CBSA Enacting Scenario 2 – COBOL changes

Introduction:

This “CBSA Enacting Scenario 2” document is provided as part of the Nazare project documentation, it provides a step by step guide to amending an existing COBOL/BMS/Db2 application. It assumes that CBSA has already been installed and is up and running (separate installation instructions are provided in the /doc folder in the repo).

The application (CBSA) is a banking system, as used by bank tellers and the application is being changed to add an extra digit onto the existing account number, increasing the length from 8 digits to 9 digits – this allows for future bank expansion.

A CICS region is provided. You will be able to logon and try out the application prior to making any changes (at the start, the account number will be 8 bytes long). The intention is to follow this script which will amend all of the affected maps, programs, and copybooks and then rebuild and try out the changes in the same CICS region – to verify the results.

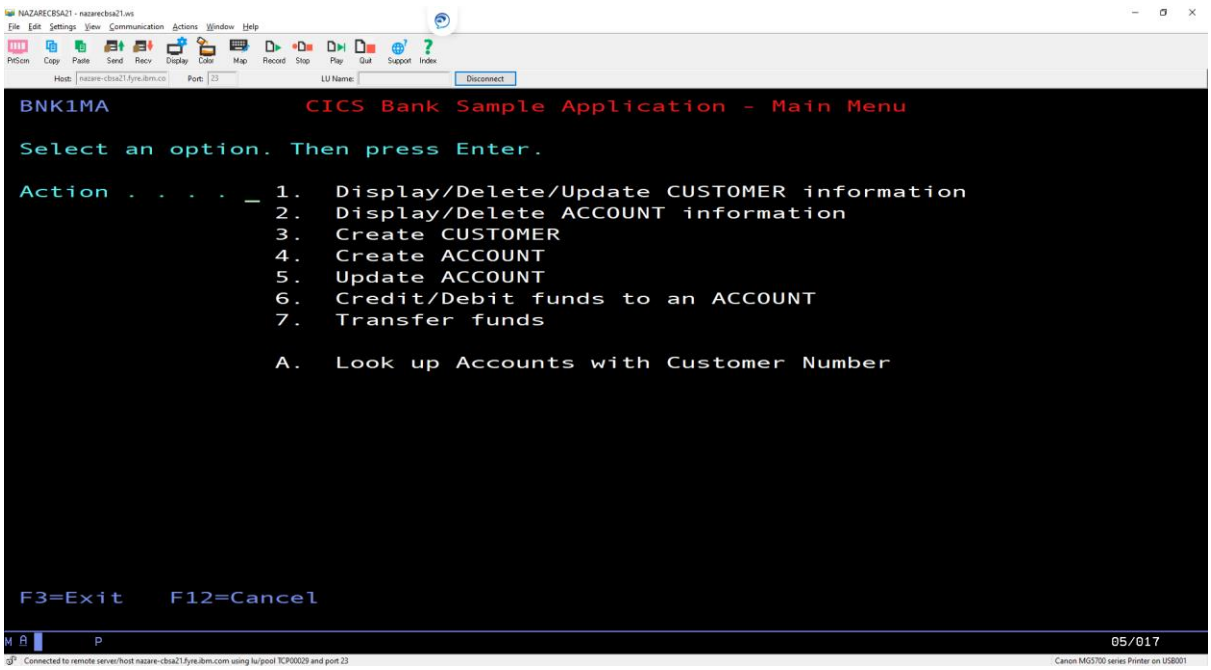
WAZI Developer for z will be the development tool used throughout this scenario.

The account number is a fundamental element of the banking application. In total the changes affect 15 programs, 7 BMS mapsets and a couple of Db2 tables. This exercise focuses on changing:

- A single BMS mapset called **BNK1CAM** which is used to create a new account.
- An associated BMS display verification program called **BNK1CAC**.
- The backend program **CREACC**, which records the new account information on to the underlying datastores.
- Multiple copybooks, which will also need changing along the way.

Explanation of the modules:

The following table of affected artefacts relate to functions found on the BMS main menu:



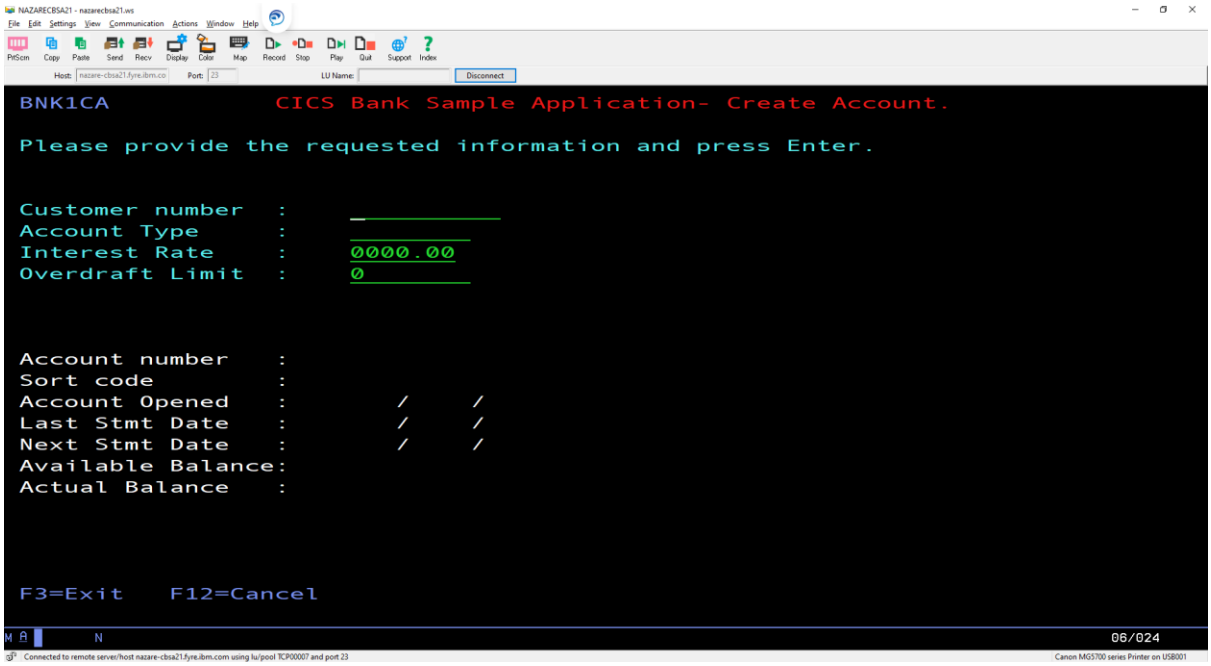
Option on the BMS main menu	***Purpose***	***Mapset and Map affected***	***BMS display program affected***	***Back end program affected***
Option 1 then pF5	Delete Customer	-	-	DELCUS DELACC
Option 2	Display Account	BNK1DAM (mapset) BNK1DA (map)	BNK1DAC	INQACC
Option 2 then pF5	Delete Account	BNK1DAM (mapset) BNK1DA (map)	BNK1DAC	DELACC
Option 4	Create Account	BNK1CAM(mapset) BNK1CA (map)	BNK1CAC	CREACC
Option 5	Update Account	BNK1UAM (mapset) BNK1UA (map)	BNK1UAC	UPDACC
Option 6	Credit/Debit funds (to an account)	BNK1CDM(mapset) BNK1CD (map)	BNK1CRA	DBCRFUN
Option 7	Transfer funds between accounts	BNK1TFM (mapset) BNK1TF (map)	BNK1TFN	XFRFUN
Option A	Look up accounts with customer number	BNK1ACC (mapset) BNK1AC (map)	BNK1CCA	INQACCCU

This exercise concentrates on changing the Create Account functionality (option 4 on the main menu).

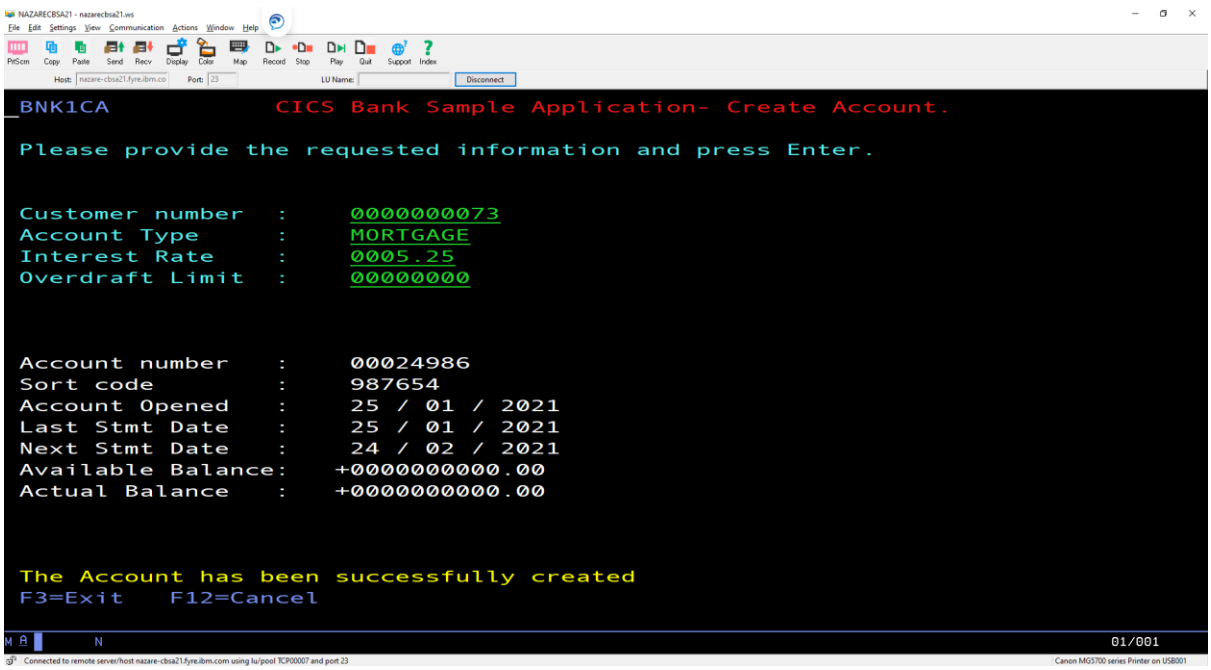
Enacting the changes:

Changing the Create Account BMS map (BNK1CAM)

1. Currently the Create Account option (option 4 from the BMS main menu) looks like this:



The Account number (shown in white but currently empty) is an output attribute and will be generated once the end user has successfully supplied a Customer Number (for the new account to be associated with), along with an Account Type, the Interest Rate and Overdraft Limit information (see below):



Here we see that the generated Account Number (00024986) is 8 bytes long.

2. Edit BMS mapset BNK1CAM from /bms folder in the Git Hub repo.
   Change the length of ACCNO from 8 to 9 bytes as highlighted in yellow (below) and also append one space onto the end of the INITIAL= statement (the INITIAL= string should now be 9 bytes worth of spaces as opposed to 8 bytes worth of spaces):

```
OVERDR    DFHMDF POS=(9,23),LENGTH=8,ATTRB=(NORM,NUM,UNPROT,FSET),      *
               COLOR=GREEN,HILIGHT=UNDERLINE,INITIAL='0',JUSTIFY=RIGHT
          DFHMDF POS=(9,32),LENGTH=1,ATTRB=(PROT,ASKIP)

          DFHMDF POS=(13,1),LENGTH=18,ATTRB=(NORM,PROT),COLOR=NEUTRAL,  *
               INITIAL='Account number   :'
ACCNO     DFHMDF POS=(13,23),LENGTH=9,ATTRB=(NORM,PROT,FSET),           *
             COLOR=NEUTRAL,INITIAL='         '
```

3. **Then save this change** (use cntrl S to save).
   As a result of changing the BMS map, the generated BMS symbolic map/DSECT (also called BNK1CAM) will automatically get changed when we rebuild the BMS map, later on.

Changing the Create Account BMS validation program (BNK1CAC):

4. Program BNK1CAC is responsible for validating the data coming from the Create Account BMS map, and the data from the backend program, being output to the BMS map. Having changed the account number to be 9 bytes on the BMS map, we now need to ensure that any account number data passed into or used by BNK1CAC is also 9 bytes long.

5. Edit COBOL program BNK1CAC from the /cobol folder in the Git Hub repo.
   Change SUBPGM-NUMBER (which passes the ACCOUNT NUMBER to and from the backend program CREACC) and amend the size from PIC 9(8) DISPLAY to PIC 9(9) DISPLAY:

```
01 SUBPGM-PARMS.
   03 SUBPGM-EYECATCHER          PIC X(4).
   03 SUBPGM-CUSTNO              PIC 9(10).
   03 SUBPGM-KEY.
      05 SUBPGM-SORTCODE         PIC 9(6) DISPLAY.
      05 SUBPGM-NUMBER           PIC 9(9) DISPLAY.
   03 SUBPGM-ACC-TYPE            PIC X(8).
      03 SUBPGM-INT-RT              PIC 9(4)V99.

   …
```

   This should be the only change required.
   {SUBPGM-NUMBER is moved to ACCNOO later in the program, but ACCNOO is the output ACCNO attribute from the BNK1CAM map (which we have already changed). So no further changes are required}

6. **Then save this change** (use cntrl S to save).

Changing the Create Account backend program (CREACC):

7. Program CREACC is responsible for creating the new account. It does so by taking data passed into it from program BNK1CAC (see above) and it allocates a new account number from the CONTROL table and writes the new ACCOUNT row onto the ACCOUNT table. If all goes well, CREACC records that a new account has been successfully created on the successfully processed transactions table (aka PROCTRAN table).

   Program CREACC accesses the Db2 ACCOUNT data using an SQL DECLARE held in copybook **ACCDB2.**

   Change the COPYBOOK ACCDB2:
   Edit ACCDB2 in the GitHub /copylib folder and make the following changes highlighted in yellow:

```
EXEC SQL DECLARE ACCOUNT TABLE
   ( ACCOUNT_EYECATCHER            CHAR(4),
     ACCOUNT_CUSTOMER_NUMBER       CHAR(10),
     ACCOUNT_SORTCODE              CHAR(6) NOT NULL,
     ACCOUNT_NUMBER                CHAR(9) NOT NULL,
     ACCOUNT_TYPE                  CHAR(8),
     ACCOUNT_INTEREST_RATE         DECIMAL(4, 2),
     ACCOUNT_OPENED                DATE,
     ACCOUNT_OVERDRAFT_LIMIT       INTEGER,
     ACCOUNT_LAST_STATEMENT        DATE,
     ACCOUNT_NEXT_STATEMENT        DATE,
     ACCOUNT_AVAILABLE_BALANCE     DECIMAL(10, 2),
     ACCOUNT_ACTUAL_BALANCE        DECIMAL(10, 2) )
END-EXEC.
```

   This amends ACCOUNT_NUMBER to be CHAR(9) instead of CHAR(8) .

   Save this change (use cntrl S to save).

   Edit COBOL program CREACC from the /cobol folder in the Git Hub repo.

   We need to change any variables in program CREACC that interact with ACCOUNT_NUMBER.

   In **WRITE-ACCOUNT-DB2 SECTION** the ACCOUNT_NUMBER gets mapped to the Db2 host variable HV-ACCOUNT-ACC-NO (as highlighted below)

```
EXEC SQL
   INSERT INTO ACCOUNT
          (ACCOUNT_EYECATCHER,
           ACCOUNT_CUSTOMER_NUMBER,
           ACCOUNT_SORTCODE,
           ACCOUNT_NUMBER,
           ACCOUNT_TYPE,
           ACCOUNT_INTEREST_RATE,
           ACCOUNT_OPENED,
           ACCOUNT_OVERDRAFT_LIMIT,
           ACCOUNT_LAST_STATEMENT,
           ACCOUNT_NEXT_STATEMENT,
           ACCOUNT_AVAILABLE_BALANCE,
           ACCOUNT_ACTUAL_BALANCE
```

```
            )
        VALUES (:HV-ACCOUNT-EYECATCHER,
                :HV-ACCOUNT-CUST-NO,
                :HV-ACCOUNT-SORTCODE,
                :HV-ACCOUNT-ACC-NO,
                :HV-ACCOUNT-ACC-TYPE,
                :HV-ACCOUNT-INT-RATE,
                :HV-ACCOUNT-OPENED,
                :HV-ACCOUNT-OVERDRAFT-LIM,
                :HV-ACCOUNT-LAST-STMT,
                :HV-ACCOUNT-NEXT-STMT,
                :HV-ACCOUNT-AVAIL-BAL,
                :HV-ACCOUNT-ACTUAL-BAL
               )
        END-EXEC.
```

It is therefore necessary to change the definition of HV-ACCOUNT-ACC-NO to be 9 bytes and change any of the variables that interact with HV-ACCOUNT-ACC-NO to be 9 bytes too.

Amend the HV-ACCOUNT-ACC-NO definition as highlighted below:

```
    …

* Get the ACCOUNT DB2 copybook
     EXEC SQL
        INCLUDE ACCDB2
     END-EXEC.

* ACCOUNT Host variables for DB2
 01 HOST-ACCOUNT-ROW.
    03 HV-ACCOUNT-EYECATCHER          PIC X(4).
    03 HV-ACCOUNT-CUST-NO             PIC X(10).
    03 HV-ACCOUNT-SORTCODE            PIC X(6).
    03 HV-ACCOUNT-ACC-NO              PIC X(9).
    03 HV-ACCOUNT-ACC-TYPE            PIC X(8).
    03 HV-ACCOUNT-INT-RATE            PIC S9(4)V99 COMP-3.
    03 HV-ACCOUNT-OPENED              PIC X(10).
    03 HV-ACCOUNT-OPENED-GROUP REDEFINES HV-ACCOUNT-OPENED.
       05 HV-ACCOUNT-OPENED-DAY       PIC XX.
       05 HV-ACCOUNT-OPENED-DELIM1    PIC X.
       05 HV-ACCOUNT-OPENED-MONTH     PIC XX.
       05 HV-ACCOUNT-OPENED-DELIM2    PIC X.
       05 HV-ACCOUNT-OPENED-YEAR      PIC X(4).
    03 HV-ACCOUNT-OVERDRAFT-LIM       PIC S9(9) COMP.
    03 HV-ACCOUNT-LAST-STMT           PIC X(10).
    03 HV-ACCOUNT-LAST-STMT-GROUP
    …
```

HV-ACCOUNT-ACC-NO is used in the **WRITE-ACCOUNT-DB2 SECTION** where the content of variable NCS-ACC-NO-DISP(9:8) is moved into HV-ACCOUNT-CUST-NO.

This needs to be changed to be NCS-ACC-NO-DISP (8:9) (as highlighted below):

```
    WRITE-ACCOUNT-DB2 SECTION.

    WAD010.
```

```
       INITIALIZE HOST-ACCOUNT-ROW.
       MOVE 'ACCT' TO HV-ACCOUNT-EYECATCHER.
       MOVE COMM-CUSTNO IN DFHCOMMAREA  TO HV-ACCOUNT-CUST-NO.
       MOVE SORTCODE  TO HV-ACCOUNT-SORTCODE.

       MOVE NCS-ACC-NO-VALUE TO NCS-ACC-NO-DISP.
       MOVE NCS-ACC-NO-DISP(8:9) TO HV-ACCOUNT-ACC-NO.
       …
```

As NCS-ACC-NO-VALUE and NCS-ACC-NO-DISP are both defined as PIC 9(16) COMP and PIC 9(16) respectively, this is sufficiently large to hold the (now) 9 byte account number and so now further action is required with these variables.

Other places which use the HV-ACCOUNT-ACC-NO are a bit further on in the same section, just after the SQL INSERT:

```
       …
      *
      *    If the INSERT was successful the WRITE to PROCTRAN datastore
      *
           MOVE HV-ACCOUNT-SORTCODE       TO STORED-SORTCODE.
           MOVE HV-ACCOUNT-ACC-NO         TO STORED-ACCNO.
           MOVE HV-ACCOUNT-CUST-NO        TO STORED-CUSTNO.
           MOVE HV-ACCOUNT-ACC-TYPE       TO STORED-ACCTYPE.
           MOVE HV-ACCOUNT-LAST-STMT(1:2) TO STORED-LST-STMT(1:2).
           MOVE HV-ACCOUNT-LAST-STMT(4:2) TO STORED-LST-STMT(3:2).
           MOVE HV-ACCOUNT-LAST-STMT(7:4) TO STORED-LST-STMT(5:4).
           MOVE HV-ACCOUNT-NEXT-STMT(1:2) TO STORED-NXT-STMT(1:2).
           MOVE HV-ACCOUNT-NEXT-STMT(4:2) TO STORED-NXT-STMT(3:2).
           MOVE HV-ACCOUNT-NEXT-STMT(7:4) TO STORED-NXT-STMT(5:4).

           PERFORM WRITE-PROCTRAN.
           …
```

Change the STORED-ACCNO definition from 8 bytes to 9 bytes as follows:

```
       01 STORED-ACCNO                      PIC X(9)  VALUE SPACES.
```

Because we have changed STORED-ACCNO we must also see where it is used. It is used in the WRITE-PROCTRAN-DB2 SECTION:

```
       WRITE-PROCTRAN-DB2 SECTION.
       WPD010.
      *
      *    Write the successfully processed transaction to the PROCTRAN
      *    table.
      *
           INITIALIZE HOST-PROCTRAN-ROW.
           INITIALIZE WS-EIBTASKN12.

           MOVE 'PRTR'   TO HV-PROCTRAN-EYECATCHER.
           MOVE SORTCODE TO HV-PROCTRAN-SORT-CODE.
           MOVE STORED-ACCNO TO HV-PROCTRAN-ACC-NUMBER.
           MOVE EIBTASKN TO WS-EIBTASKN12.
           MOVE WS-EIBTASKN12 TO HV-PROCTRAN-REF.
```

```
*
*    Populate the time and date
*
     …
```

Change the HV-PROCTRAN-ACC-NUMBER definition from 8 bytes to 9 bytes as highlighted:

```
* PROCTRAN host variables for DB2
 01 HOST-PROCTRAN-ROW.
    03 HV-PROCTRAN-EYECATCHER         PIC X(4).
    03 HV-PROCTRAN-SORT-CODE          PIC X(6).
    03 HV-PROCTRAN-ACC-NUMBER         PIC X(9).
    03 HV-PROCTRAN-DATE               PIC X(10).
    03 HV-PROCTRAN-TIME               PIC X(6).
    03 HV-PROCTRAN-REF                PIC X(12).
    03 HV-PROCTRAN-TYPE               PIC X(3).
    03 HV-PROCTRAN-DESC               PIC X(40).
    03 HV-PROCTRAN-AMOUNT             PIC S9(10)V99 COMP-3.
```

The only other variable left, that interacts with HV-ACCOUNT-ACC-NO is the variable COMM-NUMBER (see below):

```
 …
*
*    Set up the missing data in the COMMAREA ready for return
*
     MOVE HV-ACCOUNT-SORTCODE    TO COMM-SORTCODE.
     MOVE HV-ACCOUNT-ACC-NO      TO COMM-NUMBER.

     MOVE HV-ACCOUNT-OPENED-DAY(1:2)
       TO COMM-OPENED IN DFHCOMMAREA(1:2).
        MOVE HV-ACCOUNT-OPENED-MONTH(1:2)
     …
```

Save all of the previous changes that have been made so far to the CREACC program now (Cntrl S).

COMM-NUMBER is used in the COMM area which utilises COPYBOOK CREACC:

Change the COPYBOOK CREACC:
Edit CREACC in the GitHub /copylib folder, and make the following changes highlighted in yellow:

```
    03 COMM-EYECATCHER                PIC X(4).
    03 COMM-CUSTNO                    PIC 9(10).
    03 COMM-KEY.
       05 COMM-SORTCODE                PIC 9(6) DISPLAY.
       05 COMM-NUMBER                  PIC 9(9) DISPLAY.
    03 COMM-ACC-TYPE                   PIC X(8).
    03 COMM-INT-RT                     PIC 9(4)V99.
    03 COMM-OPENED                     PIC 9(8).
    03 COMM-OPENED-GROUP REDEFINES COMM-OPENED.
       05 COMM-OPENED-DAY              PIC 99.
       05 COMM-OPENED-MONTH            PIC 99.
       05 COMM-OPENED-YEAR             PIC 9999.
    03 COMM-OVERDR-LIM                 PIC 9(8).
```

```
03 COMM-LAST-STMT-DT               PIC 9(8).
03 COMM-LAST-STMNT-GROUP REDEFINES COMM-LAST-STMT-DT.
   05 COMM-LASTST-DAY                PIC 99.
   05 COMM-LASTST-MONTH              PIC 99.
   05 COMM-LASTST-YEAR               PIC 9999.
03 COMM-NEXT-STMT-DT               PIC 9(8).
03 COMM-NEXT-STMNT-GROUP REDEFINES COMM-NEXT-STMT-DT.
   05 COMM-NEXTST-DAY                PIC 99.
   05 COMM-NEXTST-MONTH              PIC 99.
   05 COMM-NEXTST-YEAR               PIC 9999.
03 COMM-AVAIL-BAL                  PIC S9(10)V99.
03 COMM-ACT-BAL                    PIC S9(10)V99.
03 COMM-SUCCESS                    PIC X.
03 COMM-FAIL-CODE                  PIC X.
```

Save the changes to the copybook (Cntrl S).

Back in program **CREACC**, we need to investigate the remaining places where COMM-NUMBER is used - there is only one other place:

```
MOVE NCS-ACC-NO-VALUE TO
COMM-NUMBER ACCOUNT-NUMBER REQUIRED-ACCT-NUMBER3.
    MOVE SPACES TO HV-CONTROL-NAME
```

Here COMM-NUMBER gets a value moved into it from the variable NCS-ACC-NO-VALUE – this variable is defined as PIC 9(16) COMP, therefore it is plenty big enough to hold a (now) 9 byte account number. So no further changes are required.

8. Program CREACC accesses the Db2 table PROCTRAN data using the SQL DECLARE held in copybook **PROCDB2**.

Change the COPYBOOK PROCDB2:
Edit PROCDB2 in the GitHub /copylib folder, and make the following changes highlighted in yellow:

```
EXEC SQL DECLARE PROCTRAN TABLE
   (
    PROCTRAN_EYECATCHER             CHAR(4),
    PROCTRAN_SORTCODE               CHAR(6) NOT NULL,
    PROCTRAN_NUMBER                 CHAR(9) NOT NULL,
    PROCTRAN_DATE                   CHAR(8),
    PROCTRAN_TIME                   CHAR(6),
    PROCTRAN_REF                    CHAR(12),
    PROCTRAN_TYPE                   CHAR(3),
    PROCTRAN_DESC                   CHAR(40),
    PROCTRAN_AMOUNT                 DECIMAL(12, 2)
   )
END-EXEC.
```

This amends PROCTRAN_NUMBER to be CHAR(9) instead of CHAR(8).

Save this change (use cntrl S to save).

Now we need to also change any variables in program **CREACC** that interact with PROCTRAN_NUMBER.

In **WRITE-PROCTRAN-DB2 SECTION** we see that PROCTRAN_NUMBER is mapped to the Db2 host variable HV-ACCOUNT-ACC-NO (as highlighted below)

```
MOVE 0                  TO HV-PROCTRAN-AMOUNT.

       EXEC SQL
          INSERT INTO PROCTRAN
                 (
                  PROCTRAN_EYECATCHER,
                  PROCTRAN_SORTCODE,
                  PROCTRAN_NUMBER,
                  PROCTRAN_DATE,
                  PROCTRAN_TIME,
                  PROCTRAN_REF,
                  PROCTRAN_TYPE,
                  PROCTRAN_DESC,
                  PROCTRAN_AMOUNT
                 )
          VALUES
                 (
                  :HV-PROCTRAN-EYECATCHER,
                  :HV-PROCTRAN-SORT-CODE,
                  :HV-PROCTRAN-ACC-NUMBER,
                  :HV-PROCTRAN-DATE,
                  :HV-PROCTRAN-TIME,
                  :HV-PROCTRAN-REF,
                  :HV-PROCTRAN-TYPE,
                  :HV-PROCTRAN-DESC,
                  :HV-PROCTRAN-AMOUNT
                 )
       END-EXEC.
```

We have already changed the definition of HV-PROCTRAN-ACC-NUMBER to be 9 bytes previously.

`HV-PROCTRAN-ACC-NUMBER` has data moved into it from variable STORED-ACCNO:

```
…
MOVE STORED-ACCNO TO HV-PROCTRAN-ACC-NUMBER.
…
```

STORED-ACCNO was changed previously, so no further changes are required.

Save this change (use cntrl S to save).

