The SQL Procedure Language (SQL PL)

David Simpson
dsimpson@themisinc.com

Themis inc.

Coding a SQL PL Procedure

- An SQL procedure consists of:
  - CREATE PROCEDURE header
  - BEGIN statement
  - Body (SQL procedural statements and / or DB2 SQL statements)
  - END statement
- Comments within an SQL procedure:
  - -- for a single line comment
  - /* to start */ end multiple-lines comments
- Statements end with semicolon

Themis inc.

A SQL PL Header

```
CREATE PROCEDURE SPA80  (IN  P_DNO  CHAR(3)
                        ,OUT P_CNT  SMALLINT
                        ,OUT P_SUMSAL DECIMAL(11,2)
                        ,OUT P_RETCODE INTEGER )
VERSION V1
```

Parameters

- Procedure Name
  - 128 byte max length
  - Unique within Schema / Collection
  - Schema / Collection ID will be supplied when create is deployed
- Parameters
  - 128 byte max length
  - Can be IN bound, OUT bound or INOUT (both directions)
  - Used to pass data between procedure and caller
  - Cannot specify a default value
- Versioning
  - 64 EBCDIC bytes max length
  - If using versioning do not use the default V1 naming convention

Themis inc.

A SQL PL Body

The body consists of 5 parts:

- SQL variable declarations
- Condition names
- Cursors
- Condition Handlers
- Code…

Themis inc.

A SQL PL Body

```
…
…
P1: BEGIN
      SET P_CNT = 0;
      SET P_SUMSAL = 0;
      SET P_RETCODE = 0;

      SELECT COUNT(*), SUM(SALARY)
        INTO P_CNT, P_SUMSAL
        FROM EMP
       WHERE DEPTNO = P_DNO;
END P1
```

Assign Values

Clear your outbound parms

Themis inc.

SQL Procedure Statements

- DECLARE Statement
- Assignment Statement
- **CALL**, **GOTO**, **LEAVE**, **RETURN**
- **IF, CASE**, **WHILE**, **LOOP**, **REPEAT**, **ITERATE, FOR**
- Compound statement
- **GET DIAGNOSTICS** statement
- **SIGNAL, RESIGNAL** statements
- SQL Statements

- Note: Sucessful Execution of any SQL statement will set
  SQLCODE variable value to 0 and
  SQLSTATE variable value to '00000'.

Themis inc.

Declaring SQL Variables

Syntax: DECLARE *SQL-variable-name data-type*
[DEFAULT *constant*] ;

- Same data types and lengths as DB2 table columns
- Parameter and variable names are not case sensitive
- SQL reserved word cannot be used as parameter name or variable name
- Variable name declarations must be first prior to other statements in the procedure body.
- Coding variable names and parameter names in a DB2 SQL statement do not require colon to precede them
- A declare statement ends with a semicolon

Themis inc.

Assignment Statement

Assigns a value to an output parameter or to an SQL variable

```
SET var1 = 10;
SET var2 = ( SELECT count(*) FROM EMP );
SET var3  = NULL;
```

- Assignment statements conform to the SQL assignment rules
- The data type of the target and source must be compatible
- An assignment statement must end with a semicolon.

V10: `SET var1 = 10, var3 = NULL;`

Themis inc.

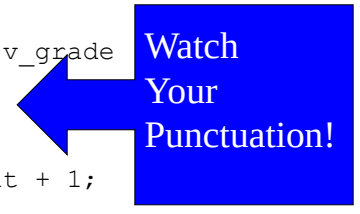
SQL Variable Example

```
…
P1: BEGIN
    DECLARE SQLCODE INTEGER DEFAULT 0;
    DECLARE V_LAST_PAID_DATE DATE;
    DECLARE V1 CHAR(25) DEFAULT 'NOT PAID';
    DECLARE V2 INTEGER;
    DECLARE V3 DECIMAL(9,2);
    DECLARE V4 DECIMAL (9,2) DEFAULT 0;

    SET V2 = 1000;
    SET V3 = 500.00;
.
.
.
END P1
```

Themis inc.

IF Statement

```
DECLARE v_grade CHAR(1);
DECLARE v_a_count INTEGER;
DECLARE v_b_count INTEGER;
DECLARE v_invalid_count INTEGER;
...
UPDATE STUDENT SET GRADE = v_grade
WHERE STUDENT_NO = var1;

IF v_grade = 'A' THEN
   set v_a_count = v_a_count + 1;
ELSEIF v_grade = 'B' THEN
   set v_b_count = v_b_count + 1;
ELSEIF . . . THEN  . . .
ELSE
   Set v_invalid_count = v_invalid_count + 1;
END IF;
```



Themis inc.

