МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ОРЛОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ И.С. ТУРГЕНЕВА»

Кафедра программной инженерии

**Отчет по лабораторной работе**

по дисциплине «Теория языков программирования и методы трансляции»

на тему: «Таблично-управляемый синтаксический разбор сверху вниз»

Студент Бажин М.И. Шифр:160582

Институт приборостроения, автоматизации и информационных технологий

Специальность 09.03.01 «Информатика и вычислительная техника»

Группа 61-ИВТ

Руководитель Гордиенко А.П.

Орел 2018

**ГРАММАТИКА**

GOAL → SQL\_CREATE\_TABLE

SQL\_CREATE\_TABLE → create\_table\_term id SQL\_STATEMENT

SQL\_STATEMENT → BRACKETS1 FIELD\_DECLORATION

KEYS\_DECLORATION BRACKETS2

FIELD\_DECLORATION → FIELD FIELD\_DECLORATION1

FIELD\_DECLORATION1 → FIELD FIELD\_DECLORATION1

→ E

FIELD → id FTYPE NULL\_VALUE ,

FTYPE → integer\_term

→ char\_term BRACKETS1 num BRACKETS2

NULL\_VALUE → null\_term

→ not\_null\_term

KEYS\_DECLORATION → PRIMARY\_KEY\_DECL FOREIGN\_KEY\_DECL

PRIMARY\_KEY\_DECL → pkey\_term FIELD\_LIST

FIELD\_LIST → BRACKETS1 id FIELD\_LIST1 BRACKETS2

FIELD\_LIST1 → , id FIELD\_LIST1

→ E

FOREIGN\_KEY\_DECL → , FOREIGN\_KEY FOREIGN\_KEY\_DECL

→ E

FOREIGN\_KEY → fkey\_term FIELD\_LIST references\_term REF\_TABLE

REF\_TABLE → id FIELD\_LIST

BRACKETS1 → (

BRACKETS2 → )

*Множество FIRST*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Грамматический символ | Шаг алгоритма | | | **FIRST** |
| 1 | 2 | 3 |
| GOAL |  |  | { create\_table\_term } | { create\_table\_term } |
| SQL\_CREATE\_TABLE |  |  | { create\_table\_term } | { create\_table\_term } |
| SQL\_STATEMENT |  |  | { ( } | { ( } |
| FIELD\_DECLORATION |  |  | {id} | {id} |
| FIELD\_DECLORATION1 |  | E | { id } | { E ; id } |
| FIELD |  |  | {id} | {id} |
| FTYPE |  |  | { integer\_term; char\_term } | { integer\_term; char\_term } |
| NULL\_VALUE |  |  | { null\_term; not\_null\_term } | { null\_term; not\_null\_term } |
| KEYS\_DECLORATION |  |  | { pkey\_term } | { pkey\_term } |
| PRIMARY\_KEY\_ DECL |  |  | { pkey\_term } | { pkey\_term } |
| FIELD\_LIST |  |  | { ( } | { ( } |
| FIELD\_LIST1 |  | E | { , } | { E ; , } |
| FOREIGN\_KEY\_DECL |  |  | { , } | { , } |
| FOREIGN\_KEY |  |  | { fkey\_term } | { fkey\_term } |
| REF\_TABLE |  |  | {id} | {id} |
| BRACKETS1 |  |  | { ( } | { ( } |
| BRACKETS2 |  |  | { ) } | { ) } |
| create\_table\_term | { create\_table\_term } |  |  | { create\_table\_term } |
| id | {id} |  |  | {id} |
| num | { num } |  |  | { num } |
| ( | { ( } |  |  | {(} |
| ) | { ) } |  |  | {)} |
| , | { , } |  |  | { , } |
| int\_term | { int\_term } |  |  | { int\_term } |
| char\_term | { char\_term } |  |  | { char\_term } |
| null\_term | { null\_term } |  |  | { null\_term } |
| not\_null\_term | { not\_null\_term } |  |  | { not\_null\_term } |
| pkey\_term | { pkey\_term } |  |  | { pkey\_term } |
| fkey\_term | { fkey\_term } |  |  | { fkey\_term } |
| references\_term | { references\_term } |  |  | { references\_term } |

