Guide for using IBM DB2 Community Edition for Windows X64 with GNUCOBOL

Chuck Haatvedt email : chuck.haatvedt@gmail.com
skype : chuck.haatvedt

This guide is for use with Windows, there is a Linux version available as well however I've not used it but documentation is available on the IBM website below. Also note that it requires a 64 bit version of Windows 10 or higher.

Db2 Community Edition helps you develop and deploy apps using the latest features of Db2 11.5. This free edition provides up to 16GB of memory, 4 cores, and no limit on database size.

Before starting I should mention that I am not a DBA although I did work on DB2 performance issues for over 15 years. I have build tables and indexes, used utilities to unload / load / reorganize tables. I also have some experience analyzing access paths using IBM Data Studio which I think is available for the Community version as well although I have not used it there.

First I'll include links to the download site as well as a couple of other useful sites and an open source Data Base Manager which works with a number of databases including DB2.

Note that you will be prompted to create an IBM userid and password to download the Community Edition of DB2.

IBM Db2® Database note that this is for 11.5.8 even though the link refers to 11.5.7

IBM Db2 Version 11.5 Knowledge Center This is the link to the knowledge center for DB2 11.5 documentation. I personally have found this to be very useful.

Dbeaver Database Manager This is a free Database Manager which supports many databases. Free multi-platform database tool for developers, database administrators, analysts and all people who need to work with databases. Supports all popular databases: MySQL, PostgreSQL, SQLite, Oracle, DB2, SQL Server, Sybase, MS Access, Teradata, Firebird, Apache Hive, Phoenix, Presto, etc.

Nirsoft - DLL Export Viewer This is a free tool to view the functions exported by a DLL on Windows. I found it useful when diagnosing linker issues.

SPFLITE - an ISPF like editor For people coming from a mainframe environment, they may prefer to use an editor similar to what they are familiar with.

WARNING !! I would strongly advise that when installing DB2 that you do NOT use the default installation directory but instead use an installation directory WITHOUT embedded spaces in the full directory path as it will make using it in the GNUCOBOL compile command line options much simpler. I found that dealing with embedded spaces in the various command line options for the compiler to be a headache best to be avoided.

Also I personally found it easier to use my Windows userid / password when prompted for this during the installation.

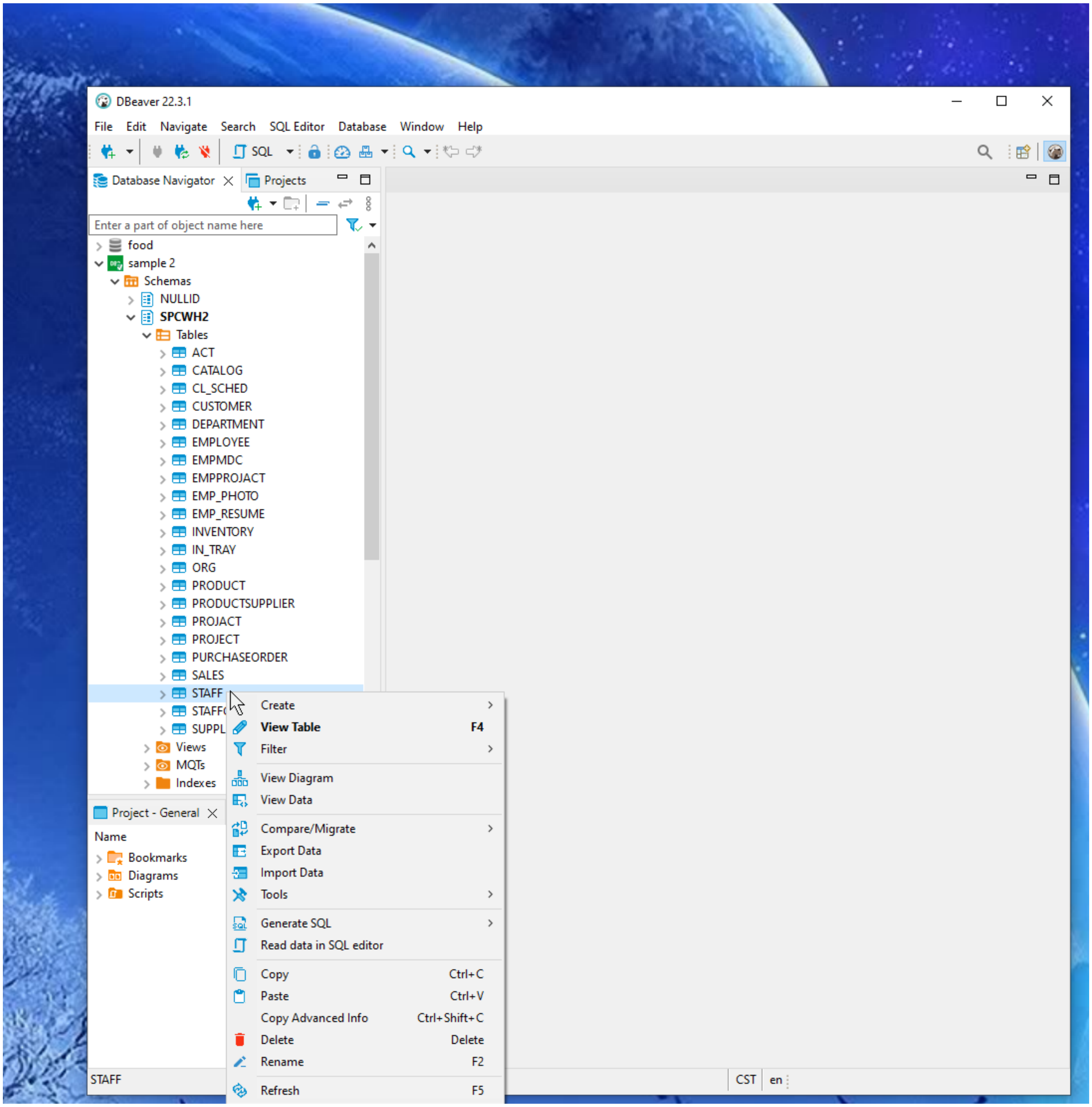
Note that the installation will include a SAMPLE database as well as a collection of sample programs to use for testing and educational purposes. I found them to be a useful starting point to explore GNUCOBOL / DB2 coding.

I prefer to not install any software on my C: drive if possible so when looking at the *.cmd files, my installation was installed to D:\DB2 and my database path was installed to F:\DB2

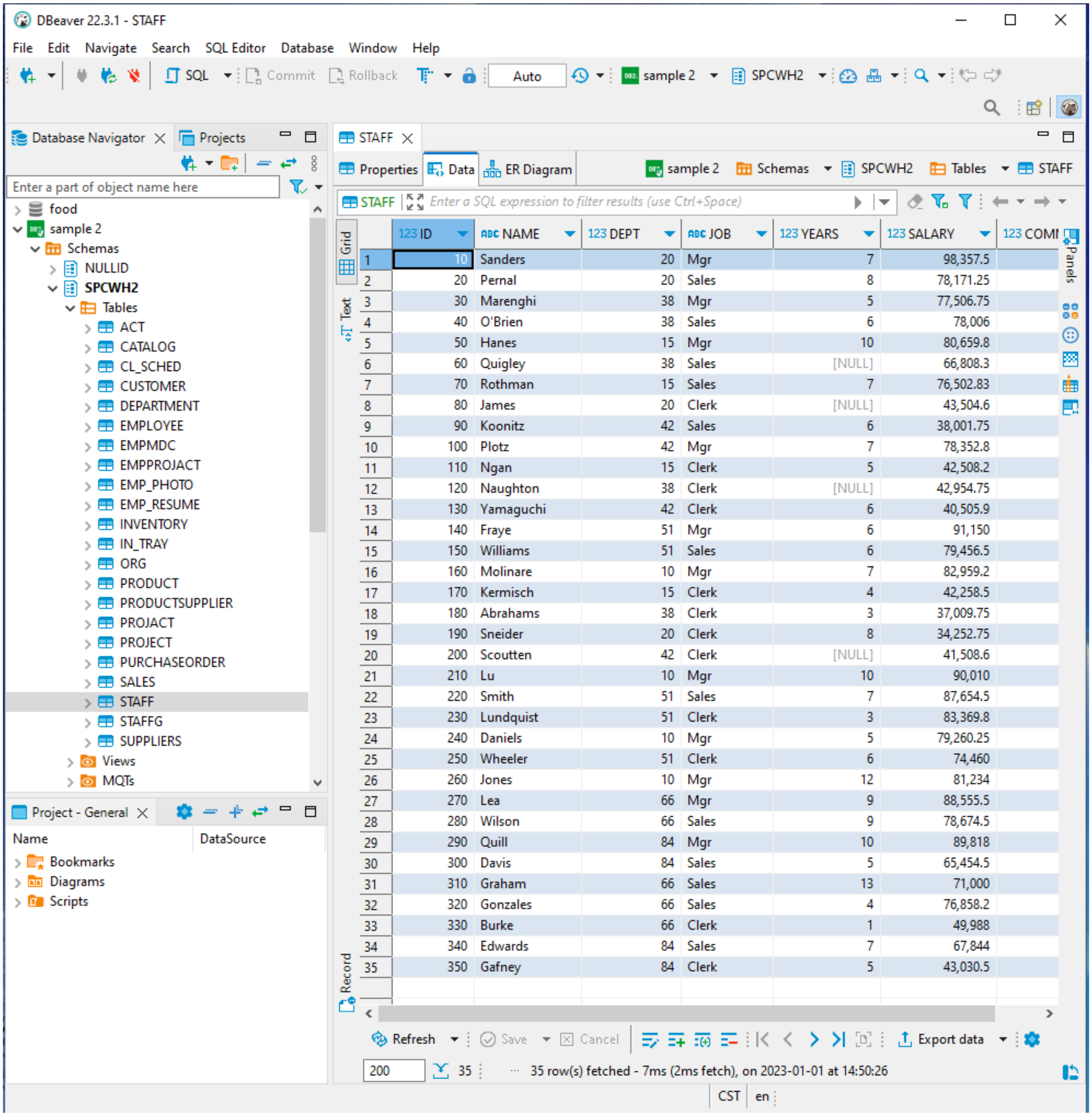
Note the D:\DB2 is the installation folder for the DB2 application
     and F:\DB2 is the location for the database storage
I personally found that using a DB Manager was very helpful as it allowed me to use a GUI interface to view the DB2 catalog as well as tools to view the contents of the SAMPLE tables. It also allows you to run SQL queries similar to SPUFI for those of you familiar with DB2 on the mainframe.

