Source Type : SQLRPGLE

1. Selecting one row.
2. Selecting all fields

Exec sql

Select \* into :fileextds from file

1. Selecting specific fields.

Select fields

Exec sql

Select fld1 , fld2 into :var1 , :var2 from file

1. To check if record exists :

Eval wempno = 0

select empno into :wempno from emppf where rating = ‘OUTSTANDING’ and empno = :imp\_Empno

If wempno = imp\_empno

Return true

endif

Checking existence of record USING SOME string CONSTANT :

Eval wRecExsit = ‘ ‘

Exec sql

select ‘Y’ into :wRecexist from emppf where rating = ‘OUTSTANDING’ and empno = :imp\_Emp

--------------sequential processing after fetching multiple records----------------------------

Cursors will be used for sequential processing

1. Declare cursor for sql query

Declare cursor for

Select \* from emppf where dept = ‘PAYROLL’ and rating = ‘OUTSTANDING’ and

Salary > 10000

Open c1

Fetch into :Ds

If sqlcode < 0

Dsply ‘error’

Endif

Dow sqlcode <> 100

Exsr Processing

Fetch next record into :ds

Enddo

DYNAMIC SQL

Consider below example where we need to insert data into empmst from different files based on application.

We can have different insert queries with different insert queries in if condition .

If applic = ‘FINANCE’

INSERT INTO EMPMST

SELECT \* FROM FINAPP

ENDIF

IF APPLICATION = ‘HR’

INSERT INTO EMPMST

SELECT \* FROM HRAPPL

ENDIF

**or we can construct a dynamic query.**

Steps

1. Construct query in string.

1. Prepare statement using prepare clause.
2. Execute statement using execute clause

EVAL FILEE = ‘HRAPPL’

Or

1. Construct query
2. Use execute immediate :variable containing query.

STRING = ‘INSERT INTO EMPMST ‘ +

‘SELECT \* FROM ‘ + %TRIM(Filename)

Prepare statement :

Exec sql prepare sqlstm1 from :string

Exec sql execute sqlstm1

OR

EXEC SQL IMMEDIATE :STRING

---------------------------------------------------------------------------------------------

**Bulk insert and update using exists**

Insert from source file where record doesn’t exists , update where record exists.

Update employee name for EMPPF records existing in EMPHRPF.

update    EMPPF a  set  
 empname = (  
  select b.empname from emphrpf b  where b.empno = a.EMPNO)  
 WHERE EXISTS  
 (select 1 from empHRPF b where a.empno =  b.empno)

Insert records into EMPPF from EMPHRPF which are not existing.

insert  into  EMPPF  
  select \* from emphrpf a where  not exists  
 (select 1 from emppf b where a.empno =  b.empno)

**Dynamic cursor**

1. **Construct a string** Example SQLSTM = ‘SELECT \* FROM ‘ + %TRIM(FILE) + ‘
2. **Prepare a sql Statement.**

Exec sql Prepare s1 from :SQLSTM

1. **Now declare cursor for that statement**

Declare cursor c1 for S1

**Dynamic query using order by:**

In a subfile screen if a user presses enter at any column subfile should be sorted based on that.

We will use rtncsrloc keyword , capture field name where enter is pressed and prepare dynamic query accordingly.

(RTNCSRLOC

Returns position of cursor to application program.

RTNCSRLOC (\*RECNAME &CSR\_REC &CSR\_FLD &CSR\_POSN)

Or

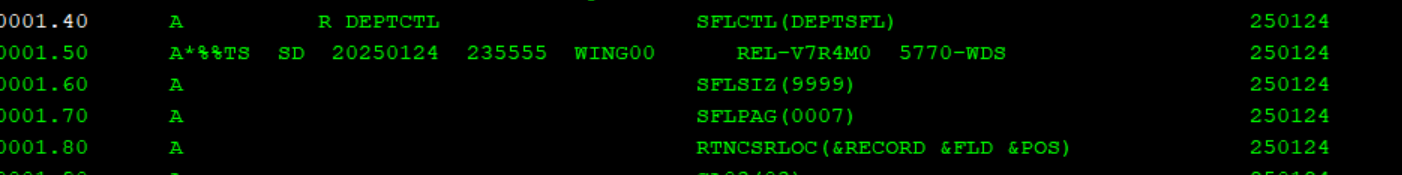
Rntcsrloc(\*MOUSE/\*window &row &col)