IF Statement

```
IF v_grade = 'A' THEN
   set v_a_count = v_a_count + 1;
   set v_counter = v_counter + 1;
ELSEIF v_grade = 'B' THEN
   set v_b_count = v_b_count + 1;
   set v_counter = v_counter + 1;
ELSEIF . . . THEN  . . .
ELSE
   Set v_invalid_count = v_invalid_count + 1;
END IF;
```

Themis inc.

CASE Statement

1. Simple CASE: Testing for value of variable

```
CASE v_grade
  WHEN 'A' THEN set v_a_count = v_a_count + 1;
                set v_counter = v_counter + 1;
  WHEN 'B' THEN set b_count = b_count + 1;
                set v_counter = v_counter + 1;
  ELSE set v_invalid_count = v_invalid_count + 1 ;
END CASE;
```

2. Searched CASE: Testing for TRUE condition

```
CASE
  WHEN v_edlevel < 12 THEN
      set v_no_diploma = v_no_diploma + 1;
  WHEN v_edlevel > 11 and v_edlevel < 16 THEN
      set v_high_school = v_high_school + 1;
END CASE;
```

Themis inc.

LOOP, LEAVE & REPEAT

```
FETCH_LOOP: LOOP
      FETCH CURSOR1 INTO VAR1, VAR2, VAR3 ;
      IF SQLCODE = 100 THEN
            LEAVE FETCH_LOOP;
      END IF;
      <process values returned by cursor>
END LOOP;
```

```
REPEAT
      FETCH CURSOR1 INTO VAR1, VAR2, VAR3;
      IF SQLCODE = 100 THEN SET V_EOF = 'Y';
        ELSE
            <process values returned by cursor>
      END IF;
  UNTIL V_EOF = 'Y'
END REPEAT;
```

Themis inc.

WHILE Statement

```
DECLARE SQLCODE          INTEGER DEFAULT 0;
DECLARE V_EOF            CHAR(1) DEFAULT 'N';
…
WHILE (V_EOF = 'N') DO
   FETCH CURSOR1 INTO VAR1, VAR2, VAR3;
   IF SQLCODE = 100 THEN
      SET V_EOF = 'Y';
   ELSE
      <process a row> . . . ;
   END IF;
END WHILE;
```

Themis inc.

FOR Statement

```
FOR FOR_ROUTINE AS
  CURSOR1 CURSOR FOR
      SELECT EMPNO, FIRSTNME, LASTNAME, SALARY
        FROM EMP
       WHERE DEPTNO = p_deptno
       ORDER BY  SALARY DESC
      DO
        SET v_numrows = v_numrows + 1;
        SET v_salarytotal = v_salarytotal + SALARY;
END FOR;
```

Variables

Column from CURSOR1

Themis inc.

GOTO Statement

The GOTO transfers control to a labeled statement. The labeled statement and the GOTO statement must be in the same scope.

```
IF V_SERVICE < 10000 THEN
   GOTO EXIT_RTN ;
END IF ;
…;
…;
EXIT_RTN :
  BEGIN
   SET P_RETURN_CODE = V_SERVICE ;
  END;
```

Themis inc.

ITERATE Statement

The ITERATE statement causes the flow of control to return to the beginning of a labeled loop.

```
WHILE_ROUTINE :
   WHILE (MORE_RESULT = 0) DO
         FETCH CURSOR1 INTO VAR1, VAR2, VAR3;
         SET MORE_RESULT = SQLCODE;
         IF VAR3 < 0 THEN
               ITERATE WHILE_ROUTINE;
         END IF ;
         . . . ;
         . . . ;
   END WHILE;
```

Themis inc.

Compound Statement

label:
BEGIN { NOT ATOMIC or ATOMIC }

[SQL-variable-declaration ; ….]
[Declare Cursor statement ; ….]
[condition-declaration ; …….]
[return-code-declaration ; ……..]

[handler-declaration ; ……..]

SQL-procedure-statement; …

END { label }

Themis inc.

Compound Statement

```
P1:
BEGIN
    DECLARE SQLCODE INTEGER ;
    DECLARE C1 CURSOR WITH RETURN ……..;
    INSERT INTO AUDIT VALUES( PARMX, PARMY, PARMZ);
    IF (SQLCODE = -803) THEN
           …;
    ELSE
           …;
    END IF;
    INSERT …;
END P1;
```

Themis inc.

