Internal Handling of Dynamic Statements

Natural automatically provides for the preparation and execution of each SQL statement and handles the opening and closing of cursors used for scanning a table.

This section covers the following topics:

- NDBIOMO
- Statement Table
- Processing of SQL Statements Issued by Natural

NDBIOMO

As each dynamic execution of an SQL statement requires a statically defined DECLARE STATEMENT and DECLARE CURSOR statement, a special I/O module (NDBIOMO) is provided which contains a fixed number of these STATEMENTs and CURSORs. This number is specified during the generation of NDBIOMO (see Step 2 in Steps Common to all Environments, Installing Natural for DB2).

Statement Table

If possible, an SQL statement is only prepared once and can then be executed several times if required. For this purpose, Natural internally maintains a table of all SQL statements that have been prepared and assigns each of these statements to a DECLAREd STATEMENT in NDBIOMO. In addition, this table maintains the cursors used by the SQL statements SELECT, FETCH, UPDATE (Positioned), and DELETE (Positioned).

Each SQL statement is uniquely identified by:

- the name of the Natural program that contains this SQL statement,
- the line number of the SQL statement in this program,
- the name of the Natural library, into which this program was stowed,
- the time stamp when this program was stowed.

Once a statement has been prepared, it can be executed several times with different variable values, using the dynamic SQL statement EXECUTE USING DESCRIPTOR or OPEN CURSOR USING DESCRIPTOR respectively.

When the full capacity of the statement table is reached, the entry for the next prepared statement overwrites the entry for a free statement whose latest execution is the least recent one.

When a new SELECT statement is requested, a free entry in the statement table with the corresponding cursor is assigned to it and all subsequent FETCH, UPDATE, and DELETE statements referring to this SELECT statement will use this cursor. Upon completion of the sequential scanning of the table, the cursor is released and free for another assignment. While the cursor is open, the entry in the statement table is marked as used and cannot be reused by another statement.

If the number of nested FIND (SELECT) statements reaches the number of entries available in the statement table, any further SQL statement is rejected at execution time and a Natural error message is returned.

The size of the statement table depends on the size specified for NDBIOMO. Since the statement table is contained in the DB2 buffer area, the DB2SIZE parameter (see Natural Parameter Modification for DB2, Installing Natural for DB2) may not be sufficient and may need to be increased.

Processing of SQL Statements Issued by Natural

The embedded SQL uses cursor logic to handle SELECT statements. The preparation and execution of a SELECT statement is done as follows:

1. The typical SELECT statement is prepared by a program flow which contains the following embedded SQL statements (note that *X* and *SQLOBJ* are SQL variables, not program labels):

   ```
   DECLARE SQLOBJ STATEMENT
   DECLARE X CURSOR FOR SQLOBJ
   INCLUDE SQLDA (copy SQL control block)
   ```

   Then, the following statement is moved into SQLSOURCE:

   ```
   SELECT PERSONNEL_ID, NAME, AGE
    FROM EMPLOYEES
    WHERE NAME IN (?, ?)
       AND AGE BETWEEN ? AND ?
   ```

   (The question marks above are parameter markers which indicate where values are to be inserted at execution time.)

   ```
   PREPARE SQLOBJ FROM SQLSOURCE
   ```

2. Then, the SELECT statement is executed as follows:

   ```
   OPEN X USING DESCRIPTOR SQLDA
   FETCH X USING DESCRIPTOR SQLDA
   ```

   The descriptor SQLDA is used to indicate a variable list of program areas. When the OPEN statement is executed, it contains the address, length, and type of each value which replaces a parameter marker in the WHERE clause of the SELECT statement. When the FETCH statement is executed, it contains the address, length, and type of all program areas which receive fields read from the table.

   When the FETCH statement is executed for the first time, it sets the Natural system variable *NUMBER to a non-zero value if at least one record is found that meets the search criteria. Then, all records satisfying the search criteria are read by repeated execution of the FETCH statement.

   To help improve performance, especially when using distributed databases, the DB2-specific FOR FETCH ONLY clause can be used. FOR FETCH ONLY is generated and executed if rows are to be retrieved only; that is, if no updating is to take place.

3. Once all records have been read, the cursor is released by executing the following statement:

```
CLOSE X
```