Here are some screenshots using Dbeaver.

This shows all the tables in the SPCWH2 schema (my DB2 userid).



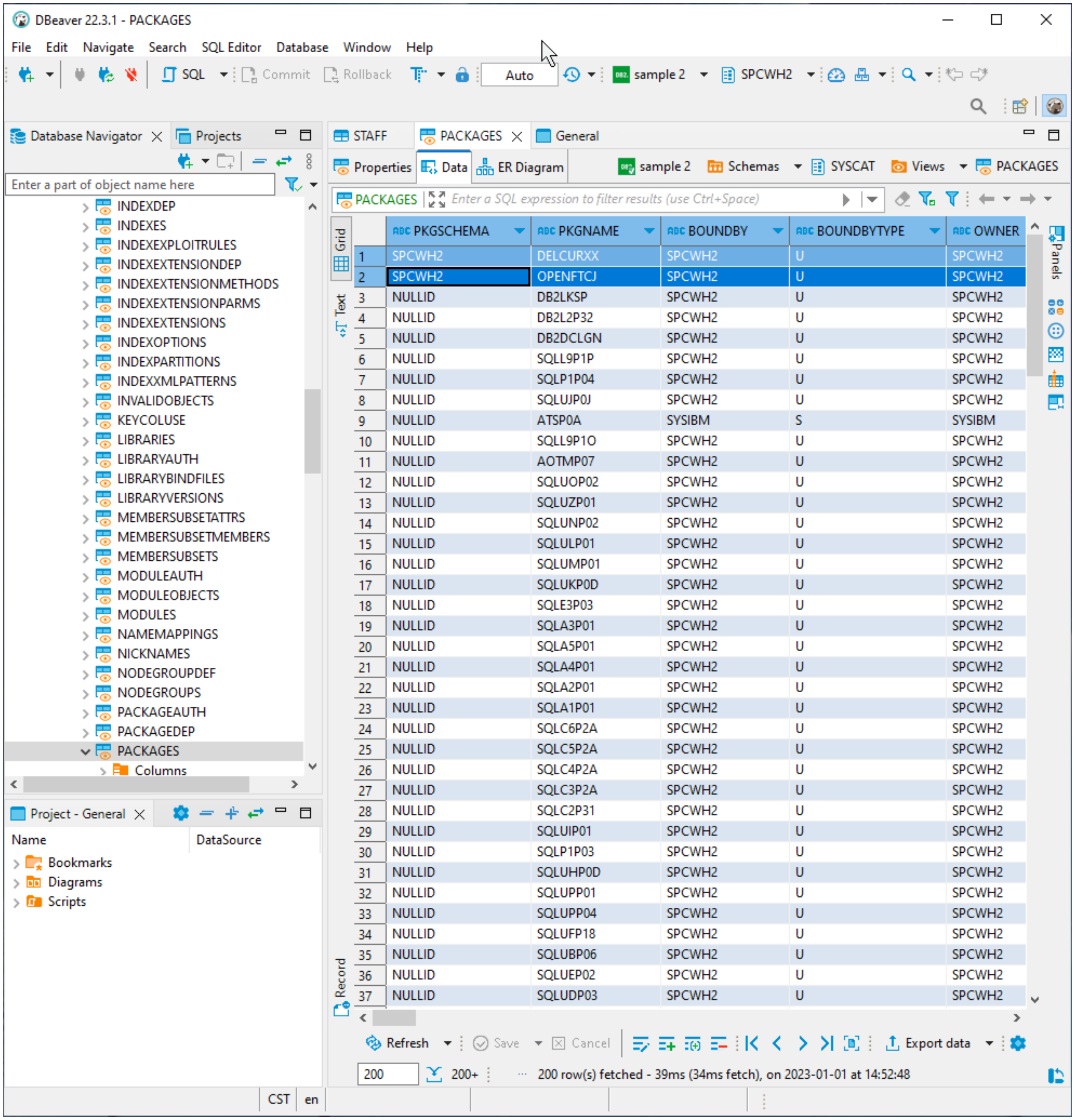
here is a screeshot showing the data in a table.



	123 ID	ABC NAME	123 DEPT	ABC JOB	123 YEARS	123 SALARY	123 COMI
1	10	Sanders	20	Mgr	7	98,357.5	
2	20	Pernal	20	Sales	8	78,171.25	
3	30	Marenghi	38	Mgr	5	77,506.75	
4	40	O'Brien	38	Sales	6	78,006	
5	50	Hanes	15	Mgr	10	80,659.8	
6	60	Quigley	38	Sales	[NULL]	66,808.3	
7	70	Rothman	15	Sales	7	76,502.83	
8	80	James	20	Clerk	[NULL]	43,504.6	
9	90	Koonitz	42	Sales	6	38,001.75	
10	100	Plotz	42	Mgr	7	78,352.8	
11	110	Ngan	15	Clerk	5	42,508.2	
12	120	Naughton	38	Clerk	[NULL]	42,954.75	
13	130	Yamaguchi	42	Clerk	6	40,505.9	
14	140	Fraye	51	Mgr	6	91,150	
15	150	Williams	51	Sales	6	79,456.5	
16	160	Molinare	10	Mgr	7	82,959.2	
17	170	Kermisch	15	Clerk	4	42,258.5	
18	180	Abrahams	38	Clerk	3	37,009.75	
19	190	Sneider	20	Clerk	8	34,252.75	
20	200	Scoutten	42	Clerk	[NULL]	41,508.6	
21	210	Lu	10	Mgr	10	90,010	
22	220	Smith	51	Sales	7	87,654.5	
23	230	Lundquist	51	Clerk	3	83,369.8	
24	240	Daniels	10	Mgr	5	79,260.25	
25	250	Wheeler	51	Clerk	6	74,460	
26	260	Jones	10	Mgr	12	81,234	
27	270	Lea	66	Mgr	9	88,555.5	
28	280	Wilson	66	Sales	9	78,674.5	
29	290	Quill	84	Mgr	10	89,818	
30	300	Davis	84	Sales	5	65,454.5	
31	310	Graham	66	Sales	13	71,000	
32	320	Gonzales	66	Sales	4	76,858.2	
33	330	Burke	66	Clerk	1	49,988	
34	340	Edwards	84	Sales	7	67,844	
35	350	Gafney	84	Clerk	5	43,030.5	