9. Program CREACC uses the COPYBOOK PROCTRAN to breakdown the PROCTRAN-DESC field.

Change the COPYBOOK PROCTRAN:
Edit PROCTRAN in the GitHub /copylib folder, and make the following changes highlighted in yellow:

```
03 PROC-TRAN-DATA.
   05 PROC-TRAN-EYE-CATCHER        PIC X(4).
   88 PROC-TRAN-VALID VALUE 'PRTR'.
   05 PROC-TRAN-LOGICAL-DELETE-AREA REDEFINES
      PROC-TRAN-EYE-CATCHER.
      07 PROC-TRAN-LOGICAL-DELETE-FLAG PIC X.
      88 PROC-TRAN-LOGICALLY-DELETED VALUE X'FF'.
      07 FILLER PIC X(3).
   05 PROC-TRAN-ID.
      07 PROC-TRAN-SORT-CODE       PIC 9(6).
      07 PROC-TRAN-NUMBER          PIC 9(9).
   05 PROC-TRAN-DATE               PIC 9(8).
   05 PROC-TRAN-DATE-GRP REDEFINES PROC-TRAN-DATE.
      07 PROC-TRAN-DATE-GRP-YYYY   PIC 9999.
      07 PROC-TRAN-DATE-GRP-MM     PIC 99.
      07 PROC-TRAN-DATE-GRP-DD     PIC 99.
   05 PROC-TRAN-TIME               PIC 9(6).
   05 PROC-TRAN-TIME-GRP REDEFINES PROC-TRAN-TIME.
      07 PROC-TRAN-TIME-GRP-HH     PIC 99.
      07 PROC-TRAN-TIME-GRP-MM     PIC 99.
      07 PROC-TRAN-TIME-GRP-SS     PIC 99.
   05 PROC-TRAN-REF                PIC 9(12).
   05 PROC-TRAN-TYPE               PIC X(3).
   88 PROC-TY-CHEQUE-ACKNOWLEDGED      VALUE 'CHA'.
   88 PROC-TY-CHEQUE-FAILURE           VALUE 'CHF'.
   88 PROC-TY-CHEQUE-PAID-IN           VALUE 'CHI'.
   88 PROC-TY-CHEQUE-PAID-OUT          VALUE 'CHO'.
   88 PROC-TY-CREDIT                   VALUE 'CRE'.
   88 PROC-TY-DEBIT                    VALUE 'DEB'.
   88 PROC-TY-WEB-CREATE-ACCOUNT       VALUE 'ICA'.
   88 PROC-TY-WEB-CREATE-CUSTOMER      VALUE 'ICC'.
   88 PROC-TY-WEB-DELETE-ACCOUNT       VALUE 'IDA'.
   88 PROC-TY-WEB-DELETE-CUSTOMER      VALUE 'IDC'.
   88 PROC-TY-BRANCH-CREATE-ACCOUNT    VALUE 'OCA'.
   88 PROC-TY-BRANCH-CREATE-CUSTOMER   VALUE 'OCC'.
   88 PROC-TY-BRANCH-DELETE-ACCOUNT    VALUE 'ODA'.
   88 PROC-TY-BRANCH-DELETE-CUSTOMER   VALUE 'ODC'.
   88 PROC-TY-CREATE-SODD              VALUE 'OCS'.
   88 PROC-TY-TRANSFER                 VALUE 'TFR'.
   05 PROC-TRAN-DESC               PIC X(40).
   05 PROC-TRAN-DESC-XFR REDEFINES PROC-TRAN-DESC.
     07 PROC-TRAN-DESC-XFR-HEADER PIC X(25).
     88 PROC-TRAN-DESC-XFR-FLAG
        VALUE 'TRANSFER'.
     07 PROC-TRAN-DESC-XFR-SORTCODE
        PIC 9(6).
     07 PROC-TRAN-DESC-XFR-ACCOUNT
        PIC 9(9).
   05 PROC-TRAN-DESC-DELACC REDEFINES PROC-TRAN-DESC.
     07 PROC-DESC-DELACC-CUSTOMER PIC 9(10).
     07 PROC-DESC-DELACC-ACCTYPE PIC X(8).
     07 PROC-DESC-DELACC-LAST-DD PIC 99.
     07 PROC-DESC-DELACC-LAST-MM PIC 99.
     07 PROC-DESC-DELACC-LAST-YYYY PIC 9999.
     07 PROC-DESC-DELACC-NEXT-DD PIC 99.
     07 PROC-DESC-DELACC-NEXT-MM PIC 99.
     07 PROC-DESC-DELACC-NEXT-YYYY PIC 9999.
     07 PROC-DESC-DELACC-FOOTER PIC X(6).
     88 PROC-DESC-DELACC-FLAG
```

```
           VALUE 'DELETE'.
       05 PROC-TRAN-DESC-CREACC REDEFINES PROC-TRAN-DESC.
         07 PROC-DESC-CREACC-CUSTOMER PIC 9(10).
         07 PROC-DESC-CREACC-ACCTYPE PIC X(8).
         07 PROC-DESC-CREACC-LAST-DD PIC 99.
         07 PROC-DESC-CREACC-LAST-MM PIC 99.
         07 PROC-DESC-CREACC-LAST-YYYY PIC 9999.
         07 PROC-DESC-CREACC-NEXT-DD PIC 99.
         07 PROC-DESC-CREACC-NEXT-MM PIC 99.
         07 PROC-DESC-CREACC-NEXT-YYYY PIC 9999.
         07 PROC-DESC-CREACC-FOOTER PIC X(6).
         88 PROC-DESC-CREACC-FLAG
           VALUE 'CREATE'.
       05 PROC-TRAN-DESC-DELCUS REDEFINES PROC-TRAN-DESC.
         07 PROC-DESC-DELCUS-SORTCODE PIC 9(6).
         07 PROC-DESC-DELCUS-CUSTOMER PIC 9(10).
         07 PROC-DESC-DELCUS-NAME     PIC X(14).
         07 PROC-DESC-DELCUS-DOB-YYYY  PIC 9999.
         07 PROC-DESC-DELCUS-FILLER    PIC X.
         88 PROC-DESC-DELCUS-FILLER-SET VALUE '-'.
         07 PROC-DESC-DELCUS-DOB-MM    PIC 99.
         07 PROC-DESC-DELCUS-FILLER2   PIC X.
         88 PROC-DESC-DELCUS-FILLER2-SET VALUE '-'.
         07 PROC-DESC-DELCUS-DOB-DD    PIC 99.
       05 PROC-TRAN-DESC-CRECUS REDEFINES PROC-TRAN-DESC.
         07 PROC-DESC-CRECUS-SORTCODE PIC 9(6).
         07 PROC-DESC-CRECUS-CUSTOMER PIC 9(10).
         07 PROC-DESC-CRECUS-NAME     PIC X(14).
         07 PROC-DESC-CRECUS-DOB-YYYY  PIC 9999.
         07 PROC-DESC-CRECUS-FILLER    PIC X.
         88 PROC-DESC-CRECUS-FILLER-SET VALUE '-'.
         07 PROC-DESC-CRECUS-DOB-MM    PIC 99.
         07 PROC-DESC-CRECUS-FILLER2   PIC X.
         88 PROC-DESC-CRECUS-FILLER2-SET VALUE '-'.
         07 PROC-DESC-CRECUS-DOB-DD    PIC 99.
       05 PROC-TRAN-AMOUNT             PIC S9(10)V99.
```

This makes the account number held in `PROC-TRAN-NUMBER` now 9 bytes long. We have also changed the account number held in `PROC-TRAN-DESC-XFR-ACCOUNT to be 9` bytes long, but to accommodate it we have reduced the length of `PROC-TRAN-DESC-XFR-HEADER` from 26 to 25 (this latter change is utilised in account transfers).

Save these changes (Cntrl S).

These 3 variables (`PROC-TRAN-NUMBER, PROC-TRAN-DESC-XFR-HEADER and PROC-TRAN-DESC-XFR-ACCOUNT`) are used later (by other programs) so no further changes to CREACC are necessary.

10. Program CREACC uses the COPYBOOK ACCOUNT.

***Change the COPYBOOK ACCOUNT**:*
Edit ACCOUNT in the GitHub /copylib folder, and make the following changes highlighted in yellow:

```
03 ACCOUNT-DATA.
   05 ACCOUNT-EYE-CATCHER        PIC X(4).
   88 ACCOUNT-EYECATCHER-VALUE        VALUE 'ACCT'.
   05 ACCOUNT-CUST-NO            PIC 9(10).
   05 ACCOUNT-KEY.
      07 ACCOUNT-SORT-CODE       PIC 9(6).
      07 ACCOUNT-NUMBER          PIC 9(9).
   05 ACCOUNT-TYPE               PIC X(8).
   05 ACCOUNT-INTEREST-RATE      PIC 9(4)V99.
   05 ACCOUNT-OPENED             PIC 9(8).
   05 ACCOUNT-OPENED-GROUP REDEFINES ACCOUNT-OPENED.
      07 ACCOUNT-OPENED-DAY       PIC 99.
      07 ACCOUNT-OPENED-MONTH     PIC 99.
      07 ACCOUNT-OPENED-YEAR      PIC 9999.
   05 ACCOUNT-OVERDRAFT-LIMIT    PIC 9(8).
   05 ACCOUNT-LAST-STMT-DATE     PIC 9(8).
   05 ACCOUNT-LAST-STMT-GROUP
      REDEFINES ACCOUNT-LAST-STMT-DATE.
      07 ACCOUNT-LAST-STMT-DAY   PIC 99.
      07 ACCOUNT-LAST-STMT-MONTH PIC 99.
      07 ACCOUNT-LAST-STMT-YEAR  PIC 9999.
   05 ACCOUNT-NEXT-STMT-DATE     PIC 9(8).
   05 ACCOUNT-NEXT-STMT-GROUP
     REDEFINES ACCOUNT-NEXT-STMT-DATE.
      07 ACCOUNT-NEXT-STMT-DAY   PIC 99.
      07 ACCOUNT-NEXT-STMT-MONTH PIC 99.
      07 ACCOUNT-NEXT-STMT-YEAR  PIC 9999.
   05 ACCOUNT-AVAILABLE-BALANCE  PIC S9(10)V99.
      05 ACCOUNT-ACTUAL-BALANCE     PIC S9(10)V99.
```

This makes the account number held in `ACCOUNT-NUMBER` now 9 bytes long.
All of the places in the code where data is moved to or moved from ACCOUNT-NUMBER have been changed previously (so no further changes are required).

Save this change (Cntrl S).

11. Program CREACC uses the COPYBOOK INQACCCU.

Change the COPYBOOK INQACCCU:
Edit INQACCCU in the GitHub /copylib folder, and make the following changes highlighted in yellow:

```
03 NUMBER-OF-ACCOUNTS        PIC S9(8) BINARY.
03 CUSTOMER-NUMBER           PIC 9(10).
03 COMM-SUCCESS              PIC X.
03 COMM-FAIL-CODE            PIC X.
03 CUSTOMER-FOUND            PIC X.
03 COMM-PCB-POINTER          POINTER.
03 ACCOUNT-DETAILS OCCURS 1 TO 20 DEPENDING ON
    NUMBER-OF-ACCOUNTS.
  05 COMM-EYE                  PIC X(4).
  05 COMM-CUSTNO               PIC X(10).
  05 COMM-SCODE                PIC X(6).
  05 COMM-ACCNO                PIC 9(9).
  05 COMM-ACC-TYPE             PIC X(8).
  05 COMM-INT-RATE             PIC 9(6).
```

```
05 COMM-OPENED              PIC 9(8).
05 COMM-OPENED-GROUP REDEFINES COMM-OPENED.
  07 COMM-OPENED-DAY PIC 99.
  07 COMM-OPENED-MONTH PIC 99.
  07 COMM-OPENED-YEAR PIC 9999.
05 COMM-OVERDRAFT            PIC 9(8).
05 COMM-LAST-STMT-DT         PIC 9(8).
05 COMM-LAST-STMT-GROUP REDEFINES COMM-LAST-STMT-DT.
  07 COMM-LAST-STMT-DAY PIC 99.
  07 COMM-LAST-STMT-MONTH PIC 99.
  07 COMM-LAST-STMT-YEAR PIC 9999.
05 COMM-NEXT-STMT-DT         PIC 9(8).
05 COMM-NEXT-STMT-GROUP REDEFINES COMM-NEXT-STMT-DT.
  07 COMM-NEXT-STMT-DAY PIC 99.
  07 COMM-NEXT-STMT-MONTH PIC 99.
  07 COMM-NEXT-STMT-YEAR PIC 9999.
05 COMM-AVAIL-BAL            PIC S9(10)V99.
05 COMM-ACTUAL-BAL           PIC S9(10)V99.
```

This makes the account number held in `COMM-ACCNO` now 9 bytes long.

Save this change (Cntrl S).

12. Program CREACC uses the COPYBOOK ACCTCTRL.

Change the COPYBOOK ACCTCTRL:
Edit ACCTCTRL in the GitHub /copylib folder, and make the following changes highlighted in yellow:

```
03 ACCOUNT-CONTROL-RECORD.
  05 ACCOUNT-CONTROL-EYE-CATCHER        PIC X(4).
  88 ACCOUNT-CONTROL-EYECATCHER-V   VALUE 'CTRL'.
  05 FILLER                             PIC 9(10).
  05 ACCOUNT-CONTROL-KEY.
     07 ACCOUNT-CONTROL-SORT-CODE       PIC 9(6).
     07 ACCOUNT-CONTROL-NUMBER          PIC 9(9).
  05 NUMBER-OF-ACCOUNTS                 PIC 9(9).
  05 LAST-ACCOUNT-NUMBER                PIC 9(9).
  05 ACCOUNT-CONTROL-SUCCESS-FLAG       PIC X.
  88 ACCOUNT-CONTROL-SUCCESS VALUE 'Y'.
  05 ACCOUNT-CONTROL-FAIL-CODE PIC X.
  05 FILLER                      PIC 9(4)V99.
  05 FILLER                      PIC 9(8).
  05 FILLER                      PIC 9(8).
  05 FILLER                      PIC 9(8).
  05 FILLER                      PIC 9(8).
  05 FILLER                      PIC S9(10)V99.
  05 FILLER                      PIC X(2).
```

This is the mechanism used to store the highest account number in use, so `ACCOUNT-CONTROL-NUMBER`, `NUMBER-OF-ACCOUNTS` and `LAST-ACCOUNT-NUMBER` all now become 9 bytes long.

Save these changes (Cntrl S).

Changing the remaining Programs, Maps and Copybooks:

As you might expect, there are many more programs, copybooks and BMS maps affected by altering the account number from 8 to 9 bytes. In this exercise you have been asked to change the Create Account related programs, copybooks and BMS map. For the sake of expediency, the changes to all of the remaining source code have been made for you. You can, should you wish to, see **Appendix A** where changes to the remaining source are fully documented (see page 44).

To expedite things, you don't need to make the remaining programs changes yourself, but you will need to follow the procedure, outlined below, to ensure that everything required gets changed:

The remaining BMS maps:

a. Go into z/OS Projects view – this is probably already open but in case it isn't, you can open it by selecting Window/Show View/zOS Projects. Go into Git Hub /bms folder and open the BNK1DAM source (click the source tab at the bottom).
b. Do a Cntrl A (to select all of the content) and click delete, to delete all of the lines of source code.
c. Next, go into the /bms_after Git Hub folder and open the **BNK1DAM.bms** source in there.
d. Do a Cntrl A (to select all of the content) and paste it (use Cntrl V to paste) in to **BNK1DAM.bms** source in the /bms folder.
e. Then use (Cntrl S) to save this change (note if there is still an asterisk against BNK1DAM.bms then the save did not work, so you will need to close the window and select 'Save' when prompted).
f. Now repeat a-e (above) substituting the content of the /bms_after map into the equivalent map in the /bms folder. Do this for these remaining BMS maps: **BNK1UAM, BNK1CDM, BNK1TFM, BNK1ACC.**

The remaining copybooks:

g. In the z/OS Project views, go into Git Hub /copylib folder and open the **INQACC.cpy** copylib source.
h. Do a Cntrl A (to select all of the content) and click delete, to delete all of the lines of source code.
i. Next, go into the /copylib_after Git Hub folder and open the INQACC source in there.
j. Do a Cntrl A (to select the content) and paste it into the **INQACC.cpy** source in the /copylib folder. Use Cntrl V to paste.
k. Then save (Cntrl S) this change.
l. Now repeat g-k (above) for the remaining copybooks: **DELACC, UPDACC, XFRFUN.**

The remaining programs:

m. Go into Git Hub /cobol folder and open the **BNK1DAC** cobol source.
n. Do a Cntrl A (to select all of the content) and click delete, to delete all of the lines of source code.

o. Next, go into the /cobol_after Git Hub folder and open the **BNK1DAC.cbl** source in there.
p. Do a Cntrl A (to select the content) and paste it into the BNK1DAC source in the /cobol folder. Use Cntrl V to paste.
q. Then save (Cntrl S) this change.
r. Now repeat m-q (above) for the remaining programs:
INQACC, DELACC, BNK1UAC, BNK1CRA, DBCRFUN, BNK1TFN, XFRFUN, BNK1CCA, INQACCCU, DELCUS, CRECUST, UPDACC.

Backing up the data on the ACCOUNT and PROCTRAN Db2 tables, making the Db2 table changes and reloading the affected tables:

Having made all of the BMS, program and copybook changes to make the account number 9 bytes, we now need to alter the Db2 ACCOUNT and PROCTRAN tables to also make these reflect a 9 byte ACCOUNT number too.

The following jobs need to be submitted in the sequence listed below:

A. Offload ACCOUNT table data, change the ACCOUNT table, reload the ACCOUNT table data:

1. Submit Job CBSA.CICSBSA.REORG(TMPACCF). This job off-loads the ACCOUNT table's data to a VSAM dataset (the account numbers are all 8 bytes long at this point) called CBSA.CICSBSA.TMPAFF. It should give a RC=0.

2. Submit CBSA.CICSBSA.REORG(DB2REDA). This job drops/removes all of the ACCOUNT related artefacts e.g. Db2 table, indexes, tablespace, and storage group. It should give a RC=0.

3. Submit CBSA.CICSBSA.REORG(DB2REDB). This job recreates the ACCOUNT table, indexes, tablespace, and storage group, but now with the ACCOUNT NUMBER at 9 bytes in length. It should give a RC=0.

4. Submit CBSA.CICSBSA.REORG(DB2BIND). This rebinds the programs against the newly changed ACCOUNT table. This should give a RC of 0 in the BIND step and a RC=8 in the GRANT step.

5. Submit CBSA.CICSBSA.REORG(TMPACCL). This takes the offloaded ACCOUNT data, reads in a record at a time, adds an additional leading '0' to the account number and INSERTs the record onto the newly revised ACCOUNT table. It should give a RC=0.

B. Offload PROCTRAN table data, change the PROCTRAN table, reload the PROCTRAN table data:

6. Submit Job CBSA.CICSBSA.REORG(TMPPROF). This job off-loads the PROCTRAN table's data to a VSAM dataset (with the account numbers all 8 bytes long at this point) called CBSA.CICSBSA.TMPPFF. It should give a RC=0.

7. Submit CBSA.CICSBSA.REORG(DB2REDC). This job drops/removes the PROCTRAN related Db2 table, tablespace, and storage group. It should give a RC=0.

8. Submit CBSA.CICSBSA.REORG(DB2REDD). This job recreates the PROCTRAN table, tablespace, and storage group, with the ACCOUNT NUMBER at 9 bytes in length. It should give a RC=0.

9. Submit CBSA.CICSBSA.REORG(DB2BIND). This rebinds the programs against the newly changed PROCTRAN table. This should give a RC of 0 in the BIND step and a RC=8 in the GRANT step.

10. Submit CBSA.CICSBSA.REORG(TMPPROL). This takes the offloaded PROCTRAN data, reads in a record at a time, adds an additional leading '0' to the account number (making the account

number 9 bytes long) and INSERTs the record onto the newly revised PROCTRAN table. It should give a RC=0.

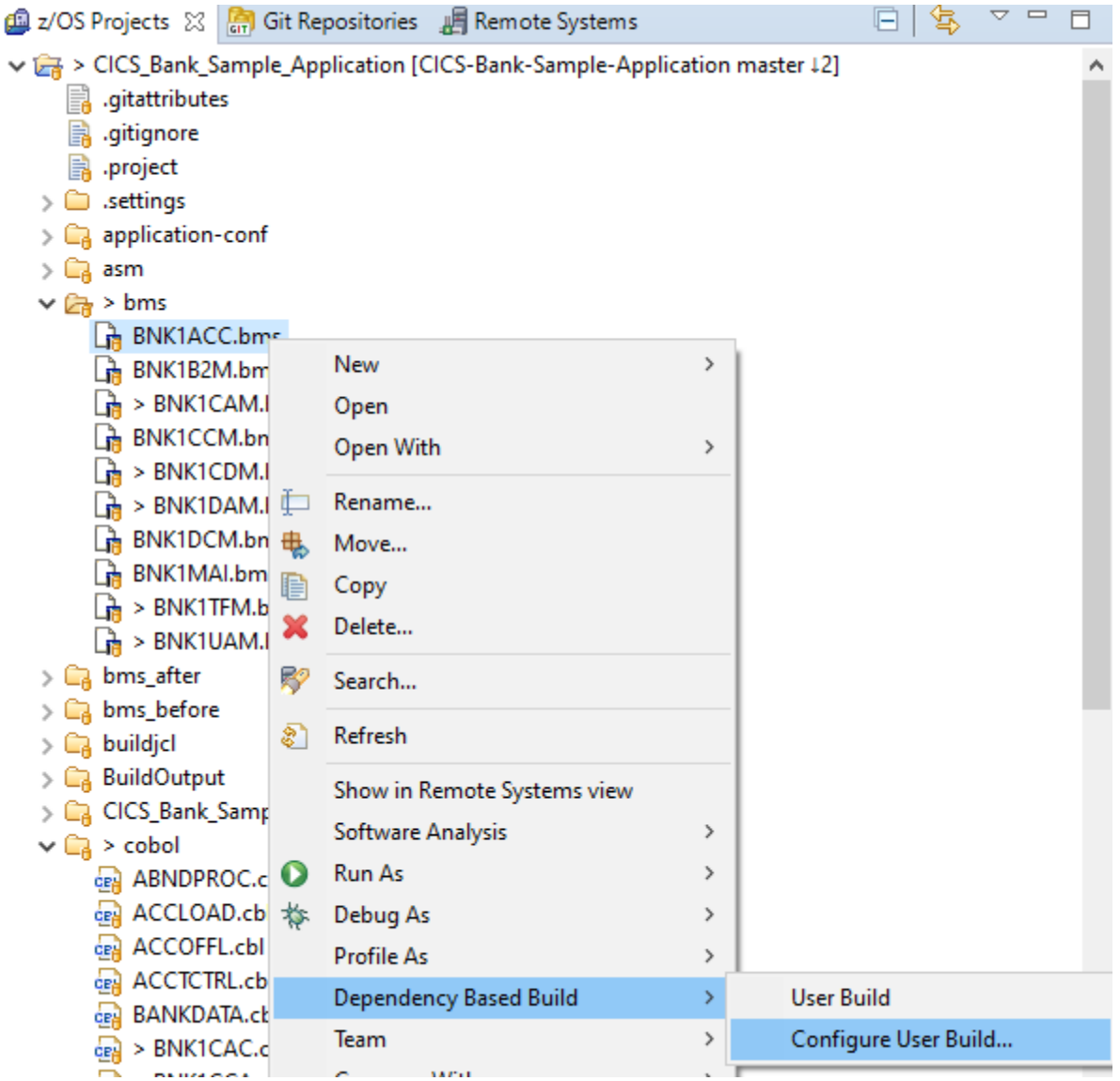
Rebuilding it all:

Rebuild the BMS maps:

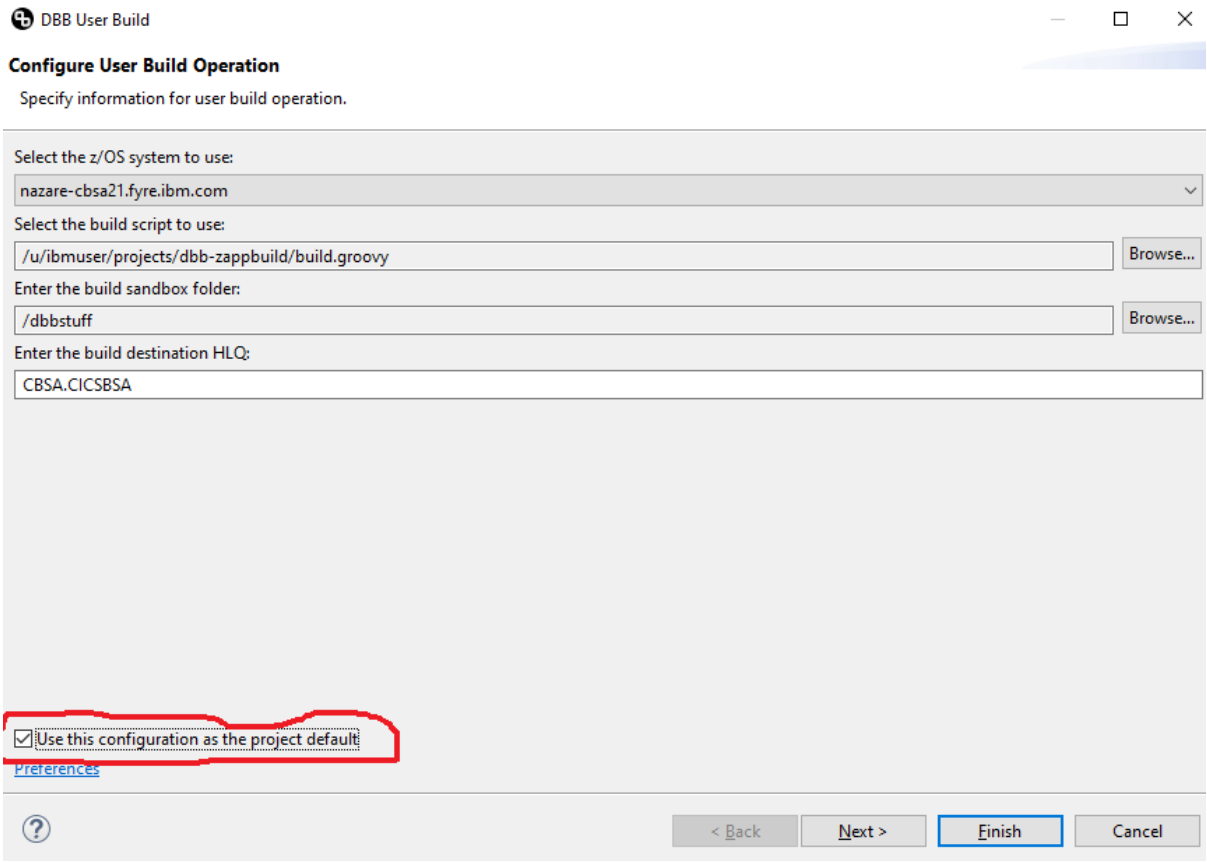
13. Having made all of the source code changes, and the Db2 changes the affected source code needs to be reassembled/recompiled. Here is the procedure:

    I. Go into the z/OS Projects view in WAZI Developer – this is probably already open but in case it isn't, you can open it by selecting Window/Show View/zOS Projects

    II. Open the /bms folder (which now contains all of the amended BMS maps) and right click on BMS map you wish to build (in this case **BNK1ACC.bms**, the first map in the list). Then select 'Dependency Based Build' and 'Configure User Build':



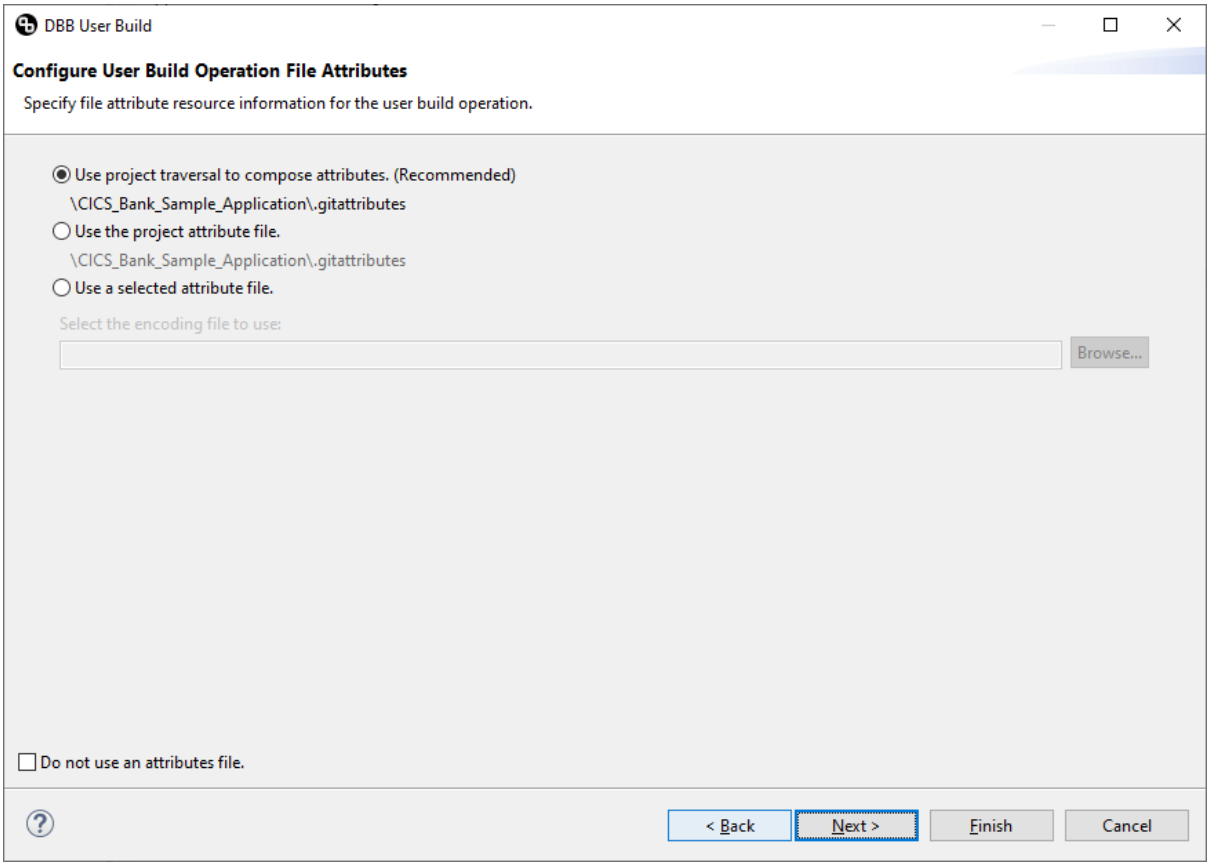
    III. On the next screen click the 'Use this configuration as the project default' box:



