How to Improve the Performance in Nested Loops

Applies to:

ABAP

Summary

In a typical scenario where we have to use nested loops using two internal tables, the volume of the data in the internal tables will have an direct impact on the run time and thereby the performance.

In this code snippet we see how the concept of a home grown cursor is useful in improving the performance of the program.

Author(s): Ravi Kanth Talagana

Company: Intelligroup Asia Pvt. Ltd.

Created on: 15 November 2006

Author Bio

Ravi Kanth Talagana is an SAP Technical consultant working as a Programmer analyst in Intelligroup Asia Pvt. Ltd. He works in the SAP NETWEAVER XI competency.

Table of Contents

How to improve the performance in nested loops?

IN ABAP programs we come across situations where we have to loop a header internal table and do some processing for each of the lines in an Item internal table which have multiple records for the same header key. For this, we use a where condition in the inner loop, which takes time to search for the match. This search always starts from the first record, resulting in more time to find the records that are at the end of the internal table. To overcome this problem, we now device an approach wherein we maintain a cursor, which holds the value of the index from where the subsequent search should start.

- Example: Say our inner internal table has 100 records. The where condition in the inner loop actually searches the internal table starting from the first record, It does the same for all the records. The time it takes for finding the record will increase for the records that have matches at the end of the internal table. That means the search will take more time to find the 100th record than the time it takes to search for the 1st or 2nd record. Using a variable, we will now maintain a cursor which will avoid the search to start allover again, from the first record. Instead it will tell the control loop from a specified index. This makes the search faster.

- Note that the precondition is that the internal tables are both sorted by the respective key fields.

Sample Code and performance data comparison

To demonstrate the improvement in the performance, I have used two internal tables which have the data fetched from the database table MARA and MAKT. Please note that there is no business logic underneath the code. Also note that I have taken 100 records in each of the internal tables. The improvement in performance is far more pronounced when the volume of the data is increased.

```
data: it_mara type table of mara,
      wa_mara type mara,
      it_makt type table of makt,
      wa_makt type makt.

select *
 up to 100 rows
  from mara
  into table it_mara.

select *
 up to 100 rows
 from makt
 into table it_makt.

perform process.
perform process_using_cursor.

*&---------------------------------------------------------------------*
*&      Form  process
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
FORM process .
  DATA: t1 TYPE i,
  t2 TYPE i,
  tmin TYPE i.
  get run time field t1.
  sort it_mara by matnr.
  sort it_makt by matnr.
  loop at it_mara into wa_mara.
    loop at it_makt into wa_makt where matnr = wa_mara-matnr.
    endloop.
  endloop.
  get run time field t2.
  tmin = t2 - t1.
  write:/ 'Time(ms):' ,tmin.
ENDFORM.                    " process
*&---------------------------------------------------------------------*
*&      Form  process_using_cursor
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
FORM process_using_cursor .
  DATA: t1 TYPE i,
  t2 TYPE i,
  tmin TYPE i,
  tabix type sy-tabix.
  get run time field t1.
  sort it_mara by matnr.
  sort it_makt by matnr.
  loop at it_mara into wa_mara.
    read table it_makt transporting no fields with key matnr = wa_mara-matnr.
    if sy-subrc = 0.
      tabix = sy-tabix.
      loop at it_makt into wa_makt from tabix.
        if wa_makt-matnr <> wa_mara-matnr.
          exit.
        endif.
      endloop.
    endif.
  endloop.
  get run time field t2.
  tmin = t2 - t1.
  write:/ 'Time(ms):' ,tmin.

ENDFORM.                    " process_using_cursor
```

Performance Comparison

No of run	Records in IT_MARA	Records in IT_MAKT	Time required in milliseconds before using the cursor Variable	Time required in milliseconds after using the cursor Variable
1	100	100	2058	836
2	100	100	2011	823
3	100	100	2263	831
Average			2044	830

Conclusion

We can improve the performance of nested loops using a home grown cursor which holds the value of the internal table index of the last match.

Disclaimer and Liability Notice