35 row(s) fetched - 7ms (2ms fetch), on 2023-01-01 at 14:50:26

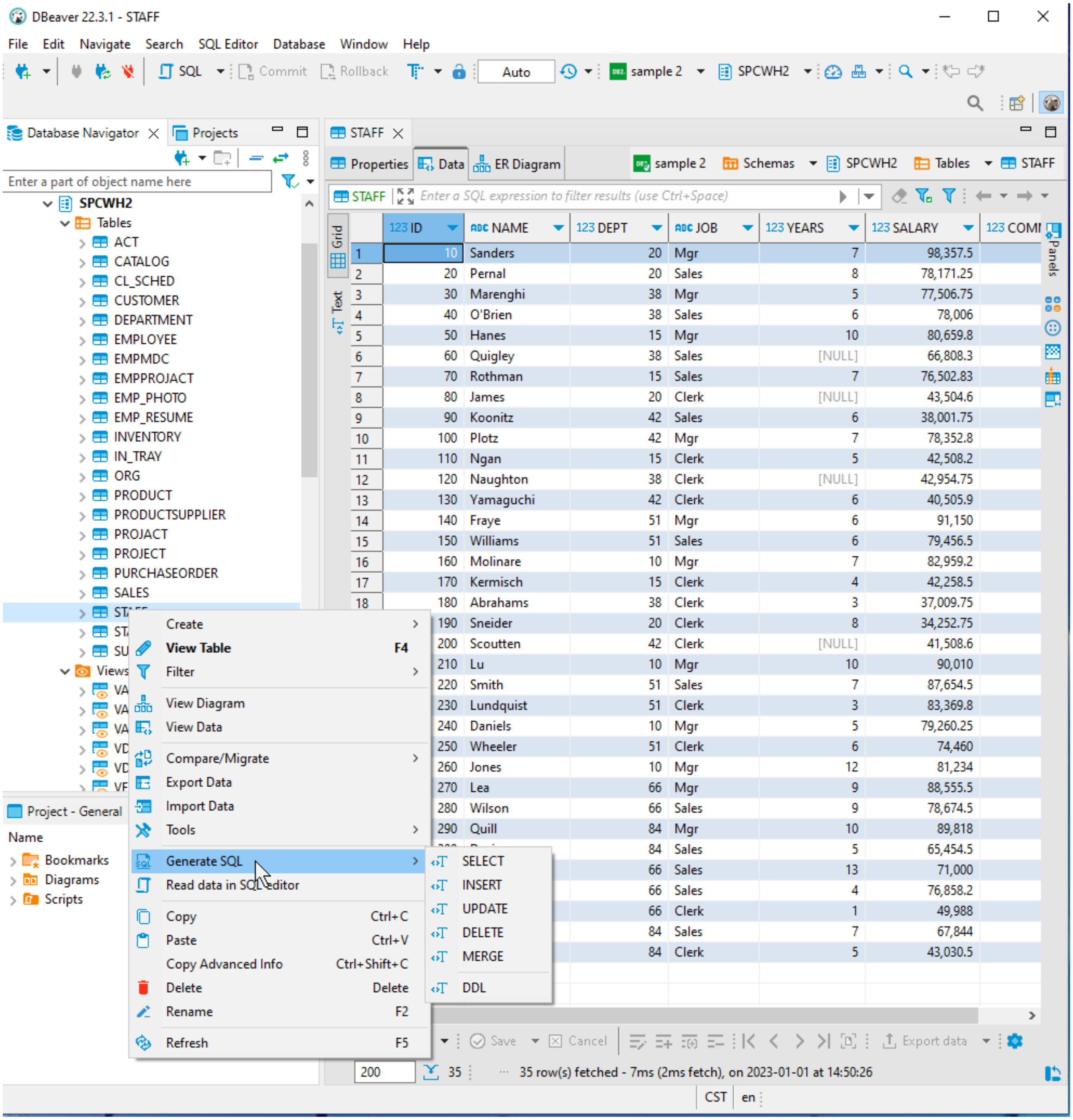
here is a screenshot of the packages… note that they are in the SYSCAT schema which is a bit different the the DB2 Z/OS catalog. So sometimes catalog data may be stored in a different location, you may have to search for it. The first two rows are packages that I created by precompiling a couple of sample programs which I modified.



	PKGSCHEMA	PKGNAME	BOUNDBY	BOUNDBYTYPE	OWNER
1	SPCWH2	DELCURXX	SPCWH2	U	SPCWH2
2	SPCWH2	OPENFTCJ	SPCWH2	U	SPCWH2
3	NULLID	DB2LKSP	SPCWH2	U	SPCWH2
4	NULLID	DB2L2P32	SPCWH2	U	SPCWH2
5	NULLID	DB2DCLGN	SPCWH2	U	SPCWH2
6	NULLID	SQLL9P1P	SPCWH2	U	SPCWH2
7	NULLID	SQLP1P04	SPCWH2	U	SPCWH2
8	NULLID	SQLUJP0J	SPCWH2	U	SPCWH2
9	NULLID	ATSP0A	SYSIBM	S	SYSIBM
10	NULLID	SQLL9P1O	SPCWH2	U	SPCWH2
11	NULLID	AOTMP07	SPCWH2	U	SPCWH2
12	NULLID	SQLUOP02	SPCWH2	U	SPCWH2
13	NULLID	SQLUZP01	SPCWH2	U	SPCWH2
14	NULLID	SQLUNP02	SPCWH2	U	SPCWH2
15	NULLID	SQLULP01	SPCWH2	U	SPCWH2
16	NULLID	SQLUMP01	SPCWH2	U	SPCWH2
17	NULLID	SQLUKP0D	SPCWH2	U	SPCWH2
18	NULLID	SQLE3P03	SPCWH2	U	SPCWH2
19	NULLID	SQLA3P01	SPCWH2	U	SPCWH2
20	NULLID	SQLA5P01	SPCWH2	U	SPCWH2
21	NULLID	SQLA4P01	SPCWH2	U	SPCWH2
22	NULLID	SQLA2P01	SPCWH2	U	SPCWH2
23	NULLID	SQLA1P01	SPCWH2	U	SPCWH2
24	NULLID	SQLC6P2A	SPCWH2	U	SPCWH2
25	NULLID	SQLC5P2A	SPCWH2	U	SPCWH2
26	NULLID	SQLC4P2A	SPCWH2	U	SPCWH2
27	NULLID	SQLC3P2A	SPCWH2	U	SPCWH2
28	NULLID	SQLC2P31	SPCWH2	U	SPCWH2
29	NULLID	SQLUIP01	SPCWH2	U	SPCWH2
30	NULLID	SQLP1P03	SPCWH2	U	SPCWH2
31	NULLID	SQLUHP0D	SPCWH2	U	SPCWH2
32	NULLID	SQLUPP01	SPCWH2	U	SPCWH2
33	NULLID	SQLUPP04	SPCWH2	U	SPCWH2
34	NULLID	SQLUFP18	SPCWH2	U	SPCWH2
35	NULLID	SQLUBP06	SPCWH2	U	SPCWH2
36	NULLID	SQLUEP02	SPCWH2	U	SPCWH2
37	NULLID	SQLUDP03	SPCWH2	U	SPCWH2

```
200 row(s) fetched - 39ms (34ms fetch), on 2023-01-01 at 14:52:48
```