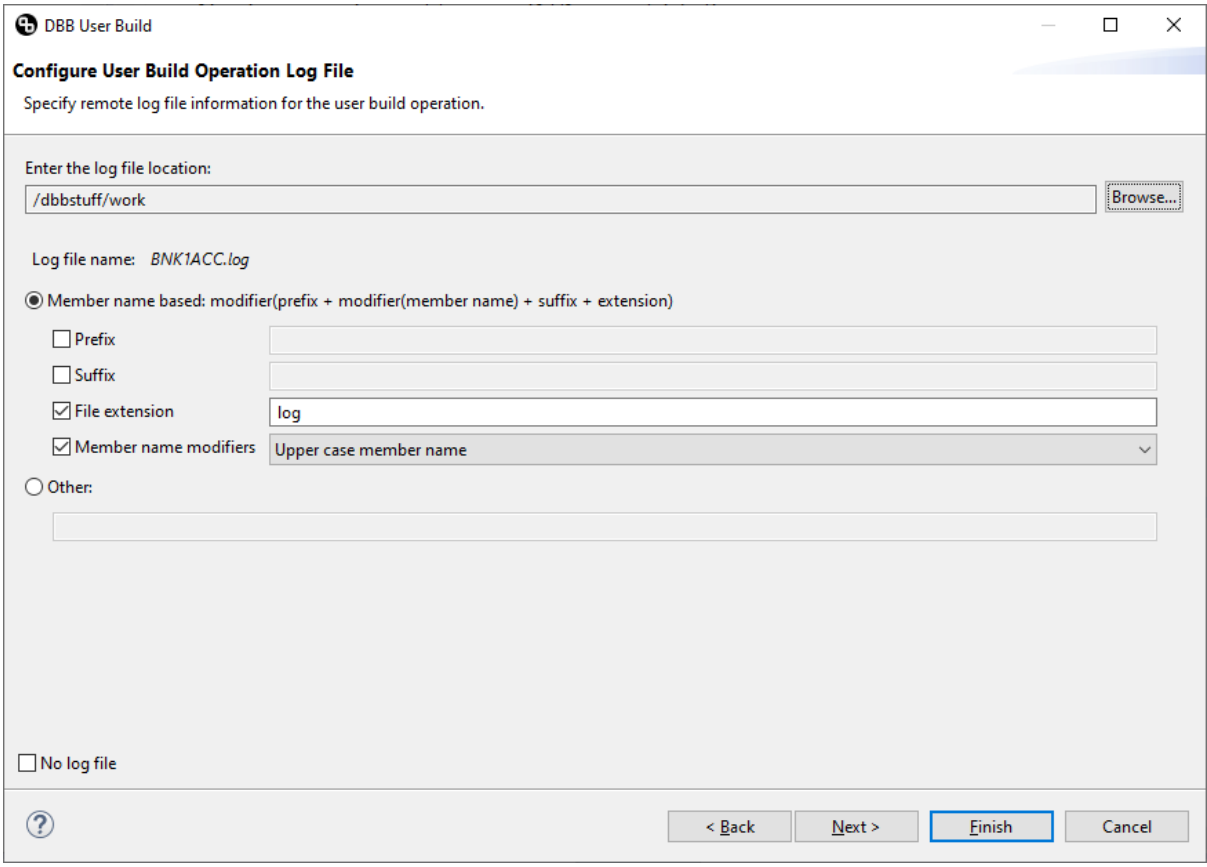
Then ensure that:

- The ‘z/OS system to use’ points at the host name for the machine that you have been allocated (it is likely to be different from the one shown in this example).
- The ‘select the build script to use:’ should be set to /u/ibmuser/projects/dbb-zappbuild/buildgroovy
- The ‘build sandbox folder’ should be /dbbstuff
- The ‘build destination HLQ’ should be CBSA.CICSBSA

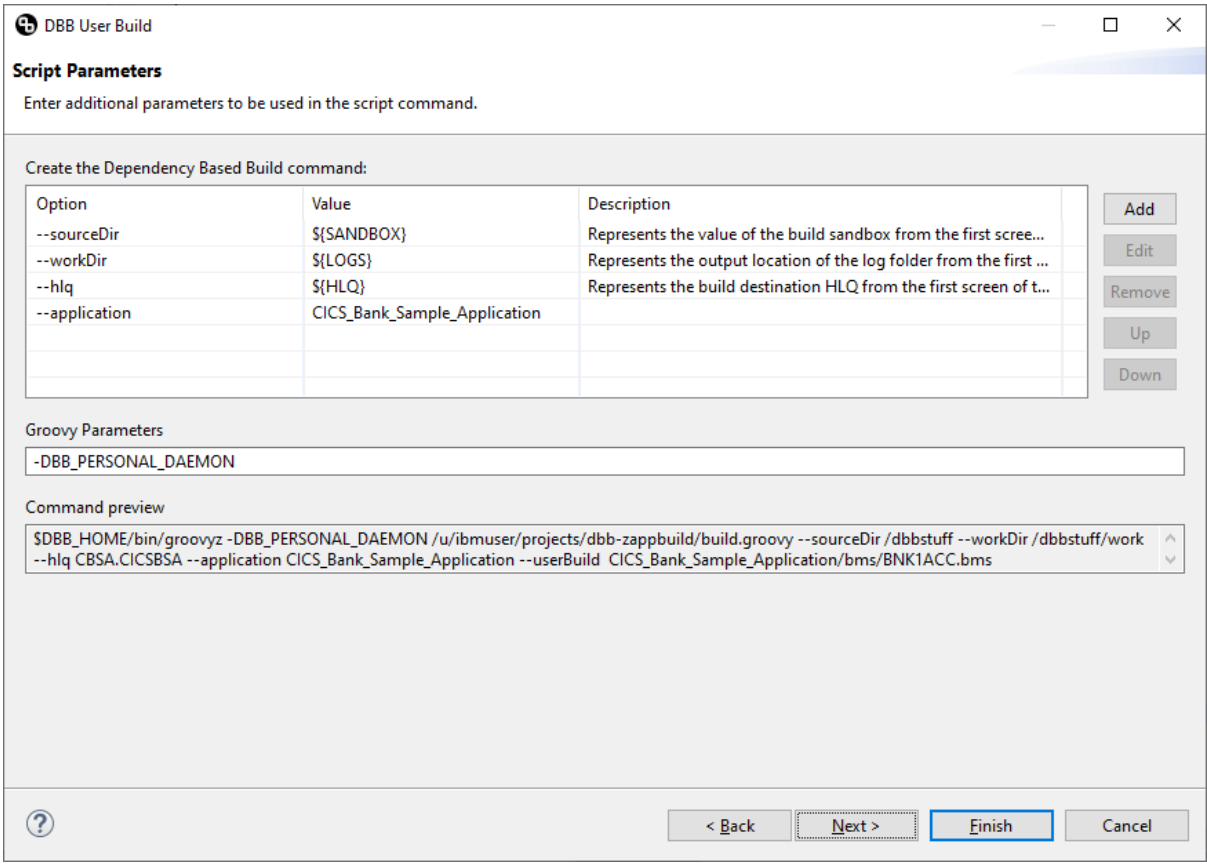
IV. Then click ‘Next’:



V. Click ‘Next’ again:

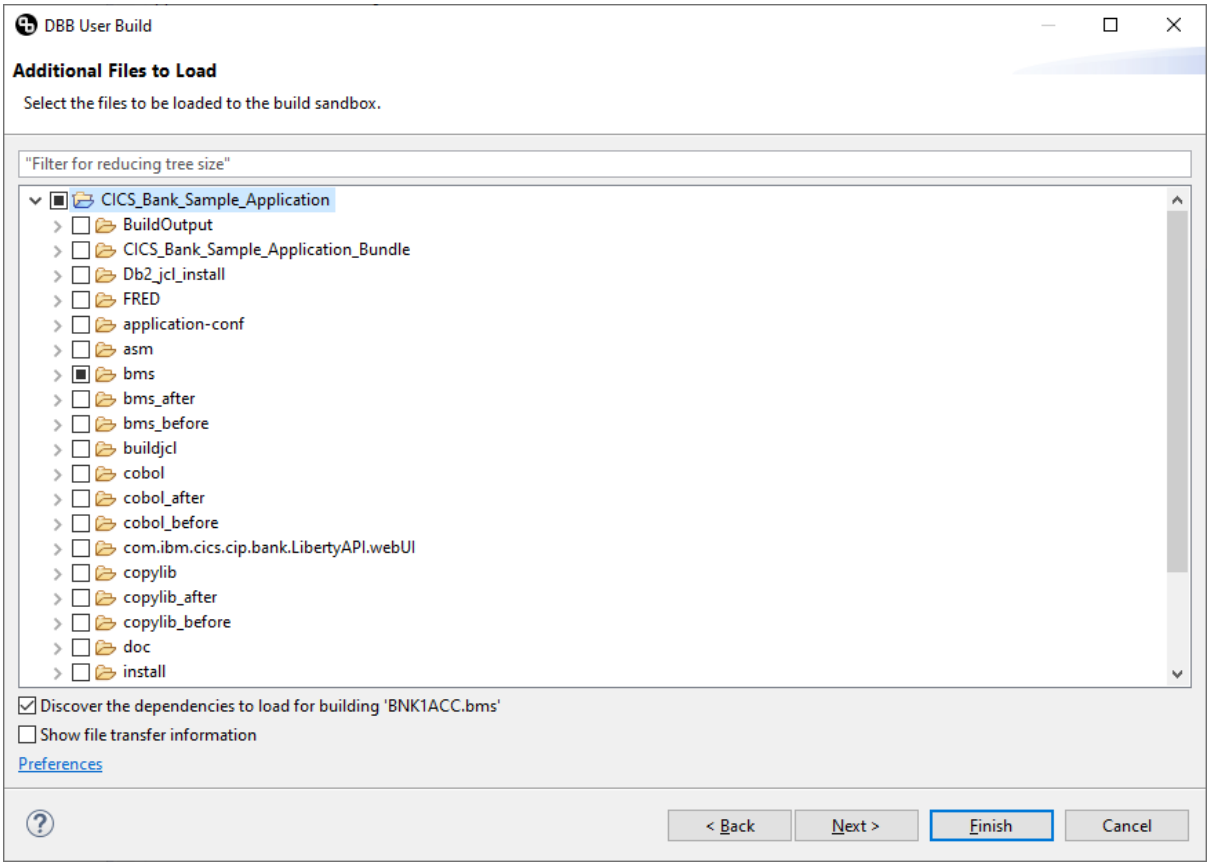


VI. And click ‘next’ again:

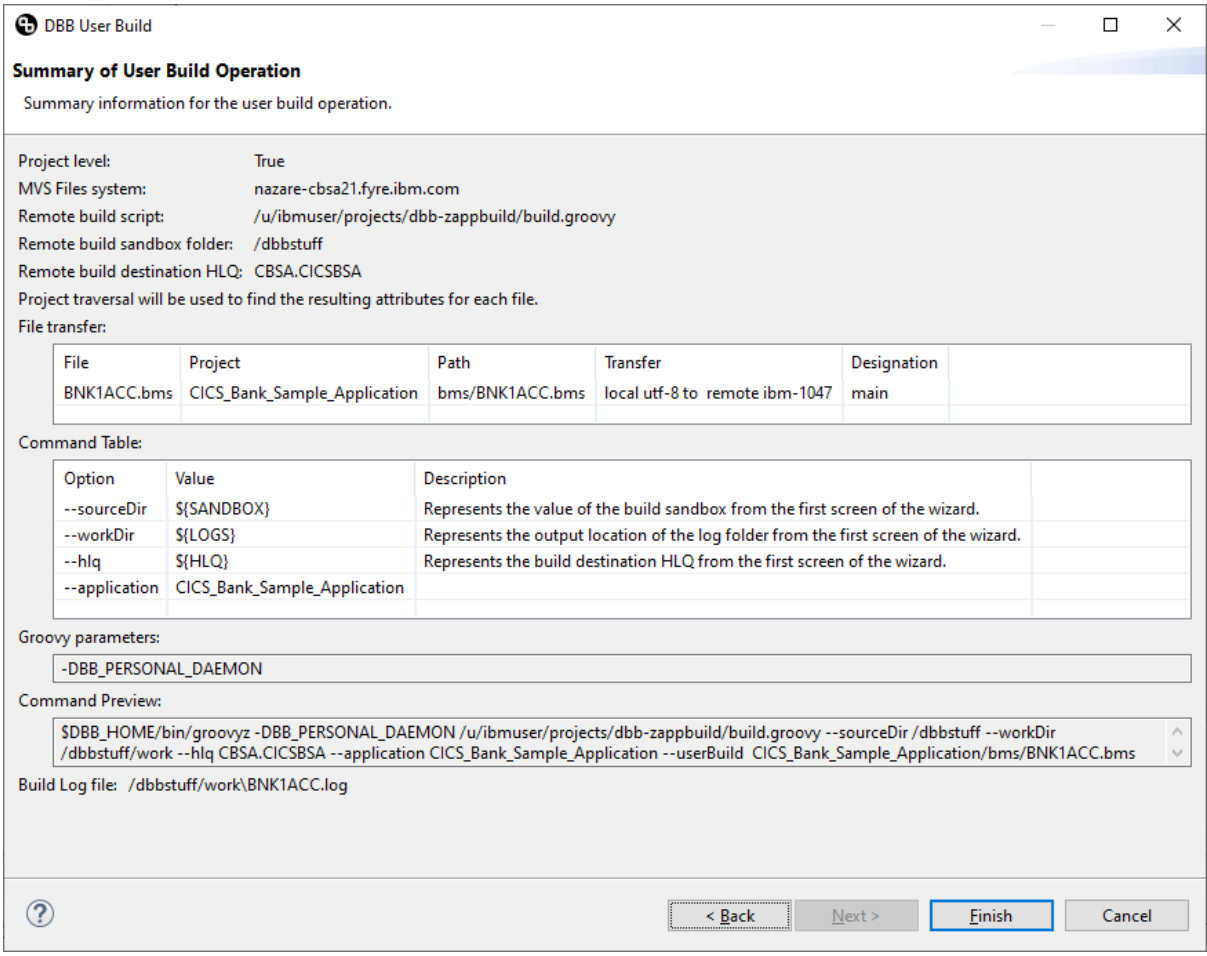


This should read as it does above, check that there are 4 variables (sourceDir, workDir, hlq and application) in the first table. Ensure that the 'Groovy Patterns' has -DBB_PERSONAL_DAEMON in it. Then click 'Next'.

VII. The next screen shows what is being built. In this particular case, it is a BMS map (you can open the bms folder should you so wish). Then click 'Next' again.

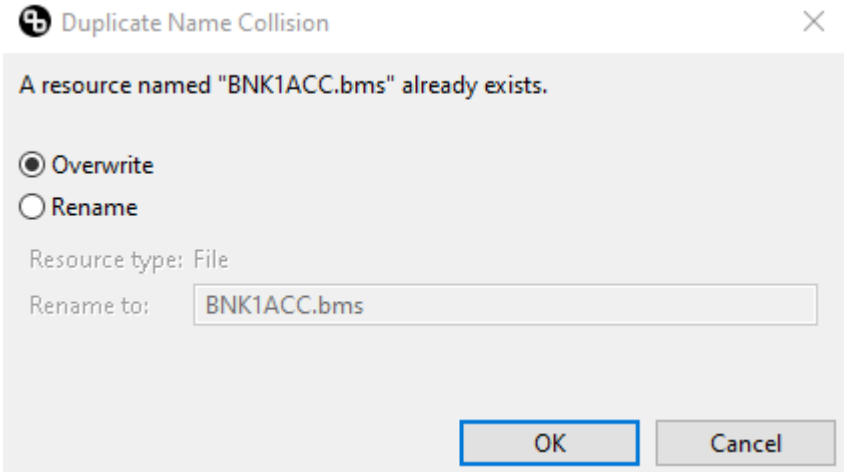


VIII. The final screen is a summary of everything that is being built:



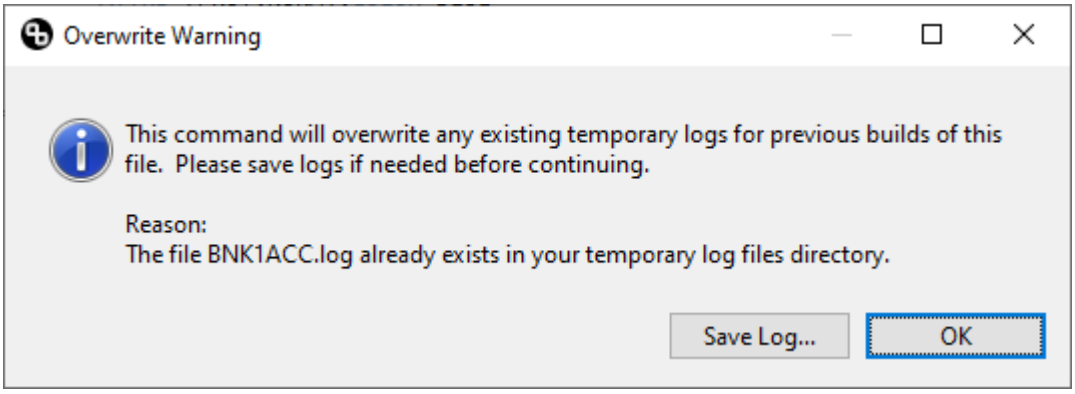
Just click ‘Finish’ to kick off the build process.

IX. You may get prompted that the source that you are building will be overwritten (this is not your source being overwritten (so don’t worry), but a copy of your source which DBB is building in its own sandbox area):

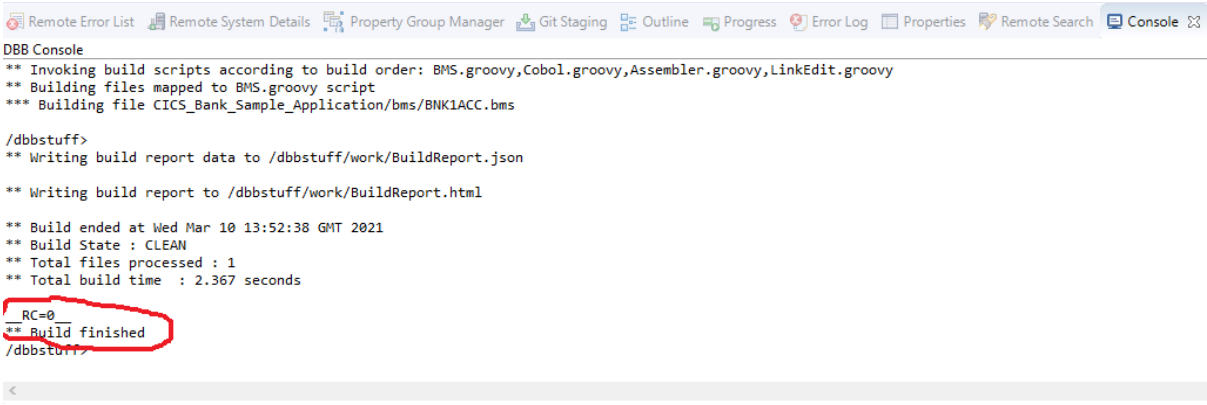


This is fine, just click OK.

X. During the build you may see messages that the temporary logs will be overwritten, this is OK, just click OK:



XI. In the 'Console' view (which can be opened if it is not already, by doing Window/Show View/Other and then put 'console' into the search box, and finally select 'General console') you should see the outcome of the build:



This should build with RC=0 (if the build was successful, if it is anything other than 0, you will need to check the edits that you made and correct as appropriate).

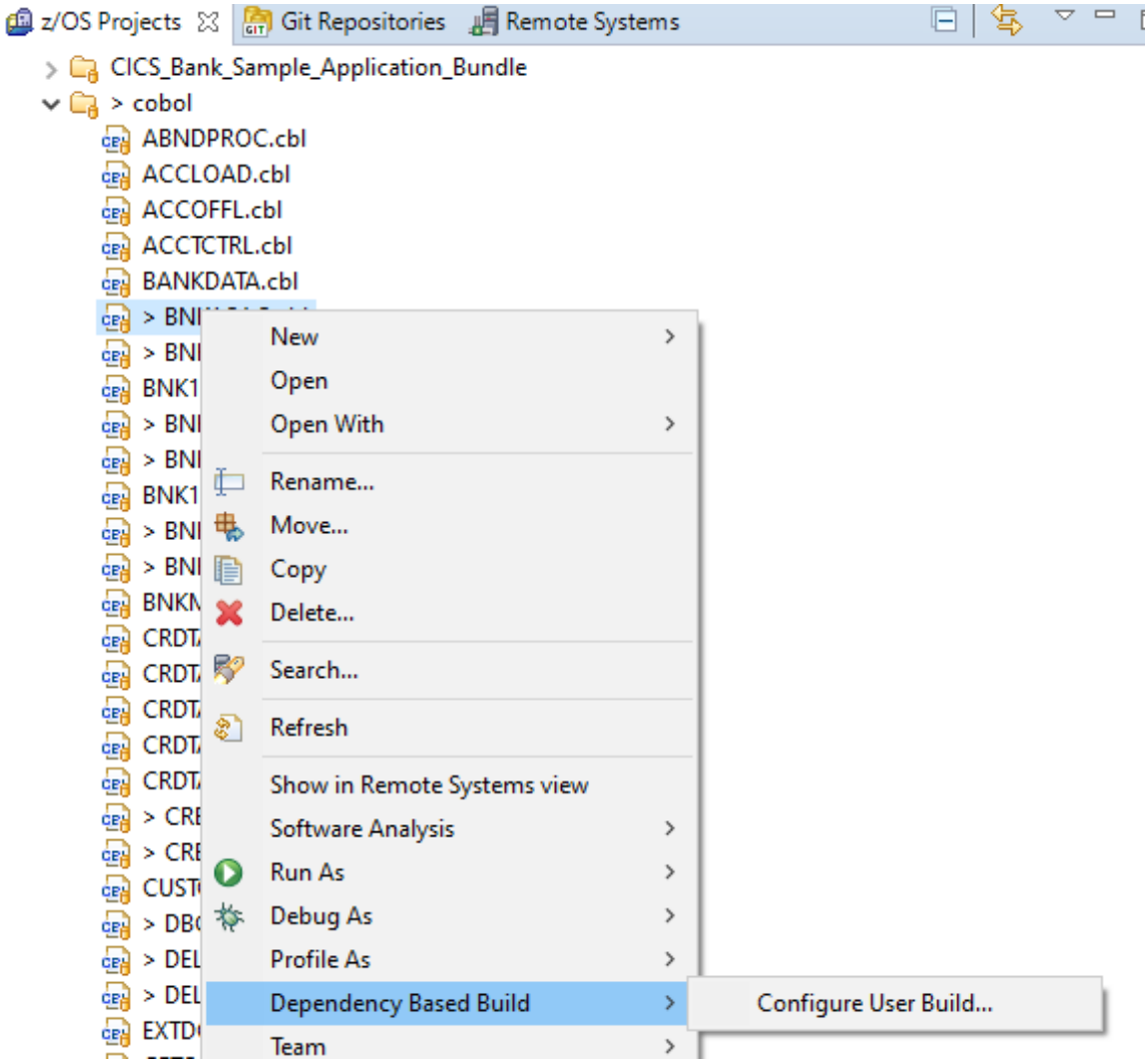
XII. Now repeat I – XII for the following BMS maps:
BNK1CAM, BNK1CDM, BNK1DAM, BNK1TFM, and BNK1UAM.

Rebuild the COBOL programs:

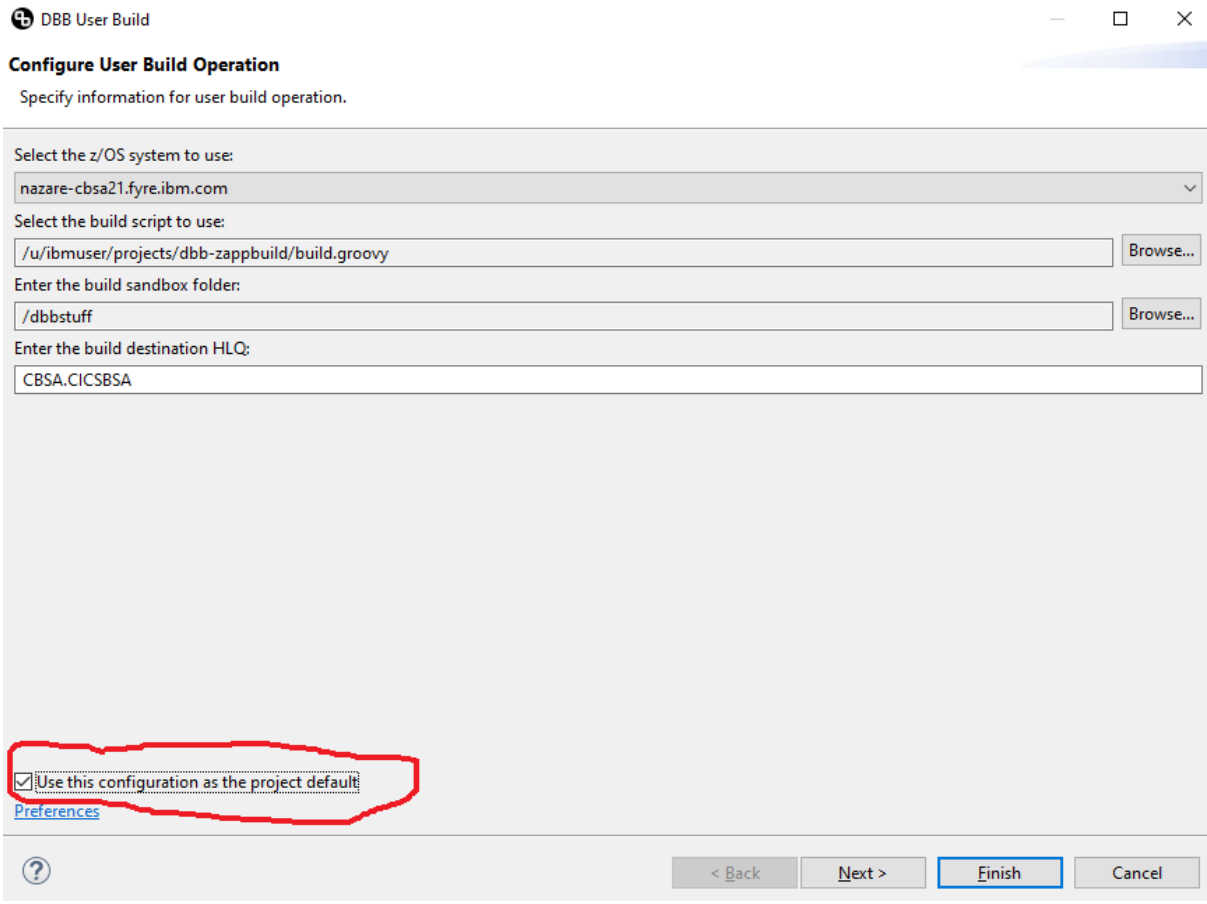
14. Having made all of the source code changes, the Db2 changes and rebuilt all of the BMS maps, the COBOL source code needs to be rebuilt:

I. Go into the z/OS Projects view in WAZI Developer – this is probably already open but in case it isn't, you can open it by selecting Window/Show View/zOS Projects

II. Open the /cobol folder (which now contains all of the amended cobol programs) and right click on a cobol program that you wish to build (in this case **BNK1CAC.cbl**). Then select 'Dependency Based Build' and 'Configure User Build':



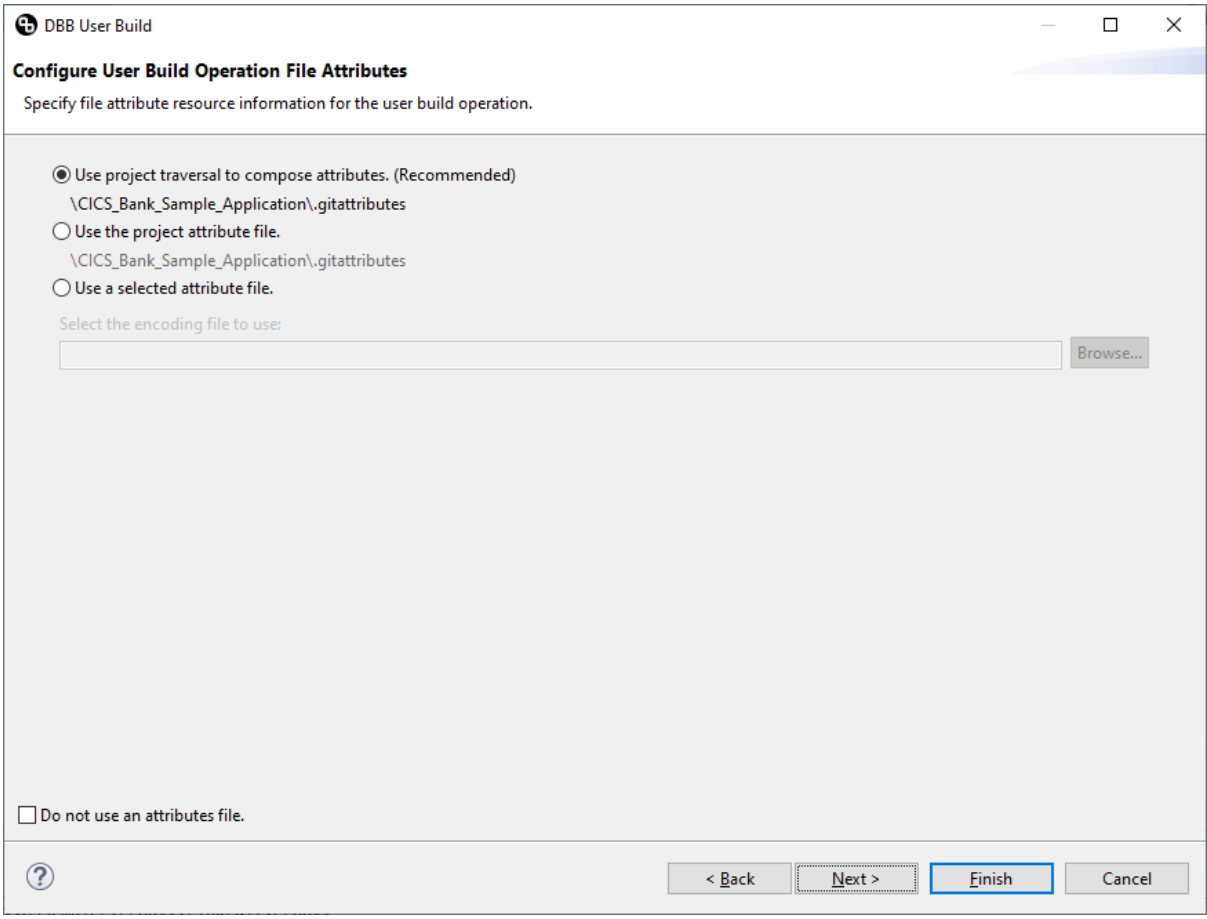
III. On the next screen click the 'Use this configuration as the project default' box:



