DB project report

Implementing DML - 2014-17184 김연우

Implementation

The last step of the project is to implement 3 kinds of DML (Data Manipulation Language): insert, delete, and select. The brief syntax of each query is as follows:

< INSERT QUERY > =

insert into < TABLE NAME > < INSERT COLUMNS AND SOURCE >

< INSERT COLUMNS AND SOURCE > =

[< COLUMN NAME LIST >] < VALUE LIST >

< DELETE QUERY > = *delete from* < TABLE NAME > [< WHERE CLAUSE >]

< SELECT QUERY > = *select* < SELECT LIST > < TABLE EXPRESSION >

< SELECT LIST > =

* | < SELECTED COLUMN > (< COMMA > < SELECTED COLUMN >)*

< TABLE EXPRESSION > = < FROM CLAUSE > [< WHERE CLAUSE >]

See *ScratchDBMSGrammar.jj* for more information on the grammar.

For each query, concrete implementation is as follows:

- **Insert query**

First, the program gets < TABLE NAME > as a string and checks if a table with the given name exists in the database. If not, it emits *NoSuchTable* error. Then, it checks if < COLUMN NAME LIST > is given in the query. If it is given, the program stores the order of the list inside each attribute in the table. When parsing the < VALUE LIST >, the program again checks if < COLUMN NAME LIST> was given in the query. If it was, the program iterates through the attribute list in the table, checks the stored order in each attribute and construct a record following the order. Else, it iterates through the attribute list of the table and constructs a record with values in the < VALUE LIST > preserving the order. During this process, *InsertTypeMismatchError, InsertColumnNonNullableError, InsertColumnExistenceError* are checked.

One of the key parts of implementing insert is to check primary key and foreign key constraints. Table class stores the indices of primary key and each foreign key it has. Thus, checking primary key constraint is simply done by retrieving the values of the primary key indices in a record and checking if the values are the same as the new primary key values. However, for foreign key, it is a bit more complicated since the order of the keys might not be preserved; the order of primary key in the referred table and that of foreign key in the referring table might not be the same. Hence, the program makes sure that all foreign keys are stored in the order of the primary key in the referred table. Then it is possible to check foreign key constraint in a similar way as the program checks primary key constraint.

- **Delete query**

Delete query uses where expression to evaluate each record in a table and delete the record that evaluate to true. Henceforward, to implement the query, I had to parse < WHERE CLAUSE > into an object that contains all the information. BooleanValueExpression, the class representing < WHERE CLAUSE >, has eval method that takes a table and a record as input, and checks if the boolean value expression evaluates to true for the given record. Regarding delete, all the errors except referential integrity error are checked before executing the query - in .jj file.

Regarding referential integrity, the program first starts by checking if the request is cascade - able. That is, it checks whether all the foreign keys of referring tables are nullable. If they are, then the program cascades all the records, with the same foreign key values with the primary key of the table user wants to delete, to null. Else, the program again checks if there are any records that actually contain the foreign key value the same as the primary key value of the requested table. Then, if not, it deletes the table.

- **Select query**

In contrast to delete query, select query can take many tables in < SELECT LIST >. For the sake of consistency with delete query, the program creates a temporary table that has all the attributes in the tables designated in < SELECT LIST >.

When doing Cartesian-product, the boolean value expression, which is the result of < WHERE CLAUSE > is passed as an argument to check if it evaluates to true for the new record. If it does, the program immediately prints the record. This process is repeated for all the Cartesian-producted records, and select query is then finished. All the errors, *SelectTableExistenceError, SelectColumnResolveError,* are dealt considering the alias of tables and attributes.

Assumption

- **Aliases are case-insensitive**
- **All the values have length shorter than 20** (**otherwise print-formatting will not work for select query**)
- **Identifier of an attribute that has type A, and value of type B are incomparable.**

How to compile & execute

Use the makefile provided.

Make jj: uses javacc to compile the given .jj file to .java files.

Make compile: compiles .jj, and compiles the whole source code.

Make run: compiles and executes the program.

To execute .jar file, type java -cp PRJ1-3_2014-17184.jar ScratchDBMSParser

Comments

It was a lot more complicated to implement DMLs, compared to implementing DDLs.