Semantic Rules for AST Creation

Group Number 45

Bhavesh Kotwani 2018B1A70614P

Shashank Tiwari 2018B1A70630P

Ashutosh Sharma 2019A7PS0040P

Ananya Bahadur Vasu 2019A7PS1135P

Palak Hemant Pariawala 2019A7PS1141P

1. program --> otherFunctions mainFunction

   program .ptr = createChildren(otherFunctions .ptr, mainFunction .ptr);

2. mainFunction --> TK_MAIN stmts TK_END

   mainFunction .ptr = createPtr('main', stmts .ptr);

3. otherFunctions --> function otherFunctions 1

   otherFunctions .ptr = createPtr(function .ptr, otherFunctions 1.ptr);

4. otherFunctions --> EPS

   otherFunctions .ptr = NULL;

5. function --> TK_FUNID input_par output_par TK_SEM stmts TK_END

   function .ptr = createPtr(TK_FUNID.value, input_par .ptr, output_par .ptr, stmts .ptr);

6. input_par --> TK_INPUT TK_PARAMETER TK_LIST TK_SQL parameter_list TK_SQR

   input_par .ptr = parameter_list .ptr;

7. output_par --> TK_OUTPUT TK_PARAMETER TK_LIST TK_SQL parameter_list TK_SQR

   output_par .ptr = parameter_list .ptr;

8. output_par --> EPS

   output_par .ptr = NULL;

9. parameter_list --> dataType TK_ID remaining_list

   newnode = createPtr(dataType .name, TK_ID.value);

   parameter_list .ptr = createPtr(newnode, remainingList .ptr);

10. dataType --> primitiveDatatype

    dataType .name = primitiveDatatype .name;

11. dataType --> constructedDatatype

    dataType .name = constructedDatatype .name;

12. primitiveDatatype --> TK_INT

    primitiveDatatype .name = ‘int’;

13. primitiveDatatype --> TK_REAL

    primitiveDatatype .name= ‘real’;

14. constructedDatatype --> TK_RECORD TK_RUID

    constructedDatatype .name = TK_RUID.name;

15. constructedDatatype --> TK_UNION TK_RUID

    constructedDatatype .name = TK_RUID.name;

16. constructedDatatype --> TK_RUID

constructedDatatype .name = TK_RUID.val;

17. remaining_list --> TK_COMMA parameter_list

remaining_list .ptr =   parameter_list .ptr;

18. remaining_list --> EPS

remaining_list .ptr = NULL;

19. stmts --> typeDefinitions   declarations   otherStmts   returnStmt

stmts .ptr = createPtr(typeDefinitions .ptr, declarations .ptr, otherStmts .ptr, returnStmt .ptr);

20. typeDefinitions --> actualOrRedefined   typeDefinitions 1

typeDefinitions .ptr = createPtr(actualOrRedefined .ptr, typeDefinitions 1..ptr);

21. typeDefinitions --> EPS

typeDefinitions .ptr = NULL;

22. actualOrRedefined --> typeDefinition

actualOrRedefined .ptr = typeDefinition .ptr;

23. actualOrRedefined --> definetypestmt

actualOrRedefined .ptr = definetypestmt .ptr;

24. typeDefinition --> TK_RECORD TK_RUID fieldDefinitions TK_ENDRECORD

typeDefinition .ptr = createPtr(TK_RUID, fieldDefinitions .ptr);

25. typeDefinition --> TK_UNION TK_RUID fieldDefinitions TK_ENDUNION

typeDefinition .ptr = createPtr(TK_RUID, fieldDefinitions .ptr);

26. fieldDefinitions --> fieldDefinition 1 fieldDefinition 2 moreFields

newnode = createPtr(fieldDefinition 2 .ptr, moreFields .ptr);

fieldDefinitions .ptr = createPtr(fieldDefinition 1.ptr, newnode);

27. fieldDefinition --> TK_TYPE fieldType TK_COLON TK_FIELDID TK_SEM

fieldDefinition .ptr = createPtr(fieldType .name , TK_FIELDID.name);