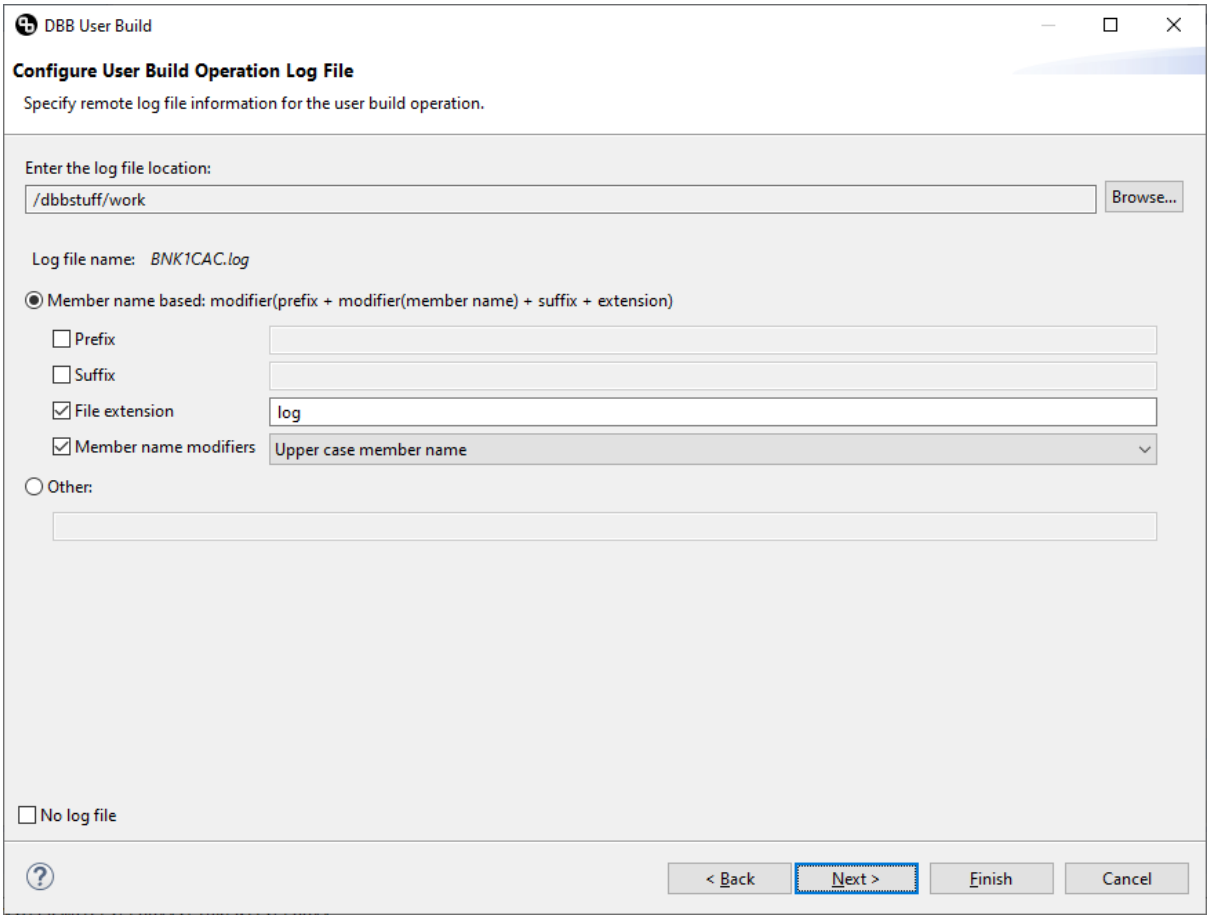
Then ensure that:

- The ‘z/OS system to use’ points at the host name for the machine that you have been allocated (it is likely to be different from the one shown in this example).
- The ‘select the build script to use:’ should be set to /u/ibmuser/projects/dbb-zappbuild/buildgroovy
- The ‘build sandbox folder’ should be /dbbstuff
- The ‘build destination HLQ’ should be CBSA.CICSBSA

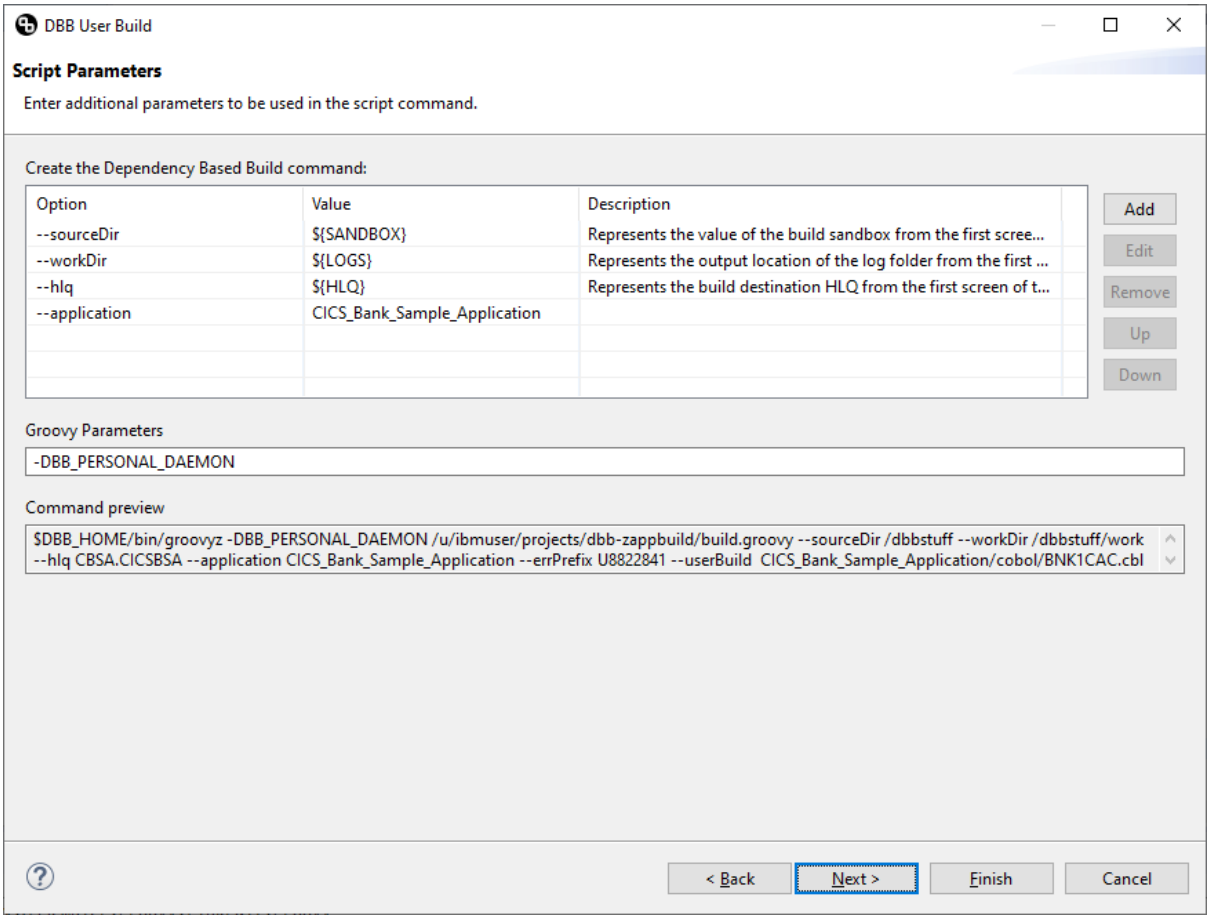
IV. Then click ‘Next’:



V. Click ‘Next’ again:

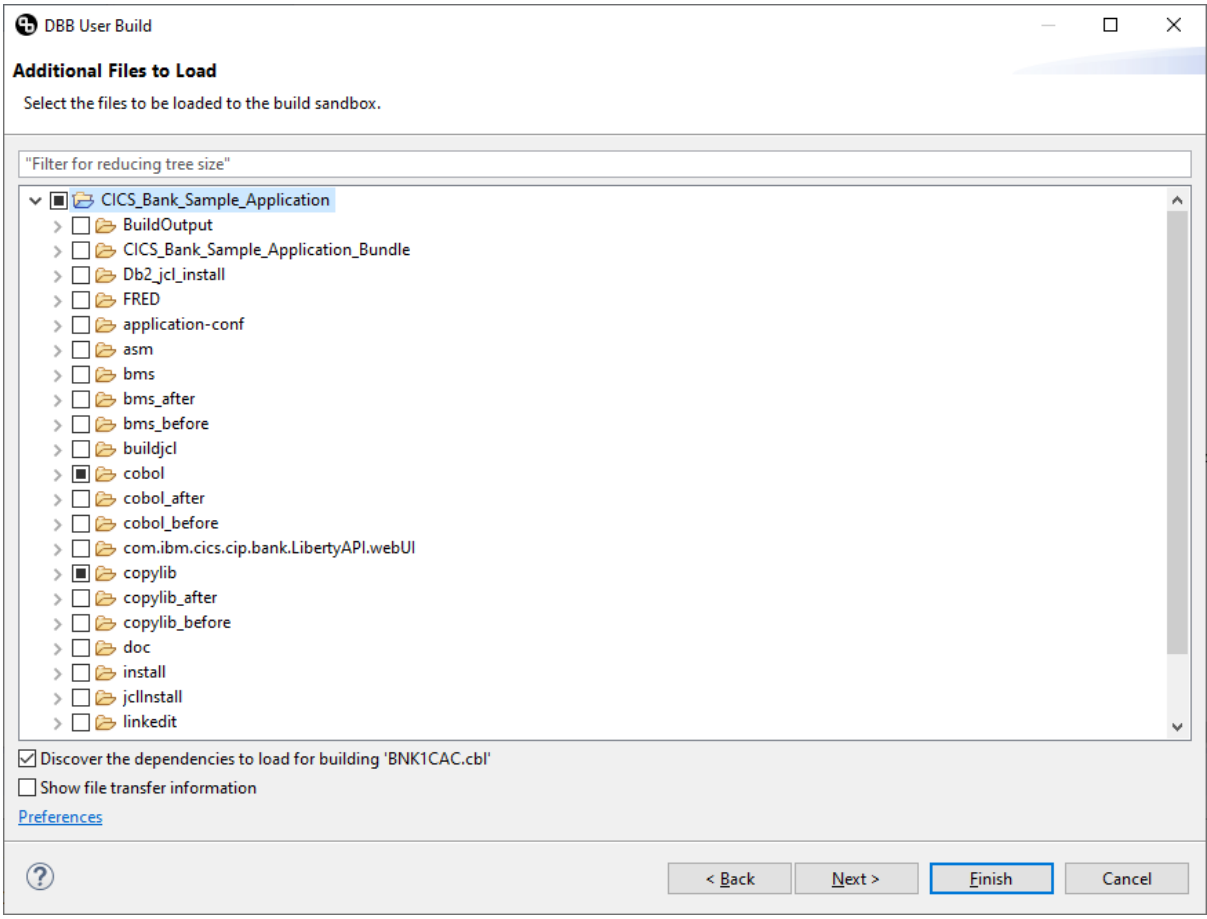


VI. And click ‘next’ again:



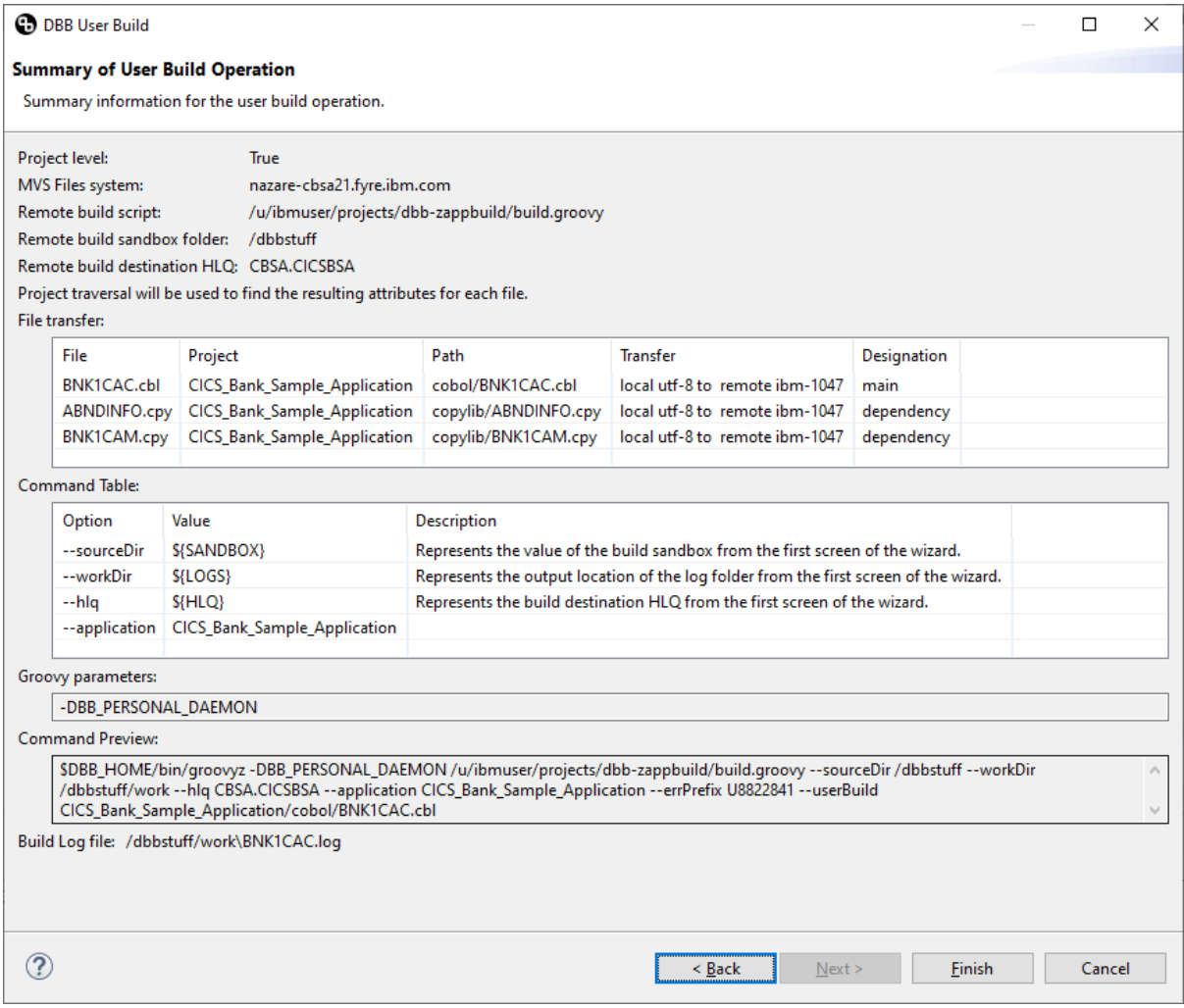
This should read as it does above, check that there are 4 variables (sourceDir, workDir, hlq and application) in the first table. Ensure that the 'Groovy Patterns' has -DBB_PERSONAL_DAEMON in it. Then click 'Next'.

VII. The next screen shows what is being built, in this particular case it is a cobol program and some copylibs (you can open these folders and check should you so wish).



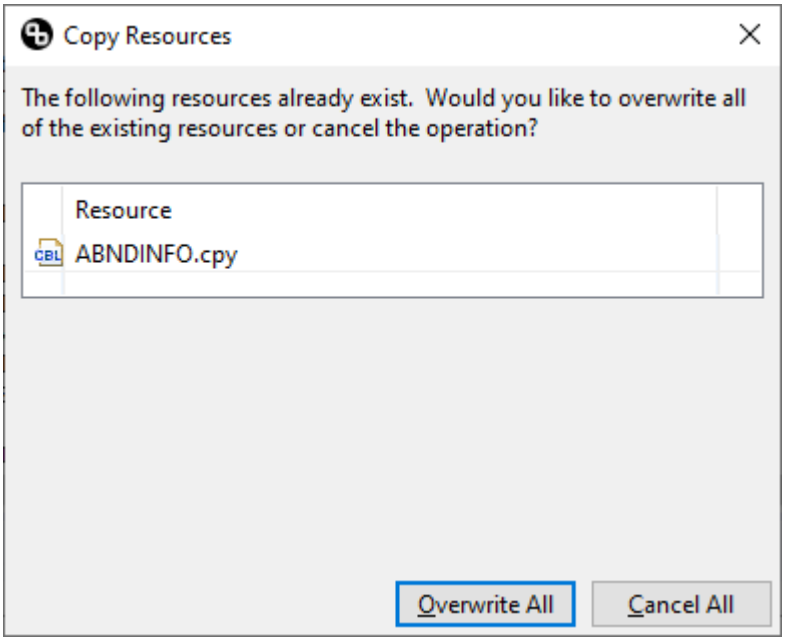
Then click ‘Next’ again.

VIII. The final screen is a summary of everything that is being built:



Just click ‘Finish’ to kick off the build process.

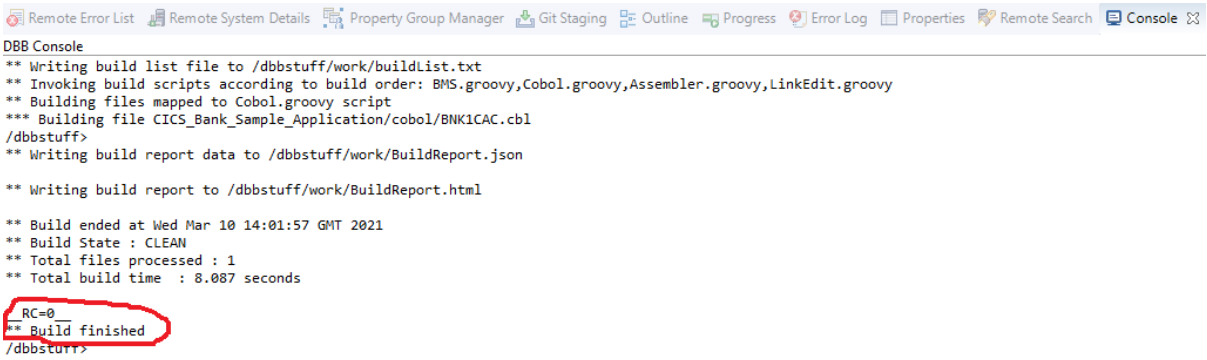
IX. You may get prompted that the cobol source or copylib members that you are building will be overwritten (this is OK, the message relates to overwriting the copy used for building in the DBB sandbox and NOT your original source):



This is fine, just click Overwrite All.

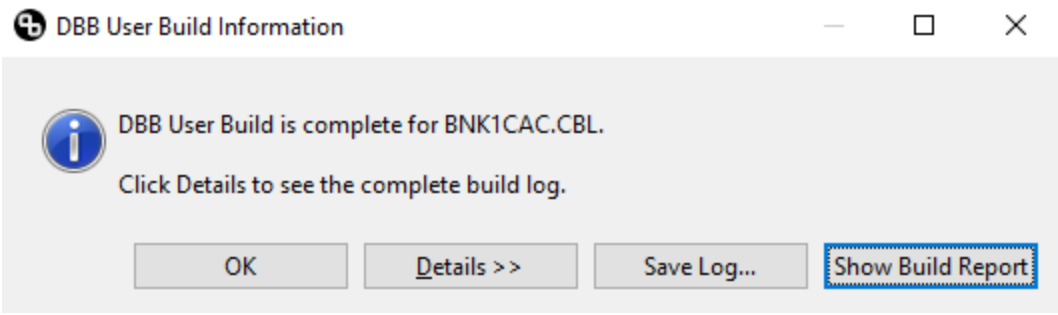
X. During the build you may see messages that the temporary logs will be overwritten, this is OK, just click OK.

XI. Open the console view (which can be opened, if it is not already, by doing Window/Show View/Other and then put 'console' into the search box, and finally select 'General console') you should see the outcome of the build:



This program should build with **RC=0** (if the build was successful, if it is anything other than 0, you will need to check the edits that you made and correct as appropriate, see XII for the compilation output).

XII. You may also see a DBB User Build is complete message:



You can look at the compilation listing by clicking the 'Details>>' button (this is useful if the build did not return RC of 0).

XIII. Once program BNK1CAC has built with an RC of 0, repeat steps I – XII (above) for the following cobol programs:

Program	Expected RC
CREACC	0
BNK1DAC	0
INQACC	0
DELACC	0
BNK1UAC	0
BNK1CRA	0
DBCRFUN	0
BNK1TFN	0
XFRFUN	0
BNK1CCA	0
INQACCCU	0
DELCUS	0
UPDACC	0
CRECUST	0

Rebind:

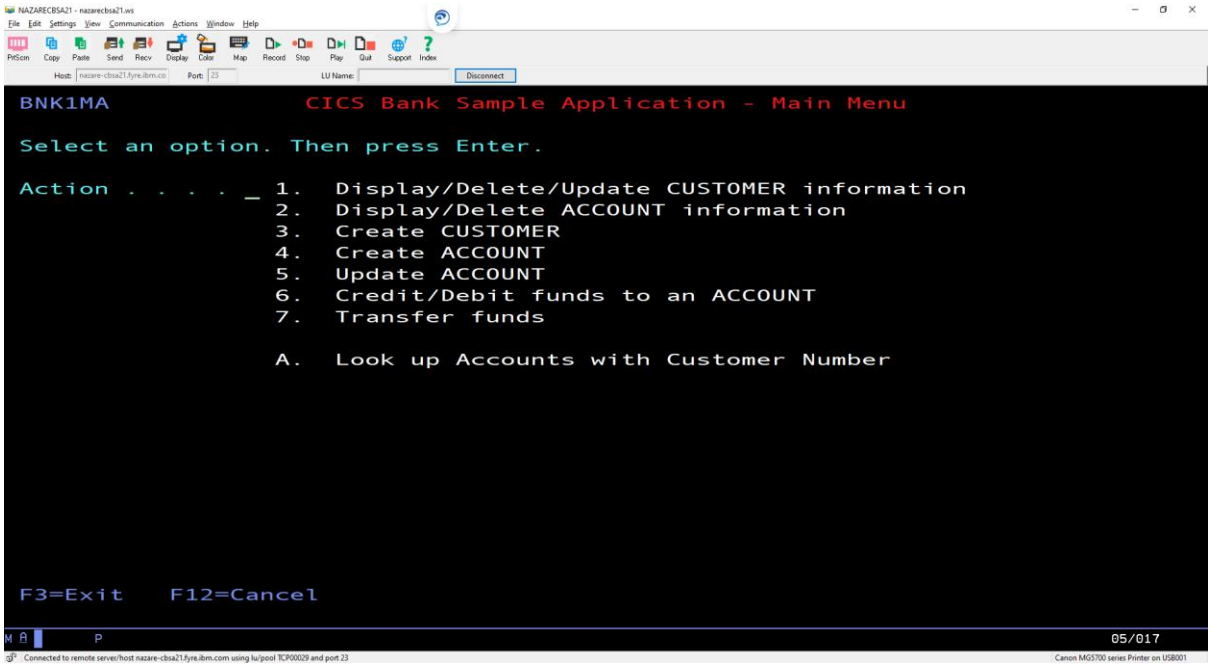
15. Having rebuilt everything, it is important to rebind against the changed Db2 structure. Go into CBSA.CICSBSA.REORG(**DB2BIND**) and submit this job (this will rebind the programs). Check the output of this job, if the return code for the BIND step was 00 (even if the return code for the GRANT step was 08) this is fine.

Restart CICS:

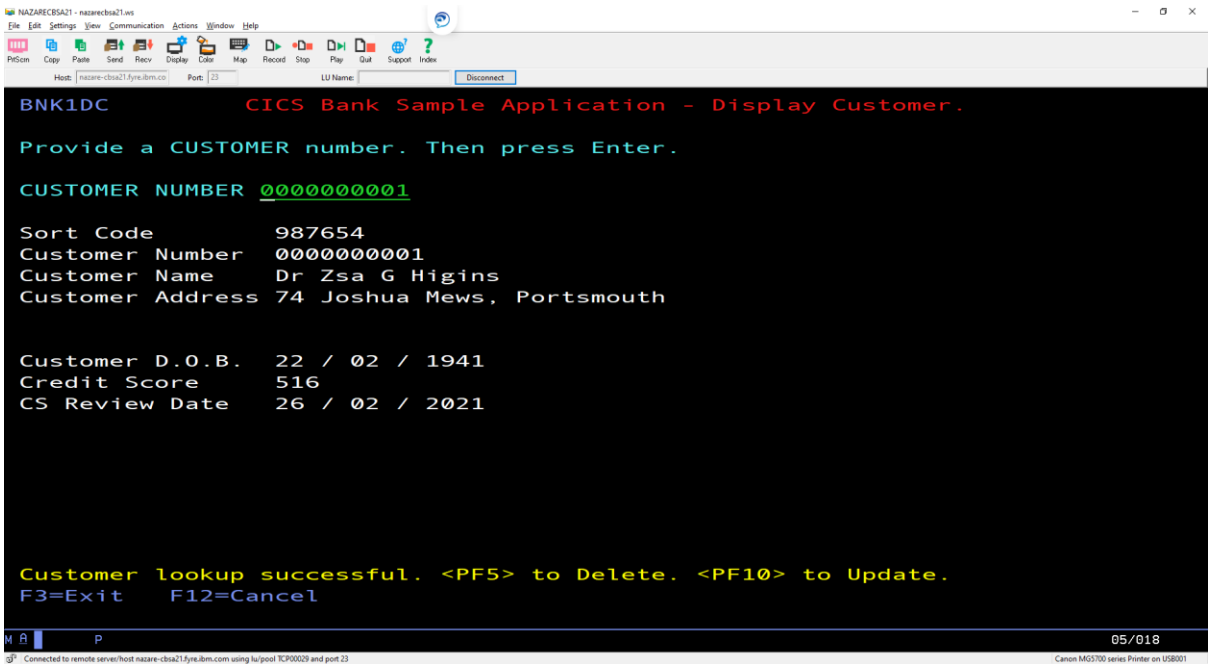
16. For CICS to pick up all of the changed maps and programs, the quickest solution is to stop and restart the CICS region. Submit job **CBSA.CICSBSA.REORG(SHUTCICS)** to shut CICS.
17. Then run job **CBSA.CICSBSA.REORG(RESTCICS)** to restart CICS again.

