**Writing a COBOL DB2 Program.**

Let us assume we are writing a cobol program to read EMPLOYEE

table and get the details of employee with the name XXXXXXX.

Let us go in step wise..

create the following table in db2 or assume it is there in db2 database.

EMPLYEE

EMPID EMPNAME DEPARTMENT SALARY DESIGNATION

1000 XXXXXXX XX 10000 SE

1001 YYYYYYY YY 9000 SE

1002 ZZZZZZZ ZZ 20000 MA

**STEP 1.** We need to declare the table structure in the

WORKING-STORAGE SECTION or LINKAGE SECTION.

EXEC SQL

DECLARE DSNXXX.EMPLOYEE

( EMPID CHAR(10) NOT NULL,

EMPNAME CHAR(30) NOT NULL,

DEPARTMENT CHAR(2) NOT NULL,

SALARY DECIMAL(10,2) NOT NULL,

DESIGNATION CHAR(4) NOT NULL )

END-EXEC.

we can use DB2 tool called DCLGEN to generate this declaration

for us and can include that copy book here.

if you create a copybook using DCLGEN. Use following sntax to include

EXEC SQL

INCLUDE < copybookname >

END-EXEC.

**STEP 2.** Declare host variables in WORKING-STORAGE SECTION.

HOST VARIABLES - A host variable is a data item declared in cobol to use

it in embedded SQL.

For EMPLOYEE table, host variable declaration is look like as follows...

01 EMPOYEE-RECORD.

05 HV-EMPID PIC X(10).

05 HV-EMPNAME PIC X(30).

05 HV-DEPARTMENT PIC X(2).

05 HV-SALARY PIC S9(8)V99 COMP-3.

05 HV-DESIGNATION PIC CHAR(4).

If you use db2 tool DCLGEN, it will automatically creates this structure also

along with table declaration specified in step1.

**STEP 3.** Include SQLCA as follows in WORKING-STORAGE SECTION.

EXEC SQL

INCLUDE SQLCA

END-EXEC.

What is SQLCA?

SQLCA - SQL communication area.

When a SQL statement executes, DB2 places a value in SQLCODE AND

SQLSTATE host variables or any other fields of SQLCA. based on

the values in these variables we can know whether sql ran

sucessfully or not.

SQLCA contains a declartion of fields like SQLCODE,SQLSTATE and

SQLERRD etc....

**STEP 4.** Add a sql statement in procdure division to get the details of employee

with the name XXXXXXX.

DISPLAY ' PROGRAM STARTED .... '

.........

EXEC SQL

SELECT SALARY

INTO :HV-SALARY

FROM EMPLOYEE

WHERE EMPNAME = 'XXXXXXX'

END-EXEC.

IF SQLCODE = 0

DISPLAY ' SQL EXECUTED SUCESSFULLY '

DISPLAY ' EMPLOYEE SALARY IS ' HV-SALARY

ELSE

DISPLAY ' SQL FAILED '

DISPLAY ' SQL CODE ' SQLCODE

END-IF.

....

....

DISPLAY ' PROGRAM ENDED'.

Here SQLCODE = 0 means, sql ran sucessfully without any issues. Hence

we are displaying the HV-SALARY into the spool.

If SQLCODE NOT = 0 , there is issue in executing the sql statement.

Now we have compeleted coding a cobol-db2 program. our next step is to

compile the program.

**SAMPLE COMPILE JCL**

[Click here](http://www.mainframegurukul.com/tutorials/database/db2_tutorials/sample-db2-cobol-compile-jcl.html) to see the compile JCL. Use this compile jcl to compile the program.

**SAMPLE RUN JCL**

[Click here](http://www.mainframegurukul.com/tutorials/database/db2_tutorials/sample-db2-cobol-run-jcl.html) to see the sample run jcl. Use this run jcl to run the program.