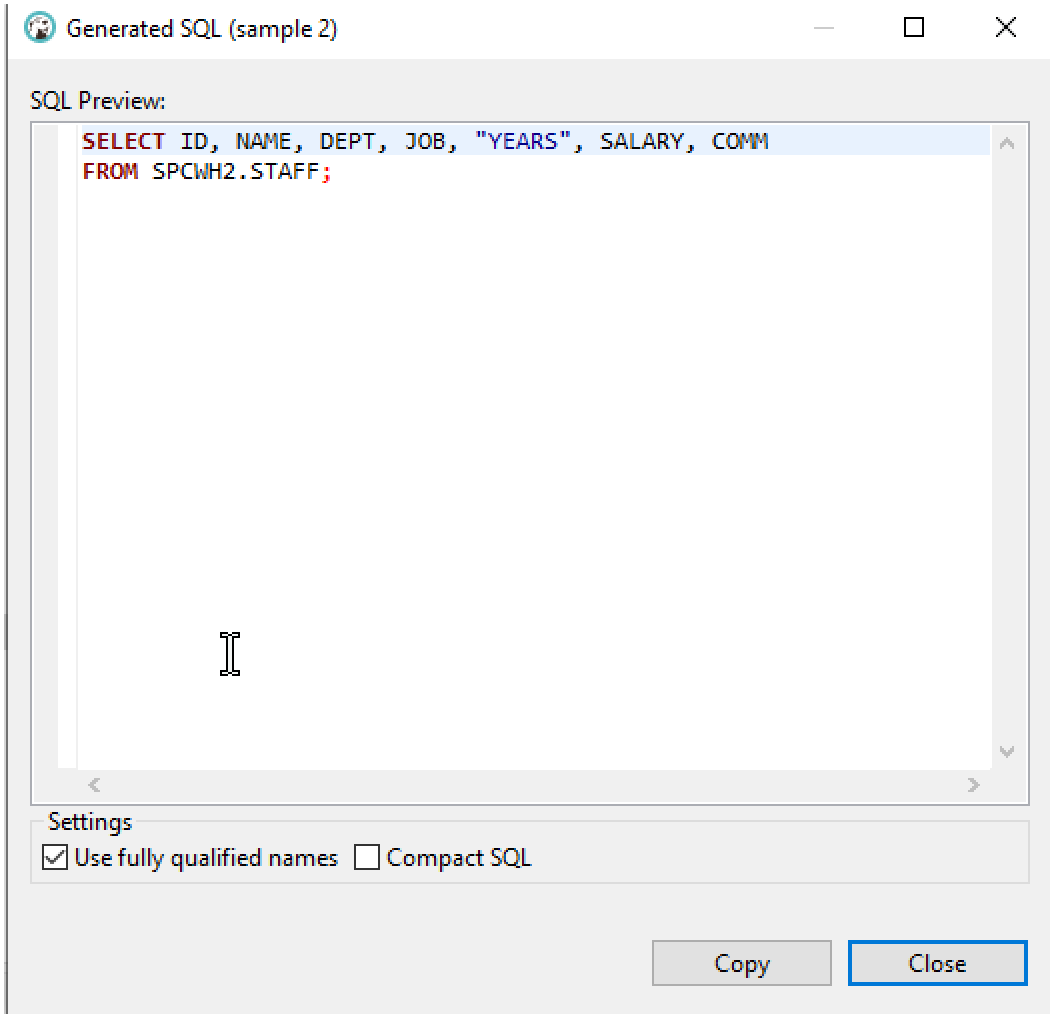
here is a option to generate sql…



	123 ID	ABC NAME	123 DEPT	ABC JOB	123 YEARS	123 SALARY	123 COMM
1	10	Sanders	20	Mgr	7	98,357.5	
2	20	Pernal	20	Sales	8	78,171.25	
3	30	Marenghi	38	Mgr	5	77,506.75	
4	40	O'Brien	38	Sales	6	78,006	
5	50	Hanes	15	Mgr	10	80,659.8	
6	60	Quigley	38	Sales	[NULL]	66,808.3	
7	70	Rothman	15	Sales	7	76,502.83	
8	80	James	20	Clerk	[NULL]	43,504.6	
9	90	Koonitz	42	Sales	6	38,001.75	
10	100	Plotz	42	Mgr	7	78,352.8	
11	110	Ngan	15	Clerk	5	42,508.2	
12	120	Naughton	38	Clerk	[NULL]	42,954.75	
13	130	Yamaguchi	42	Clerk	6	40,505.9	
14	140	Fraye	51	Mgr	6	91,150	
15	150	Williams	51	Sales	6	79,456.5	
16	160	Molinare	10	Mgr	7	82,959.2	
17	170	Kermisch	15	Clerk	4	42,258.5	
18	180	Abrahams	38	Clerk	3	37,009.75	
	190	Sneider	20	Clerk	8	34,252.75	
	200	Scoutten	42	Clerk	[NULL]	41,508.6	
	210	Lu	10	Mgr	10	90,010	
	220	Smith	51	Sales	7	87,654.5	
	230	Lundquist	51	Clerk	3	83,369.8	
	240	Daniels	10	Mgr	5	79,260.25	
	250	Wheeler	51	Clerk	6	74,460	
	260	Jones	10	Mgr	12	81,234	
	270	Lea	66	Mgr	9	88,555.5	
	280	Wilson	66	Sales	9	78,674.5	
	290	Quill	84	Mgr	10	89,818	
			84	Sales	5	65,454.5	
			66	Sales	13	71,000	
			66	Sales	4	76,858.2	
			66	Clerk	1	49,988	
			84	Sales	7	67,844	
			84	Clerk	5	43,030.5	

35 row(s) fetched - 7ms (2ms fetch), on 2023-01-01 at 14:50:26

here is the generated sql



```
SELECT ID, NAME, DEPT, JOB, "YEARS", SALARY, COMM
FROM SPCWH2.STAFF;
```

Settings

☑ Use fully qualified names ☐ Compact SQL

It should be noted that all of my testing has been done on my local machine where the DB2 server is installed. I have not connected to my DB2 server from a remote machine. I would expect that this is possible to do but it is outside my experience.

All of my precompiles / compiles and executions have been done from a Windows CMD window. Actually the “db2cmd” command can open the DB2 command window as another separate window and I find that somewhat easier to work with.

For those of you who have worked with DB2 on the mainframe I will WARN you that some of the functionality of the current DB2 release on the mainframe has NOT been implemented on the DB2 Community Edition for LUW (perhaps not the paid version either).

This is the most significant item I have found as it relates to multi-row processing functionality.

- Db2 / 11.5

Support of Multi-row Fetch Db2 Precompiler and embedded Runtime **support** the fetch of multiple rows by using a single FETCH statement.,The Db2 server and client does not **support ROWSET** cursors. Hence the Db2 precompiler strips off the WITH **ROWSET** POSITIONING keyword from the DECLARE CURSOR statement, and the NEXT **ROWSET** keyword from the FETCH...

I would suggest that you read the following section before using multi-row sql in your applications. This is especially true if you are attempting to use ROWSET cursors with either an UPDATE WHERE CURRENT OF   or a DELETE WHERE CURRENT OF because I have found that they do NOT work as expected with DB2 Z/OS.

Embedded SQL/COBOL Support for MRI and MRF

On DB2 Z/OS there are 4 steps to execute a typical batch DB2 program as follows.

1. Precompile the source program with embedded SQL
2. Compile the output source code created by the DB2 precompiler
3. BIND the DBRM file created by the precompiler
4. Execute the DB2 programs

On DB2 LUW 11.5.8 it is similar however the PRECOMPILE command has the option to create a DB2 package OR a *.bnd file to be used as input to the BIND command.

Note that if the BINDFILE parm is not included in the PRECOMPILE command it will actually create a DB2 package so the BIND step can be skipped. There may be reasons for you to create a *.bnd file and do a separate BIND command. For example if you create a new index on a table and you want your program to use that index it would require that the package be updated with possibly a new access path. For that you could just do another BIND with the REPLACE parm without having to recompile your program.

So with DB2 LUW 11.5.8 step 3 above is optional depending on the PRECOMPILE command presence of the BNDFILE parm.

PRECOMPILE command -- link to the documentation for Precompile command

This page contains the text of some of my Windows cmd script files.

NOTE that the ^ symbol is a line continuation character in the Windows cmd files. I prefer not to use very wide command lines. This way I can have one parameter per line in the cmd file.

Note that the cyan highlighting is for file paths specific to my environment and you will need to change that to match your environment.
Note that my compile cmd files have a "-v" at the end to create verbose output. You could remove the "-v" if you wish. I found it useful when finding errors.

SET_COB32.CMD – this sets the GNUCOBOL environment for the 32 bit compiler

```
set "save_loc=%cd%
cd /d D:\GNUCOBOL-V32-X32
call set_env.cmd
echo returning from set_env
echo value of save_loc is
set save_loc
cd /d %save_loc%
set save_loc=
```

SET_COB64.CMD – this sets the GNUCOBOL environment for the 64 bit compiler

```
set "save_loc=%cd%
cd /d D:\GNUCOBOL-V32-X64
call set_env.cmd
echo returning from set_env
echo value of save_loc is
set save_loc
cd /d %save_loc%
set save_loc=
```

Note ==> the sample cobol programs are in this directory.

D:\DB2\SQLLIB\samples\cobol_mf\

The COBOL source programs with embedded SQL have an extension of “SQB”

COMPILE-MSYS2_32STATIC.CMD – this compiles a GNUCOBOL program in 32 bit

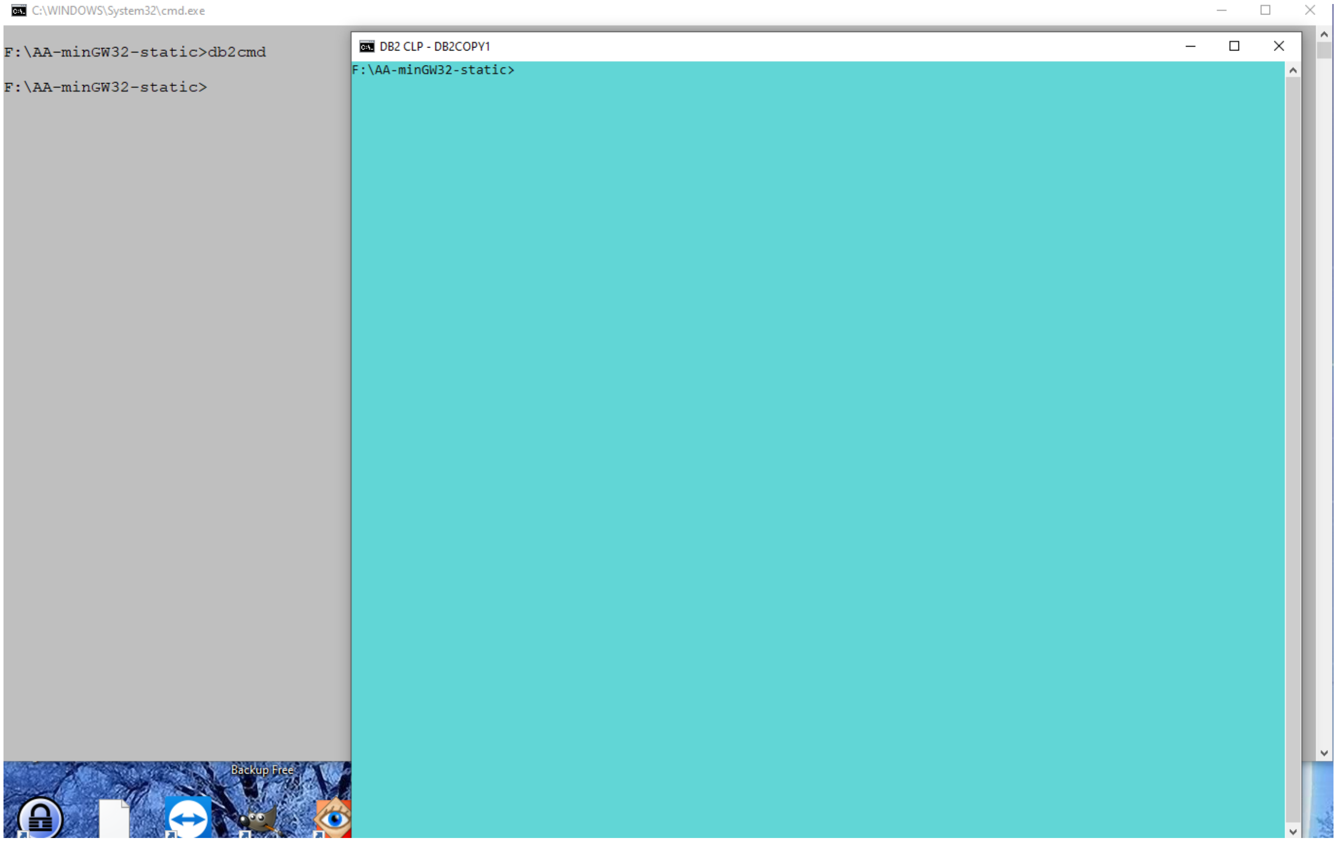
```
cobc -x %1.cbl checkerr.cbl ^
-std=mf ^
-Wall ^
-T %1.lst ^
-fixed ^
-fstatic-call ^
-save-temps ^
-fno-gen-c-decl-static-call ^
-Id:/DB2/SQLLIB/include/cobol_mf ^
-A "-v -I D:/DB2/SQLLIB/include -include sqlaprep.h" ^
-Q "D:\DB2\SQLLIB\BIN\db2agapi.dll" -v
```