Try out the changes:

18. After leaving a few minutes to allow the CICS region to fully restart, logon to the CICS region, with the IBMUSER User id and its password.
19. Clear the screen and type OMEN and enter, to initiate the CBSA Main Menu:

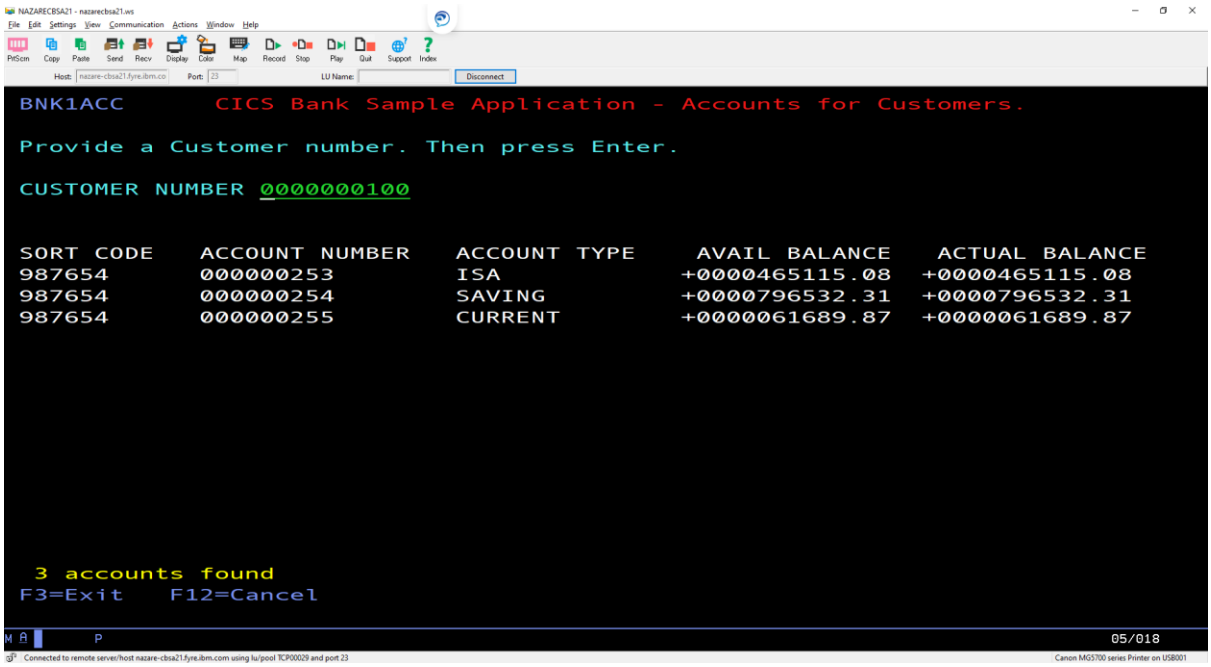


20. Then go into option 1 (Display customer) and supply a customer number. Check that this still works:



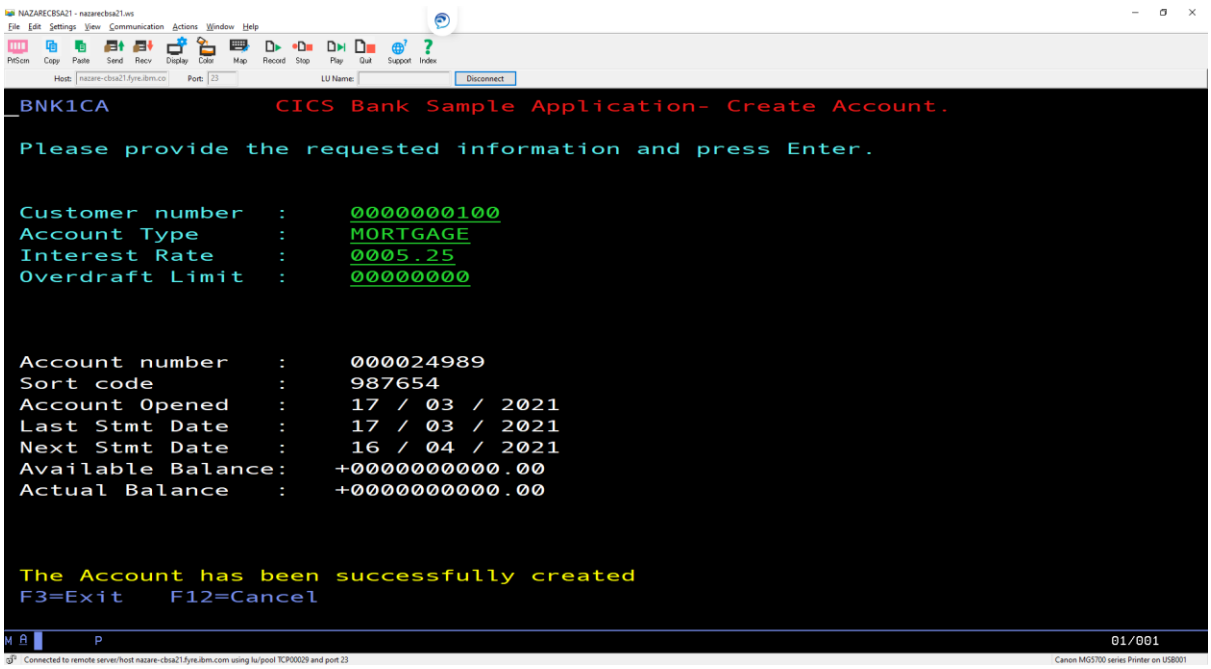
(This option was not directly changed but should not have been regressed).

21. Go back to the main menu, and then select option A (Look up account for customer) and supply a customer number:



In this example we have selected customer 100, which has 3 accounts associated with it. Notice that the account numbers returned are now 9 bytes in length - this is working correctly.

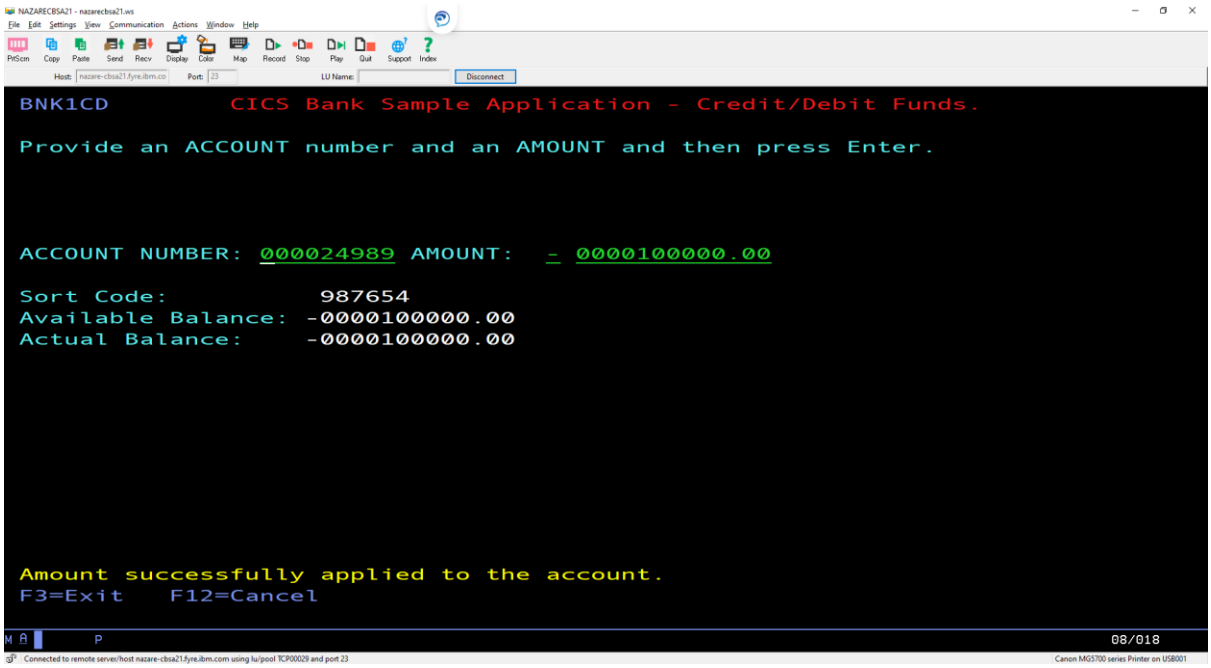
22. Go back to the main menu and select option 4 (Create an account):



In this example a new MORTGAGE account has been requested for customer 100. Notice that the allocated account number (in white) is now 9 bytes in length.

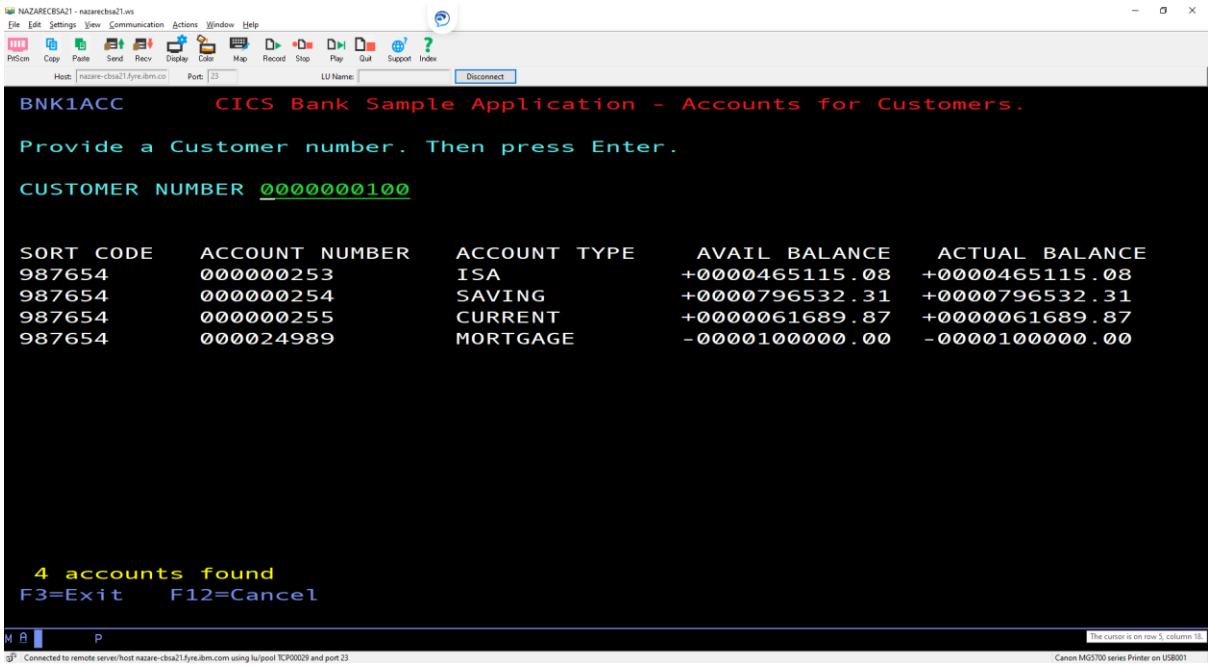
23. Go back to the main menu, and select option 6 (Credit/Debit funds) and enter a value of

-100,000.00 for the account created previously:



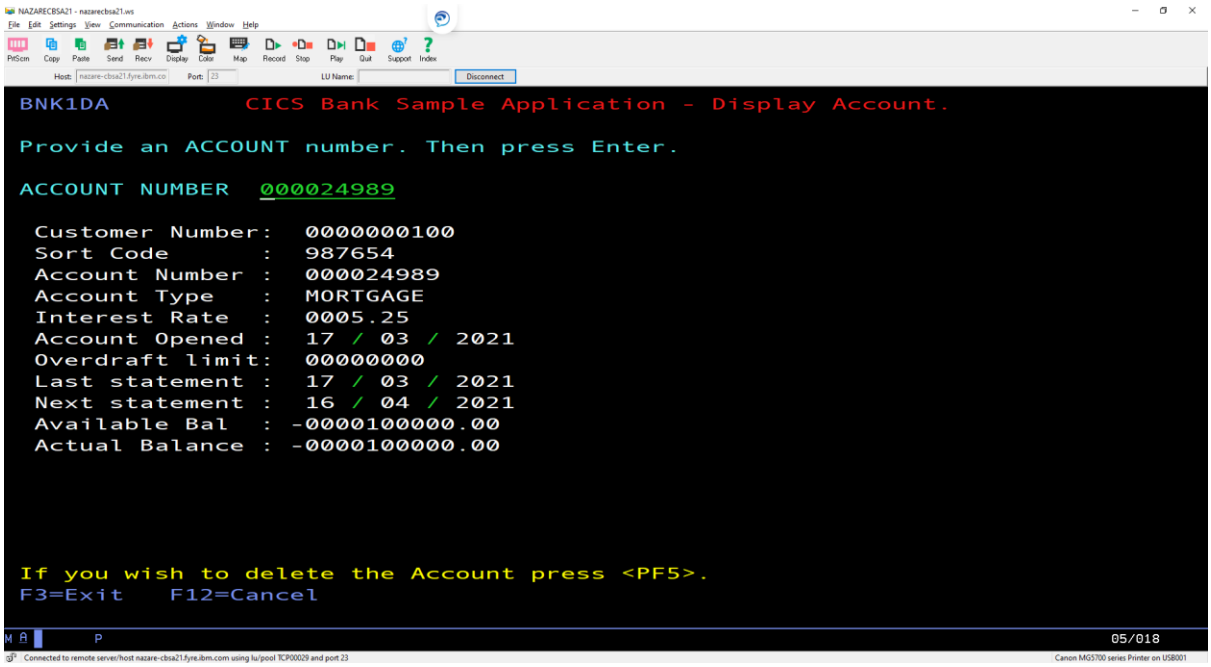
Notice that the input account number (in green) is now 9 bytes and that the amount was successfully debited from the new mortgage account.

24. Return to the main menu, and select option A (Look up account with customer number) again:



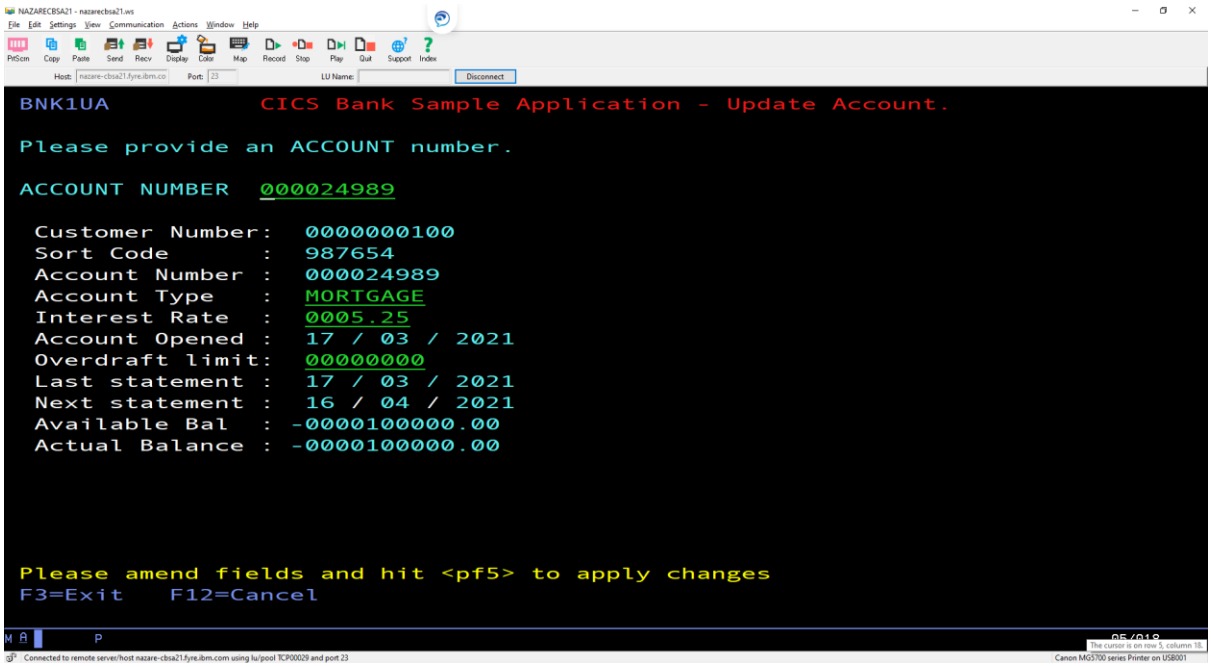
Notice that the new MOTRGAGE account is correctly shown for customer 100 and it contains minus £100,000.

25. Go back to the main menu, and select option 2 (Display/delete account):



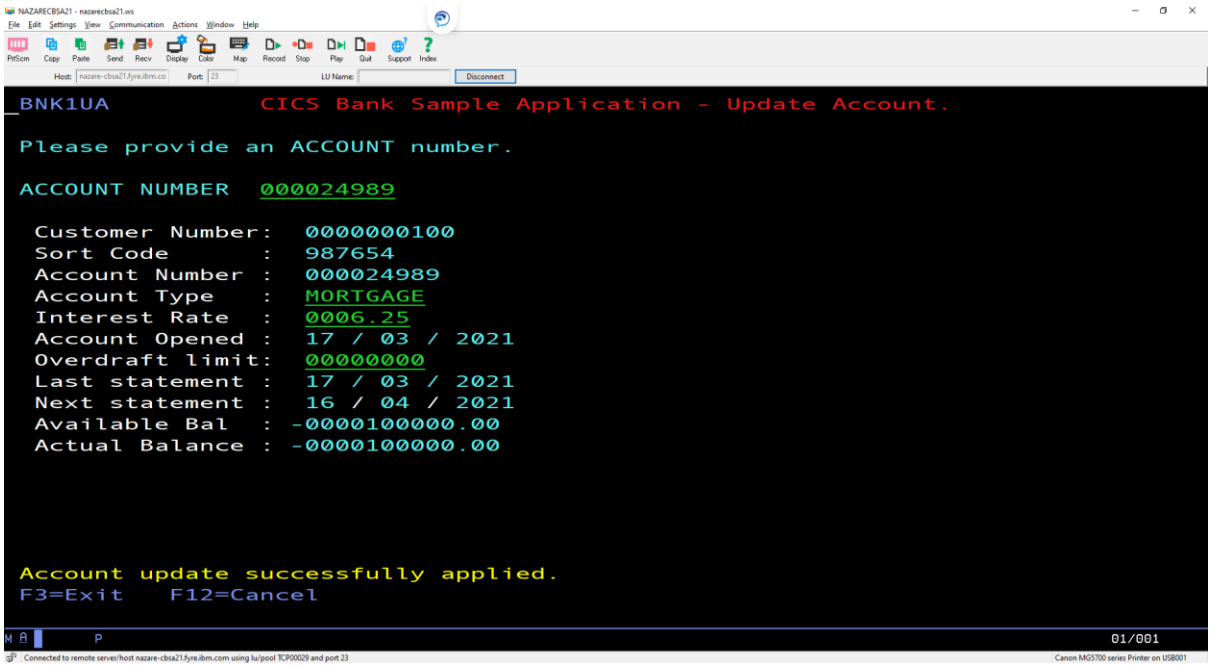
Notice that the input account number (in green) is now 9 bytes. The returned account number (in white) is now also 9 bytes in length.

26. Go back to the main menu and select option 5 (Update account):

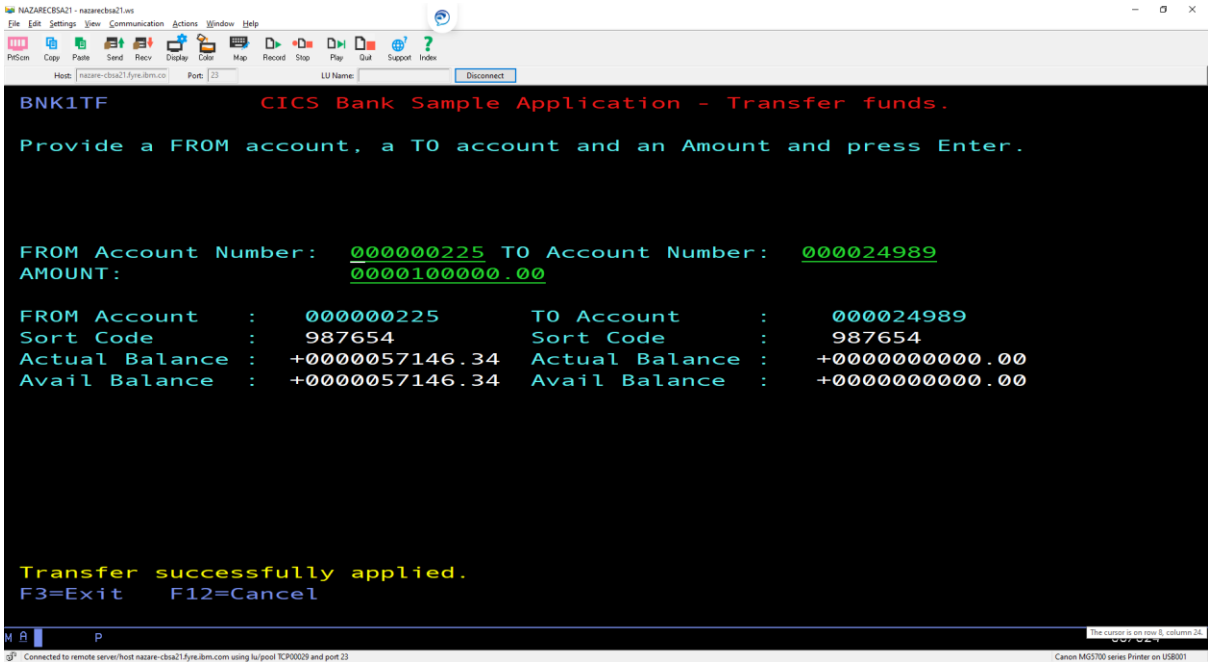


Notice that both account numbers are now 9 bytes in length.

Change the interest rate from (say) 5.25 to 6.25 and hit pf5 to update the record:

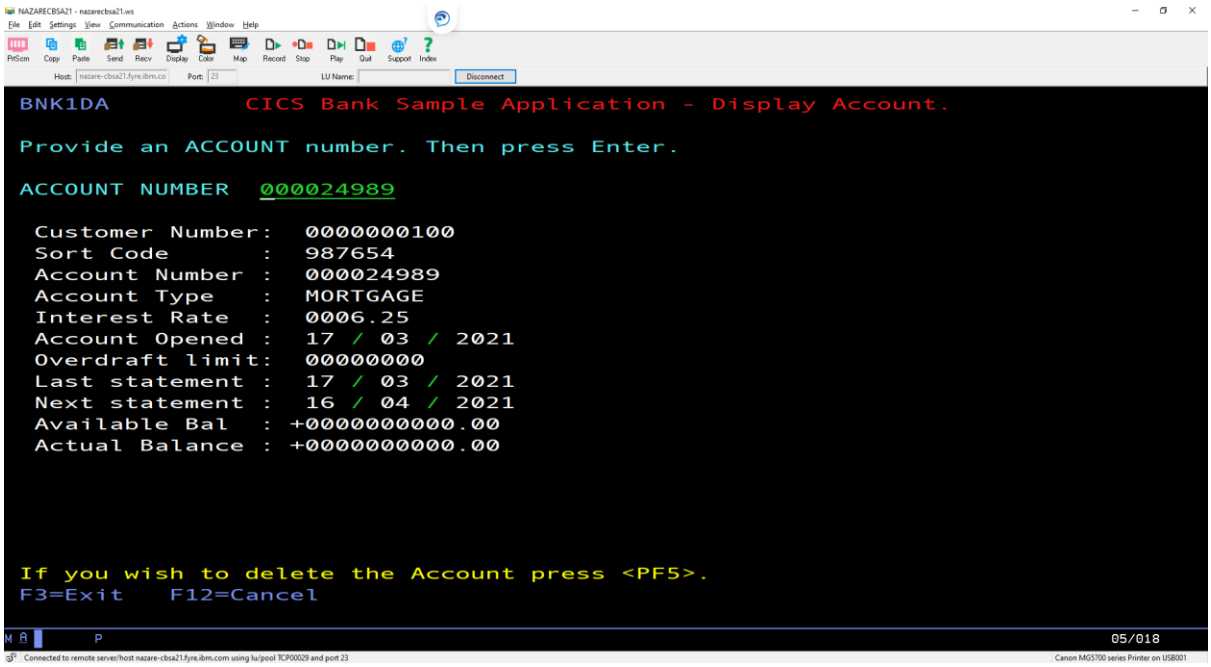


27. Return to the main menu, and select option 7 (Transfer funds):

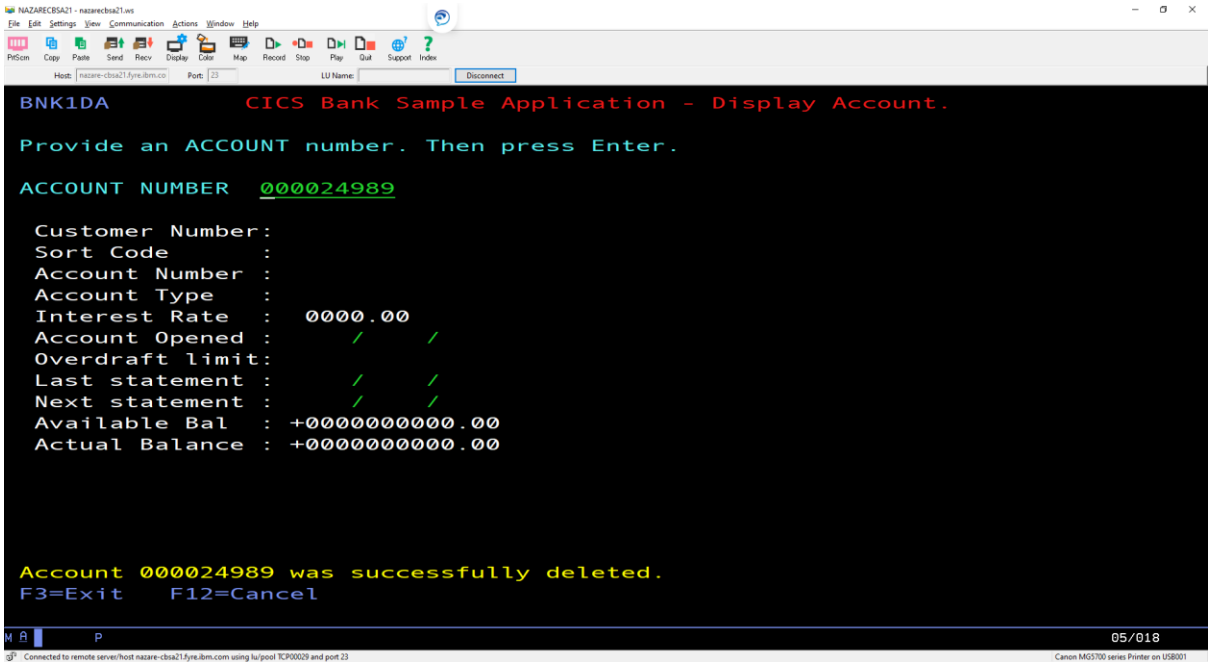


This transfer, between accounts, has paid off the £100,000 mortgage for the account created previously. Notice that the account number is now 9 bytes long wherever it is displayed on this map.

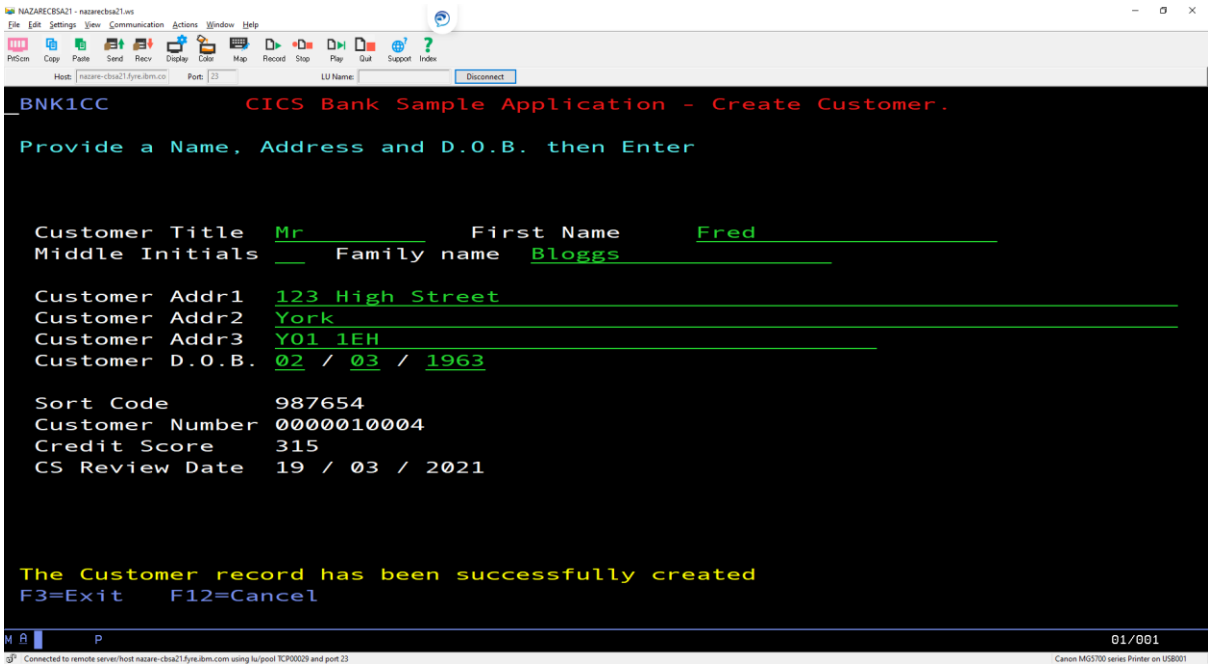
28. Go back to the main menu and select option 2 again (Display/Delete account):



Notice that the account now has a balance of 0. This time hit pf5 to delete the account.

