SELECT SINGLE vs. SELECT UP TO 1 ROWS

June 11, 2016

Horst Keller

Some time ago I´ve written a small block Selecting One Row From a Database Table in order to explain the usage of SELECT SINGLE and SELECT UP TO 1 ROWS. Obviously, there are still discussions about this. OK, let´s derive some rules based on my previous blog:

TASK: You want to select one row with a fully specified key into a tabular result

You need neither SINGLE nor UP TO 1 ROWS:

```
SELECT …
       FROM dbtab
       WHERE full_key
       INTO TABLE itab.
```

The result is tabular containing one line. It is written to an internal table.

TASK: You want to select one row with a fully specified key into a structure

You use SINGLE:

```
SELECT SINGLE …
       FROM dbtab
       WHERE full_key
       INTO wa.
```

The result is tabular containing one line. You use SINGLE to prevent a SELECT loop. The line is copied directly to wa.

TASK: You want to select one row with a partly specified key

You use UP TO 1 ROWS:

```
SELECT …
       FROM dbtab
       WHERE part_key
       ORDER BY …
       INTO TABLE itab
       UP TO 1 ROWS.
```

or

```
SELECT …
       FROM dbtab
       WHERE part_key
       ORDER BY …
       INTO wa
       UP TO 1 ROWS.
  …
ENDSELECT.
```

The result is tabular containing one line for which the usage of ORDER BY is highly recommended. Usage of SINGLE is not appropriate here, because the resulting line is undefined. The line can be written to an internal table or a workarea.

TASK: You want to check the existence of a row

You use SINGLE:

```
SELECT SINGLE col
       FROM dbtab
       WHERE any_key
       INTO (field)
       ##warn_ok.
IF sy-subrc = 0.
   …
ENDIF.
```

The result is written to a single data object. There is no need for a tabular evaluation of the result and the usage of ORDER BY. From 7.40, SP05 on, you can even specify a literal ´X´ or a constant for col in order to prevent any data transport from DB to ABAP:

```
SELECT SINGLE 'X'
       FROM dbtab
       WHERE any_key
       INTO (field)
       ##warn_ok.
IF sy-subrc = 0.
   …
ENDIF.
```

If you or your Q-Manager don´t like the pragma `##warn_ok`**, you can also use:**

```
SELECT 'X'
       FROM dbtab
       WHERE any_key
       INTO (field)
       UP TO 1 ROWS.
ENDSELECT.
IF sy-subrc = 0.
   …
ENDIF.
```

Practically there is no difference in performance.
The paper is open for discussion.