DB2 Cursors

Cursors

Up to this point we have retrieved at most one row at a time. If a program issues a SELECT ... INTO statement to retrieve several rows all at once then this would result in an error. However DB2 has a mechanism, called a cursor, that enables an application to retrieve a set of rows and then process the set, one row at a time.

Cursor Functions

We can assume that DB2 builds a results table to hold all the rows retrieved by executing a SELECT statement. DB2 uses a cursor to make the rows, from the results table, available to the application program. A cursor identifies the current row of the results table. When you use a cursor, the program can retrieve each row sequentially from the results table until end-of-data (i.e the not found condition SQLCODE=100). The SELECT statement used must be within a DECLARE CURSOR statement and cannot include an INTO clause. The DECLARE CURSOR statement defines and names the cursor, identifying the set of rows to be retrieved with the SELECT statement of the cursor.

The results table is processed very much like a sequential data set. The cursor must be opened (with an OPEN statement) before any rows are retrieved. A FETCH statement is used to retrieve the cursor's current row. FETCH can be executed repeatedly until all rows have been retrieved. When the end-of-data condition occurs the cursor must be closed with a CLOSE statement.

A program can have several cursors. Each cursor requires its own:

- DECLARE CURSOR statement to define the cursor.
- OPEN and CLOSE statements to open and close the cursor.
- FETCH statement to retrieve rows from the cursor's results table.

Declarations for host variables that are referenced in a DECLARE CURSOR statement must precede the DECLARE CURSOR statement. The DECLARE CURSOR statement must precede any statement that references the cursor.

You can use cursors to FETCH, UPDATE or DELETE a row from a table but you cannot use them to insert a row into a table.

The DECLARE CURSOR Statement

To define and identify a set of rows to be accessed with a cursor, issue a DECLARE CURSOR statement. This statement names a cursor and specifies a SELECT statement. The SELECT statement defines a set of rows that make up the results table.

The DECLARE CURSOR statement has the following format:

```
EXEC SQL
        DECLARE  cursor-name  CURSOR FOR
                SELECT column1, column2, ..
                FROM  table-name
                WHERE   condition
        FOR UPDATE OF  column, ...
        FOR FETCH ONLY
END-EXEC.
```

In the above example the SELECT statement shown is very simple, however, you can code far more complex SELECT statements within a DECLARE CURSOR statement.

The DECLARE CURSOR statement is usually placed in Working-Storage but it can go in the Procedure Division.

The FOR UPDATE OF Clause

This clause must be specified if you intend to update any (or all) of the rows in the identified table. In this clause you name each column you intend to update. If you do not specify the names of columns you will later update, you will receive an error code in the SQLCODE field when you try to update them.

A column of the identified table can be updated even though it is not part of the results table. That is, it can be specified in the FOR UPDATE OF clause even if it was not in the SELECT clause.

The FOR FETCH ONLY Clause

This clause has been available since the release of version 2.2. It is used to indicate that there is no intention to update the table. When it is coded DB2 may perform a block FETCH to improve efficiency.

Read-Only Results Table

The results table is read-only if the SELECT statement includes the DISTINCT keyword, a UNION operator, a column function, a GROUP BY clause, a HAVING clause or an ORDER BY clause. It is also read-only if the FROM clause identifies a read-only view or identifies more than one table or view.

If the results table is read-only then you cannot code the FOR UPDATE OF clause. This is a particular problem when a file of updates has a particular order to its records and you would like to code an ORDER BY clause so a merge can be performed.

The OPEN Statement

To tell DB2 you are ready to process the first row of the results table, issue an OPEN statement in your application program. When this occurs, DB2 processes the SELECT statement within the DECLARE CURSOR statement to identify the results table. This table is placed in Virtual Storage.

The format of the OPEN statement is

```
EXEC SQL
        OPEN  cursor-name
END-EXEC.
```

If any host variables are specified in the SELECT statement then the values in them at the time of the OPEN are used in creating the results table.

