Manual

Shuai Yuan

The program "translation" parse SQL phrases into Why3ML program. It mainly contains the following parts:

- header of Why3ML program, mainly commands of importing modules.
- parser for the SQL table definition, this part translate the "CREATE TABLE" phrase into the definition of the type of the corresponding table's tuple.
- parser for the SQL assertion, this part translate the "CREATE ASSERTION" phrase into "predicate" in the Why3ML program.
- parser for the SQL INSERT command, this part translate the SQL INSERT command into a method in the Why3ML program.
- parser for the SQL DELETE command, this part translate the SQL INSERT command into a method in the Why3ML program.
- parser for the SQL UPDATE command, this part translate the SQL INSERT command into a method in the Why3ML program.

1 Parser for the SQL Table Definition

The source language of SQL table definition is expressed in the following grammar:

<table definition>	::=	CREATE TABLE <*table name*>
		(<table element list>)
<table element list>	::=	<table element>
	|	<table element list>, <table element>
<table element>	::=	<*column name*> <data type>
<data type>	::=	INTEGER
		SMALLINT
		FLOAT
		NUMERIC
		BOOLEAN

2 Parser for the SQL Assertion

The grammar of SQL assertion is:

CREATE ASSERTION <*assertion name*>
CHECK <exists predicate>

<exists predicate>	::=	[NOT] EXISTS (<query expression>)
<query expression>	::=	SELECT *
		FROM <table list>
		WHERE <search condition>
<table list>	::=	<*table name*> <*tuple name*>
		<table list>, <*table name*> <*tuple name*>
<search condition>	::=	<boolean term>
	|	<search condition> OR <boolean term>
<boolean term>	::=	<boolean factor>
	|	<boolean term> AND <boolean factor>
<boolean factor>	::=	<predicate>
	|	[NOT] (<search condition>)
<predicate>	::=	<exists predicate>
	|	<comparison predicate>
	|	<between predicate>
	|	<in predicate>
	|	<null predicate>
<comparison predicate>	::=	<expression$_1$ > <comp op> <expression$_2$ >
<comp op>	::=	= | <> | < | ≤ | > | ≥
<expression>	::=	<term>
	|	<expression> {+ | −} <term>
<term>	::=	<factor>
	|	<term> {∗ | /} <factor>
<factor>	::=	(<expression>)
	|	[+ | −] <*constant*>
	|	[+ | −] <column>
<column>	::=	<*tuple name*>.<*attribute name*>
<between predicate>	::=	<expression> [NOT]
		BETWEEN <*constant*$_1$ > AND <*constant*$_2$ >
<in predicate>	::=	<expression>[NOT] IN (<in value list>)
<in value list>	::=	<*constant*>
	|	<in value list>, <*constant*>
<null predicate>	::=	<column> IS [NOT] NULL

3 Parser for the SQL INSERT statement

The grammar of SQL insert statement is:

<insert statement>	::=	INSERT INTO <*table name*> VALUES (<column value list>)
	|	INSERT INTO <*table name*> (<column name list>)
		VALUES (<column value list>)
<column name list>	::=	<*column name*>
	|	<column name list>, <*column name*>
<column value list>	::=	<column value>
	|	<column value list>, <*column value*>

4 Parser for the SQL DELETE statement

The grammar of SQL delete statement is:

<delete statement>	::=	DELETE FROM <*target table name*> [USING <table reference list>] [WHERE <search condition>]
<table reference list>	::=	<*table name*>
	|	<table reference list>, <*table name*>
<search condition>	::=	<boolean term>
	|	<search condition> OR <boolean term>
<boolean term>	::=	<boolean factor>
	|	<boolean term> AND <boolean factor>
<boolean factor>	::=	<predicate>
	|	[NOT] (<search condition>)
<predicate>	::=	<comparison predicate>
	|	<between predicate>
	|	<in predicate>
	|	<null predicate>

The left parts are the same as those in the SQL assertion, so they are omitted in this manual.

5 Parser for the SQL UPDATE statement

The grammar of SQL update statement is:

<update statement>	::=	UPDATE <*target table name*> SET <set clause list> [FROM <table reference list>] [WHERE <search condition>]
<set clause list>	::=	<set clause>
		<set clause list>, <set clause>
<set clause>	::=	<set column> = <*constant*>
<set column>	::=	<*table name*>.<*attribute name*>

The left parts are the same as those in the SQL delete statement grammar, so they are omitted in this manual.