*Множество FOLLOW*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Нетерминал | Шаг алгоритма | | | | **FOLLOW1** |
| 1 | 2 | 3 | 4 |
| GOAL | { eof } |  |  |  | { eof } |
| SQL\_CREATE\_TABLE |  |  | { eof } |  | { eof } |
| SQL\_STATEMENT |  |  | { eof } |  | { eof } |
| FIELD\_DECLORATION |  | { pkey\_term } |  |  | { pkey\_term } |
| FIELD\_DECLORATION1 |  |  | { pkey\_term } |  | { pkey\_term } |
| FIELD |  | { id } |  | { pkey\_term } | { id ; pkey\_term } |
| FTYPE |  | { null\_term ; not\_null\_term } |  |  | { null\_term ; not\_null\_term } |
| NULL\_VALUE |  | { , } |  |  | { , } |
| KEYS\_DECLORATION |  | { ) } |  |  | { ) } |
| PRIMARY\_KEY\_ DECL |  | { , } |  | { ) } | { , ; ) } |
| FIELD\_LIST |  | { references\_term } | { , } |  | { , ; references\_term } |
| FIELD\_LIST1 |  | { ) } | { ) } |  | { ) } |
| FOREIGN\_KEY\_DECL |  |  | { ) } |  | { ) } |
| FOREIGN\_KEY |  | { , } |  | { ) } | { , ; ) } |
| REF\_TABLE |  |  | { , } |  | { , } |
| BRACKETS1 |  | { id ; num } |  |  | { id ; num } |
| BRACKETS2 |  |  | { eof ; null\_term ; not\_null\_term ; , ; references\_term } |  | { eof ; null\_term ; not\_null\_term ; , ; references\_term } |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Нетерминал | Шаг алгоритма | | | | **FOLLOW2** |
| 1 | 2 | 3 | 4 |
| GOAL |  |  |  |  | { eof } |
| SQL\_CREATE\_TABLE |  |  | { eof } |  | { eof } |
| SQL\_STATEMENT |  |  | { eof } |  | { eof } |
| FIELD\_DECLORATION |  |  |  |  | { pkey\_term } |
| FIELD\_DECLORATION1 |  |  | { pkey\_term } |  | { pkey\_term } |
| FIELD |  |  |  | { pkey\_term } | { id ; pkey\_term } |
| FTYPE |  |  |  |  | { null\_term ; not\_null\_term } |
| NULL\_VALUE |  |  |  |  | { , } |
| KEYS\_DECLORATION |  |  |  |  | { ) } |
| PRIMARY\_KEY\_ DECL |  |  |  | { ) } | { , ; ) } |
| FIELD\_LIST |  |  | { , ; ) } |  | { , ; references\_term; ) } |
| FIELD\_LIST1 |  |  | { ) } |  | { ) } |
| FOREIGN\_KEY\_DECL |  |  | { ) } |  | { ) } |
| FOREIGN\_KEY |  |  |  | { ) } | { , ; ) } |
| REF\_TABLE |  |  | { , ; ) } |  | { , ; ) } |
| BRACKETS1 |  |  |  |  | { id ; num } |
| BRACKETS2 |  |  | { eof ; null\_term ; not\_null\_term ; , ; references\_term; ) } |  | { eof ; null\_term ; not\_null\_term ; , ; references\_term } |