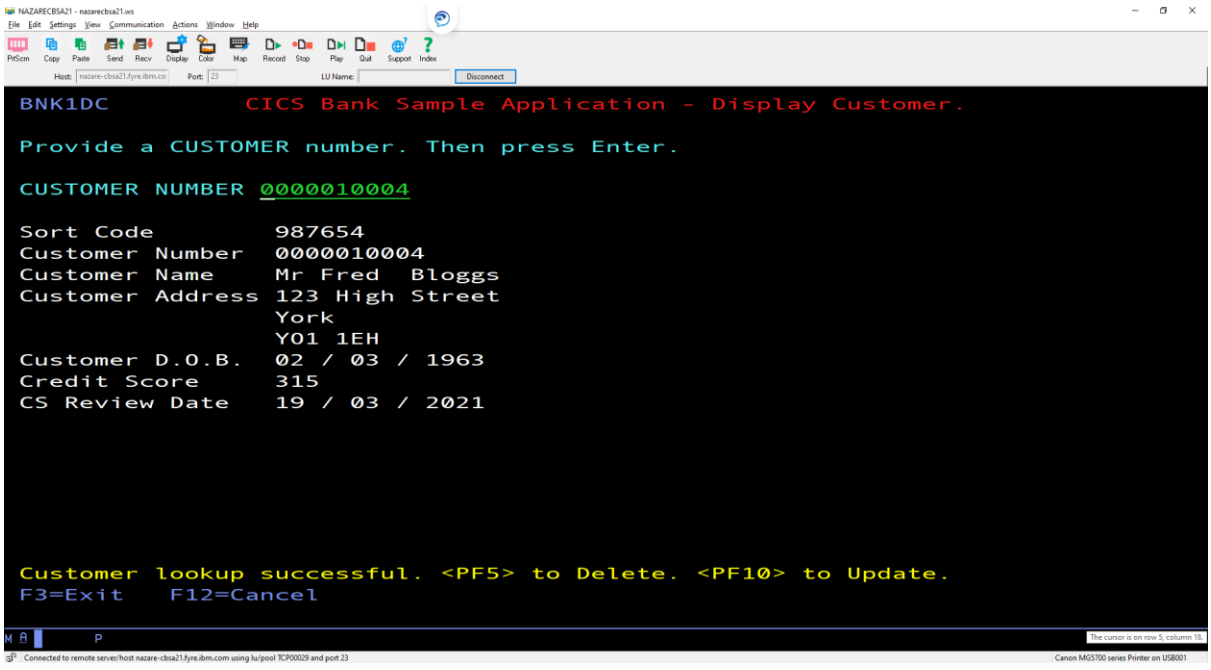


29. Return to the main menu and select option 3 (Create Customer) and enter details for a new customer:

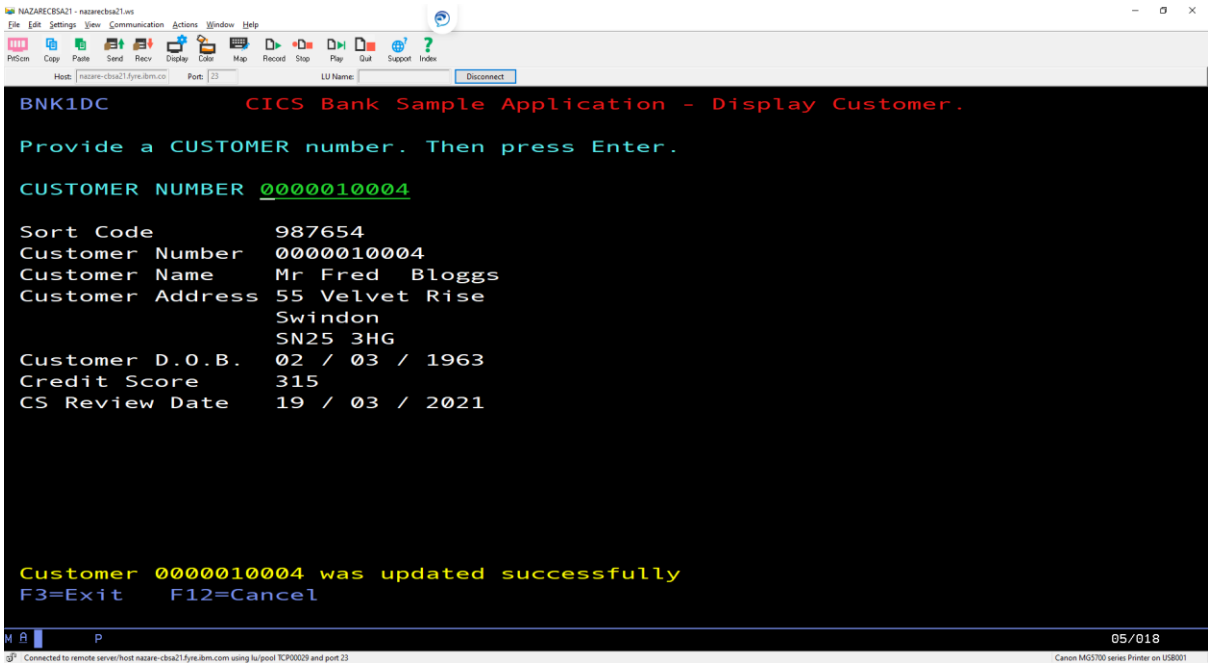


Although we did not change this view it demonstrates that the functionality still works and has not been regressed.

30. Return to the main menu and select option 1 (Display Customer):

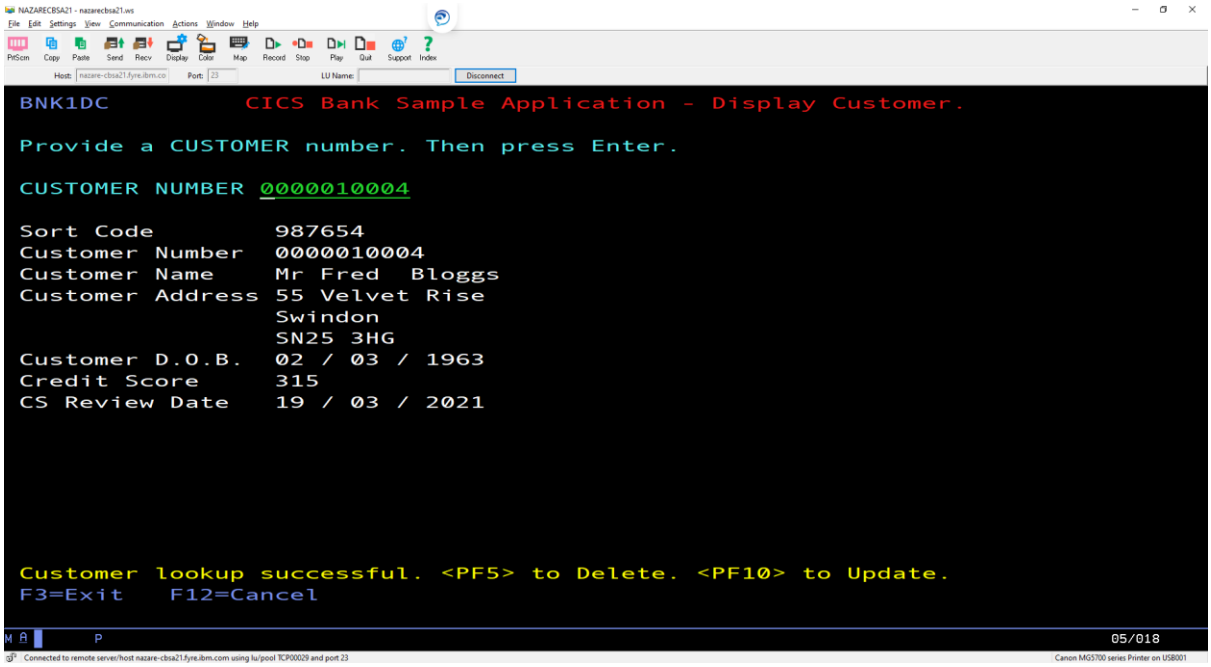


Hit pf10 and update and enter a new address, then hit enter to update:

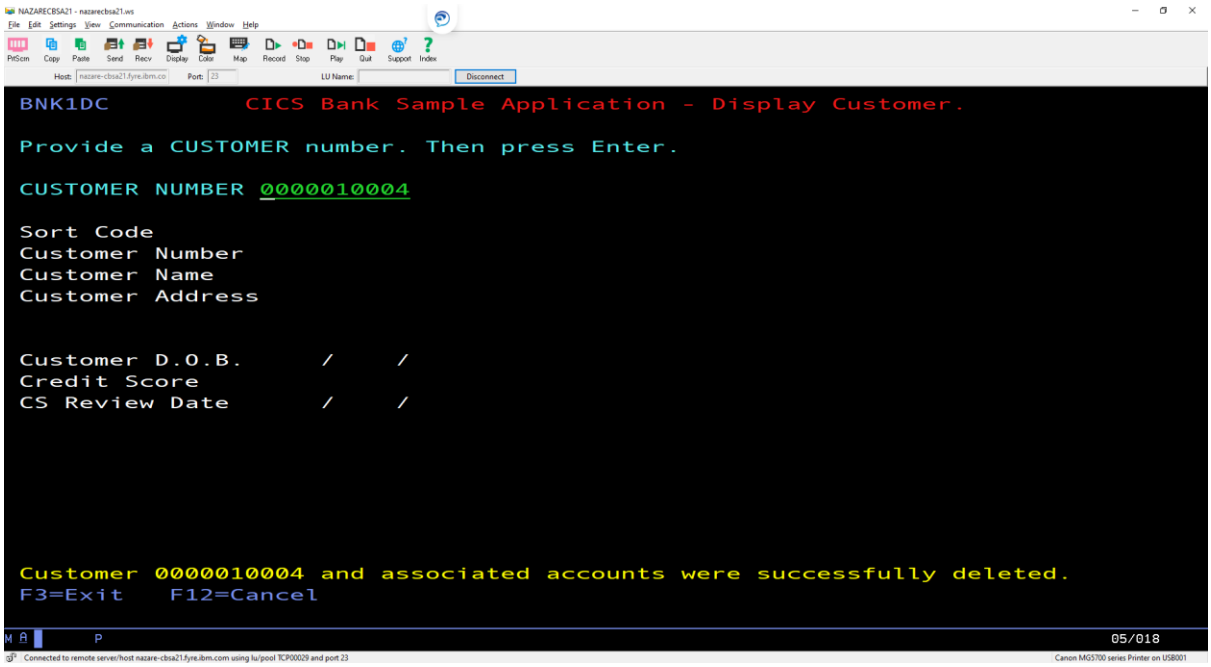


This function was not amended during the exercise but doing this proves that updates to customer details are not regressed.

31. Finally, return to the main menu and select option 1 to display the customer:



This time hit pf5 to delete the customer:



This has proven that the delete customer functionality has not been regressed.

Congratulations you have completed the exercise.

Appendix A

For the sake of expediency the exercise focuses on changing one BMS mapset, one BMS display validation program and one backend program. However, this Appendix documents the changes (line by line) to the remaining BMS mapsets, BMS display verification programs, copybooks and all of the remaining backend programs.

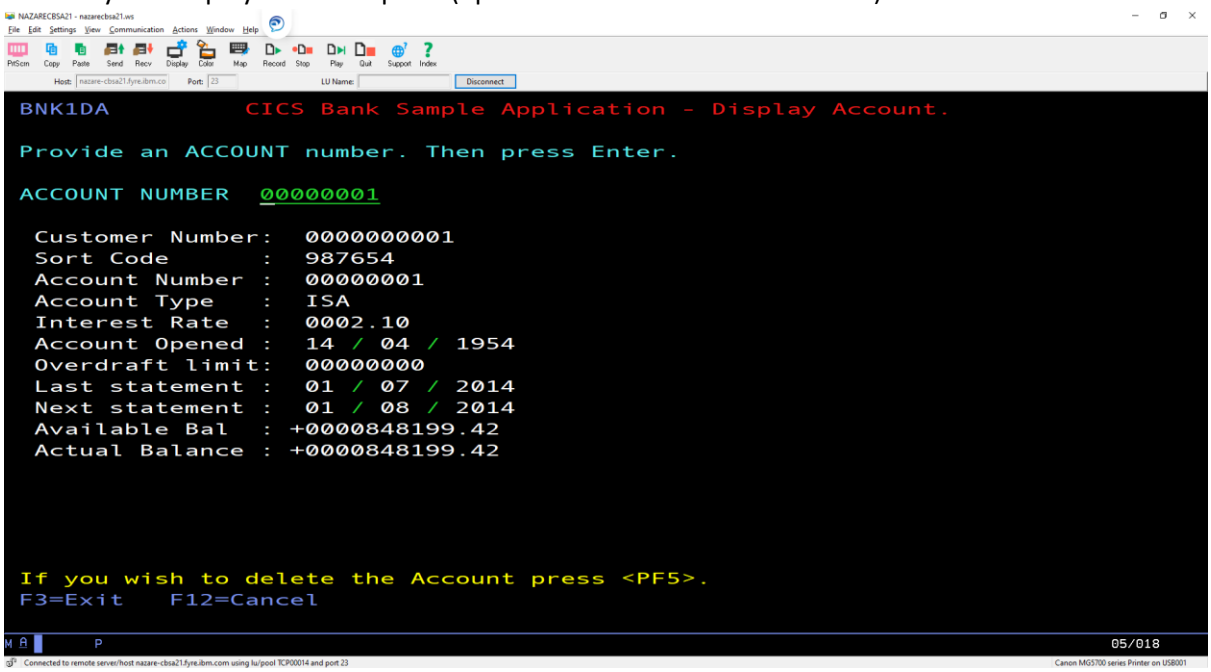
If you have followed the previous instructions these changes are implemented as a series of ‘cut’ and ‘paste’ operations but are shown here for reference.

The Display Account functionality:

Option on the BMS main menu	*Purpose*	*Mapset and Map affected*	*BMS display program affected*	*Back end program affected*
Option 2	Display Account	BNK1DAM (mapset) BNK1DA (map)	BNK1DAC	INQACC

Changing the Display Account BMS map (BNK1DAM)

1. Currently the Display Account option (option 2 from the BMS main menu) looks like this:



   The end user enters an account number and then the information is retrieved and displayed. There are several things that will need to change if we are making the account number larger:

   - The input account number (in green) needs to be able to accommodate 9 bytes instead of 8.
   - The Account Number returned (shown in white) will need to become 9 bytes.
   - Any account data passed to or from the BMS validation program will need to be 9 bytes.

2. Edit BMS mapset BNK1DAM from /bms folder in the Git Hub repo.
   Change the length of ACCNO from 8 to 9 bytes as highlighted in yellow below. Then amend the position of the stop byte from 5,26 to 5,27

```
...

DFHMDF POS=(3,1),LENGTH=44,ATTRB=(NORM,PROT),COLOR=TURQUOISE, *
      INITIAL='Provide an ACCOUNT number. Then press Enter.'
DFHMDF POS=(5,1),LENGTH=15,ATTRB=(NORM,PROT),COLOR=TURQUOISE, *
```

```
                INITIAL='ACCOUNT NUMBER'
ACCNO     DFHMDF POS=(5,17),LENGTH=9,ATTRB=(NORM,NUM,IC),COLOR=GREEN,    *
                HILIGHT=UNDERLINE
          DFHMDF POS=(5,27),LENGTH=1,ATTRB=(PROT,ASKIP)
          …
```

Next, we would expect to amend the length of ACCNO2 to make it 9 bytes, but it is already defined as having LENGTH(10), so no changes are necessary:

```
ACCNO2    DFHMDF POS=(9,20),LENGTH=10,                                   *
                ATTRB=(PROT,ASKIP,FSET,NORM),COLOR=NEUTRAL
```

3. **Then save this change** (use control S to save). **NOTE** if the bms source does not save with Cntrl S, close the tab and you will be prompted to save at that point (click the Save button).

   As a result of changing the BMS map, the generated BMS symbolic map/DSECT (also called BNK1DAM) will automatically get changed when the BMS map is rebuilt.

Changing the Display Account BMS validation program (BNK1DAC):

4. Program BNK1DAC is responsible for validating the data coming from the Display Account BMS map and also data from the backend program, being output onto the BMS map. Since we have now changed the BMS map to allow an extra byte for each instance of the account number, it follows that we will also need to ensure that any account number data passed into or used by BNK1DAC is also 9 bytes long.

5. Edit COBOL program **BNK1DAC** from the /cobol folder in the Git Hub repo. This pulls in a couple of copybooks:
   - The BNK1DAM copybook, should be regenerated (with 9 byte account numbers) when the BMS map (above) gets recompiled – so no further action is necessary.
   - And the INQACC copybook

   Change PARMS-SUBPGM-ACCNO (which passes the ACCOUNT NUMBER to and from the backend program INQACC) and amend the size from PIC 9(8) to PIC 9(9) as highlighted:

```
01 PARMS-SUBPGM.
   03 PARMS-SUBPGM-EYE          PIC X(4).
   88 PARMS-SUBPGM-EYE-VALID        VALUE 'ACCT'.
   03 PARMS-SUBPGM-CUSTNO       PIC X(10).
   03 PARMS-SUBPGM-SCODE        PIC X(6).
   03 PARMS-SUBPGM-ACCNO        PIC 9(9).
   03 PARMS-SUBPGM-ACC-TYPE     PIC X(8).
   03 PARMS-SUBPGM-INT-RATE     PIC 9(4)V99.
   03 PARMS-SUBPGM-OPENED       PIC 9(8).
   03 PARMS-SUBPGM-OVERDRAFT    PIC 9(8).
   03 PARMS-SUBPGM-LAST-STMT-DT PIC 9(8).
   03 PARMS-SUBPGM-NEXT-STMT-DT PIC 9(8).
```

```
   03 PARMS-SUBPGM-AVAIL-BAL    PIC S9(10)V99.
   03 PARMS-SUBPGM-ACTUAL-BAL   PIC S9(10)V99.
   03 PARMS-SUBPGM-SUCCESS      PIC X.
   03 PARMS-SUBPGM-FAIL-CD      PIC X.
   03 PARMS-SUBPGM-DEL-SUCCESS  PIC X.
   03 PARMS-SUBPGM-DEL-FAIL-CD  PIC X.
   03 PARMS-SUBPGM-DEL-APPLID   PIC X(8).
   03 PARMS-SUBPGM-DEL-PCB1     POINTER.
   03 PARMS-SUBPGM-DEL-PCB2     POINTER.
   03 PARMS-SUBPGM-DEL-PCB3     POINTER.
```

BNK1DAC utilises copybook INQACC.

Change the COPYBOOK INQACC:
Edit INQACC in the GitHub /copylib folder, and make the following change highlighted in yellow:

```
01 INQACC-COMMAREA.
   03 INQACC-EYE                  PIC X(4).
   03 INQACC-CUSTNO               PIC 9(10).
   03 INQACC-SCODE                PIC 9(6).
   03 INQACC-ACCNO                PIC 9(9).
   03 INQACC-ACC-TYPE             PIC X(8).
   03 INQACC-INT-RATE             PIC 9(4)V99.
   03 INQACC-OPENED               PIC 9(8).
   03 INQACC-OPENED-GROUP REDEFINES INQACC-OPENED.
     05 INQACC-OPENED-DAY         PIC 99.
     05 INQACC-OPENED-MONTH         PIC 99.
     05 INQACC-OPENED-YEAR        PIC 9999.
   03 INQACC-OVERDRAFT            PIC 9(8).
   03 INQACC-LAST-STMT-DT         PIC 9(8).
   03 INQACC-LAST-STMT-GROUP REDEFINES INQACC-LAST-STMT-DT.
     05 INQACC-LAST-STMT-DAY         PIC 99.
     05 INQACC-LAST-STMT-MONTH         PIC 99.
     05 INQACC-LAST-STMT-YEAR        PIC 9999.
   03 INQACC-NEXT-STMT-DT         PIC 9(8).
   03 INQACC-NEXT-STMT-GROUP REDEFINES INQACC-NEXT-STMT-DT.
     05 INQACC-NEXT-STMT-DAY         PIC 99.
     05 INQACC-NEXT-STMT-MONTH         PIC 99.
     05 INQACC-NEXT-STMT-YEAR        PIC 9999.
   03 INQACC-AVAIL-BAL            PIC S9(10)V99.
   03 INQACC-ACTUAL-BAL           PIC S9(10)V99.
   03 INQACC-SUCCESS              PIC X.
   03 INQACC-PCB1-POINTER         POINTER.
```

Then save this change (use cntrl S to save).

Back in program BNK1DAC, modify WS-COMM-AREA (used to hold a copy of comm area data) to be as follows:

```
01 WS-COMM-AREA.
   03 WS-COMM-EYE                 PIC X(4).
   03 WS-COMM-CUSTNO              PIC X(10).
   03 WS-COMM-SCODE               PIC X(6).
   03 WS-COMM-ACCNO               PIC 9(9).
   03 WS-COMM-ACC-TYPE            PIC X(8).
```

```
03 WS-COMM-INT-RATE          PIC 9(4)V99.
03 WS-COMM-OPENED            PIC 9(8).
03 WS-COMM-OVERDRAFT         PIC 9(8).
03 WS-COMM-LAST-STMT-DT      PIC 9(8).
03 WS-COMM-NEXT-STMT-DT      PIC 9(8).
03 WS-COMM-AVAIL-BAL         PIC S9(10)V99.
03 WS-COMM-ACTUAL-BAL        PIC S9(10)V99.
03 WS-COMM-SUCCESS           PIC X.
03 WS-COMM-FAIL-CD           PIC X.
03 WS-COMM-DEL-SUCCESS       PIC X.
03 WS-COMM-DEL-FAIL-CD       PIC X.
```

Then amend the commarea itself DFHCOMMAREA, as highlighted:

```
01 DFHCOMMAREA.
   03 COMM-EYE                PIC X(4).
   03 COMM-CUSTNO             PIC X(10).
   03 COMM-SCODE              PIC X(6).
   03 COMM-ACCNO              PIC 9(9).
   03 COMM-ACC-TYPE           PIC X(8).
   03 COMM-INT-RATE           PIC 9(6).
   03 COMM-OPENED             PIC 9(8).
   03 COMM-OVERDRAFT          PIC 9(8).
   03 COMM-LAST-STMT-DT       PIC 9(8).
   03 COMM-NEXT-STMT-DT       PIC 9(8).
   03 COMM-AVAIL-BAL          PIC S9(10)V99.
   03 COMM-ACTUAL-BAL         PIC S9(10)V99.
   03 COMM-SUCCESS            PIC X.
   03 COMM-FAIL-CD            PIC X.
   03 COMM-DEL-SUCCESS        PIC X.
   03 COMM-DEL-FAIL-CD        PIC X.
```

Having just modified variables;
PARMS-SUBPGM-ACCNO,
INQACC-ACCNO,
WS-COMM-ACCNO,
COMM-ACCNO
(and also indirectly the **ACCNO** and **ACCNO2** in the BNK1DAM copy book), we need to look at all places in program BNK1DAC where these variables have been utilised, to ensure that where these variables get values assigned from, can cope with a 9 byte account number, and where these variables get assigned to other variables, that those other variables can cope with a 9 byte long account number. In this instance, the variables that they interact with are already 9 bytes, so no further changes need to be made.

Finally, because we added an extra byte to the account number in the commarea we need to increase the commarea length by 1 (from 102 to 103), as highlighted below:

```
…
EXEC CICS
   RETURN TRANSID('ODAC')
   COMMAREA(WS-COMM-AREA)
   LENGTH(103)
   RESP(WS-CICS-RESP)
   RESP2(WS-CICS-RESP2)
END-EXEC.
```

Save these changes (use cntrl S to save).

Changing the Inquire Account backend program (INQACC):

6. Program INQACC is responsible for getting the account data from the Db2 ACCOUNT table and either returning those values or returning an appropriate error condition back to the calling program (BNK1DAC).

   Program INQACC accesses the Db2 ACCOUNT data using an SQL DECLARE held in copybook **ACCDB2** and also the ACCOUNT data held in the **ACCOUNT** copybook. Both of these copybooks have already been changed, so no further action is required.

   However, the matching host variable for ACCDB2 needs to be changed. Edit program INQACC in the GitHub /cobol folder and change HV-ACCOUNT-ACC-NO from PIC X(8) to PIC X(9), as highlighted below:

```
* ACCOUNT Host variables for DB2
 01 HOST-ACCOUNT-ROW.
    03 HV-ACCOUNT-EYECATCHER     PIC X(4).
    03 HV-ACCOUNT-CUST-NO        PIC X(10).
    03 HV-ACCOUNT-SORTCODE       PIC X(6).
    03 HV-ACCOUNT-ACC-NO         PIC X(9).
    03 HV-ACCOUNT-ACC-TYPE       PIC X(8).
    03 HV-ACCOUNT-INT-RATE       PIC S9(4)V99 COMP-3.
    03 HV-ACCOUNT-OPENED         PIC X(10).
    03 HV-ACCOUNT-OVERDRAFT-LIM  PIC S9(9) COMP.
    03 HV-ACCOUNT-LAST-STMT      PIC X(10).
    03 HV-ACCOUNT-NEXT-STMT      PIC X(10).
    03 HV-ACCOUNT-AVAIL-BAL      PIC S9(10)V99 COMP-3.
    03 HV-ACCOUNT-ACTUAL-BAL     PIC S9(10)V99 COMP-3.
```

   Having just modified variables;
   HV-ACCOUNT-ACC-NO
   (and also indirectly the **ACCNO** and **ACCNO2** in the BNK1DAM copy book
   We need to look at all places in program INQACC where these variables have been utilised, to ensure that where these variables get values assigned from, can cope with a 9 byte account number, and where these variables get assigned to other variables, that those other variables can cope with a 9 byte long account number. In this instance, the variables that they interact with are already 9 bytes, so no further changes need to be made.

   Variables INQACC-ACCNO and REQUIRED-ACC-NUMBER2 both moved data into **HV-ACCOUNT-ACC-NO. INQACC-ACCNO** comes from the ACCOUNT copybook, which has already been modified – so no change is required for that.

   In program INQACC **change REQUIRED-ACC-NUMBER2** from **PIC 9(8)** to **PIC 9(9)**, as highlighted below:

```
01 ACCOUNT-KY2.
```

```
03 REQUIRED-SORT-CODE2        PIC 9(6) VALUE 0.
03 REQUIRED-ACC-NUMBER2       PIC 9(9) VALUE 0.
```

REQUIRED-ACC-NUMBER2 is also used in other places but these variables have already been amended.

No further changes are required in INQACC for **ACCNO** and **ACCNO2**.

Save these changes (use cntrl S to save).

The Delete Account functionality:

Option on the BMS main menu	***Purpose***	***Mapset and Map affected***	***BMS display program affected***	***Back end program affected***
Option 2 then pF5	Delete Account	BNK1DAM (mapset) BNK1DA (map)	BNK1DAC	DELACC

The mapset and the BMS validation program are the same as for Display Account (which have been amended previously). All that remains to be changed is the backend program DELACC.

Changing the Delete Account backend program (DELACC):

1. Program DELACC is responsible for deleting the account data from the Db2 ACCOUNT table and either returning a confirmation that the delete was successful or an appropriate error condition back to the calling program (BNK1DAC).

   Program DELACC accesses the Db2 ACCOUNT data using an SQL DECLARE held in copybook **ACCDB2** and also the ACCOUNT data held in the **ACCOUNT** copybook. Both of these copybooks have already been changed, so no further action is required.
   Copybook DELACC is utilised as a commarea.

   Change the COPYBOOK DELACC:
   Edit DELACC in the GitHub /copylib folder, and make the following change highlighted in yellow:

```
01 DELACC-COMMAREA.
  03 DELACC-EYE                  PIC X(4).
  03 DELACC-CUSTNO               PIC X(10).
  03 DELACC-SCODE                PIC X(6).
  03 DELACC-ACCNO                PIC 9(9).
  03 DELACC-ACC-TYPE             PIC X(8).
  03 DELACC-INT-RATE             PIC 9(4)V99.
  03 DELACC-OPENED               PIC 9(8).
  03 DELACC-OPENED-GROUP REDEFINES DELACC-OPENED.
    05 DELACC-OPENED-DAY   PIC 99.
    05 DELACC-OPENED-MONTH PIC 99.
    05 DELACC-OPENED-YEAR  PIC 9999.
  03 DELACC-OVERDRAFT            PIC 9(8).
  03 DELACC-LAST-STMT-DT         PIC 9(8).
  03 DELACC-LAST-STMT-GROUP REDEFINES DELACC-LAST-STMT-DT.
    05 DELACC-LAST-STMT-DAY   PIC 99.
    05 DELACC-LAST-STMT-MONTH PIC 99.
    05 DELACC-LAST-STMT-YEAR  PIC 9999.
  03 DELACC-NEXT-STMT-DT         PIC 9(8).
  03 DELACC-NEXT-STMT-GROUP REDEFINES DELACC-NEXT-STMT-DT.
    05 DELACC-NEXT-STMT-DAY   PIC 99.
    05 DELACC-NEXT-STMT-MONTH PIC 99.
    05 DELACC-NEXT-STMT-YEAR  PIC 9999.
  03 DELACC-AVAIL-BAL            PIC S9(10)V99.
  03 DELACC-ACTUAL-BAL           PIC S9(10)V99.
  03 DELACC-SUCCESS              PIC X.
  03 DELACC-FAIL-CD              PIC X.
```

```
      03 DELACC-DEL-SUCCESS          PIC X.
      03 DELACC-DEL-FAIL-CD          PIC X.
      03 DELACC-DEL-APPLID           PIC X(8).
      03 DELACC-DEL-PCB1             POINTER.
      03 DELACC-DEL-PCB2             POINTER.
      03 DELACC-DEL-PCB3             POINTER.
```

Then save this change (use cntrl S to save).