28. fieldtype --> primitiveDatatype

fieldType .ptr = primitiveDatatype .ptr;

29. fieldtype --> TK_RUID

fieldType .name = TK_RUID.val;

30. moreFields --> fieldDefinition moreFields 1

moreFields .ptr = createPtr(fieldDefinition .ptr, moreFields 1.ptr);

31. moreFields --> EPS

moreFields .ptr = NULL;

32. declarations --> declaration declarations 1

declarations .ptr = createPtr(declaration .ptr, declarations 1.ptr);

33. declarations --> EPS

declarations .ptr = NULL;

34. declaration --> TK_TYPE dataType TK_COLON TK_ID global_or_not TK_SEM

declaration .ptr = createPtr(dataType .name, TK_ID.value, global_or_not .global);

35. global_or_not --> TK_COLON TK_GLOBAL

global_or_not .global = True;

36. global_or_not --> EPS

global_or_not .global = False;

37. otherStmts --> stmt otherStmts

otherStmts .ptr = createPtr(stmt .ptr, otherStmts .ptr);

38. otherStmts --> EPS

otherStmts .ptr = NULL;

39. stmt --> assignmentStmt

stmt .ptr = assignmentStmt .ptr;

40. stmt --> iterativeStmt

stmt .ptr = iterativeStmt .ptr;

41. stmt --> conditionalStmt

stmt .ptr = conditionalStmt .ptr;

42. stmt --> ioStmt

stmt .ptr = ioStmt .ptr;

43. stmt --> funCallStmt

stmt .ptr = funCallStmt .ptr;

44. assignmentStmt --> SingleOrRecId TK_ASSIGNOP arithmeticExpression TK_SEM

assignmentStmt .ptr = createPtr(‘ <---’, SingleOrRecId .ptr, arithmeticExpression .ptr);

45. SingleOrRecId --> TK_ID option_single_constructed

SingleOrRecId .ptr = createPtr(TK_ID.value, option_single_constructed .ptr);

46. option_single_constructed --> EPS

option_single_constructed .ptr = NULL;

47. option_single_constructed --> oneExpansion   moreExpansions

option_single_constructed .ptr = createPtr(oneExpansion .ptr, moreExpansions .ptr);

48. oneExpansion --> TK_DOT TK_FIELDID

oneExpansion .ptr = TK_FIELDID.ptr;

49. moreExpansions --> oneExpansion   moreExpansions 1

moreExpansions .ptr = createPtr(oneExpansion .ptr, moreExpansions 1.ptr);

50. funCallStmt --> outputParameters TK_CALL TK_FUNID TK_WITH TK_PARAMETERS
inputParameters TK_SEM

funCallStmt .ptr = createPtr(TK_FUNID.value, outputParameters .ptr, inputParameters .ptr);

51. outputParameters --> TK_SQL idList TK_SQR TK_ASSIGNOP

outputParameters .ptr = createPtr(‘< ---’, idList .ptr);

52. outputParameters --> EPS

outputParameters .ptr = NULL;

53. inputParameters --> TK_SQL idList TK_SQR

inputParameters .ptr = idList .ptr;

54. iterativeStmt --> TK_WHILE TK_OP booleanExpression TK_CL stmt   otherStmts TK_ENDWHILE

newnode = createPtr(stmt .ptr,   otherStmts .ptr);

iterativeStmt .ptr = createPtr('while',   booleanExpression .ptr, newnode);

55. conditionalStmt --> TK_IF TK_OP booleanExpression TK_CL TK_THEN stmt   otherStmts elsePart

newnode = createPtr(stmt .ptr,   otherStmts .ptr);

conditionalStmt = createPtr('if', booleanExpression .ptr, newnode, elsePart .ptr);

56. elsePart --> TK_ELSE stmt   otherStmts TK_ENDIF

newnode = createPtr(stmt .ptr,   otherStmts .ptr);

elsePart .ptr = createPtr('else', newnode);