COMPILE-MSYS2_64STATIC.CMD – this compiles a GNUCOBOL program in 64 bit

```
cobc -x %1.cbl checkerr.cbl ^
-std=mf ^
-Wall ^
-T %1.lst ^
-fixed ^
-fstatic-call ^
-fno-gen-c-decl-static-call ^
-Id:/DB2/SQLLIB/include/cobol_mf ^
-A "-I D:/DB2/SQLLIB/include --include sqlsystm.h --include sqlaprep.h" ^
-Q "D:\DB2\SQLLIB\BIN\db2agapi64.dll" -v
```

Note that to PRECOMPILE or BIND, those commands must be done in a DB2 Command window. To do that you need to create it by executing the following command in a Windows CMD window

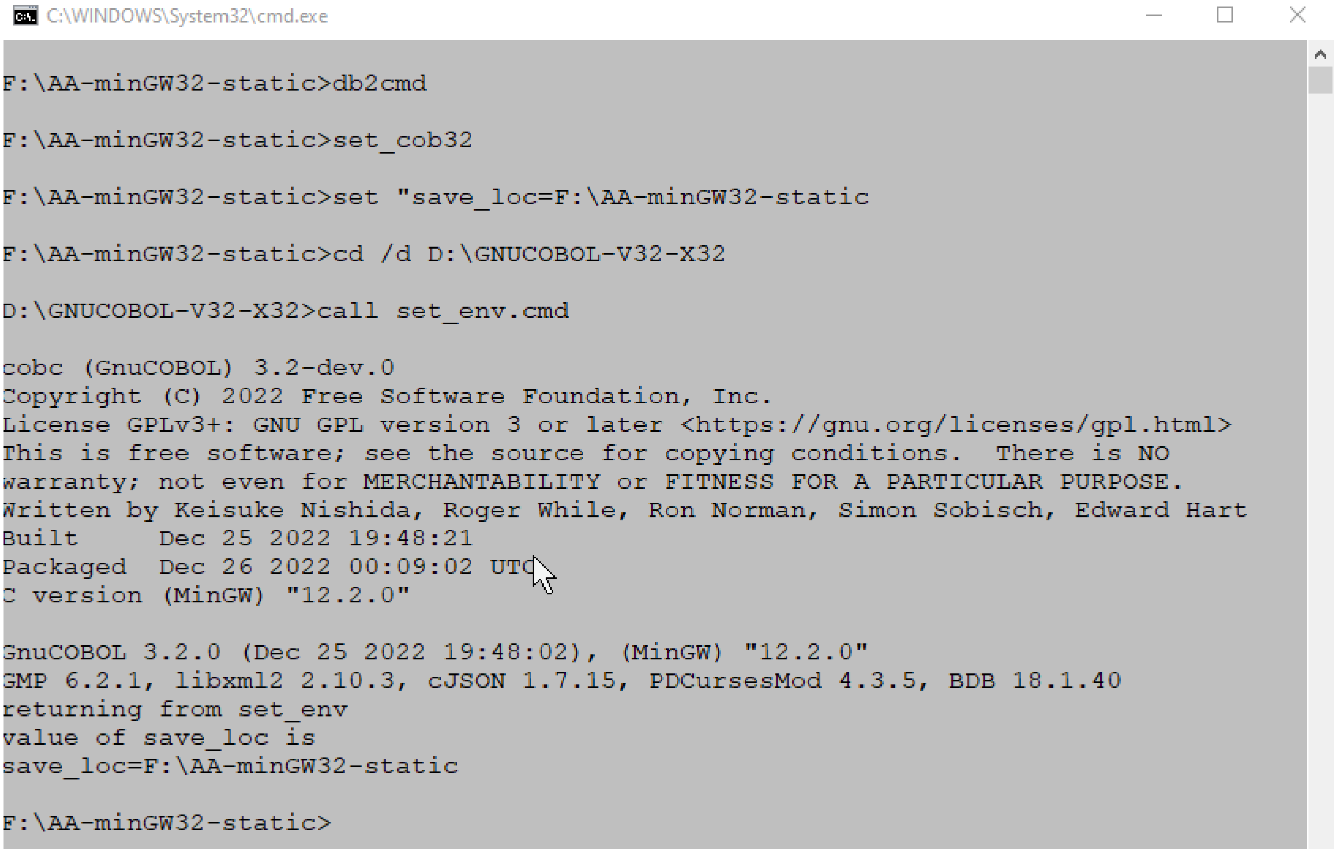
Note we first open a Windows CMD window and enter the command “db2cmd” and hit enter. I then opens a DB2 command window while leaving the Windows CMD available for use as well.



```
F:\AA-minGW32-static>db2cmd

F:\AA-minGW32-static>
```

```
F:\AA-minGW32-static>
```

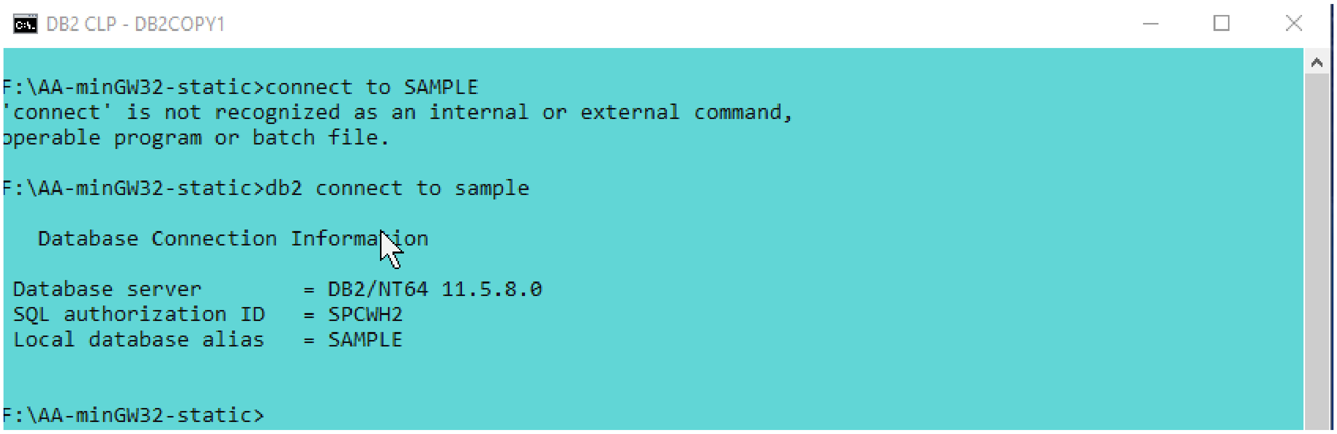
Then enter the command in the Windows CMD to set the environment



```
F:\AA-minGW32-static>db2cmd

F:\AA-minGW32-static>set_cob32

F:\AA-minGW32-static>set "save_loc=F:\AA-minGW32-static

F:\AA-minGW32-static>cd /d D:\GNUCOBOL-V32-X32

D:\GNUCOBOL-V32-X32>call set_env.cmd

cobc (GnuCOBOL) 3.2-dev.0
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <https://gnu.org/licenses/gpl.html>
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
Written by Keisuke Nishida, Roger While, Ron Norman, Simon Sobisch, Edward Hart
Built     Dec 25 2022 19:48:21
Packaged  Dec 26 2022 00:09:02 UTC
C version (MinGW) "12.2.0"

GnuCOBOL 3.2.0 (Dec 25 2022 19:48:02), (MinGW) "12.2.0"
GMP 6.2.1, libxml2 2.10.3, cJSON 1.7.15, PDCursesMod 4.3.5, BDB 18.1.40
returning from set_env
value of save_loc is
save_loc=F:\AA-minGW32-static

F:\AA-minGW32-static>
```