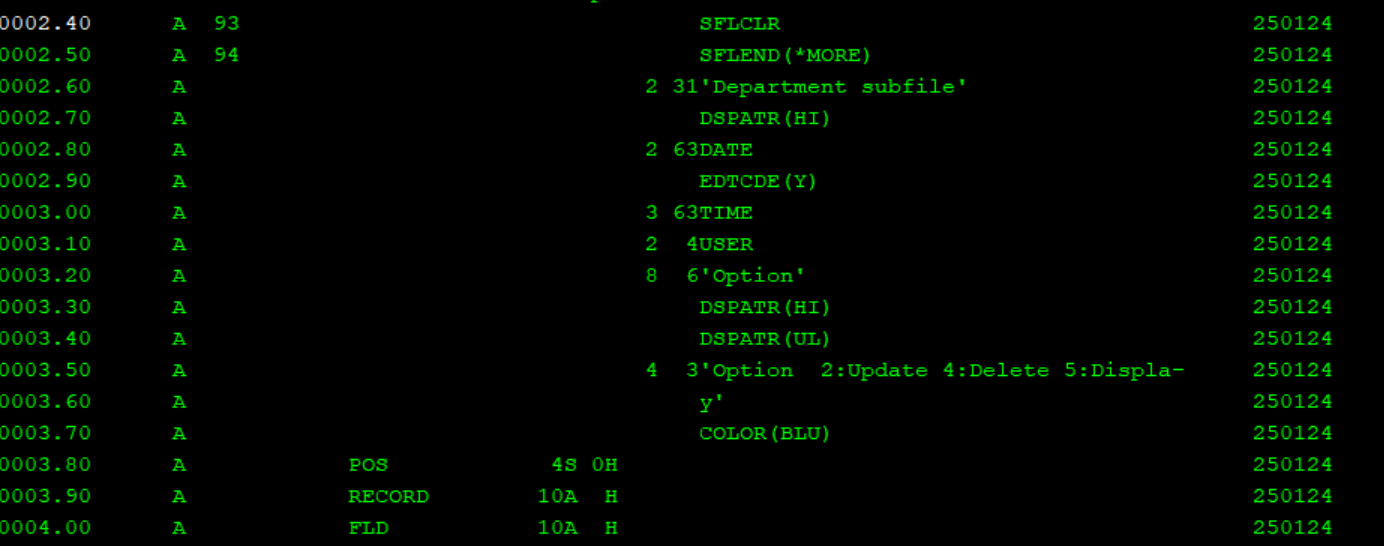
Sample declaration :

RTNCSRLOC(&RECORD &FLD &POS)   :Record level keyword to get record name , fld name and position where cursor was placed

This has to be at record level .

And fields RECORD , FLD will be defined as character in record format and POS as 4S 0 (Zoned)





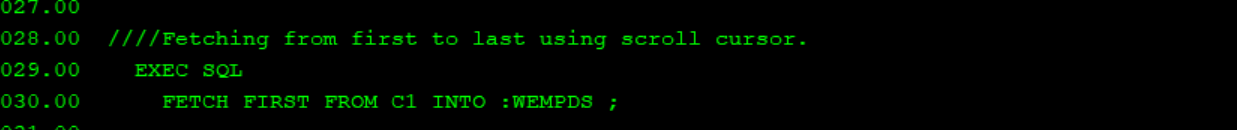
**Scrollable cursors:**

Scrollable cursor is a cursor that can be moved in both directions Forward/Backward.

When we open cursor is placed at first row, we can specify positions based on different orientation of fetch statement.

Before: Before first row

First: On first



Last: On Last



After: After last row

Absolute: On absolute row from first row forward or last row backward

Example

5th row Forward

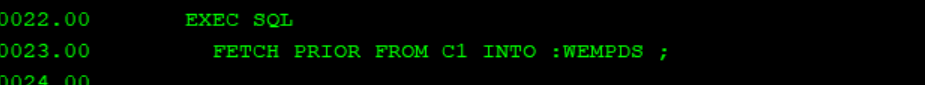
EXEC SQL FETCH ABSOLUTE +5 C1 INTO :empNO, :empNAME, :city;

EXEC SQL FETCH ABSOLUTE -5 C1 INTO :empNO, :empNAME, :city;

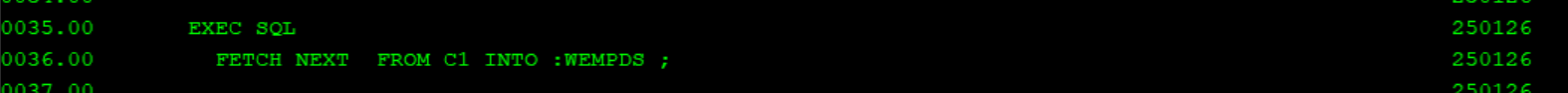
Relative : On row forward or backward that is relative no of rows from current row.