PROGRAM-2

**000100 IDENTIFICATION DIVISION.**

**000200 PROGRAM-ID. 'DB2CRSR1'.**

**000300\* SAMPLE COBOL PROGRAM**

**000400\* USES A CURSOR TO READ AND DISPLAY EVERY ROW IN APPLICANT**

**000500 ENVIRONMENT DIVISION.**

**000600 INPUT-OUTPUT SECTION.**

**000700 FILE-CONTROL.**

**000800 DATA DIVISION.**

**000900 FILE SECTION.**

**001000 WORKING-STORAGE SECTION.**

**001100\* REGULAR WORKING STORAGE THINGS GO HERE AS ALWAYS**

**001200 01 DISPLAY-SQLCODE PIC Z(8)9-.**

**001300**

**001400 01 SWITCHES.**

**001500 05 CURSOR-AT-END PIC X VALUE 'N'.**

**001600**

**001700 01 ERR-MESS-DATA.**

**001800 05 ERR-MESS-LEN PIC S9(4) BINARY VALUE +960.**

**001900 05 ERR-MESS-TEXT PIC X(120) OCCURS 8 TIMES**

**002000 INDEXED BY ERR-INDEX.**

**002100 01 ERR-TEXT-LEN PIC S9(9) BINARY VALUE +120.**

**002200\***

**002600 EXEC SQL**

**002700 INCLUDE SQLCA**

**002800 END-EXEC.**

**002900 EXEC SQL**

**003000 INCLUDE APPLICAN**

**003100 END-EXEC.**

**003200**

**003300 EXEC SQL**

**003400 DECLARE APPLICAN\_CUR CURSOR FOR**

**003500 SELECT TEMPID, NAME, ADDRESS, EDLEVEL, COMMENTS**

**003600 FROM APPLICANT**

**003700 END-EXEC.**

**8003800**

**003900 PROCEDURE DIVISION.**

**004000 PERFORM INIT**

**004100 PERFORM GET-ALL-ROWS UNTIL CURSOR-AT-END = 'Y'**

**004200 PERFORM TERM**

**004300 GOBACK.**

**004400**

**004500 INIT.**

**004600 DISPLAY 'STARTING PROGRAM DB2CRSR1'**

**004700 DISPLAY 'GOING TO OPEN CURSOR'.**

**004800 EXEC SQL OPEN APPLICAN\_CUR END-EXEC.**

**004900 EVALUATE TRUE**

**005000 WHEN SQLCODE = 0**

**005100 DISPLAY 'SUCCESSFUL OPEN '**

**005200 WHEN SQLCODE > 0 OR SQLWARN0 = 'W'**

**005300 PERFORM WARNING-PARAGRAPH**

**005400 WHEN SQLCODE < 0 GO TO ERROR-EXIT**

**005500 END-EVALUATE**

**005600 PERFORM FETCH-PAR.**

**005700**

**005800 TERM.**

**005900 EXEC SQL CLOSE APPLICAN\_CUR END-EXEC.**

**006000**

**006100 GET-ALL-ROWS.**

**006200 DISPLAY 'ROW FROM TABLE:'**

**006300 DISPLAY TEMPID, NAME-X, ADDRESS-X,**

**006400 EDLEVEL, COMMENTS**

**006500 PERFORM FETCH-PAR.**

**006600**

**006700 FETCH-PAR.**

**006800 DISPLAY 'GOING TO FETCH'**

**006900 MOVE SPACES TO NAME-X, ADDRESS-X, COMMENTS**

**007000 MOVE ZEROS TO EDLEVEL**

**007100 EXEC SQL**

**007200 FETCH APPLICAN\_CUR**

**007300 INTO :TEMPID, :NAME-X, :ADDRESS-X, :EDLEVEL, :COMMENTS**

**007400 END-EXEC**

**007500**

**007600 EVALUATE TRUE**

**007700 WHEN SQLCODE = 0**

**007800\* DISPLAY 'SUCCESSFUL FETCH '**

**007900 CONTINUE**

**008000 WHEN SQLCODE = +100**

**008100 DISPLAY 'CURSOR AT END'**

**008200 MOVE 'Y' TO CURSOR-AT-END**

**008300 WHEN SQLCODE > 0 OR SQLWARN0 = 'W'**

**008400 PERFORM WARNING-PARAGRAPH**

**008500 WHEN SQLCODE < 0 GO TO ERROR-EXIT**

**008600 END-EVALUATE.**

**008700**

**008800 ERROR-EXIT.**

**008900\*\*\*\***

**009000\*\*UPDATES, MAINFRAME INFO AT:**

**009100\*\*HTTP://theamericanprogrammer.com/db2-sql/index.html**

**009200 MOVE SQLCODE TO DISPLAY-SQLCODE.