57. elsePart --> TK_ENDIF

elsePart .ptr = NULL;

58. ioStmt --> TK_READ TK_OP var TK_CL TK_SEM

ioStmt .ptr = createPtr('read', var .ptr);

59. ioStmt --> TK_WRITE TK_OP var TK_CL TK_SEM

ioStmt .ptr = createPtr('write', var .ptr);

60. arithmeticExpression --> term   expPrime

arithmeticExpression .ptr = createPtr(term .ptr, expPrime .ptr)

61. expPrime --> lowPrecedenceOperators term expPrime 1

expPrime 1.ptr = createPtr(lowPrecedenceOperators .name, expPrime .ptr, term .ptr);

expPrime .ptr = expPrime 1.ptr;

62. expPrime --> EPS

expPrime .ptr = NULL;

63. term --> factor termPrime

term .ptr = createPtr(factor .ptr, termPrime .ptr)

64. termPrime --> highPrecedenceOperators factor termPrime 1

termPrime 1.ptr = createPtr(highPrecedenceOperators .name, termPrime .ptr, factor .ptr);

termPrime .ptr = termPrime 1.ptr;

65. termPrime --> EPS

termPrime .ptr = NULL;

66. factor --> TK_OP arithmeticExpression TK_CL

factor .ptr = arithmeticExpression .ptr;

67. factor --> var

factor .ptr = var .ptr;

68. highPrecedenceOperators --> TK_MUL

highPrecedenceOperators .name = ‘*’;

69. highPrecedenceOperators --> TK_DIV

highPrecedenceOperators .name = ‘/’;

70. lowPrecedenceOperators --> TK_PLUS

lowPrecedenceOperators .name = ‘+’;

71. lowPrecedenceOperators --> TK_MINUS

lowPrecedenceOperators .name = ‘-’;

72. booleanExpression --> TK_OP booleanExpression 1 TK_CL logicalOp TK_OP booleanExpression 2 TK_CL

booleanExpression .ptr = createPtr(logicalOp .name, booleanExpression 1.ptr, booleanExpression 2.ptr);

73. booleanExpression --> var 1 relationalOp var 2

booleanExpression .ptr = createPtr(relationalOp .ptr, var 1.ptr, var 2.ptr);

74. booleanExpression --> TK_NOT TK_OP booleanExpression 1 TK_CL

booleanExpression .ptr = createPtr(‘~’, booleanExpression 1.ptr);

75. var --> singleOrRecId

var .ptr = singleOrRecId .ptr;

76. var --> TK_NUM

var .ptr = createPtr(TK_NUM.value);

77. var --> TK_RNUM

var .ptr = createPtr(TK_RNUM.value);

78. logicalOp --> TK_AND

logicalOp .name = ‘&&&’;

79. logicalOp --> TK_OR

logicalOp .name = ‘@@@’;

80. relationalOp --> TK_LT

relationalOp .name = ‘< ’;

81. relationalOp --> TK_LE

relationalOp .name = ‘ <=’;

82. relationalOp --> TK_EQ

relationalOp .name = ‘==’;

83. relationalOp --> TK_GT

relationalOp .name = ‘> ’;

84. relationalOp --> TK_GE

relationalOp .name = ‘ >=’;

85. relationalOp --> TK_NE

relationalOp .name = ‘!=’;

86. returnStmt --> TK_RETURN optionalReturn TK_SEM

returnStmt .ptr = optionalReturn .ptr;

87. optionalReturn --> TK_SQL idList TK_SQR

optionalReturn .ptr = idList .ptr;

88. optionalReturn --> EPS

optionalReturn .ptr = NULL;

89. idList --> TK_ID more_ids

idList .ptr = createPtr(TK_ID.value,   more_ids .ptr);

90. more_ids --> TK_COMMA idList

more_ids .ptr = idList .ptr;

91. more_ids --> EPS

more_ids .ptr = NULL;