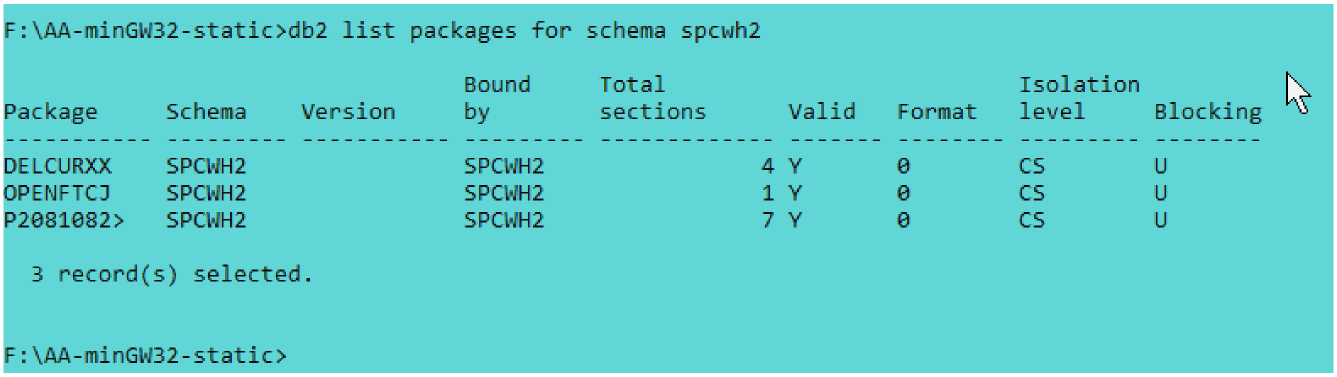
Then in the DB2 Command window you need to connect to the database SAMPLE before entering any commands.

Note that the command has to be prefaced with "db2" I forgot that to illustrate it.



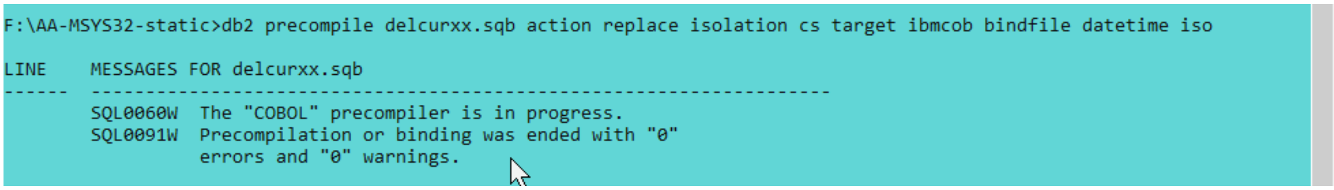
```
F:\AA-minGW32-static>connect to SAMPLE
'connect' is not recognized as an internal or external command,
operable program or batch file.

F:\AA-minGW32-static>db2 connect to sample

   Database Connection Information

 Database server        = DB2/NT64 11.5.8.0
 SQL authorization ID   = SPCWH2
 Local database alias   = SAMPLE


F:\AA-minGW32-static>
```

here is an example of the command to list packages.



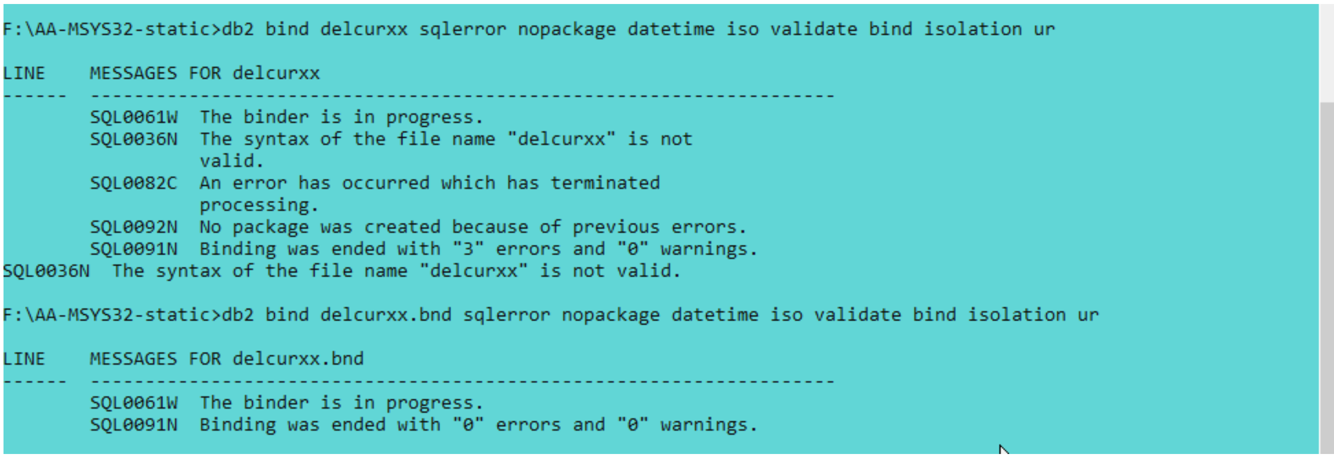
```
F:\AA-minGW32-static>db2 list packages for schema spcwh2

                                  Bound     Total                          Isolation
Package     Schema    Version     by        sections      Valid   Format   level     Blocking
----------- --------- ----------- --------- ------------- ------- -------- --------- --------
DELCURXX    SPCWH2                SPCWH2                4 Y       0        CS        U
OPENFTCJ    SPCWH2                SPCWH2                1 Y       0        CS        U
P2081082>   SPCWH2                SPCWH2                7 Y       0        CS        U

  3 record(s) selected.


F:\AA-minGW32-static>
```

Here is an example of a precompile command creating a bndfile and iso datetime. Notice that this does NOT create a packagae so a BIND command is needed for that.



```
F:\AA-MSYS32-static>db2 precompile delcurxx.sqb action replace isolation cs target ibmcob bindfile datetime iso

LINE    MESSAGES FOR delcurxx.sqb
------  --------------------------------------------------------------------
        SQL0060W  The "COBOL" precompiler is in progress.
        SQL0091W  Precompilation or binding was ended with "0"
                  errors and "0" warnings.
```

Here is an example of a DB2 BIND command, notice that the first command has an error and the second command has corrected the error.



```
F:\AA-MSYS32-static>db2 bind delcurxx sqlerror nopackage datetime iso validate bind isolation ur

LINE    MESSAGES FOR delcurxx
------  --------------------------------------------------------------------
        SQL0061W  The binder is in progress.
        SQL0036N  The syntax of the file name "delcurxx" is not
                  valid.
        SQL0082C  An error has occurred which has terminated
                  processing.
        SQL0092N  No package was created because of previous errors.
        SQL0091N  Binding was ended with "3" errors and "0" warnings.
SQL0036N  The syntax of the file name "delcurxx" is not valid.

F:\AA-MSYS32-static>db2 bind delcurxx.bnd sqlerror nopackage datetime iso validate bind isolation ur

LINE    MESSAGES FOR delcurxx.bnd
------  --------------------------------------------------------------------
        SQL0061W  The binder is in progress.
        SQL0091N  Binding was ended with "0" errors and "0" warnings.
```

Note that for mult-row fetch cursors if you want the same behavior as DB2 Z/OS you need to set the following environment variable