Back in program DELACC (edit program DELACC in the GitHub /cobol folder) , the matching host variable for ACCDB2 needs to be changed. Change HV-ACCOUNT-ACC-NO from PIC X(8) to PIC X(9), as highlighted below:

```
* ACCOUNT Host variables for DB2
 01 HOST-ACCOUNT-ROW.
      03 HV-ACCOUNT-EYECATCHER     PIC X(4).
      03 HV-ACCOUNT-CUST-NO        PIC X(10).
      03 HV-ACCOUNT-SORTCODE       PIC X(6).
      03 HV-ACCOUNT-ACC-NO         PIC X(9).
      03 HV-ACCOUNT-ACC-TYPE       PIC X(8).
      03 HV-ACCOUNT-INT-RATE       PIC S9(4)V99 COMP-3.
      03 HV-ACCOUNT-OPENED         PIC X(10).
      03 HV-ACCOUNT-OVERDRAFT-LIM  PIC S9(9) COMP.
      03 HV-ACCOUNT-LAST-STMT      PIC X(10).
      03 HV-ACCOUNT-NEXT-STMT      PIC X(10).
      03 HV-ACCOUNT-AVAIL-BAL      PIC S9(10)V99 COMP-3.
      03 HV-ACCOUNT-ACTUAL-BAL     PIC S9(10)V99 COMP-3.
```

DELACC also records information about the account being deleted on the PROCTRAN Db2 table. This utilises copybook PROCDB2 which has already been changed.
We do however need to amend the PROCTRAN host variable HV-PROCTRAN-ACC-NUMBER from PIC X(8) to PIC X(9), as highlighted below:

```
* PROCTRAN host variables for DB2
 01 HOST-PROCTRAN-ROW.
      03 HV-PROCTRAN-EYECATCHER    PIC X(4).
      03 HV-PROCTRAN-SORT-CODE     PIC X(6).
      03 HV-PROCTRAN-ACC-NUMBER    PIC X(9).
      03 HV-PROCTRAN-DATE          PIC X(10).
      03 HV-PROCTRAN-TIME          PIC X(6).
      03 HV-PROCTRAN-REF           PIC X(12).
      03 HV-PROCTRAN-TYPE          PIC X(3).
      03 HV-PROCTRAN-DESC          PIC X(40).
      03 HV-PROCTRAN-AMOUNT        PIC S9(10)V99 COMP-3.
```

Having just modified variables;
HV-ACCOUNT-ACC-NO
HV-PROCTRAN-ACC-NUMBER
DELACC-ACCNO
We need to look at all places in program DELACC where these variables have been utilised, to ensure that where these variables get values assigned from, can cope with a 9 byte account number, and where these variables get assigned to other variables, that those other variables can cope with a 9 byte long account number.

In this instance, the variables that they interact with are already 9 bytes, so no further changes need to be made to program DELACC.

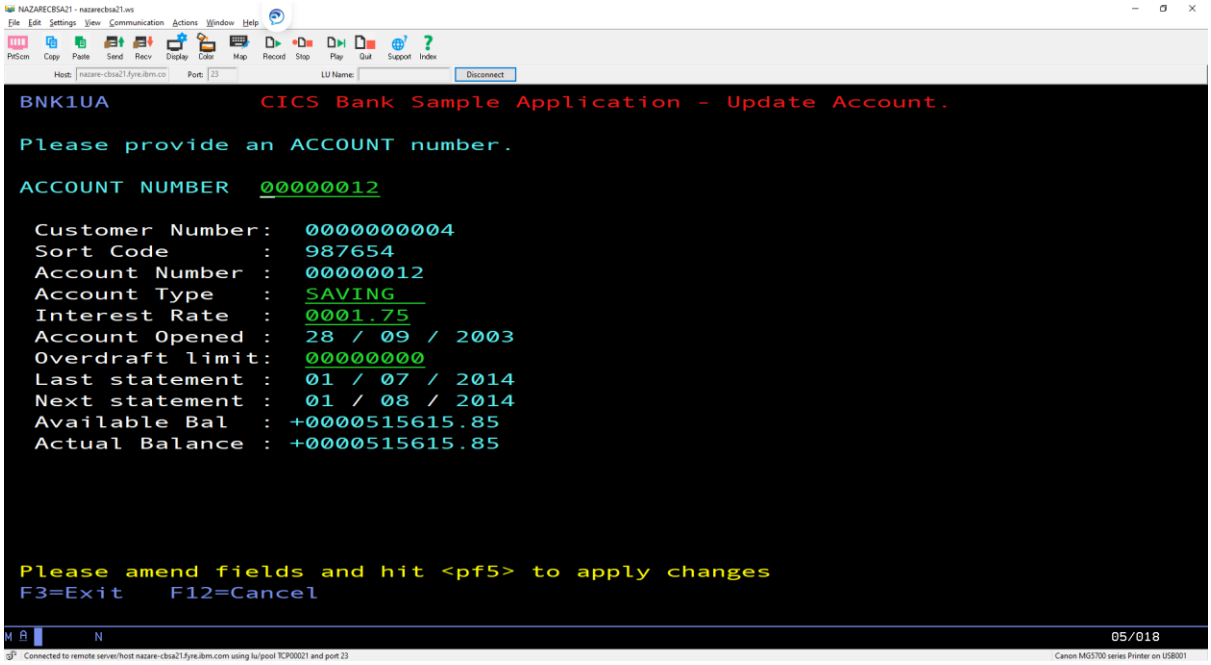
Save these changes (use cntrl S to save).

The Update Account functionality:

Option on the BMS main menu	***Purpose***	***Mapset and Map affected***	***BMS display program affected***	***Back end program affected***
Option 5	Update Account	BNK1UAM (mapset) BNK1UA (map)	BNK1UAC	UPDACC

Changing the Update Account BMS map (BNK1UAM)

1. Currently the Update Account option (option 5 from the BMS main menu) looks like this:



   The end user enters an account number and then the information is retrieved and displayed. There are several things that will need to change:

   - The input account number (in green) needs to be able to accommodate 9 bytes instead of 8.
   - The Account Number returned (shown in blue) will need to become 9 bytes.
   - Any account data passed to or from the BMS validation program will need to be 9 bytes.

2. Edit BMS mapset BNK1UAM from /bms folder in the Git Hub repo.

   Change the length of ACCNO from 8 to 9 bytes as highlighted in yellow below. Then amend the position of the stop byte from 5,26 to 5,27

```
                …

                DFHMDF POS=(5,1),LENGTH=15,ATTRB=(NORM,PROT),COLOR=TURQUOISE, *
                      INITIAL='ACCOUNT NUMBER'
ACCNO           DFHMDF POS=(5,17),LENGTH=9,ATTRB=(NORM,NUM,IC,UNPROT),       *
                      COLOR=GREEN,                                          *
```

```
            HILIGHT=UNDERLINE
DFHMDF  POS=(5,27),LENGTH=1,ATTRB=(PROT,ASKIP)
DFHMDF  POS=(7,1),LENGTH=18,ATTRB=(NORM,PROT),                  *
                COLOR=NEUTRAL,INITIAL=' Customer Number:'
…
```

Next, we would expect to amend the length of ACCNO2 to make it 9 bytes, but it is already defined as having LENGTH(10):

```
ACCNO2    DFHMDF  POS=(9,20),LENGTH=10,                         *
            ATTRB=(PROT,NORM),COLOR=TURQUOISE
```

so no further changes are necessary:

3. **Then save this change** (use cntrl S to save). **NOTE** if the bms source does not save with Cntrl S, close the tab and you will be prompted to save at that point (click the Save button).

   As a result of changing the BMS map, the generated BMS symbolic map/DSECT (also called BNK1UAM) will automatically get changed when the BMS map is rebuilt.

Changing the Update Account BMS validation program (BNK1UAC):

4. Program BNK1UAC is responsible for validating the data coming from the Update Account BMS map and data, from the backend program, being output to the BMS map. Since the BMS map has now been changed to allow an extra byte for each instance of the account number, it follows that we will also need to ensure that any account number data passed into or used by BNK1UAC is also 9 bytes long.

5. Edit COBOL program **BNK1UAC** from the /cobol folder in the Git Hub repo. This pulls in the copybook:
   - BNK1UAM copybook. This copybook will get regenerated (with 9 byte account numbers) when the BMS map (above) gets built – so no further action is necessary.

   Change WS-COMM-AREA (which holds a copy of the COMMAREA) and amend the size of WS-COMM-ACCNO from PIC 9(8) to PIC 9(9), as highlighted:

```
01 WS-COMM-AREA.
   03 WS-COMM-EYE              PIC X(4).
   03 WS-COMM-CUSTNO           PIC X(10).
   03 WS-COMM-SCODE            PIC X(6).
   03 WS-COMM-ACCNO            PIC 9(9).
   03 WS-COMM-ACC-TYPE         PIC X(8).
   03 WS-COMM-INT-RATE         PIC 9(4)V99.
   03 WS-COMM-OPENED           PIC 9(8).
```

```
           03 WS-COMM-OVERDRAFT         PIC 9(8).
           03 WS-COMM-LAST-STMT-DT      PIC 9(8).
           03 WS-COMM-NEXT-STMT-DT      PIC 9(8).
           03 WS-COMM-AVAIL-BAL         PIC S9(10)V99.
           03 WS-COMM-ACTUAL-BAL        PIC S9(10)V99.
           03 WS-COMM-SUCCESS           PIC X.
```

Change the COMMAREA itself and amend COMM-ACCNO in the same way:

```
       LINKAGE SECTION.
       01 DFHCOMMAREA.
          03 COMM-EYE                  PIC X(4).
          03 COMM-CUSTNO               PIC X(10).
          03 COMM-SCODE                PIC X(6).
          03 COMM-ACCNO                PIC 9(9).
          03 COMM-ACC-TYPE             PIC X(8).
          03 COMM-INT-RATE             PIC 9(4)V99.
          03 COMM-OPENED               PIC 9(8).
          03 COMM-OVERDRAFT            PIC 9(8).
          03 COMM-LAST-STMT-DT         PIC 9(8).
          03 COMM-NEXT-STMT-DT         PIC 9(8).
          03 COMM-AVAIL-BAL            PIC S9(10)V99.
          03 COMM-ACTUAL-BAL           PIC S9(10)V99.
          03 COMM-SUCCESS              PIC X.
          03 COMM-PCB1-POINTER         POINTER.
```

Having just modified variables;
WS-COMM-ACCNO,
COMM-ACCNO,
(and also indirectly the **ACCNO** and **ACCNO2** variables in the BNK1UAM copy book), we need to look at all places in program BNK1UAC where these variables have been utilised, to ensure that where these variables get values assigned from, can cope with a 9 byte account number, and where these variables get assigned to other variables, that those other variables can cope with a 9 byte long account number. In this instance, the variables that they interact with are already 9 bytes, so no further changes are needed for this.

Finally, because we added an extra byte to the account number in the commarea we need to increase the commarea length by 1 (from 99 to 100), as highlighted below:

```
           …
          EXEC CICS
             RETURN TRANSID('OUAC')
             COMMAREA(WS-COMM-AREA)
             LENGTH(100)
             RESP(WS-CICS-RESP)
             RESP2(WS-CICS-RESP2)
          END-EXEC.
           …
```

Save these changes (use cntrl S to save).

Changing the Update Account backend program (UPDACC):

6. Program UPDACC is responsible for updating the account data on the Db2 ACCOUNT table and either returning those values or returning an appropriate error condition back to the calling program (BNK1UAC).

   Program UPDACC accesses the Db2 ACCOUNT data using an SQL DECLARE held in copybook **ACCDB2** and also the ACCOUNT data held in the **ACCOUNT** copybook. Both of these copybooks have already been changed, so no further action is required.

   However, the matching host variable for ACCDB2 needs to be changed.

   Edit COBOL program UPDACC from the /cobol folder in the Git Hub repo, then change HV-ACCOUNT-ACC-NO from PIC X(8) to PIC X(9), as highlighted below:

```
* ACCOUNT Host variables for DB2
 01 HOST-ACCOUNT-ROW.
    03 HV-ACCOUNT-EYECATCHER      PIC X(4).
    03 HV-ACCOUNT-CUST-NO         PIC X(10).
    03 HV-ACCOUNT-KEY.
       05 HV-ACCOUNT-SORTCODE     PIC X(6).
       05 HV-ACCOUNT-ACC-NO       PIC X(9).
    03 HV-ACCOUNT-ACC-TYPE        PIC X(8).
    03 HV-ACCOUNT-INT-RATE        PIC S9(4)V99 COMP-3.
    03 HV-ACCOUNT-OPENED          PIC X(10).
    03 HV-ACCOUNT-OVERDRAFT-LIM   PIC S9(9) COMP.
    03 HV-ACCOUNT-LAST-STMT       PIC X(10).
    03 HV-ACCOUNT-NEXT-STMT       PIC X(10).
    03 HV-ACCOUNT-AVAIL-BAL       PIC S9(10)V99 COMP-3.
    03 HV-ACCOUNT-ACTUAL-BAL      PIC S9(10)V99 COMP-3.
```

   Program UPDACC has a copybook defined for the COMMAREA called UPDACC.

   Change the COPYBOOK UPDACC:
   Edit UPDACC in the GitHub /copylib folder, and make the following change highlighted in yellow:

```
    03 COMM-EYE                  PIC X(4).
    03 COMM-CUSTNO               PIC X(10).
    03 COMM-SCODE                PIC X(6).
    03 COMM-ACCNO                PIC 9(9).
    03 COMM-ACC-TYPE             PIC X(8).
    03 COMM-INT-RATE             PIC 9(4)V99.
    03 COMM-OPENED               PIC 9(8).
    03 COMM-OPENED-GROUP REDEFINES COMM-OPENED.
       05 COMM-OPENED-DAY              PIC 99.
       05 COMM-OPENED-MONTH            PIC 99.
       05 COMM-OPENED-YEAR             PIC 9999.
    03 COMM-OVERDRAFT            PIC 9(8).
    03 COMM-LAST-STMT-DT         PIC 9(8).
    03 COMM-LAST-STMNT-GROUP REDEFINES COMM-LAST-STMT-DT.
       05 COMM-LASTST-DAY              PIC 99.
       05 COMM-LASTST-MONTH            PIC 99.
       05 COMM-LASTST-YEAR             PIC 9999.
```

```
          03 COMM-NEXT-STMT-DT         PIC 9(8).
          03 COMM-NEXT-STMNT-GROUP REDEFINES COMM-NEXT-STMT-DT.
             05 COMM-NEXTST-DAY              PIC 99.
             05 COMM-NEXTST-MONTH            PIC 99.
             05 COMM-NEXTST-YEAR             PIC 9999.
          03 COMM-AVAIL-BAL            PIC S9(10)V99.
          03 COMM-ACTUAL-BAL           PIC S9(10)V99.
          03 COMM-SUCCESS              PIC X.
```

Then save this change (use cntrl S to save).

Back in program UPDACC. Having just modified variables;
HV-ACCOUNT-ACC-NO
COMM-ACCNO

(and also indirectly the **ACCNO** and **ACCNO2** in the BNK1UAM copy book).

We need to look at all places in program UPDACC where these variables have values assigned and where these variables are used to assign values to other variables (now that these variables have been modified to be 9 bytes long).

Variable DESIRED-ACC-NO gets moved into **HV-ACCOUNT-ACC-NO.** Change **DESIRED-ACC-NO** from **PIC 9(8)** to **PIC 9(9)**, as highlighted below:

```
      *
      * Pull in the input and output data structures
      *
       01 DESIRED-ACC-KEY.
          03 DESIRED-SORT-CODE          PIC 9(6).
          03 DESIRED-ACC-NO             PIC 9(9).
```

No further changes are required in UPDACC for **COMM-ACCNO**, **ACCNO** and **ACCNO2**.

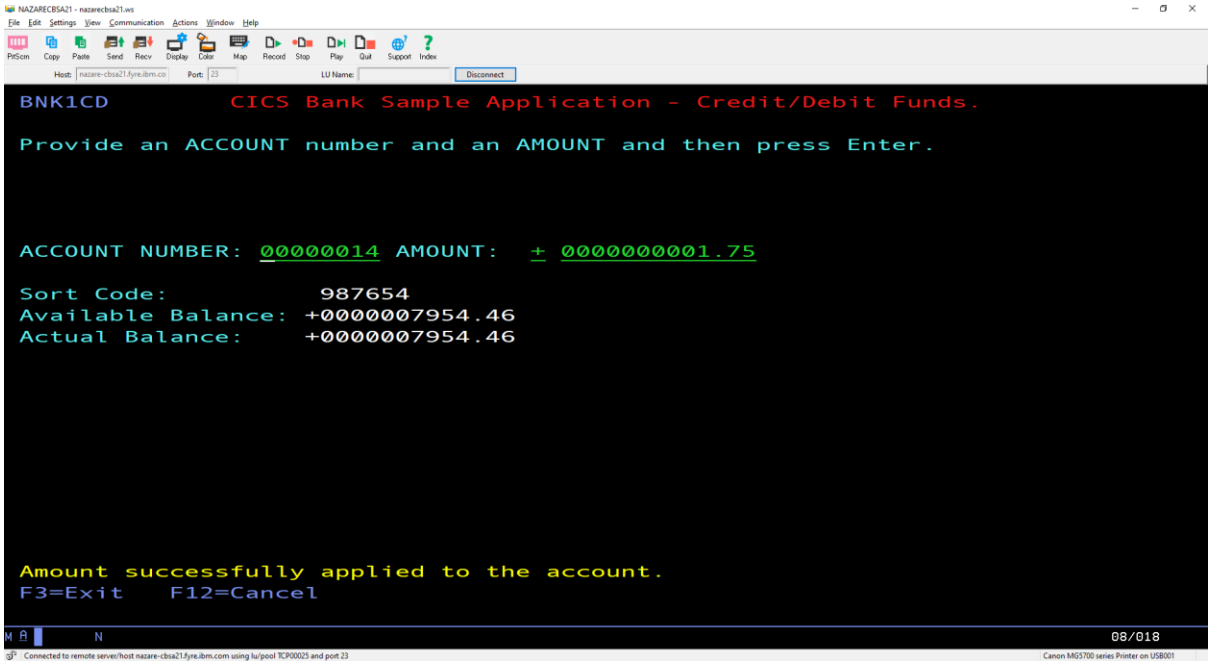
Save these changes (use cntrl S to save).

The Credit/Debit Funds (to an account) functionality:

Option on the BMS main menu	***Purpose***	***Mapset and Map affected***	***BMS display program affected***	***Back end program affected***
Option 6	Credit/Debit funds (to an account)	BNK1CDM(mapset) BNK1CD (map)	BNK1CRA	DBCRFUN

Changing the Credit/Debit Account BMS map (BNK1CDM)

1. Currently the Credit/Debit funds option (option 6 from the BMS main menu) looks like this:



   The end user enters an account number, a + or - sign (for credit or debit), followed by an amount. The update is then applied and assuming it is successful the updated balance is displayed. The fields in green are the input fields. We will need to change the input account number to accommodate 9 bytes instead of 8.

2. Edit BMS mapset **BNK1CDM** from /bms folder in the Git Hub repo.

   Change the length of ACCNO from 8 to 9 bytes. Then amend the position of the next attribute from POS=(8,26) to (8,27), and everything else on line 8 must then be pushed along by one byte. These changes are highlighted below:

```
...
         DFHMDF POS=(8,1),LENGTH=15,ATTRB=(NORM,PROT),COLOR=TURQUOISE, *
               INITIAL='ACCOUNT NUMBER:'
ACCNO    DFHMDF POS=(8,17),LENGTH=9,ATTRB=(NORM,NUM,IC,FSET),          *
               COLOR=GREEN,                                           *
               HILIGHT=UNDERLINE
         DFHMDF POS=(8,27),LENGTH=7,ATTRB=(NORM,PROT,ASKIP),           *
               COLOR=TURQUOISE,                                       *
               INITIAL='AMOUNT:'
```

```
SIGN      DFHMDF POS=(8,36),LENGTH=1,ATTRB=(NORM,UNPROT,FSET),          *
               COLOR=GREEN,INITIAL='+',HILIGHT=UNDERLINE

AMT       DFHMDF POS=(8,38),LENGTH=13,ATTRB=(NORM,UNPROT,FSET),         *
               COLOR=GREEN,INITIAL='0000000000.00',HILIGHT=UNDERLINE
          DFHMDF POS=(8,52),LENGTH=1,ATTRB=(PROT,ASKIP),                *
               COLOR=GREEN,INITIAL=' '
          DFHMDF POS=(10,1),LENGTH=16,ATTRB=(NORM,PROT),COLOR=TURQUOISE,*
                     INITIAL='Sort Code: '

…
```

There are no other changes required on this map.

7. **Save this change** (use cntrl S to save). **NOTE** if the bms source does not save with Cntrl S, close the tab and you will be prompted to save at that point (click the Save button).

   As a result of changing the BMS map, the generated BMS symbolic map/DSECT (also called BNK1CDM) will automatically get changed when the BMS map is rebuilt.

Changing the Credit/Debit Account BMS validation program (BNK1CRA):

3. Program BNK1CRA is responsible for validating the data coming from the Credit/Debit Account BMS map and data, from the backend program, being output to the BMS map. Since we have now changed the BMS map to allow an extra byte for each instance of the account number, it follows that we will also need to ensure that any account number data passed into or used by BNK1CRA is also 9 bytes long.

4. Edit COBOL program **BNK1CRA** from the /cobol folder in the Git Hub repo. This pulls in the following copybooks:
   - BNK1CDM copybook, should be regenerated (with 9 byte account numbers) when the BMS map (above) gets reassembled – so no further action is necessary.
   - ABNDPROC copybook, has no account numbers in it, so further action is necessary.

   Change WS-COMM-AREA (which holds a copy of the COMMAREA) and amend the size of WS-COMM-ACCNO from PIC X(8) to PIC X(9), as highlighted:

```
01 WS-COMM-AREA.
   03 WS-COMM-ACCNO             PIC X(9).
   03 WS-COMM-SIGN               PIC X.
   03 WS-COMM-AMT                PIC 9(12).
```

   We also need to change the COMMAREA itself and amend COMM-ACCNO in the same way:

```
LINKAGE SECTION.
01 DFHCOMMAREA.
   03 COMM-ACCNO                 PIC X(9).
```

```
        03 COMM-SIGN                 PIC X.
        03 COMM-AMT                  PIC 9(12).
```

Having just modified variables;
WS-COMM-ACCNO,
COMM-ACCNO,
(and also indirectly the **ACCNO** variable in the BNK1CDM copy book), we need to look at all places in program BNK1CRA where these variables have been utilised, to ensure that where these variables get values assigned from, can cope with a 9 byte account number, and where these variables get assigned to other variables, that those other variables can cope with a 9 byte account number.

In the PREMIERE SECTION amend the WHEN OTHER statement to set the length of the ACCNOL to be 9 instead of 8 :

```
   WHEN OTHER
      MOVE LOW-VALUES TO BNK1CDO
      MOVE 'Invalid key pressed.' TO MESSAGEO
      MOVE 9 TO ACCNOL
      SET SEND-DATAONLY-ALARM TO TRUE
       PERFORM SEND-MAP

   END-EVALUATE.
```

Variable SUBPGM-ACCNO is a variable passed to another program, so this should be amended to be 9 bytes too (as highlighted):

```
    01 SUBPGM-PARMS.
       03 SUBPGM-ACCNO              PIC X(9).
       03 SUBPGM-AMT                PIC S9(10)V99.
       03 SUBPGM-SORTC              PIC 9(6).
       03 SUBPGM-AV-BAL             PIC S9(10)V99.
       03 SUBPGM-ACT-BAL            PIC S9(10)V99.
       03 SUBPGM-SUCCESS            PIC X.
       03 SUBPGM-FAIL-CODE          PIC X.
```

Finally, because we added an extra byte to the account number in the commarea we need to increase the commarea length by 1 (from 21 to 22), as highlighted below:

```
        …
       EXEC CICS
          RETURN TRANSID('OCRA')
          COMMAREA(WS-COMM-AREA)
          LENGTH(22)
          RESP(WS-CICS-RESP)
          RESP2(WS-CICS-RESP2)
       END-EXEC.
        …
```

Save these changes (use cntrl S to save).

Changing the Credit/Debit Account backend program (DBCRFUN):

5. Program DBCRFUN is responsible for applying the debit or credit, updating the account data on the Db2 ACCOUNT table, and recording the successful transaction on the PROCTRAN (Successfully Processed Transactions) table.

   Program DBCRFUN accesses the Db2 ACCOUNT data using an SQL DECLARE held in copybook **ACCDB2** and also the ACCOUNT data held in the **ACCOUNT** copybook. Both of these copybooks have already been changed, so no further action is required.

   However, the matching host variable for ACCDB2 needs to be changed.

   Edit COBOL program **DBCRFUN** from the /cobol folder in the Git Hub and change HV-ACCOUNT-ACC-NO from PIC X(8) to PIC X(9), as highlighted below:

```
* ACCOUNT Host variables for DB2
 01 HOST-ACCOUNT-ROW.
    03 HV-ACCOUNT-EYECATCHER      PIC X(4).
    03 HV-ACCOUNT-CUST-NO         PIC X(10).
    03 HV-ACCOUNT-KEY.
       05 HV-ACCOUNT-SORTCODE     PIC X(6).
       05 HV-ACCOUNT-ACC-NO       PIC X(9).
    03 HV-ACCOUNT-ACC-TYPE        PIC X(8).
    03 HV-ACCOUNT-INT-RATE        PIC S9(4)V99 COMP-3.
    03 HV-ACCOUNT-OPENED          PIC X(10).
    03 HV-ACCOUNT-OVERDRAFT-LIM   PIC S9(9) COMP.
    03 HV-ACCOUNT-LAST-STMT       PIC X(10).
    03 HV-ACCOUNT-NEXT-STMT       PIC X(10).
    03 HV-ACCOUNT-AVAIL-BAL       PIC S9(10)V99 COMP-3.
    03 HV-ACCOUNT-ACTUAL-BAL      PIC S9(10)V99 COMP-3.
```

   Program DBCRFUN accesses the Db2 PROCTRAN data using an SQL DECLARE held in copybook **PROCDB2** and also the PROCTRAN data held in the **PROCTRAN** copybook. Both of these copybooks have already been changed, so no further action is required.

   However, the matching host variable for PROCDB2 needs to be changed.

   In program DBCRFUN change HV-PROCTRAN-ACC-NUMBER from PIC X(8) to PIC X(9), as highlighted below:

```
* PROCTRAN host variables for DB2
 01 HOST-PROCTRAN-ROW.
    03 HV-PROCTRAN-EYECATCHER     PIC X(4).
    03 HV-PROCTRAN-SORT-CODE      PIC X(6).
    03 HV-PROCTRAN-ACC-NUMBER     PIC X(9).
    03 HV-PROCTRAN-DATE           PIC X(10).
    03 HV-PROCTRAN-TIME           PIC X(6).
    03 HV-PROCTRAN-REF            PIC X(12).
```

```
   03 HV-PROCTRAN-TYPE           PIC X(3).
   03 HV-PROCTRAN-DESC           PIC X(40).
   03 HV-PROCTRAN-AMOUNT         PIC S9(10)V99 COMP-3.
```

The remaining copybooks do not need to be changed.

DBCRFUN utilises a COMMAREA containing an account number, this should be changed as highlighted:

```
LINKAGE SECTION.
01 DFHCOMMAREA.
   03 COMM-ACCNO                 PIC X(9).
   03 COMM-AMT                   PIC S9(10)V99.
   03 COMM-SORTC                 PIC 9(6).
   03 COMM-AV-BAL                PIC S9(10)V99.
   03 COMM-ACT-BAL               PIC S9(10)V99.
   03 COMM-SUCCESS               PIC X.
    03 COMM-FAIL-CODE            PIC X.
```

Having just modified variables;
HV-ACCOUNT-ACC-NO
HV-PROCTRAN-ACC-NUMBER
COMM-ACCNO

We need to look at all places in program DBCRFUN where these variables have values assigned and where these variables are used to assign values to other variables (now that these variables have been modified to be 9 bytes long).

Variable DESIRED-ACC-NO gets moved into **HV-ACCOUNT-ACC-NO.** Change **DESIRED-ACC-NO** from **PIC 9(8)** to **PIC 9(9)**, as highlighted below:

```
01 DESIRED-ACC-KEY.
   03 DESIRED-SORT-CODE          PIC 9(6).
   03 DESIRED-ACC-NO             PIC 9(9).
```

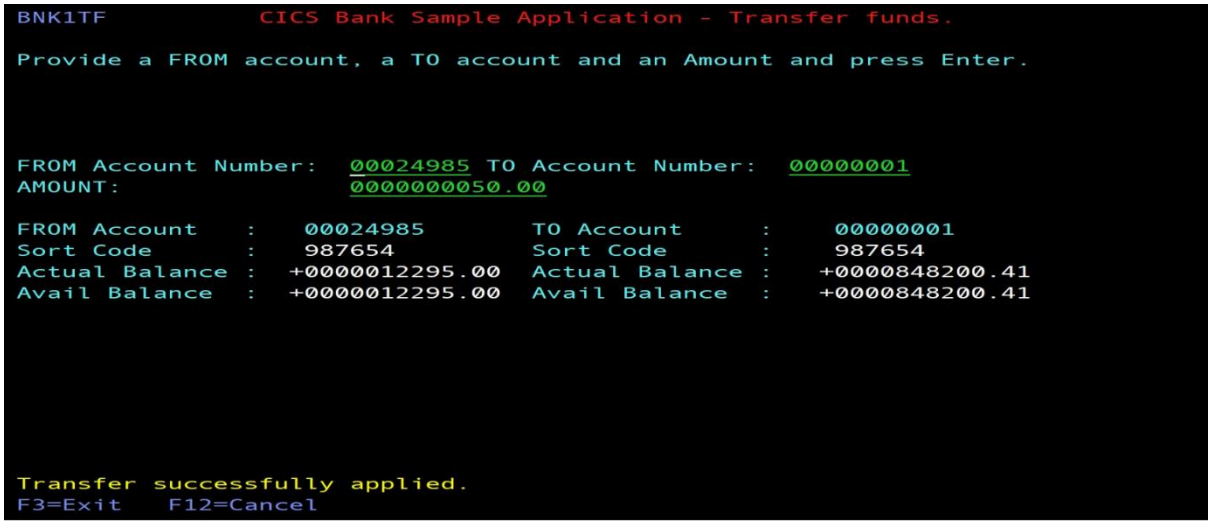
Save these changes (use cntrl S to save).

The Transfer funds between accounts functionality:

Option on the BMS main menu	*Purpose*	*Mapset and Map affected*	*BMS display program affected*	*Back end program affected*
Option 7	Transfer funds between accounts	BNK1TFM (mapset) BNK1TF (map)	BNK1TFN	XFRFUN

Changing the Transfer funds BMS map (BNK1TFM)

1. Currently the Transfer Funds option (option 7 from the BMS main menu) looks like this:



   The end user enters a FROM account number, and an AMOUNT (to be transferred) and a TO account number (the recipient account). The transfer is made and assuming it is successful, the balances of both accounts are displayed below. The fields in green are the input fields. We will need to change the input FROM account number, the input TO account number and the FROM and TO confirmation account numbers to accommodate 9 bytes instead of 8.