If you use CURRENT TIMESTAMP with cursors, CURRENT TIMESTAMP is evaluated once, at the time of then OPEN statement; this value is then used on all subsequent FETCH statements.

The FETCH Statement

To move the contents of a selected row into your program, use the FETCH statement. The SELECT statement within the DECLARE CURSOR statement defines the results table, but DB2 does not retrieve any data for the application program until a FETCH is issued.

When a FETCH statement is issued in an application program, DB2 uses the cursor to point to the next row in the results table, making it the current row. DB2 then moves the current row's contents into the program. This sequence is repeated each time a FETCH statement is issued until all the rows in the results table have been processed.

DB2 maintains the position of the current row until the next FETCH statement for the cursor is issued. Neither the UPDATE or DELETE statements alter the position of the current row within the results table, the DELETE statement simply marks the current row for deletion.

The FETCH statement has the following format:

```
EXEC SQL
        FETCH  cursor-name
        INTO  :host-variable1, :host-variable2, ...
END-EXEC.
```

You do not have to place the retrieved data into host variables, although this is the most commonly adopted method. It is possible to retrieve the data into column names as they have been defined in the program via the DECLARE statement or DCLGEN statement.

The CLOSE Statement

When you have finished processing the rows of the results table and you want to use the cursor again, issue a CLOSE statement to close the cursor.

The format of the CLOSE statement is as follows:

```
EXEC SQL
        CLOSE   cursor-name
END-EXEC.
```

You can let DB2 automatically close the cursor when the application program terminates but this practice is not recommended.

The UPDATE Statement

You can update the data of the current row retrieved by a program by using the UPDATE statement. To do this, issue an UPDATE ... WHERE CURRENT OF statement which has the following format:

```
EXEC SQL
        UPDATE  table-name
                SET  column-A = value, column-B = value, ...
        WHERE CURRENT OF  cursor-name
END-EXEC.
```

When used with a cursor, the UPDATE statement differs from the one seen previously:

- Only one row is updated, the current row.
- The WHERE clause identifies the cursor that points to the row to be updated.
- Each column to be updated must have been named previously in the FOR UPDATE OF clause of the DECLARE CURSOR statement.

After a row has been updated the cursor's position remains on that row until the next FETCH statement is issued.

As stated previously, the FOR UPDATE OF clause cannot be coded in a DECLARE CURSOR statement for a read-only results table. Also, updating with a cursor, with its need for a cursor declaration, FETCH...INTO...statement and UPDATE...WHERE CURRENT OF statement, requires considerably more coding than the simple embedded UPDATE statement. Finally, using the cursor update also precludes the use of indexes on columns specified in the FOR UPDATE OF clause and hence has performance implications. For these reasons think carefully before using cursor updates.

The DELETE Statement

You can delete the current row retrieved by a program by using the DELETE statement. To do this, issue a DELETE ... WHERE CURRENT OF statement which has the following format:

```
EXEC SQL
        DELETE FROM table-name
        WHERE CURRENT OF cursor-name
END-EXEC.
```

When used with a cursor, the DELETE statement differs from the one seen previously:

- Only one row is deleted, the current row.
- The WHERE clause identifies the cursor that points to the row to be deleted.

After a row has been deleted, you cannot update or delete

another row using that cursor until you issue a FETCH statement for the next row.

An example of using a Cursor

```
EXEC SQL
        DECLARE CURS-A CURSOR FOR
                SELECT EMPNO, LASTNAME, WORKDEPT
                FROM EMPLTABLE
                WHERE WORKDEPT = 'D11'
END-EXEC.

EXEC SQL
        OPEN CURS-A
END-EXEC.

EXEC SQL
        FETCH CURS-A
        INTO :EMP-NUM, :NAME2, :DEPT
END-EXEC.
```

perform the following until end-of data

print a report line

```
EXEC SQL
        FETCH CURS-A
        INTO :EMP-NUM, :NAME2, :DEPT
END-EXEC.

EXEC SQL
        CLOSE CURS-A
END-EXEC.
```