Таблица разбора

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | create\_table\_term | id | num | ( | ) | , | integer\_term | char\_term | null\_term | not\_null\_term | pkey\_term | fkey\_term | references\_term | eof |
| GOAL | GOAL → SQL\_CREATE\_TABLE |  |  |  |  |  |  |  |  |  |  |  |  |  |
| SQL\_CREATE\_TABLE | SQL\_CREATE\_TABLE → create\_table\_term id SQL\_STATEMENT |  |  |  |  |  |  |  |  |  |  |  |  |  |
| SQL\_STATEMENT |  |  |  | SQL\_STATEMENT → BRACKETS1 FIELD\_DECLORATION  KEYS\_DECLORATION BRACKETS2 |  |  |  |  |  |  |  |  |  |  |
| FIELD\_DECLORATION |  | FIELD\_DECLORATION → FIELD FIELD\_DECLORATION1 |  |  |  |  |  |  |  |  |  |  |  |  |
| FIELD\_DECLORATION1 |  | FIELD\_DECLORATION1 → FIELD FIELD\_DECLORATION1 |  |  |  |  |  |  |  |  | FIELD\_DECLORATION1 → E |  |  |  |
| FIELD |  | FIELD → id TYPE NULL\_VALUE , |  |  |  |  |  |  |  |  |  |  |  |  |
| TYPE |  |  |  |  |  |  | TYPE → integer\_term | TYPE  →  char\_term BRACKETS1 num BRACKETS2 |  |  |  |  |  |  |
| NULL\_VALUE |  |  |  |  |  |  |  |  | NULL\_VALUE → null\_term | NULL\_VALUE → not\_null\_term |  |  |  |  |
| KEYS\_DECLORATION |  |  |  |  |  |  |  |  |  |  | KEYS\_DECLORATION → PRIMARY\_KEY\_DECL FOREIGN\_KEY\_DECL |  |  |  |
| PRIMARY\_KEY\_ DECL |  |  |  |  |  |  |  |  |  |  | PRIMARY\_KEY\_DECL →  pkey\_term FIELD\_LIST |  |  |  |
| FIELD\_LIST |  |  |  | FIELD\_LIST → BRACKETS1 id FIELD\_LIST1 BRACKETS2 |  |  |  |  |  |  |  |  |  |  |
| FIELD\_LIST1 |  |  |  |  | FIELD\_LIST1 → E | FIELD\_LIST1 → , id FIELD\_LIST1 |  |  |  |  |  |  |  |  |
| FOREIGN\_KEY\_DECL |  |  |  |  | FOREIGN\_KEY\_DECL → E | FOREIGN\_KEY\_DECL1 → , FOREIGN\_KEY FOREIGN\_KEY\_DECL |  |  |  |  |  |  |  |  |
| FOREIGN\_KEY |  |  |  |  |  |  |  |  |  |  |  | FOREIGN\_KEY → f\_key FIELD\_LIST references\_term REF\_TABLE |  |  |
| REF\_TABLE |  | REF\_TABLE → id FIELD\_LIST |  |  |  |  |  |  |  |  |  |  |  |  |
| BRACKETS1 |  |  |  | BRACKETS1 → ( |  |  |  |  |  |  |  |  |  |  |
| BRACKETS2 |  |  |  |  | BRACKETS2 → ) |  |  |  |  |  |  |  |  |  |

Листинг программы

type

symb = (goal, sql\_create\_table, sql\_statement, field\_decloration, field\_decloration1, field, ftype, null\_value, keys\_decloration, primary\_key\_decl, field\_list, field\_list1, foreign\_key\_decl, foreign\_key, ref\_table, bracket1, bracket2,

t\_crt, t\_id, t\_num, t\_bracket1, t\_bracket2, t\_comma, t\_int, t\_char, t\_null, t\_nnull, t\_pkey, t\_fkey, t\_ref, eof);

const

MTable: array[goal..bracket2, t\_crt..eof] of string =

(

('sql\_create\_table','er','er','er','er','er','er','er','er','er','er','er','er','er'),

('t\_crt t\_id sql\_statement','er','er','er','er','er','er','er','er','er','er','er','er','er'),

('er','er','er','( field\_decloration keys\_decloration )','er','er','er','er','er','er','er','er','er','er'),

('er','field field\_decloration1','er','er','er','er','er','er','er','er','er','er','er','er'),

('er','field field\_decloration1','er','er','er','er','er','er','er','er','','er','er','er'),

('er','t\_id ftype null\_value ,','er','er','er','er','er','er','er','er','er','er','er','er'),

('er','er','er','er','er','er','t\_int','t\_char ( t\_num )','er','er','er','er','er','er'),

('er','er','er','er','er','er','er','er','t\_null','t\_nnull','er','er','er','er'),

('er','er','er','er','er','er','er','er','er','er','primary\_key\_decl foreign\_key\_decl','er','er','er'),

('er','er','er','er','er','er','er','er','er','er','t\_pkey field\_list','er','er','er'),

('er','er','er','( t\_id field\_list1 )','er','er','er','er','er','er','er','er','er','er'),

('er','er','er','er','',', t\_id field\_list1','er','er','er','er','er','er','er','er'),

('er','er','er','er','',', foreign\_key foreign\_key\_decl','er','er','er','er','er','er','er','er'),

