

# 编译原理大作业

Deadline:

- 1、词法分析器（包括代码及要求的文档）2022 年 4 月 29 日 11: 59PM (GMT+8)
- 2、语法分析器（包括代码及要求的文档）2022 年 4 月 29 日 11: 59PM (GMT+8)

## 一、目标

本次大作业为编写一个编译器前端（包括词法分析器和语法分析器），（1）使用自动机理论编写词法分析器，（2）自上而下或者自下而上的语法分析方法编写语法分析器。

1、编写 SQL--语言的词法分析器，理解词法分析器的工作原理，熟练掌握词法分析器的工作流程并编写源代码，识别出单词的二元属性，填写符号表。

2、编写 SQL--语言的语法分析器，理解自上而下/自下而上的语法分析算法的工作原理；理解词法分析与语法分析之间的关系。语法分析器的输入为 SQL--语言源代码，输出为按扫描顺序进行推导或归约的正确/错误的判别结果，以及按照最左推导顺序/规范规约顺序生成语法树所用的产生式序列。

## 二、软件需求

### 1、词法分析器

（1）完成 SQL--语言的词法分析器，要求采用课程教授方法，实现有限自动机**确定化**，**最小化**算法。词法分析器的输入为 SQL--语言源代码，输出识别出单词的二元属性，填写符号表。单词符号的类型包括关键字，标识符，界符，运算符，整数，浮点数，字符串。每种单词符号的具体要求如下：

**关键字（KW，不区分大小写）包括：**

类别	语法关键字
查询表达式（4 个）	(1) SELECT, (2) FROM, (3) WHERE, (4) AS
插入表达式（3 个）	(5) INSERT, (6) INTO, (7) VALUES
更新表达式（1 个）	(8) UPDATE
删除表达式（1 个）	(9) DELETE
连接操作（3 个）	(10) JOIN, (11) LEFT, (12) RIGHT
聚合操作（4 个）	(13) MIN, (14) MAX, (15) AVG, (16) SUM
集合操作（2 个）	(17) UNION, (18) ALL
组操作（4 个）	(19) GROUP BY, (20) HAVING, (21) DISTINCT, (22) ORDER BY
条件语句（5 个）	(23) TRUE, (24) FALSE, (25) IS, (26) NOT, (27) NULL

表 1 SQL--关键字

**运算符（OP）包括：**

运算符类型	语法关键字
比较运算符（7个）	(1) =, (2) >, (3) <, (4) >=, (5) <=, (6) !=, (7) <=>
逻辑运算符（5个）	(8) AND, (9) &&, (10)   , (11) OR, (12) XOR
属性运算符（1个）	(13) .

表 2 SQL--运算符

界符 (SE) 包括:

类型	语法关键字
界符 (3个)	(1) (, (2) ), (3) ,

表 3 SQL--界符

标识符 (IDN) 为字母、数字和下划线 ( ) 组成的不以数字开头的串  
整数 (INT)、浮点数 (FLOAT) 的定义与 C 语言相同

字符串 (STR) 定义与 C 语言相同

- (2) 实现语言: C/C++/Java/Python;
- (3) 操作目的: 生成符号表; 将待分析代码转化为语法分析器可接受的序列。

2、语法分析器

(1) 完成 SQL--语言的语法分析器, 语法分析器的输入为 SQL--语言代码的 Token 序列, 输出用最左推导或规范规约产生语法树所用的**产生式序列**。SQL--语言文法须包含以下操作: (具体语法详见附件):

- ① 查询语句;
- ② 插入语句;
- ③ 更新语句;
- ④ 删除语句;
- ⑤ 表连接操作 (JOIN, LEFT JOIN, RIGHT JOIN)
- ⑥ 聚合操作 (MIN, MAX, SUM, COUNT)
- ⑦ 组操作 (GROUP BY, HAVING, ORDER BY, DISTINCT)
- ⑧ 集合操作 (UNION, UNION ALL)

例如: 采用 LL (1) 语法分析方法构建语法分析器需要完成以下内容:

- ① FIRST 集合、FOLLOW 集合;
- ② 对待编译代码规约使用的产生式序列。

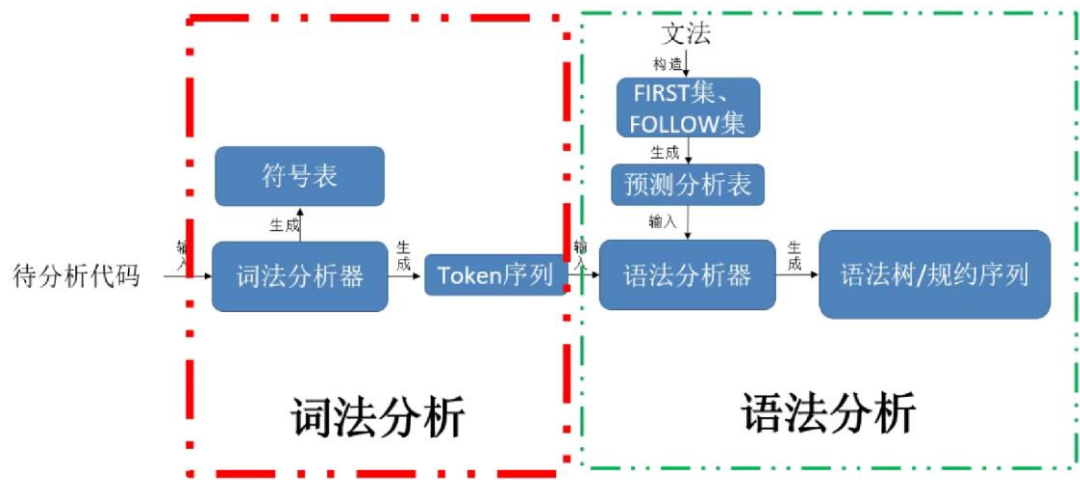


图 1 LL (1) 词法语法分析器流程图

- (2) 实现语言: C/C++/Java/Python。

### 三、输出示例

#### 1、词法分析输出示例

```
SELECT t.c FROM t WHERE t.a > 0
```

代码 1 待测 SQL 代码

(1) 【必须按规定格式输出】输出 Token 序列：

Token 输出格式：

**[待测代码中的单词符号] [TAB] <[单词符号种别],[单词符号内容]>**

其中，单词符号种别为 KW（关键字）、OP（运算符）、SE（界符）、IDN（标识符）INT（整形数）、FLOAT（浮点数）、STR（字符串）；单词符号内容 KW、OP、SE 为其编号（见表 1、表 2、表 3），其余类型为其值。

```
SELECT <KW,1>
t    <IDN,t>
.    <OP,13>
c    <IDN,c>
FROM  <KW,2>
t    <IDN,t>
WHERE <KW,3>
t    <IDN,t>
.    <OP,13>
a    <IDN,a>
>    <OP,2>
0    <INT,0>
```

代码 2 Token 序列示例

#### 2、语法分析输出示例（以预测分析为例）

(1) 【可选】输出 FIRST 集和 FOLLOW 集（此处仅展示一部分）：

**FIRST:**

```
functionCall=[AVG, MAX, MIN, SUM],
selectClause=[FROM, GROUP, HAVING, ORDER],
expression=[NOT, !, IDN, -, STR, FLOAT, INT, TRUE, FALSE, AVG, MAX, MIN, SUM],
comparisonOperator=[=, >, <, <=, >=, !=, <=>],
selectElementHead=[*, IDN, AVG, MAX, MIN, SUM],
.....
```

**FOLLOW:**

```
functionCall=[AS, IDN, ,, FROM, GROUP, HAVING, ORDER, =, >, <, <=, >=, !=, <=>, IS, AND, &&, XOR, OR,
||, UNION, JOIN, WHERE, LEFT, RIGHT, []],
selectElements=[FROM, GROUP, HAVING, ORDER],
expression=[ORDER, UNION, ,, HAVING, JOIN, WHERE, GROUP, LEFT, RIGHT, []],
logicalOperator=[NOT, !, IDN, -, STR, FLOAT, INT, TRUE, FALSE, AVG, MAX, MIN, SUM],
.....
```