SQL Statements

Most DB2 SQL statements are supported :

- SELECT INTO
- DECLARE CURSOR / OPEN / FETCH / CLOSE
- INSERT
- UPDATE
- DELETE
- MERGE

Themis inc.

Sample Stored Procedure

```
CREATE PROCEDURE SPA80   (OUT P_CNT1     SMALLINT
                         ,OUT P_SUMSAL   DECIMAL(11,2)
                         ,OUT P_RETCODE  INTEGER
                         )
   VERSION V1               ASUTIME 500000
   ISOLATION LEVEL CS       VALIDATE BIND
   PACKAGE OWNER DBTHM80    QUALIFIER THEMIS1
   RESULT SETS 0            LANGUAGE SQL

P1: BEGIN
   DECLARE SQLCODE INTEGER DEFAULT 0;

   SELECT COUNT(*), SUM(SALARY)
     INTO P_CNT1, P_SUMSAL
     FROM EMP;

   SET P_RETCODE = SQLCODE;
END P1
```

Themis inc.

Returning Result Sets

```
CREATE PROCEDURE SPB80   (OUT P_RETCODE INTEGER)

   VERSION V1               ASUTIME 500000
   ISOLATION LEVEL CS       VALIDATE BIND
   PACKAGE OWNER DBTHM80    QUALIFIER THEMIS1
   RESULT SETS 1            LANGUAGE SQL
P1: BEGIN
   DECLARE SQLCODE INTEGER DEFAULT 0;

   DECLARE CURSOR1 CURSOR WITH RETURN FOR
       SELECT EMPNO, LASTNAME, MIDINIT, FIRSTNME,
              SALARY, DEPTNO
         FROM EMP
       ORDER BY DEPTNO, EMPNO
        FOR FETCH ONLY;

   OPEN CURSOR1;
   SET P_RETCODE = SQLCODE;
END P1
```

Themis inc.

Processing a Cursor

```
P1: BEGIN
   DECLARE SQLCODE INTEGER DEFAULT 0;
   DECLARE V_EOC   CHAR(1) DEFAULT 'N';
   DECLARE V_SAL   DECIMAL(9,2);
   DECLARE C1 CURSOR FOR
     SELECT SALARY FROM EMP
      WHERE DEPTNO = P_DEPTNO;

   OPEN C1;

   SET P_SUM = 0;
   REPEAT
      FETCH C1 INTO V_SAL;
      IF SQLCODE = 100 THEN
         SET V_EOC = 'Y';
      END IF;
      SET P_SUM = P_SUM + V_SAL;
    UNTIL V_EOC = 'Y'
   END REPEAT;

   CLOSE C1;
END P1
```

Themis inc.

Capturing SQLCODE & SQLSTATE

```
P1: BEGIN
     DECLARE SQLCODE  INTEGER DEFAULT 0;
     DECLARE SQLSTATE CHAR(5) DEFAULT '00000';

     ...
     FETCH CURSOR1 INTO V1, V2;
     IF SQLCODE = 100 THEN
          SET EOF = 'Y';
```

Themis inc.

Unhandled Errors

```
CREATE PROCEDURE SPERR (OUT P_RETCODE INTEGER)
     VERSION VERSION1
P1: BEGIN
     DECLARE SQLCODE     INTEGER DEFAULT 0;
     DECLARE V1          DATE;

     SET P_RETCODE = 0;
     SELECT DATE('2016-12-32')
       INTO V1
       FROM SYSIBM.SYSDUMMY1;

     SET P_RETCODE = SQLCODE;
END P1
```

Execution Stops Here!!

Themis inc.

Handlers

```
DECLARE { EXIT | CONTINUE } HANDLER FOR { SQLEXCEPTION | SQLWARNING | NOT FOUND | SQLSTATE 'value' }
```

Themis inc.

Exit Handler – Compound Statement

```
P1: BEGIN
      ...
      DECLARE EXIT HANDLER FOR SQLEXCEPTION
           BEGIN
             SET P_RETCODE = SQLCODE;
             GET DIAGNOSTICS CONDITION 1
                V_SQLMSG = MESSAGE_TEXT;
           END;
      ...
END P1
```

Variable from the SP

GET DIAGNOSTICS keyword

Handler code executes and procedure Exits

Themis inc.