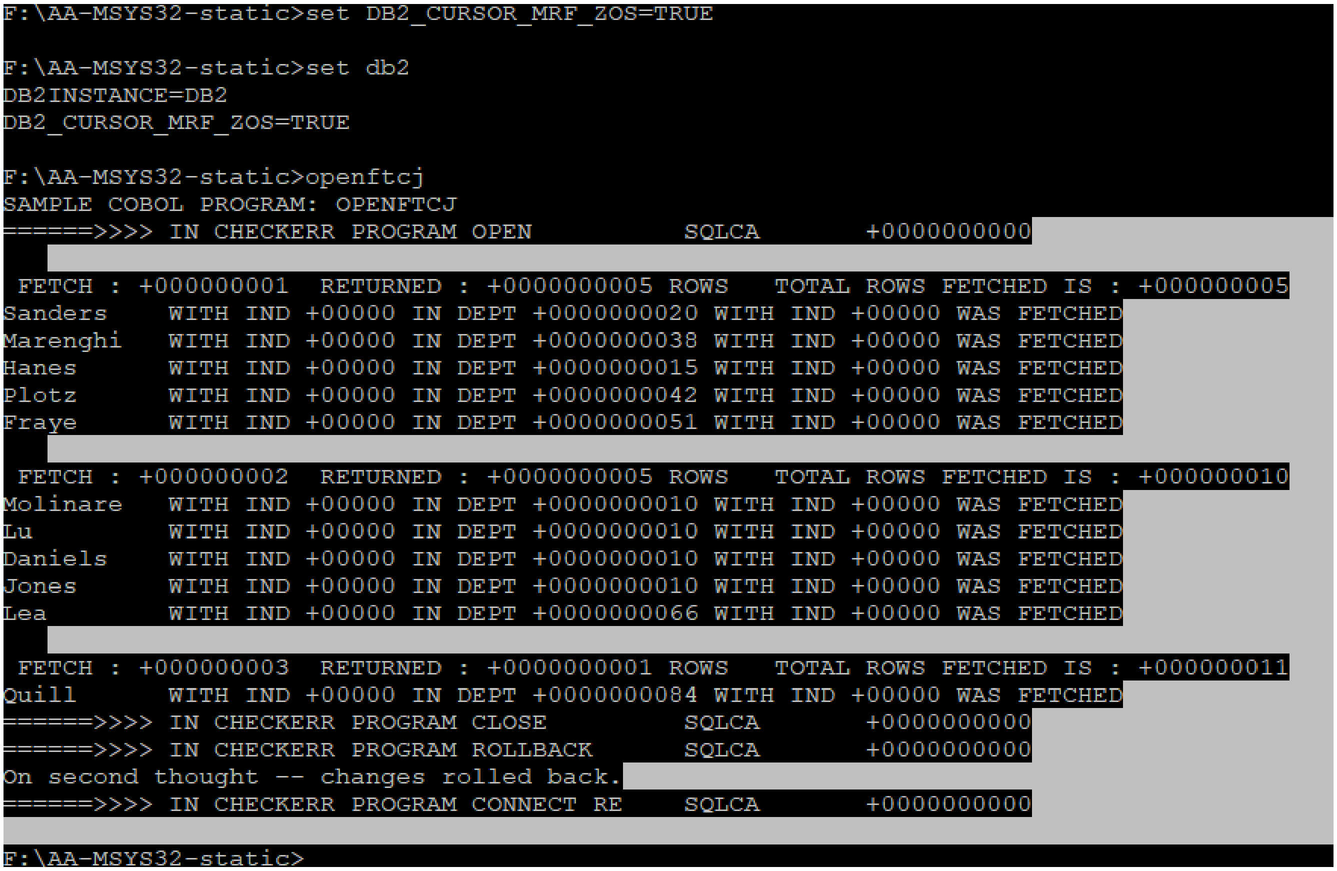
```
set DB2_CURSOR_MRF_ZOS=TRUE
```

with this on if the last fetch returned all the remaining rows, then the SQLCODE will be set to +100.

If not set it will execute another fetch after all the rows are returned before setting the SQLCODE to +100.

Here is an example with a rowset returning 5 rows and the cursor returning 11 rows total.

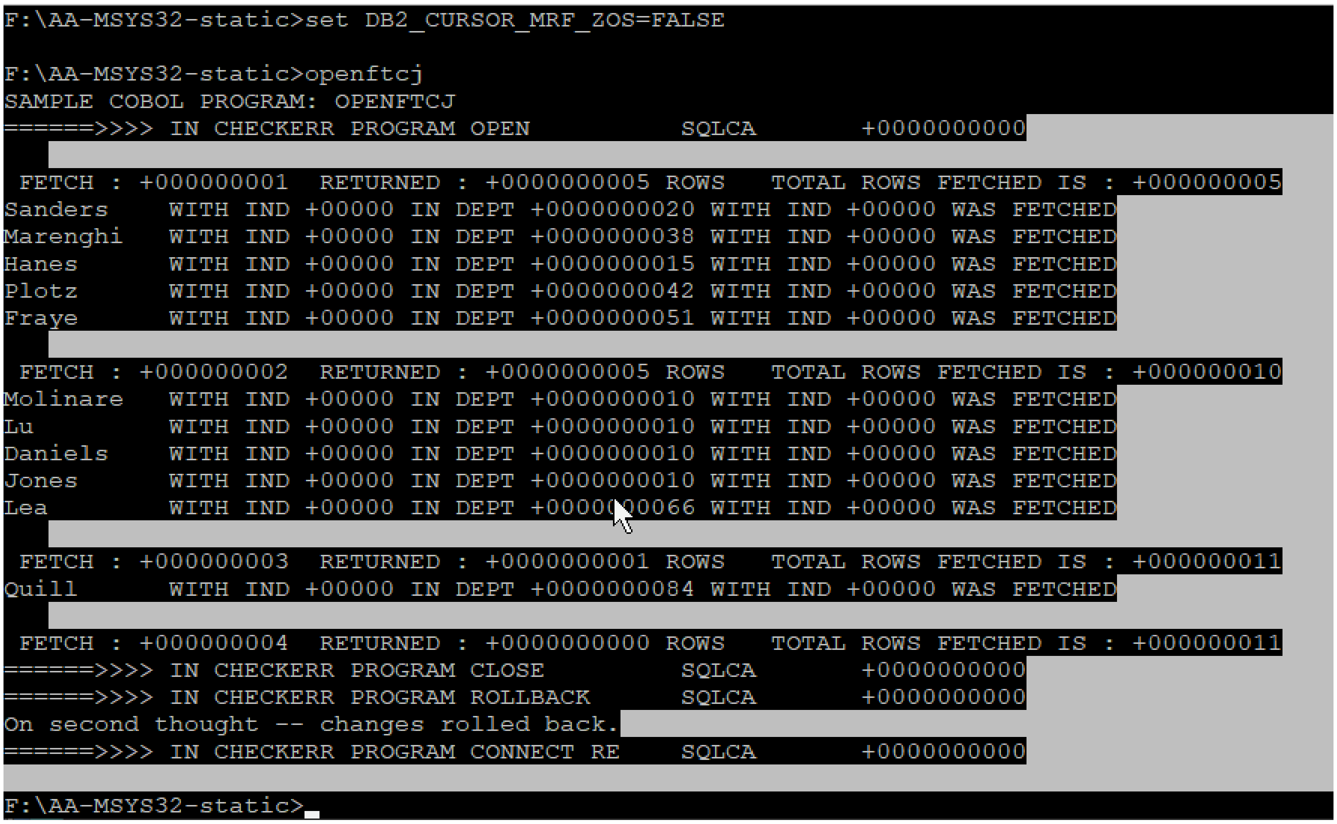
Notice that it executed 3 FETCH commands



```
F:\AA-MSYS32-static>set DB2_CURSOR_MRF_ZOS=TRUE

F:\AA-MSYS32-static>set db2
DB2INSTANCE=DB2
DB2_CURSOR_MRF_ZOS=TRUE

F:\AA-MSYS32-static>openftcj
SAMPLE COBOL PROGRAM: OPENFTCJ
======>>>> IN CHECKERR PROGRAM OPEN          SQLCA       +0000000000

 FETCH : +000000001  RETURNED : +0000000005 ROWS   TOTAL ROWS FETCHED IS : +000000005
Sanders    WITH IND +00000 IN DEPT +0000000020 WITH IND +00000 WAS FETCHED
Marenghi   WITH IND +00000 IN DEPT +0000000038 WITH IND +00000 WAS FETCHED
Hanes      WITH IND +00000 IN DEPT +0000000015 WITH IND +00000 WAS FETCHED
Plotz      WITH IND +00000 IN DEPT +0000000042 WITH IND +00000 WAS FETCHED
Fraye      WITH IND +00000 IN DEPT +0000000051 WITH IND +00000 WAS FETCHED

 FETCH : +000000002  RETURNED : +0000000005 ROWS   TOTAL ROWS FETCHED IS : +000000010
Molinare   WITH IND +00000 IN DEPT +0000000010 WITH IND +00000 WAS FETCHED
Lu         WITH IND +00000 IN DEPT +0000000010 WITH IND +00000 WAS FETCHED
Daniels    WITH IND +00000 IN DEPT +0000000010 WITH IND +00000 WAS FETCHED
Jones      WITH IND +00000 IN DEPT +0000000010 WITH IND +00000 WAS FETCHED
Lea        WITH IND +00000 IN DEPT +0000000066 WITH IND +00000 WAS FETCHED

 FETCH : +000000003  RETURNED : +0000000001 ROWS   TOTAL ROWS FETCHED IS : +000000011
Quill      WITH IND +00000 IN DEPT +0000000084 WITH IND +00000 WAS FETCHED
======>>>> IN CHECKERR PROGRAM CLOSE         SQLCA       +0000000000
======>>>> IN CHECKERR PROGRAM ROLLBACK      SQLCA       +0000000000
On second thought -- changes rolled back.
======>>>> IN CHECKERR PROGRAM CONNECT RE    SQLCA       +0000000000

F:\AA-MSYS32-static>
```