代码 3 FIRST 集、FOLLOW 集部分输出示例

(2)【必须按规定格式输出】输出规约序列（此处仅展示一部分）：

**输出格式：**

**[序号] [TAB] [选用规则序号] [TAB] [栈顶符号]#[面临输入符号] [TAB] [执行动作]**

其中，选用规则序号见附件文法规则；执行动作为“reduction”（归约），“move”（LL 分析的跳过或 LR 分析的移进），“accept”（接受）或“error”（错误）。

```
1  1  root#SELECT  reduction
2  2  dmlStatement#SELECT  reduction
3  6  selectStatement#SELECT reduction
4  /  SELECT#SELECT move
5  14 unionType#IDN reduction
6  26 selectElements#IDN  reduction
7  28 selectElementHead#IDN reduction
8  31 selectElement#IDN reduction
9  49 fullColumnName#IDN  reduction
10 48 uid#IDN  reduction
11 /  IDN#IDN  move
12 50 dottedId#.  reduction
13 /  .#.  move
14 48 uid#IDN  reduction
15 /  IDN#IDN  move
.....
```

代码 4 归约序列部分输出示例

注：分析栈左端为栈顶，输入串过长没有在输出实例中进行展示，输入串即词法分析器生成的 Token 序列。

## 四、提交要求

### 1、源代码

包括词法分析器、语法分析器

### 2、开发报告

对项目的开发过程进行详细的描述，包括（1）词法分析器算法描述，输出格式说明，源程序编译步骤；（2）语法分析器的算法描述，创建的分析表（预测分析表、LR 分析表等），输出格式说明，源程序编译步骤。

### 3、测试报告

在给出的测试用例上分析后词法分析器以及语法分析器的输出截图（**注意需要按照要求格式输出**）。

## 五、展示验收

每组 10 分钟时间，通过 PPT+演示的形式展示本组完成的大作业。具体时间地点待定。

## 六、注意事项：

- (1) 采用附录中给出的文法编写程序。
- (2) 不得借助 Lex/Yacc, ANTLR 等编译器自动生成工具。
- (3) 所有出现在报告、代码、展示 PPT 中的内容, 若非原创, 需要明确标明内容来源, 并给出正规的引用。未标明引用而采纳他人内容的情况, **视为抄袭**, 遵照天津大学相关条例处理。
- (4) 小组内各位组员需合理分工配合完成大作业。PPT 展示时需明确每位组员负责完成的工作内容。

## 附录

注: 文法中出现的 $\epsilon$ 代表 $\epsilon$ , 文法中出现的 IDN (标识符)、INT (整数)、FLOAT (浮点数)、STR (字符串) 参照词法分析器中定义, 所有的关键字、界符、运算符均以原本形式存在。  
//表示注释, 为了方便理解语义所用, 在分析过程中可以删去。

### SQL 文法:

```
1. root -> dmlStatement

2. dmlStatement -> selectStatement
3. dmlStatement -> insertStatement
4. dmlStatement -> updateStatement
5. dmlStatement -> deleteStatement

6. selectStatement -> querySpecification unionStatements
7. unionStatements -> unionStatement unionStatements
8. unionStatements -> $

9. unionStatement -> unionStatementKey unionStatementQuery
10. unionStatementKey -> UNION unionType
11. unionStatementQuery -> querySpecification

12. unionType -> ALL
13. unionType -> DISTINCT
14. unionType -> $

15. querySpecification -> SELECT unionType selectElements selectClause
16. querySpecification -> ( querySpecification )
17. selectClause -> fromClause groupByClause havingClause orderByClause

18. fromClause -> FROM tableSources whereExpression
19. fromClause -> $

20. groupByClause -> GROUP BY expressions
21. groupByClause -> $

22. havingClause -> HAVING expression
23. havingClause -> $

24. orderByClause -> ORDER BY expressions
25. orderByClause -> $

26. selectElements -> selectElementHead selectElementListRec
27. selectElementHead -> *
28. selectElementHead -> selectElement
29. selectElementListRec -> , selectElement selectElementListRec
30. selectElementListRec -> $

31. selectElement -> fullColumnName elementNameAlias
32. selectElement -> functionCall elementNameAlias
```

```

33. elementNameAlias -> uid
34. elementNameAlias -> AS uid
35. elementNameAlias -> $

36. tableSources -> tableSource tableSourceListRec
37. tableSourceListRec -> , tableSource tableSourceListRec
38. tableSourceListRec -> $

39. tableSource -> tableSourceItem joinParts
40. joinParts -> joinPart joinParts
41. joinParts -> $

42. tableSourceItem -> tableName elementNameAlias
43. tableSourceItem -> ( tableSources )

44. tableName -> uid

45. uidList -> uid uidListRec
46. uidListRec -> , uid uidListRec
47. uidListRec -> $

48. uid -> IDN

49. fullColumnName -> uid dottedId

50. dottedId -> . uid
51. dottedId -> $

52. expressions -> expression expressionRec
53. expressionRec -> , expression expressionRec
54. expressionRec -> $

// 为便于理解语义，将含有左递归以及二义性的原 MySQL 文法规则以注释形式给出，此处已修改为 LL(1)的形式
// expression -> opposite expression
// expression -> expression logicalOperator expression
// expression -> predicate IS oppositeOrNot trueValue
// expression -> predicate
55. expression -> opposite expression
56. expression -> predicate expressionRight
57. expressionRight -> logicalOperator expression
58. expressionRight -> IS oppositeOrNot trueValue
59. expressionRight -> $
60. opposite -> NOT
61. opposite -> !
62. oppositeOrNot -> NOT
63. oppositeOrNot -> $
64. trueValue -> TRUE
65. trueValue -> FALSE
66. trueValue -> UNKNOWN

// 为便于理解语义，将含有左递归以及二义性的原 MySQL 文法规则以注释形式给出，此处已修改为 LL(1)的形式
// predicate -> predicate comparisonOperator predicate
// predicate -> expressionAtom
67. predicate -> expressionAtom predicateRight
68. predicateRight -> comparisonOperator predicate
69. predicateRight -> $

70. expressionAtom -> constant
71. expressionAtom -> fullColumnName
72. expressionAtom -> functionCall

73. constant -> stringLiteral
74. constant -> decimalLiteral
75. constant -> - decimalLiteral
76. constant -> booleanLiteral

77. decimalLiteral -> FLOAT
78. decimalLiteral -> INT