**

**009300 DISPLAY 'SQLCODE FOLLOWS' DISPLAY-SQLCODE**

**EVALUATE TRUE**

**WHEN SQLCODE = 0**

**\* DISPLAY 'SUCCESSFUL EXECUTION'**

**CONTINUE**

**WHEN SQLCODE = +100**

**\* DISPLAY 'NOT FOUND'**

**CONTINUE**

**WHEN SQLCODE = -180**

**DISPLAY 'BAD DATA IN DATE/TIME/TIMESTAMP'**

**WHEN SQLCODE = -181**

**DISPLAY 'BAD DATA IN DATE/TIME/TIMESTAMP'**

**WHEN SQLCODE = -305**

**DISPLAY 'NO NULL INDICATOR'**

**WHEN SQLCODE = -311**

**DISPLAY 'LENGTH OF VARIABLE WRONG'**

**WHEN SQLCODE = -501**

**DISPLAY 'CURSOR NOT OPEN ON FETCH'**

**WHEN SQLCODE = -530**

**DISPLAY 'RI INS/UPD'**

**WHEN SQLCODE = -532**

**DISPLAY 'RI DELETE'**

**WHEN SQLCODE = -803**

**DISPLAY 'DUP ROW '**

**WHEN SQLCODE = -805**

**DISPLAY 'DBRM NOT FOUND IN PLAN'**

**WHEN SQLCODE = -811**

**DISPLAY 'MORE THAN 1 ROW ON SELECT INTO '**

**WHEN SQLCODE = -818**

**DISPLAY 'TIMESTAMP MISMATCH, LOAD MOD/PLAN'**

**WHEN SQLCODE = -904**

**DISPLAY 'UNAVAIL RESOURCE'**

**WHEN SQLCODE = -911**

**DISPLAY 'DEADLOCK/TIMEOUT, ROLLBACK DONE'**

**WHEN SQLCODE = -913**

**DISPLAY 'DEADLOCK/TIMEOUT VICTIM, NO ROLLBACK'**

**WHEN OTHER**

**DISPLAY 'SEVERE SQL ERROR'**

**END-EVALUATE**

**011800**

**011900 CALL 'DSNTIAR' USING SQLCA ERR-MESS-DATA ERR-TEXT-LEN**

**012000**

**012100 PERFORM ERROR-EXIT-PRINT-ERROR**

**012200 VARYING ERR-INDEX FROM 1 BY 1 UNTIL ERR-INDEX > 8**

**012300**

**012400\* IN REAL LIFE YOU WOULD CALL AN ABORT ROUTINE**

**012500 EXEC SQL**

**012600 ROLLBACK**

**012700 END-EXEC**

**012800**

**012900 DISPLAY 'ROLLBACK DONE'**

**013000**

**013100 GOBACK.**

**013200**

**013300 ERROR-EXIT-PRINT-ERROR.**

**013400 IF ERR-MESS-TEXT(ERR-INDEX) NOT = SPACES**

**013500 THEN DISPLAY ERR-MESS-TEXT(ERR-INDEX).**

**013600**

**013700 WARNING-PARAGRAPH.**

**013800 IF SQLWARN1 = 'W'**

**013900 THEN DISPLAY 'CHARACTER DATA TRUNCATED'**

**014000 'SQLWARN1 = W'**

**014100 END-IF**

**014200**

**014300 IF SQLWARN2 = 'W'**

**014400 THEN DISPLAY 'A FUNCTION HANDLED A NULL BY IGNORING IT'**

**014500 'SQLWARN2 = W'**

**014600 END-IF**

**014700**

**014800 IF SQLWARN3 = 'W'**

**014900 THEN DISPLAY 'THE NUMBER OF HOST VARIABLES IS LESS '**

**015000 'THAN THE NUMBER OF COLUMNS SELECTED '**

**015100 'SQLWARN3 = W'**

**015200 END-IF**

**015300**

**015400 IF SQLWARN4 = 'W'**

**015500 THEN DISPLAY 'A DYNAMIC SQL UPDATE/DELETE DOES NOT '**

**015600 'CONTAIN A WHERE CLAUSE '**

**015700 'SQLWARN4 = W'**

**015800 END-IF**

**015900**

**016000 IF SQLWARN5 = 'W'**

**016100 THEN DISPLAY 'DYNAMIC SQL DOES NOT CONTAIN VALID SQL'**

**016200 'SQLWARN5 = W'**

**016300 END-IF**

**016400**

**016500 IF SQLWARN6 = 'W'**

**016600 THEN DISPLAY 'DATE/TIMESTAMP ARITHMETIC '**

**016700 'PRODUCES AN INVALID DATE EX: NOV 31'**

**016800 'IT IS CHANGED TO LAST DAY OF MONTH EX: NOV 30'**

**016900 'SQLWARN6 = W'**

**017000 END-IF**

**017100**

**017200 IF SQLWARN7 = 'W'**

**017300 THEN DISPLAY 'CHARACTER DATA TRUNCATED '**

**017400 'POSSIBLE LOW ORDER TRUNCATION '**

**017500 'SQLWARN7 = W'**

**017600 END-IF**

**017700**

**017800 IF SQLWARN8 = 'W'**

**017900 THEN DISPLAY 'A CHARACTER COULD NOT BE CONVERTED '**

**018000 'SQLWARN8 = W'**

**018100 END-IF**

**018200**

**018300 IF SQLWARN9 = 'W'**

**018400 THEN DISPLAY 'ARITHMETIC DATA ERRORS FOUND'**

**018500 'WHILE DOING A COUNT(DISTINCT) '**

**018600 'SQLWARN9 = W'**

**018700 END-IF**

**018800**

**018900 IF SQLWARNA = 'W'**

**019000 THEN DISPLAY 'CHARACTER CONVERSION ERROR'**

**019100 'IN SQLCA OR SQLDA. THE CODE WILL BE INVALID'**

**019200 'SQLWARNA = W'**

**019300 END-IF.**