Current row : On current row

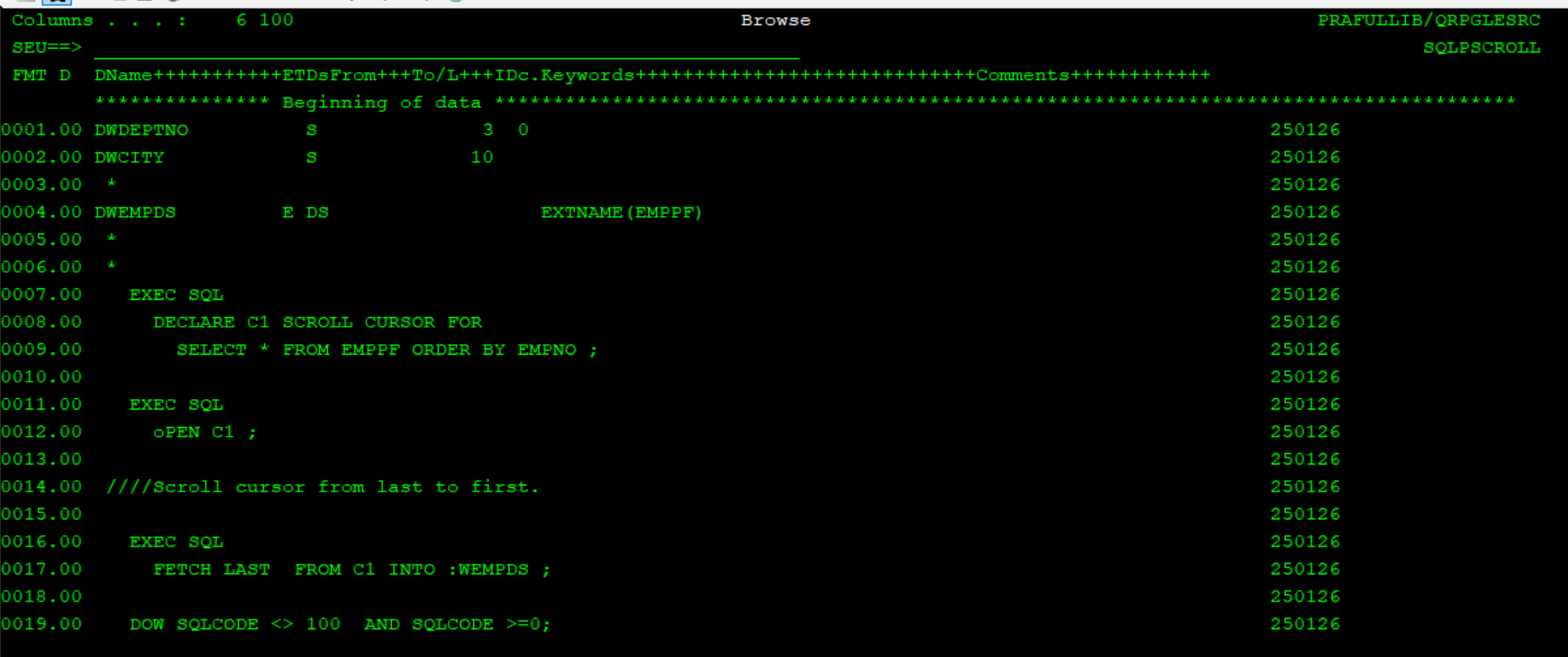
Prior : prior to current row

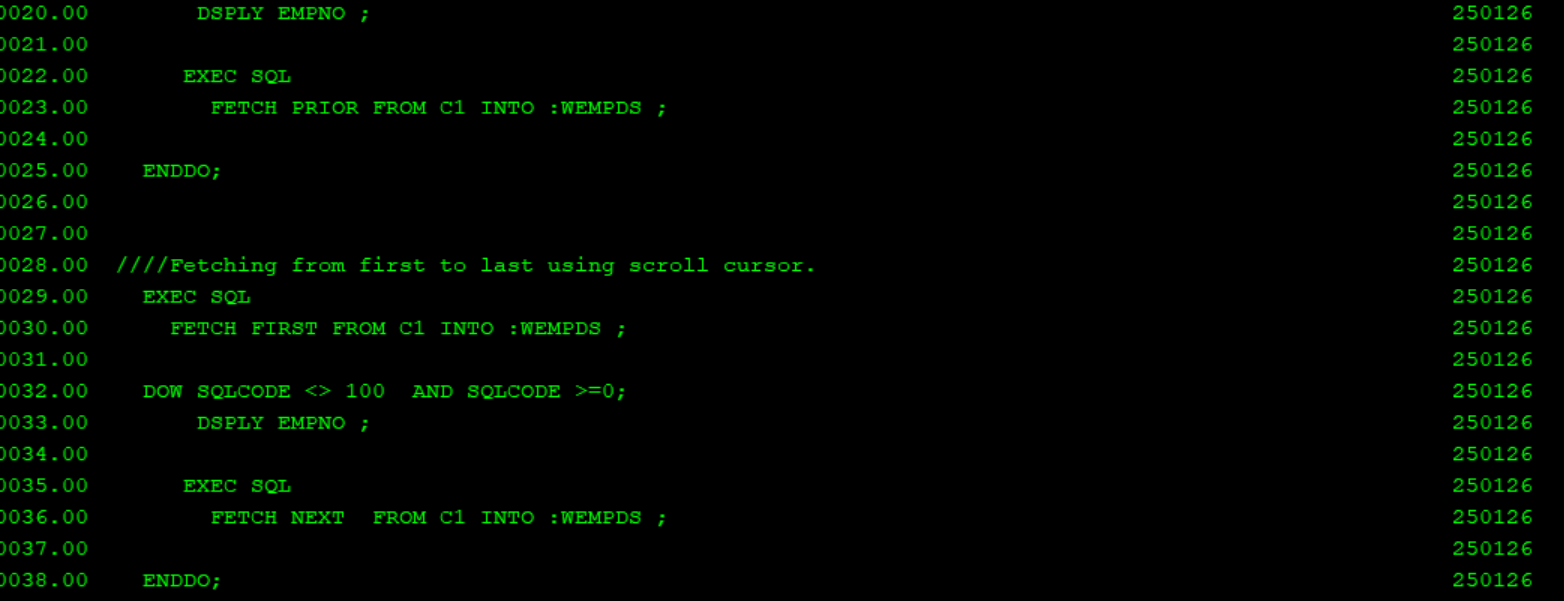


Next : Fetch next row By default simple fetch implies fetch next.



Example program to fetch from start to end and end to start using scroll cursors.

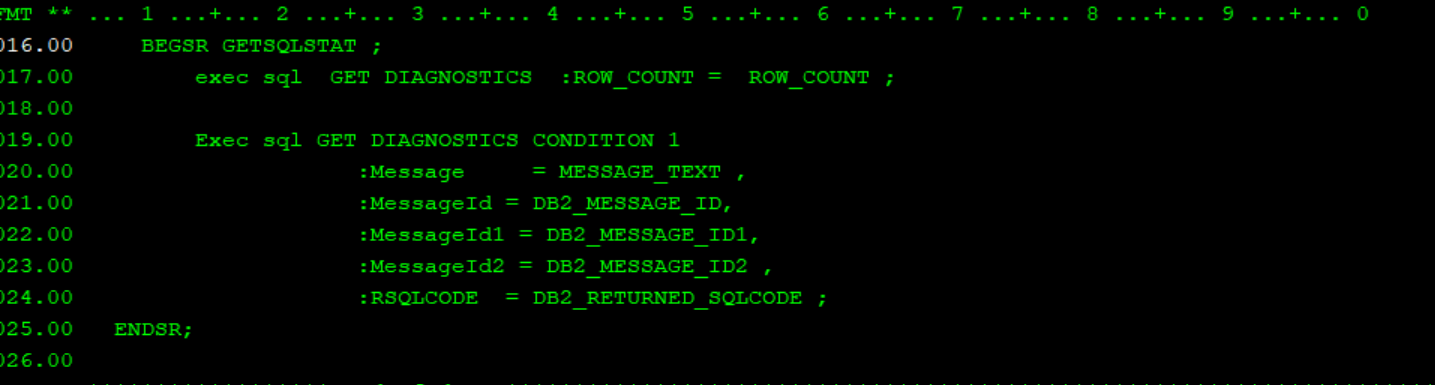




Get diagnostics :

|  |  |
| --- | --- |
| **Keyword** | **Description** |
| DB2\_MESSAGE\_ID | Message id for the error in MESSAGE\_TEXT (below) |
| DB2\_MESSAGE\_ID1 | Underlying escape CPF message that originally caused this error |
| DB2\_MESSAGE\_ID2 | Underlying diagnostic CPD message that originally caused this error |
| DB2\_RETURNED\_SQLCODE | SQL code (SQLCOD) |
| MESSAGE\_LENGTH | Length of the message in MESSAGE\_TEXT |
| MESSAGE\_TEXT | Message text for the error |
| RETURNED\_SQLSTATE | SQL state (SQLSTATE) |

GET DIAGNOSTICS example



Referential constraint / commitment control operations in SQLRPGLE needs file to be journalled . Below are journalling steps.

1. Create journal receiver (Object type \*JRNRCV)

Crtjrnrcv command to create journal receiver.

CRTJRNRCV JRNRCV(TESTRCV1)

1. Create journal (\*JRN)

Crtjrn command to create journal and use journal receiver created in first step.

CRTJRN JRN(TESTJRN1) JRNRCV(TESTRCV1)

1. Start journalling physical file or tables.

STRJRNPF FILE(EMPTABLE) JRN(TESTJRN) IMAGES(\*BOTH) OMTJRNE(\*OPNCLO)

Checking data from journals : DSPJRN command and we can take output into outfile.

Journal codes :

PT/PX : Insert

UB : Before update

UP : Update

DL : DELETE