```

```

79. comparisonOperator -> =
80. comparisonOperator -> >
81. comparisonOperator -> <
82. comparisonOperator -> <=
83. comparisonOperator -> >=
84. comparisonOperator -> !=
85. comparisonOperator -> <=>

86. logicalOperator -> AND
87. logicalOperator -> &&
88. logicalOperator -> XOR
89. logicalOperator -> OR
90. logicalOperator -> ||

91. stringLiteral -> STRING

92. booleanLiteral -> TRUE
93. booleanLiteral -> FALSE

94. functionCall -> aggregateWindowedFunction

95. aggregateWindowedFunction -> function ( unionType fullColumnName )
96. function -> AVG
97. function -> MAX
98. function -> MIN
99. function -> SUM

100. joinPart -> JOIN tableSourceItem joinRightPart
101. joinRightPart -> joinDirection JOIN tableSourceItem ON expression
102. joinRightPart -> ON expression joinDirection JOIN tableSourceItem ON expression
103. joinDirection -> LEFT
104. joinDirection -> RIGHT

105. insertStatement -> insertKeyword tableName insertStatementRight
106. insertStatementRight -> insertStatementValue
107. insertStatementRight -> ( uidList ) insertStatementValue
108. insertKeyword -> INSERT into
109. into -> INTO
110. into -> $

111. insertStatementValue -> insertFormat ( expressionsWithDefaults ) expressionsWithDefaultsListRec
112. insertFormat -> VALUES
113. insertFormat -> VALUE
114. expressionsWithDefaultsListRec -> , ( expressionsWithDefaults ) expressionsWithDefaultsListRec
115. expressionsWithDefaultsListRec -> $

116. expressionsWithDefaults -> expressionOrDefault expressionOrDefaultListRec
117. expressionsWithDefaults -> $
118. expressionOrDefaultListRec -> , expressionOrDefault expressionOrDefaultListRec
119. expressionOrDefaultListRec -> $

120. expressionOrDefault -> expression
121. expressionOrDefault -> DEFAULT

122. updateStatement -> UPDATE tableName elementNameAlias SET updatedElement updatedElementListRec
whereExpression
123. updatedElementListRec -> , updatedElement updatedElementListRec
124. updatedElementListRec -> $
125. whereExpression -> WHERE expression
126. whereExpression -> $

127. updatedElement -> fullColumnName = expressionOrDefault

128. deleteStatement -> DELETE FROM tableName deleteStatementRight
129. deleteStatementRight -> whereExpression
130. deleteStatementRight -> ( uidList ) whereExpression

```