Same program execution with it set to FALSE

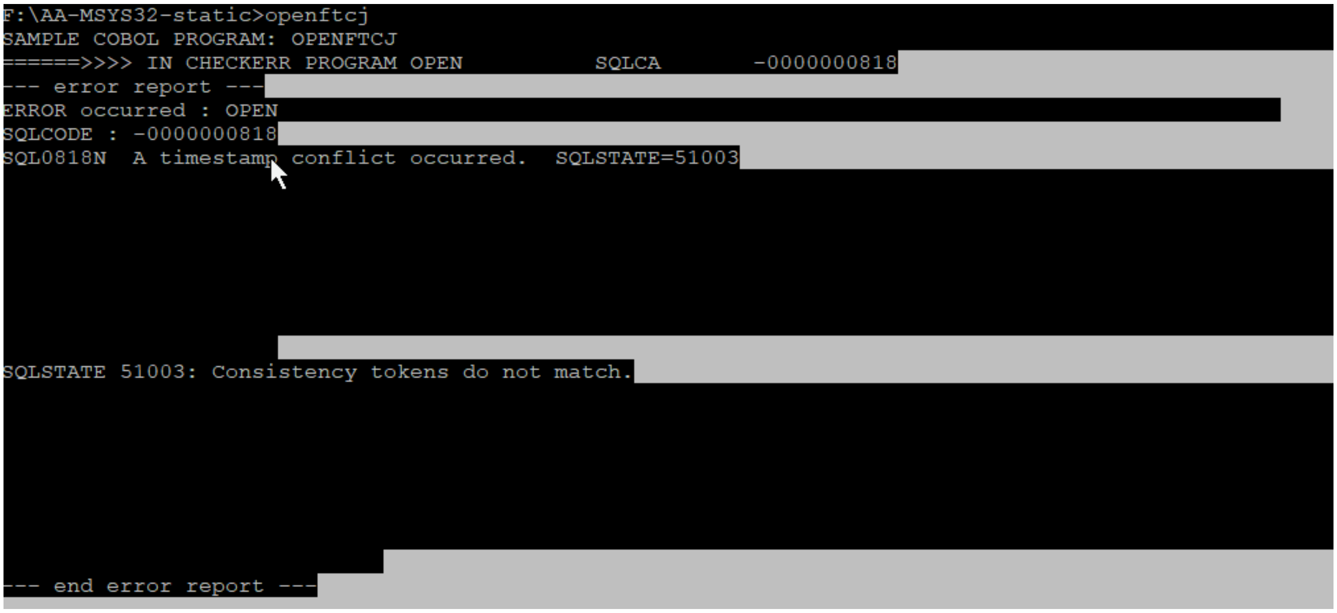
Notice that it executed 4 FETCH commands



```
F:\AA-MSYS32-static>set DB2_CURSOR_MRF_ZOS=FALSE

F:\AA-MSYS32-static>openftcj
SAMPLE COBOL PROGRAM: OPENFTCJ
======>>>> IN CHECKERR PROGRAM OPEN          SQLCA       +0000000000

 FETCH : +000000001  RETURNED : +0000000005 ROWS   TOTAL ROWS FETCHED IS : +000000005
Sanders    WITH IND +00000 IN DEPT +0000000020 WITH IND +00000 WAS FETCHED
Marenghi   WITH IND +00000 IN DEPT +0000000038 WITH IND +00000 WAS FETCHED
Hanes      WITH IND +00000 IN DEPT +0000000015 WITH IND +00000 WAS FETCHED
Plotz      WITH IND +00000 IN DEPT +0000000042 WITH IND +00000 WAS FETCHED
Fraye      WITH IND +00000 IN DEPT +0000000051 WITH IND +00000 WAS FETCHED

 FETCH : +000000002  RETURNED : +0000000005 ROWS   TOTAL ROWS FETCHED IS : +000000010
Molinare   WITH IND +00000 IN DEPT +0000000010 WITH IND +00000 WAS FETCHED
Lu         WITH IND +00000 IN DEPT +0000000010 WITH IND +00000 WAS FETCHED
Daniels    WITH IND +00000 IN DEPT +0000000010 WITH IND +00000 WAS FETCHED
Jones      WITH IND +00000 IN DEPT +0000000010 WITH IND +00000 WAS FETCHED
Lea        WITH IND +00000 IN DEPT +0000000066 WITH IND +00000 WAS FETCHED

 FETCH : +000000003  RETURNED : +0000000001 ROWS   TOTAL ROWS FETCHED IS : +000000011
Quill      WITH IND +00000 IN DEPT +0000000084 WITH IND +00000 WAS FETCHED

 FETCH : +000000004  RETURNED : +0000000000 ROWS   TOTAL ROWS FETCHED IS : +000000011
======>>>> IN CHECKERR PROGRAM CLOSE         SQLCA       +0000000000
======>>>> IN CHECKERR PROGRAM ROLLBACK      SQLCA       +0000000000
On second thought -- changes rolled back.
======>>>> IN CHECKERR PROGRAM CONNECT RE    SQLCA       +0000000000

F:\AA-MSYS32-static>_
```

Here is an example of a timestamp error…

precompile / compile / execute   ===> missing bind



```
F:\AA-MSYS32-static>openftcj
SAMPLE COBOL PROGRAM: OPENFTCJ
======>>>> IN CHECKERR PROGRAM OPEN          SQLCA       -0000000818
--- error report ---
ERROR occurred : OPEN
SQLCODE : -0000000818
SQL0818N  A timestamp conflict occurred.  SQLSTATE=51003



SQLSTATE 51003: Consistency tokens do not match.



--- end error report ---
```

Here is the OPENFTCJ.SQB source code for you to test with.

```
       Identification Division.
       PROGRAM-ID. "OPENFTCJ".

       Data Division.
       Working-Storage Section.

         copy "sqlca.cbl".
         EXEC SQL BEGIN DECLARE SECTION END-EXEC.
       01 dept-rec.
         03 pname             pic x(10) OCCURS 5 TIMES.
         03 dept              pic s9(9) comp-5 OCCURS 5 TIMES.
         03 IND-PNAME         PIC S9(4) COMP-5 OCCURS 5 TIMES.
         03 IND-DEPT          PIC S9(4) COMP-5 OCCURS 5 TIMES.
         03 cnt               pic s9(9) comp-5.
       01 userid            pic x(8).
       01 passwd.
         49 passwd-length   pic s9(4) comp-5 value 0.
         49 passwd-name     pic x(18).

         EXEC SQL END DECLARE SECTION END-EXEC.

       77 errloc          pic x(80).
       77 cntr-fetch                   pic s9(9)  binary   value +0.
       77 cntr-rows                    pic s9(9)  binary   value +0.
       77 sub                          pic s9(9)  binary   value +0.

       Procedure Division.
       0000-mainline.
         DISPLAY "SAMPLE COBOL PROGRAM: OPENFTCJ".

         EXEC SQL CONNECT TO sample END-EXEC

         MOVE 5 TO cnt.
         EXEC SQL DECLARE c1 CURSOR
         WITH ROWSET POSITIONING
         FOR SELECT NAME, DEPT
             FROM staff
             WHERE job='Mgr' END-EXEC.

         EXEC SQL OPEN c1 END-EXEC.
         move "OPEN" to errloc.
         call "checkerr" using SQLCA errloc.

      **call the FETCH and UPDATE/DELETE loop.
         perform 1000-Fetch-Loop thru 1000-exit
           until SQLCODE not equal 0.

         EXEC SQL CLOSE c1 END-EXEC.
         move "CLOSE" to errloc.
         call "checkerr" using SQLCA errloc.

         EXEC SQL ROLLBACK END-EXEC.
         move "ROLLBACK" to errloc.
         call "checkerr" using SQLCA errloc.
         display "On second thought -- changes rolled back.".
         EXEC SQL CONNECT RESET END-EXEC.
         move "CONNECT RESET" to errloc.
         call "checkerr" using SQLCA errloc.
       0000-exit.
         go to End-Prog.

       1000-Fetch-Loop.

         EXEC SQL FETCH NEXT ROWSET FROM C1
         FOR :CNT ROWS
         INTO :PNAME     :IND-PNAME,
              :DEPT      :IND-DEPT
         END-EXEC.

           add +1                      to cntr-fetch.
           add SQLERRD (3)             to cntr-rows.

           display '   '.
           display ' FETCH : ' cntr-fetch
               '  RETURNED : ' SQLERRD (3) ' ROWS'
               '   TOTAL ROWS FETCHED IS : ' cntr-rows.
           move +0                     to sub.
           perform SQLERRD (3) times
               perform 2000-print-row thru 2000-exit
           end-perform.

       1000-exit.  exit.


       2000-print-row.

           add +1                      to sub.
         DISPLAY PNAME(sub) " WITH IND " IND-PNAME(sub)
         " IN DEPT "
         DEPT(sub) " WITH IND " IND-DEPT(sub)
         " WAS FETCHED".

       2000-exit.  exit.


       End-Prog.
         stop run.
```