('er','er','er','er','er','er','er','er','er','er','er','t\_fkey field\_list t\_ref ref\_table','er','er'),

('er','t\_id field\_list','er','er','er','er','er','er','er','er','er','er','er','er'),

('er','er','er','(','er','er','er','er','er','er','er','er','er','er'),

('er','er','er','er',')','er','er','er','er','er','er','er','er','er')

);

var

Form1: TForm1;

stack: array[1..100] of symb;

p\_s: integer;

token: symb;

procedure TForm1.FormCreate(Sender: TObject);

begin

p\_s:=0;

end;

procedure TForm1.Parse;

var prod: string;

er\_flag: boolean;

begin

er\_flag:=false;

push(eof);

push(goal);

token:=lextosymb(yylex);

repeat

if stack[p\_s] in [t\_crt..t\_ref] then //вершина стека - терминал

if token = stack[p\_s] then

begin

pop;

token:=lextosymb(yylex);

end

else

begin

showmessage('Ошибка');

er\_flag:=true;

break;

end

else if stack[p\_s] in [goal..bracket2] then //вершина стека - нетерминал

if MTable[stack[p\_s],token] <> 'er' then

begin

prod:=MTable[stack[p\_s],token];

pop;

if prod <> '' then pushRightProd(prod); //разбивает правую часть продукции на термы и заносит в стек в обратном порядке

end

else

begin

showmessage('Ошибка');

er\_flag:=true;

break;

end

else //в стеке посторонний символ

begin

showmessage('Ошибка');

er\_flag:=true;

break;

end;

until stack[p\_s] = eof;

if not er\_flag then

if token <> eof then showmessage('Ошибка')

else showmessage('Успех!');

end;

procedure TForm1.pushRightProd(production: string);

var i:integer;

term:string;

begin

for i:=length(production) downto 1 do

begin

if (production[i] = ' ') then

begin

push(strtosymb(term));

term:='';

end

else insert(production[i],term,1);

end;

push(strtosymb(term)); //после цикла последний терм не будет добавлен стек, поэтому добавляем

end;

procedure TForm1.push(value: symb);

var t:symb;

begin

p\_s:=p\_s+1;

stack[p\_s]:=value;

end;

procedure TForm1.pop;

begin

p\_s:=p\_s-1;

end;

function TForm1.strtosymb(value:string): symb;

begin

case value of

//НЕТЕРМИНАЛЫ

'goal': result:=goal;

'sql\_create\_table': result:=sql\_create\_table;

'sql\_statement': result:=sql\_statement;

'field\_decloration': result:=field\_decloration;

'field\_decloration1': result:=field\_decloration1;

'field': result:=field;

'ftype': result:=ftype;

'null\_value': result:=null\_value;

'keys\_decloration': result:=keys\_decloration;

'primary\_key\_decl': result:=primary\_key\_decl;

'field\_list': result:=field\_list;

'field\_list1': result:=field\_list1;

'foreign\_key\_decl': result:=foreign\_key\_decl;

'foreign\_key': result:=foreign\_key;

'ref\_table': result:=ref\_table;

'bracket1': result:=bracket1;

'bracket2': result:=bracket2;

//ТЕРМИНАЛЫ

't\_num': result:=t\_num;

't\_id': result:=t\_id;

't\_int': result:=t\_int;

't\_char': result:=t\_char;

't\_crt': result:=t\_crt;

't\_null': result:=t\_null;

't\_nnull': result:=t\_nnull;

't\_pkey': result:=t\_pkey;

't\_fkey': result:=t\_fkey;

't\_ref': result:=t\_ref;

',': result:=t\_comma;

'(': result:=t\_bracket1;

')': result:=t\_bracket2;

end;

end;

function TForm1.lextosymb(value:integer): symb;

begin

case value of

num: result:=t\_num;

id: result:=t\_id;

int\_term: result:=t\_int;

char\_term: result:=t\_char;

create\_table\_term: result:=t\_crt;

null\_term: result:=t\_null;

not\_null\_term: result:=t\_nnull;

pkey\_term: result:=t\_pkey;

fkey\_term: result:=t\_fkey;

references\_term: result:=t\_ref;

ord(','): result:=t\_comma;

ord('('): result:=t\_bracket1;

ord(')'): result:=t\_bracket2;

0: result:=eof;

end;

end;

end.