2. Edit BMS mapset **BNK1TFM** from /bms folder in the Git Hub repo.

   Change the length of FACCNO from 8 to 9 bytes. Then amend the next field's position from POS=(8,32) to (8,33). Next amend the position of TACCNO from POS(8,52) to POS=(8,53) and also change the length of TACCNO from 8 bytes to 9 bytes. Then amend the stop byte from POS=(8,61) to POS=(8,63). These changes are highlighted below:

```
   …

FACCNO   DFHMDF POS=(8,23),LENGTH=9,ATTRB=(NORM,NUM,IC),COLOR=GREEN,    *
               HILIGHT=UNDERLINE
         DFHMDF POS=(8,33),LENGTH=19,ATTRB=(NORM,PROT,ASKIP),           *
               COLOR=TURQUOISE,                                         *
               INITIAL='TO Account Number:'
TACCNO   DFHMDF POS=(8,53),LENGTH=9,ATTRB=(NORM,NUM),COLOR=GREEN,       *
               HILIGHT=UNDERLINE
         DFHMDF POS=(8,63),LENGTH=1,ATTRB=(PROT,ASKIP)
         DFHMDF POS=(9,1),LENGTH=7,ATTRB=(NORM,PROT),COLOR=TURQUOISE,   *
                     INITIAL='AMOUNT:'
```

Lower down in the BNK1TFM code change the lengths of variables FACCNO2 and TACCNO2 from 8 to 9 bytes. These changes are highlighted below:

```
...

FACCNO2   DFHMDF POS=(11,20),LENGTH=9,ATTRB=(NORM,NUM,PROT),            *
               COLOR=TURQUOISE
          DFHMDF POS=(11,35),LENGTH=16,ATTRB=(NORM,PROT),               *
               COLOR=TURQUOISE,INITIAL='TO Account     :'
TACCNO2   DFHMDF POS=(11,55),LENGTH=9,ATTRB=(NORM,PROT),COLOR=TURQUOISE
```

3. **Save this change** (use cntrl S to save). **NOTE** if the bms source does not save with Cntrl S, close the tab and you will be prompted to save at that point (click the Save button).

   As a result of changing the BMS map, the generated BMS symbolic map/DSECT (also called BNK1TFM) will automatically get changed when the BMS map is rebuilt.

Changing the Transfer Funds BMS validation program (BNK1TFN):

4. Program BNK1TFN is responsible for validating the data coming from the Transfer Funds BMS map and data, from the backend program, being output to the BMS map. Since we have now changed the BMS map to allow an extra byte for each instance of the account number, it follows that we will also need to ensure that any account number data passed into or used by BNKTFN is also 9 bytes long. Additionally, we will need to change the validation to cope with an entered account number of 9 bytes.

5. Edit COBOL program **BNK1TFN** from the /cobol folder in the Git Hub repo. This pulls in the following copybooks:
   - BNK1TFM copybook, should be regenerated (with 9 byte account numbers) when the BMS map (above) gets reassembled – so no further action is necessary.
   - ABNDINFO copybook, has no account numbers in it, so further action is necessary.

   Change COMMAREA-FACCNO AND COMMAREA-TACCNO in DFHCOMMAREA from PIC 9(8) to PIC 9(9), as highlighted:

```
01 DFHCOMMAREA.
   03 COMMAREA-FACCNO             PIC 9(9).
   03 COMMAREA-TACCNO             PIC 9(9).
   03 COMMAREA-AMT                PIC 9(12).
```

   Also amend the working storage copy of the COMM-AREA:

```
01 WS-COMMAREA.
   03 WS-COMMAREA-FACCNO          PIC 9(9).
   03 WS-COMMAREA-TACCNO          PIC 9(9).
   03 WS-COMMAREA-AMT             PIC 9(12).
```

Having just modified variables;
COMMAREA-FACCNO,
COMMAREA-TACCNO,
WS-COMMAREA-FACCNO,
WS-COMMAREA-TACCNO
(and also indirectly the **FACCNO, TACCNO, FACCNO2 and TACCNO2** variables in the BNK1TFM copy book), we need to look at all places in program BNK1TFN where these variables have been utilised, to ensure that where these variables get values assigned from, can cope with a 9 byte account number, and where these variables get assigned to other variables, that those other variables can cope with a 9 byte long account number.

In the EDIT-DATA SECTION amend the following 'if' statement to check whether the account number contains all '0's by adding an additional zero, as highlighted:

```
IF FACCNOI = '000000000' OR TACCNOI = '000000000'
   MOVE 'Account no 000000000 is not valid          ' TO
      MESSAGEO
   MOVE 'N' TO VALID-DATA-SW
   GO TO ED999
END-IF.
```

FACCNO and TACCNO get moved to SUBPGM-FACCNO and SUBPGM-TACCNO respectively, so we will need to change those:

```
01 SUBPGM-PARMS.
   03 SUBPGM-FACCNO              PIC 9(9).
   03 SUBPGM-FSCODE              PIC 9(6).
   03 SUBPGM-TACCNO              PIC 9(9).
   03 SUBPGM-TSCODE              PIC 9(6).
   03 SUBPGM-AMT                 PIC S9(10)V99.
   03 SUBPGM-FAVBAL              PIC S9(10)V99.
   03 SUBPGM-FACTBAL             PIC S9(10)V99.
   03 SUBPGM-TAVBAL              PIC S9(10)V99.
   03 SUBPGM-TACTBAL             PIC S9(10)V99.
   03 SUBPGM-FAIL-CODE           PIC X.
    03 SUBPGM-SUCCESS             PIC X.
```

Finally, because we added two extra bytes to the account numbers in the WS-COMMAREA we need to increase the length by 2 (from 29 to 31), as highlighted below:

```
…
 EXEC CICS
    RETURN TRANSID('OTFN')
    COMMAREA(WS-COMMAREA)
    LENGTH(31)
    RESP(WS-CICS-RESP)
    RESP2(WS-CICS-RESP2)
 END-EXEC.
…
```

Save these changes (use cntrl S to save).

Changing the Transfer Funds backend program (XFRFUN):

6. Program XFRFUN is responsible for transferring funds between the FROM account and the TO account, updating the account data (balances etc.) on the Db2 ACCOUNT table, and recording the successful transaction on the PROCTRAN (Successfully Processed Transactions) table.

   Program XFRFUN accesses the Db2 ACCOUNT data using an SQL DECLARE held in copybook **ACCDB2** and also the ACCOUNT data held in the **ACCOUNT** copybook. Both of these copybooks have already been changed, so no further action is required.
   It also accesses the Db2 PROCTRAN data using an SQL DECLARE held in copybook **PROCDB2** and also the PROCTRAN data held in in the **PROCTRAN** copybook. These have also been changed previously.

   However, the matching host variable for ACCDB2 needs to be changed.

   Edit COBOL program **XFRFUN** from the /cobol folder in the Git Hub repo and change HV-ACCOUNT-ACC-NO from PIC X(8) to PIC X(9), as highlighted below:

```
* ACCOUNT Host variables for DB2
 01 HOST-ACCOUNT-ROW.
    03 HV-ACCOUNT-EYECATCHER      PIC X(4).
    03 HV-ACCOUNT-CUST-NO         PIC X(10).
    03 HV-ACCOUNT-KEY.
       05 HV-ACCOUNT-SORTCODE     PIC X(6).
       05 HV-ACCOUNT-ACC-NO       PIC X(9).
    03 HV-ACCOUNT-ACC-TYPE        PIC X(8).
    03 HV-ACCOUNT-INT-RATE        PIC S9(4)V99 COMP-3.
    03 HV-ACCOUNT-OPENED          PIC X(10).
    03 HV-ACCOUNT-OVERDRAFT-LIM   PIC S9(9) COMP.
    03 HV-ACCOUNT-LAST-STMT       PIC X(10).
    03 HV-ACCOUNT-NEXT-STMT       PIC X(10).
    03 HV-ACCOUNT-AVAIL-BAL       PIC S9(10)V99 COMP-3.
    03 HV-ACCOUNT-ACTUAL-BAL      PIC S9(10)V99 COMP-3.
```

   Change the host variable for PROCDB2 as follows:

```
* PROCTRAN host variables for DB2
 01 HOST-PROCTRAN-ROW.
    03 HV-PROCTRAN-EYECATCHER     PIC X(4).
    03 HV-PROCTRAN-SORT-CODE      PIC X(6).
    03 HV-PROCTRAN-ACC-NUMBER     PIC X(9).
    03 HV-PROCTRAN-DATE           PIC X(10).
    03 HV-PROCTRAN-TIME           PIC X(6).
    03 HV-PROCTRAN-REF            PIC X(12).
    03 HV-PROCTRAN-TYPE           PIC X(3).
    03 HV-PROCTRAN-DESC           PIC X(40).
    03 HV-PROCTRAN-AMOUNT         PIC S9(10)V99 COMP-3.
```

   The copybook ABNDINFO has no account numbers in it, so no further action is necessary.

   Program XFRFUN has a copybook defined for the COMMAREA called XFRFUN.

Change the COPYBOOK XFRFUN:
Edit XFRFUN in the GitHub /copylib folder, and make the following changes highlighted in yellow:

```
* Licensed Materials - Property of IBM
*
* (c) Copyright IBM Corp. 2017.
*
* US Government Users Restricted Rights - Use, duplication or
* disclosure restricted by GSA ADP Schedule Contract
* with IBM Corp.
    03 COMM-FACCNO                PIC 9(9).
    03 COMM-FSCODE                PIC 9(6).
    03 COMM-TACCNO                PIC 9(9).
    03 COMM-TSCODE                PIC 9(6).
    03 COMM-AMT                   PIC S9(10)V99.
    03 COMM-FAVBAL                PIC S9(10)V99.
    03 COMM-FACTBAL               PIC S9(10)V99.
    03 COMM-TAVBAL                PIC S9(10)V99.
    03 COMM-TACTBAL               PIC S9(10)V99.
    03 COMM-FAIL-CODE             PIC X.
     03 COMM-SUCCESS              PIC X.
```

Then save this change (use cntrl S to save).

Back in program XFRFUN. Having just modified variables;
HV-ACCOUNT-ACC-NO
HV-PROCTRAN-ACC-NUMBER
COMM-FACCNO
COMM-TACCNO

We need to look at all places in program XFRFUN where these variables have values assigned and where these variables are used to assign values to other variables (now that these variables have been modified to be 9 bytes long).

Variable DESIRED-ACC-NO gets moved to HV-ACCOUNT-ACC-NO, so we must change it as follows:

```
01 DESIRED-ACC-KEY.
   03 DESIRED-SORT-CODE          PIC 9(6).
   03 DESIRED-ACC-NO             PIC 9(9).
```

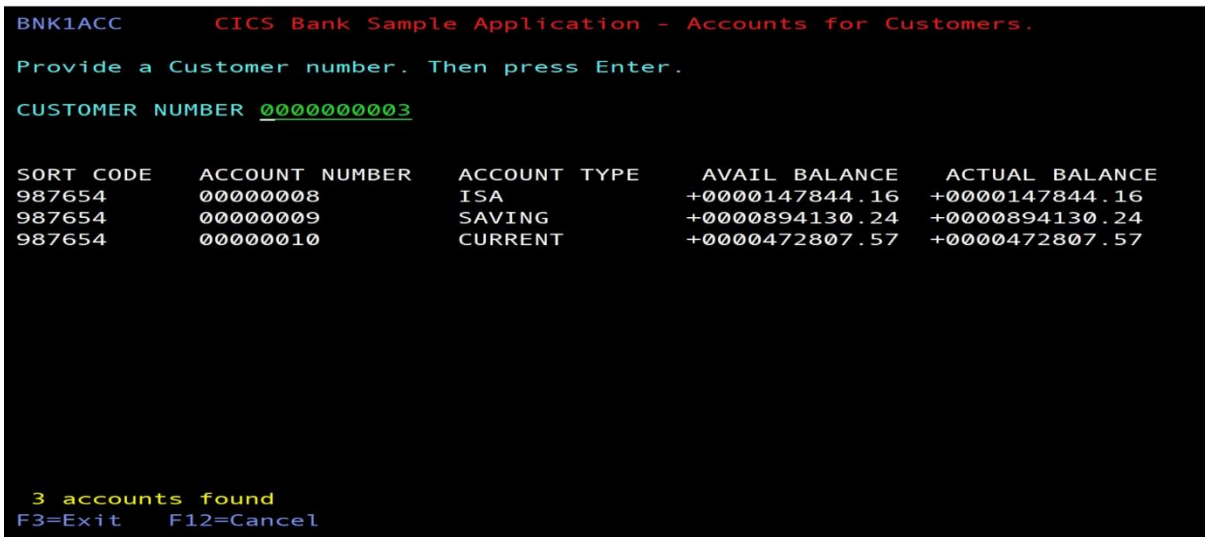
Save these changes (use cntrl S to save).

The Look Up Accounts with customer number functionality:

Option on the BMS main menu	*Purpose*	*Mapset and Map affected*	*BMS display program affected*	*Back end program affected*
Option A	Look up accounts with customer number	BNK1ACC (mapset) BNK1AC (map)	BNK1CCA	INQACCCU

Changing the Look up accounts with customer number BMS map (BNK1ACC)

1. Currently the Look up accounts with customer number option (option A from the BMS main menu) looks like this:



   The end user enters a customer number and all of the accounts associated with the customer number get returned. The ACCOUNT NUMBERs returned are currently displayed in an 8 byte format, this must be changed.

2. Changing the BMS mapset **BNK1ACC** in /bms folder in the Git Hub repo should **NOT** be necessary. This is because the details returned for each account get displayed as a single line of text information (so the account number is not specifically referenced), as can be seen in the variable ACCOUNT below.

```
BNK1ACC  DFHMSD TYPE=&SYSPARM,MODE=INOUT,LANG=COBOL,STORAGE=AUTO,       *
               CTRL=FREEKB,EXTATT=YES,TERM=3270-2,TIOAPFX=YES,          *
               MAPATTS=(COLOR,HILIGHT,OUTLINE,PS,SOSI),                 *
               DSATTS=(COLOR,HILIGHT,OUTLINE,PS,SOSI)
BNK1ACC  DFHMDI SIZE=(24,80),                                           *
               COLUMN=1,LINE=1
         DFHMDF POS=(1,1),LENGTH=7,INITIAL='BNK1ACC',                   *
               ATTRB=(PROT,NORM),COLOR=BLUE
COMPANY  DFHMDF POS=(1,14),LENGTH=60,ATTRB=(NORM,PROT),COLOR=RED,       *
               INITIAL='CICS Bank Sample Application - Accounts for Cus*
               tomers.'
*****************************************************************
         DFHMDF POS=(3,1),LENGTH=44,ATTRB=(NORM,PROT),COLOR=TURQUOISE, *
               INITIAL='Provide a Customer number. Then press Enter.'
         DFHMDF POS=(5,1),LENGTH=15,ATTRB=(NORM,PROT),COLOR=TURQUOISE, *
```

```
                INITIAL='CUSTOMER NUMBER'
CUSTNO    DFHMDF POS=(5,17),LENGTH=10,ATTRB=(NORM,NUM,IC),COLOR=GREEN,   *
                HILIGHT=UNDERLINE
          DFHMDF POS=(5,28),LENGTH=1,ATTRB=(NORM,PROT,ASKIP)
          DFHMDF POS=(8,1),LENGTH=41,ATTRB=(NORM,PROT),                  *
                COLOR=NEUTRAL,                                           *
                INITIAL='SORT CODE   ACCOUNT NUMBER   ACCOUNT TYPE'
          DFHMDF POS=(8,43),LENGTH=33,ATTRB=(NORM,PROT),                 *
                COLOR=NEUTRAL,                                           *
                INITIAL='   AVAIL BALANCE   ACTUAL BALANCE'
ACCOUNT   DFHMDF POS=(9,1),LENGTH=79,ATTRB=(NORM,PROT,FSET,ASKIP),       *
                COLOR=NEUTRAL,OCCURS=10
*-----------------------------------------------------------------------*
MESSAGE   DFHMDF POS=(23,1),LENGTH=79,COLOR=YELLOW,                      *
                ATTRB=(BRT,PROT,ASKIP)
          DFHMDF POS=(24,1),LENGTH=20,ATTRB=(NORM,PROT,ASKIP),           *
                COLOR=BLUE,                                              *
                INITIAL='F3=Exit   F12=Cancel'
DUMMY     DFHMDF POS=(24,79),LENGTH=1,ATTRB=(DRK,PROT,ASKIP,FSET),       *
                INITIAL=' '
*-----------------------------------------------------------------------*
          DFHMSD TYPE=FINAL
                 END
```

Changing the Look up accounts with a customer number BMS validation program (BNK1CCA):

3. Program BNK1CCA is responsible for validating the data coming from the Look up accounts with a customer number BMS map and data, from the backend program, being output to the BMS map. In this instance we have not changed the BMS map, but we will still need to amend the place where the ACCOUNT variable is referenced (i.e. the line of account data displayed on the BMS map), to reflect that the account number needs to be 9 bytes instead of 9.

4. Edit COBOL program **BNK1CCA** from the /cobol folder in the Git Hub repo. This pulls in the following copybooks:
    - BNK1ACC copybook, should be unchanged because we did not amend the BMS map - so no further action is necessary.
    - ABNDINFO copybook, has no account numbers in it, so no further action is necessary.
    - INQACCCU copybook, this copybook has been changed previously, so no further action is necessary.

    Variable COMM-ACCNO (in copybook INQACCCU) has previously been changed to be 9 bytes, however we need to look at all places in program BNK1CCA where COMM-ACCNO get assigned to other variables, and ensure that those other variables can also cope with a 9 byte long account number.

    COMM-ACCNO is moved to variable ACCNO-CHAR. Change this to PIC X(9) as highlighted:

```
01 ACCNO-CHAR                  PIC X(9).
```

To allow for an extra digit being added to the account number (in ACCNO-CHAR) we must also reduce the spacing between the account number and the next variable (which is the account type) on the account line. Amend this bit of code:

```
STRING
   SCODE-CHAR  DELIMITED BY SIZE
   '      '    DELIMITED BY SIZE
   ACCNO-CHAR  DELIMITED BY SIZE
   '         ' DELIMITED BY SIZE
   COMM-ACC-TYPE(WS-INDEX)
               DELIMITED BY SIZE
   '       '
               DELIMITED BY SIZE
   WS-AVAIL-BAL-SIGN
               DELIMITED BY SIZE
   WS-AVAIL-BAL-X-PND
               DELIMITED BY SIZE
   '.'
               DELIMITED BY SIZE
   WS-AVAIL-BAL-X-PNCE
               DELIMITED BY SIZE
   '  '
               DELIMITED BY SIZE
   WS-ACT-BAL-SIGN
               DELIMITED BY SIZE
   WS-ACT-BAL-X-PND
              DELIMITED BY SIZE
   '.'
              DELIMITED BY SIZE
   WS-ACT-BAL-X-PNCE
              DELIMITED BY SIZE
INTO ACCOUNTO(WS-INDEX)
```

Where the highlighted field (below) has been reduced in size by one byte and now contains 8 instead of 9 spaces (padding characters):

```
STRING
   SCODE-CHAR  DELIMITED BY SIZE
   '      '    DELIMITED BY SIZE
   ACCNO-CHAR  DELIMITED BY SIZE
   '        ' DELIMITED BY SIZE
   COMM-ACC-TYPE(WS-INDEX)
               DELIMITED BY SIZE
   '       '
               DELIMITED BY SIZE
   WS-AVAIL-BAL-SIGN
               DELIMITED BY SIZE
   WS-AVAIL-BAL-X-PND
               DELIMITED BY SIZE
   '.'
               DELIMITED BY SIZE
   WS-AVAIL-BAL-X-PNCE
               DELIMITED BY SIZE
   '  '
               DELIMITED BY SIZE
   WS-ACT-BAL-SIGN
               DELIMITED BY SIZE
   WS-ACT-BAL-X-PND
              DELIMITED BY SIZE
```

```
           '.'
                      DELIMITED BY SIZE
           WS-ACT-BAL-X-PNCE
                      DELIMITED BY SIZE
       INTO ACCOUNTO(WS-INDEX)
```

Save these changes (use cntrl S to save).

Changing the Look up accounts with a customer number backend program (INQACCCU):

5. Program INQACCCU is responsible for talking an incoming customer number and looking up the associated accounts for that customer on the ACCOUNT Db2 table.

   Program INQACCCU accesses the Db2 ACCOUNT data using an SQL DECLARE held in copybook **ACCDB2** and also the ACCOUNT data held in the **ACCOUNT** copybook. Both of these copybooks have already been changed, so no further action is required.

   However, the matching host variable for ACCDB2 needs to be changed.

   Edit COBOL program **INQACCCU** from the /cobol folder in the Git Hub repo and change HV-ACCOUNT-ACC-NO from PIC X(8) to PIC X(9), as highlighted below:

```
* ACCOUNT Host variables for DB2
 01 HOST-ACCOUNT-ROW.
    03 HV-ACCOUNT-EYECATCHER        PIC X(4).
    03 HV-ACCOUNT-CUST-NO           PIC X(10).
    03 HV-ACCOUNT-SORTCODE          PIC X(6).
    03 HV-ACCOUNT-ACC-NO            PIC X(9).
    03 HV-ACCOUNT-ACC-TYPE          PIC X(8).
    03 HV-ACCOUNT-INT-RATE          PIC S9(4)V99 COMP-3.
    03 HV-ACCOUNT-OPENED            PIC X(10).
    03 HV-ACCOUNT-OVERDRAFT-LIM     PIC S9(9) COMP.
    03 HV-ACCOUNT-LAST-STMT         PIC X(10).
    03 HV-ACCOUNT-NEXT-STMT         PIC X(10).
    03 HV-ACCOUNT-AVAIL-BAL         PIC S9(10)V99 COMP-3.
    03 HV-ACCOUNT-ACTUAL-BAL        PIC S9(10)V99 COMP-3.
```

   Program INQACCCU has a copybook defined for the COMMAREA called INQACCCU, this too has been changed previously, to reflect a 9 byte long account number.

   Having just modified variable;
   HV-ACCOUNT-ACC-NO

   We need to look at all places in program INQACCCU where this variable has a value assigned and where this variable is used to assign a value to other variables (now that this variable has been modified to be 9 bytes long). There are no assignments to/from this variable that are still 8 bytes, so no further action is required.

   Save these changes (use cntrl S to save).

The Delete Customer functionality:

Option on the BMS main menu	***Purpose***	***Mapset and Map affected***	***BMS display program affected***	***Back end program affected***
Option 1 then pF5	Delete Customer	-	-	DELCUS DELACC

It may seem odd that we need to look at the Delete Customer functionality, provided by option 1 from the BMS main menu, when we are changing the account number. However, when you elect to delete a customer, under the covers, it also deletes all of the accounts associated with that customer too.

Account numbers are not utilised in the BMS map for delete Customer, we need only look at the backend program DELCUS.

Changing the Delete Customer backend program (DELCUS):

1. Program DELCUS is responsible for deleting the customer but it also deletes all of the accounts belonging to that customer too. It utilises program INQCUST to validate the customer (which doesn't need to change) and it utilises program INQACCCU (which has been changed previously and now copes with 9 byte account numbers). So the only thing that we need to do is to deal with any references of account which are still 8 bytes – this will be the commarea utilised in DELCUS, used to pass information to/from program INQACCCU and any 8 byte account numbers held in arrays in Working Storage etc.

   Program DELCUS accesses the Db2 PROCTRAN table using an SQL DECLARE held in copybook **PROCDB2** and also the PROCTRAN data held in in the **PROCTRAN** copybook. These have been changed previously so no further action is required.

   However, the matching host variables for PROCDB2 need to be changed.

   Edit COBOL program **DELCUS** from the /cobol folder in the Git Hub repo and change `HV-PROCTRAN-ACC-NUMBER` from PIC X(8) to PIC X(9), as highlighted below:

```
* PROCTRAN host variables for DB2
 01 HOST-PROCTRAN-ROW.
     03 HV-PROCTRAN-EYECATCHER   PIC X(4).
     03 HV-PROCTRAN-SORT-CODE    PIC X(6).
     03 HV-PROCTRAN-ACC-NUMBER   PIC X(9).
     03 HV-PROCTRAN-DATE         PIC X(10).
     03 HV-PROCTRAN-TIME         PIC X(6).
     03 HV-PROCTRAN-REF          PIC X(12).
     03 HV-PROCTRAN-TYPE         PIC X(3).
     03 HV-PROCTRAN-DESC         PIC X(40).
     03 HV-PROCTRAN-AMOUNT       PIC S9(10)V99 COMP-3.
```

Program DELCUS also utilises numerous copybooks including ACCOUNT, PROCTRAN, INQACCCU and ABNDINFO – these have all be changed previously or do not require amendment, so no further action is required.

Having just modified variables;
HV-PROCTRAN-ACC-NUMBER
COMM-ACCNO (via the INQACCCU copybook which was changed as part of changing program CREACC)

We need to look at all the places in program DELCUS where these variables have a value assigned and where these variables are used to assign a value to other variables (now that these variables have been modified to be 9 bytes long).

Variable COMM-ACCNO gets moved to DELACC-COMM-ACCNO, so we must change it as highlighted below:

```
01 DELACC-COMMAREA.
   03 DELACC-COMM-EYE           PIC X(4).
   03 DELACC-COMM-CUSTNO        PIC X(10).
   03 DELACC-COMM-SCODE         PIC X(6).
   03 DELACC-COMM-ACCNO         PIC 9(9).
   03 DELACC-COMM-ACC-TYPE      PIC X(8).
   03 DELACC-COMM-INT-RATE      PIC 9(4)V99.
   03 DELACC-COMM-OPENED        PIC 9(8).
   03 DELACC-COMM-OVERDRAFT     PIC 9(8).
   03 DELACC-COMM-LAST-STMT-DT  PIC 9(8).
   03 DELACC-COMM-NEXT-STMT-DT  PIC 9(8).
   03 DELACC-COMM-AVAIL-BAL     PIC S9(10)V99.
   03 DELACC-COMM-ACTUAL-BAL    PIC S9(10)V99.
   03 DELACC-COMM-SUCCESS       PIC X.
   03 DELACC-COMM-FAIL-CD       PIC X.
   03 DELACC-COMM-DEL-SUCCESS   PIC X.
   03 DELACC-COMM-DEL-FAIL-CD   PIC X.
   03 DELACC-COMM-APPLID        PIC X(8).
   03 DELACC-COMM-PCB1          POINTER.
   03 DELACC-COMM-PCB2          POINTER.
```

Save these changes (use cntrl S to save).

2. Program DELCUS links to program DELACC passing it a commarea (changed in the above step) to delete the accounts associated with the customer. We have previously amended program DELACC, so no further changes are required. And that completes the changes to program DELCUS.

The other program changes…..

We have made the changes that directly affect the account number moving to 9 bytes to all of the associated account related programs. However, because the account number is contained within the PROCTRAN Db2 table it is necessary for us to include that into our development, but it also means that anywhere where PROCTRAN is utilised in CBSA also needs to change. The section contains the any 'other' programs which need to be amended.

Changing the Create Customer backend program (CRECUST)

1. Program CRECUST is responsible for taking validated in-coming customer details to create a new customer on the CUSTOMER datastore. If this is successful, a successfully processed transaction is written to the PROCTRAN Db2 table. As the account number is a variable on the PROCTRAN table, and we have now changed this from 8 to 9 bytes, the data passed into it from CRECUST must now also be changed to be 9 bytes in length.

   Program CRECUST accesses the Db2 PROCTRAN data using an SQL DECLARE held in copybook **PROCDB2**. This copybook has already been changed, so no further action is required.

   However, the matching host variable for PROCDB2 needs to be changed.

   Edit COBOL program **CRECUST** from the /cobol folder in the Git Hub repo and change HV-PROCTRAN-ACC-NUMBER from PIC X(8) to PIC X(9), as highlighted below:

```
* PROCTRAN host variables for DB2
 01 HOST-PROCTRAN-ROW.
    03 HV-PROCTRAN-EYECATCHER     PIC X(4).
    03 HV-PROCTRAN-SORT-CODE      PIC X(6).
    03 HV-PROCTRAN-ACC-NUMBER     PIC X(9).
    03 HV-PROCTRAN-DATE           PIC X(10).
    03 HV-PROCTRAN-TIME           PIC X(6).
    03 HV-PROCTRAN-REF            PIC X(12).
    03 HV-PROCTRAN-TYPE           PIC X(3).
    03 HV-PROCTRAN-DESC           PIC X(40).
    03 HV-PROCTRAN-AMOUNT         PIC S9(10)V99 COMP-3.
```

   Having just modified variable;
   HV-PROCTRAN-ACC-NUMBER

   We need to look at all places in program CRECUST where this variable has a value assigned and where this variable is used to assign a value to other variables (now that this variable has been modified to be 9 bytes long). In this case the only thing moved into HV-PROCTRAN-ACC-NUMBER is zeros, so no further action is required.

   Save these changes (use cntrl S to save).

To complete the change exercise……

If you have completed all of the program changes detailed in this appendix, to complete the exercise you now need to complete sections:

- “Backing up the data on the ACCOUNT and PROCTRAN Db2 tables, making the Db2 table changes and reloading the affected tables” (on page 17).
- “Rebuilding it all” (on page 19).
- “Rebind” (on page 34).
- “Restart CICS” (on page 34).
- “Try out the changes” (on page 34).