES DELETE  CHARACTER INTEGER DATE  SHOW TABLES  EXIT |

### 在yacc文件中，终端符名约定俗成地用大写表示。因此在lex中将识别到的小写终端符名统一返回其大写形式。其中，char、CHAR返回CHARACTER，int、INT返回INTEGRER。

### 2.1.3识别出的操作符及优先级定义

|  |
| --- |
| AND  COMPARISION /\* < <= > >= <> = \*/  '+' '-'  '\*' '/' |

### 2.1.4识别标示符

### 对于字符集合[-+\*/:(),.;]，匹配到其中的任何一个符号，都将直接输出；

### 对于字符集合[0-9]+，匹配一个或多个的0~9之间的数值，都将返回为NUMBER；

### 对于字符集合[a-zA-Z][a-zA-Z0-9\_]\*，匹配以字母开头的，可重复多次的该模式，都返回成标识符NMAE；

### 2.1.5 主要接口

|  |
| --- |
| sql:  ';'  | table\_def //创建table命令  | table\_drop //删除table命令  | index\_create //创建索引命令  | index\_drop //删除索引命令  | select\_stat //select语句  | insert\_stat //insert语句  | delete\_stat //delete语句  | table\_show //显示所有创建的表  ; |

## 2.2 主要的语句语法

### CREATE TABLE

table\_def: //创建表

CREATE TABLE table '(' table\_attr\_list ')' ';'

{

before = clock();

create\_table\_lqp(tbname, attrcnt, attr);

after = clock();

print\_runtime(before, after);

}

;

table: //表名的定义

NAME

{

tbname = $1;

rel[tbcnt] = $1;

}

;

column\_def:

column data\_type //列名+数据类型

{

attrcnt++; //列的数量

}

;

**测试样例：**CREATE TABLE test4(id int, name char(20));

### INSERT

insert\_stat: //insert 语句

INSERT INTO table VALUES '(' insert\_list ')' ';'

{

before = clock();

insert\_lqp(tbname, valcnt, val);

after = clock();

print\_runtime(before, after);

}

;

insert\_list: //插入值列表

NUMBER //插入数字

{

val[valcnt].type = INT;

memcpy(val[valcnt].data, &$1, 4);

valcnt++;

}

| STRING //插入字符串

{

val[valcnt].type = CHAR;

strcpy(val[valcnt].data, $1);

valcnt++;

}

| insert\_list ',' NUMBER //插入多个数字

{

val[valcnt].type = INT;

memcpy(val[valcnt].data, &$3, 4);

valcnt++;

}

| insert\_list ',' STRING //插入多个string

{

val[valcnt].type = CHAR;

strcpy(val[valcnt].data, $3);

valcnt++;

}

;

**测试样例：**

insert into test4 values(1, ‘andy’);

insert into test4 values(2, ‘Shirley’);

insert into test4 values(3, ‘Nemo’);

### SELECT

select\_clause: //选择列

SELECT unique sellist

| SELECT unique '\*'

{

is\_ares = 1;

}

;

select\_stat:

select\_clause FROM fromlist where\_clause order\_clause ';'

{

before = clock();

select\_lqp(selcnt, selattrs, tbcnt, rel, condcnt, cond, orderby, orderattr, is\_uni, is\_ares);

after = clock();

print\_runtime(before, after);

}

;

select\_clause: //选择列

SELECT unique sellist

| SELECT unique '\*'

{

is\_ares = 1;

}

;

fromlist: //from之后tables

table //单个表

{

is\_where = 1;

tbcnt++;

}

| fromlist ',' table //多个表

{

is\_where = 1;

tbcnt++;

}

;

sellist: //select 属性列

column //单个列显示

{

selcnt++;

}

| sellist ',' column //多个列显示

{

selcnt++;

}

;

**测试样例：**SELECT unique/distinct/unique \* FROM test;

### DELETE

delete\_stat: // delete 语句

DELETE FROM table where\_clause ';'

{

before = clock();

delete\_lqp(tbname, condcnt, cond);

after = clock();

print\_runtime(before, after);

}

;

**测试样例：**delete from test4 where id=3;

### TABLE SHOW

table\_show:

SHOW TABLES ';'

{

before = clock();

table\_show();

after = clock();

print\_runtime(before, after);

}

;

**测试样例：**SHOW TABLES;

### DROP TABLE

table\_drop: //删除表

DROP TABLE table ';'

{

before = clock();

drop\_table\_lqp(tbname);

after = clock();

print\_runtime(before, after);

}

;

**测试样例：**DROP TABLE test4;

## 3 文件结构

|  |  |
| --- | --- |
| 文件 | 说明 |
| sql.y | sql语法配置文件 |
| sql.l | sql词法配置文件 |
| y.tab.h + y.tab.c | 语法文件用过yacc生成的解析器 |
| lex.yy.c | 词法文件通过lex生成的解析器 |