

DL/1 & COBOL

To compile, link and run a COBOL program in batch accessing an IMS database, the following JCL is needed, the load module being represented by @@@@ :

```
// EXEC COB2LEX,MEMB=@@@@@,LLIB='DEV@@TR.TEST.LOADLIB'  
//STEPA.SYSIN DD *  
    *TITLE DL1PROG - COBOL II  
  
/*  
//STEPB.SYSIN DD *  
    ENTRY DLITCBL  
/*  
// EXEC PGM=DFSRRC00,PARM='DLI,@@@@@,ECAPPSBR'  
//STEPLIB DD DSN=&&EXECLIB,DISP=OLD  
//      DD DSN=DEV@@TR.TEST.LOADLIB,DISP=SHR  
//      DD DSN=SA.IMS.LOADLIB,DISP=SHR  
//      DD DSN=SYS1.RUNTIME,DISP=SHR  
//IMS   DD DSN=UB.IMS.TESTPSB,DISP=SHR  
//      DD DSN=UB.IMS.TESTDBD,DISP=SHR  
//IMSACB DD DSN=SYS1.TESTACB,DISP=SHR  
//DFSRESLB DD DSN=SA.IMS.LOADLIB,DISP=SHR  
//SYSUDUMP DD DUMMY  
/* DL1 BUFFERING PARAMETERS FOLLOW  
//DFSVSAMP DD *  
    2048,6  
    DLITRACE DDNAME=DFSTROUT  
/*  
//DFSTROUT DD SYSOUT=*  
//DDP   DD DSN=DEV@@TR.DEV@@TR.TRDBK2,DISP=SHR  
//DDS   DD DSN=DEV@@TR.DEV@@TR.TRDBE2,DISP=SHR  
//IEFRDER DD DUMMY  
//SYSOUT DD SYSOUT=*  
//SYSABOUT DD SYSOUT=*
```

The first statement in the Procedure Division must be :-

```
ENTRY 'DLITCBL' USING PCB-MASK.
```

As this line defines the entry point to the program, it must not occur in a called paragraph.

To compile, link and run a COBOL program in BMP batch accessing an IMS database, the following JCL is needed, the load module being represented by @@@@ @@@@ :

```
// EXEC COB2LEX,MEMB=@@@@@,LLIB='DEV@@TR.TEST.LOADLIB'  
//STEPA.SYSIN DD *  
    *TITLE DL1PROG - COBOL II  
  
/*  
//STEPB.SYSIN DD *  
ENTRY DLITCBL  
/*  
//BMP EXEC PGM=DFSRR00,  
// REGION=4096K,  
// PARM=(BMP,DL1EX1,ECAPPSBR,,,N00010,,1,,1,2,,,DL1D)  
/*  
//STEPLIB DD DSN=&&EXECLIB,DISP=OLD  
// DD DSN=DEV@@TR.TEST.LOADLIB,DISP=SHR  
// DD DSN=SA.IMS.V5.LOADLIB,DISP=SHR  
// DD DSN=SYS1.RUNTIME,DISP=SHR  
//SYSUDUMP DD DUMMY  
//SYSOUT DD SYSOUT=*
```

The first statement in the Procedure Division must be :-

```
ENTRY 'DLITCBL' USING IO-PCB, PCB-MASK.
```

As this line defines the entry point to the program, it must not occur in a called paragraph.

PCB-MASK is defined in the Linkage Section of the program.

e.g. 01 COURSE-PCB.

03	DBD-NAME	PIC X(8).
03	SEGMENT-LEVEL-NO	PIC XX.
03	STATUS-CODE	PIC XX.
03	PROCESSING-OPTIONS	PIC X(4).
03	RESERVED-FOR-DL1	PIC S9(5) COMP.
03	SEGMENT-NAME	PIC X(8).
03	LENGTH-OF-KEY-FEEDBACK-AREA	PIC S9(5) COMP.
03	NO-SENSITIVE-SEGMENTS	PIC S9(5) COMP.
03	KEY-FEEDBACK-AREA	PIC X(17).

For BMP batch programs, another PCB mask area needs to be defined in the Linkage Section (IO-PCB).

e.g. 01 IO-PCB.

03	FILLER	PIC X(10).
03	IO-STAT-CODE	PIC XX.
03	FILLER	PIC X(20).
01	COURSE-PCB.	
03	DBD-NAME	PIC X(8).
03	SEGMENT-LEVEL-NO	PIC XX.
03	STATUS-CODE	PIC XX.
03	PROCESSING-OPTIONS	PIC X(4).
03	RESERVED-FOR-DL1	PIC S9(5) COMP.
03	SEGMENT-NAME	PIC X(8).
03	LENGTH-OF-KEY-FEEDBACK-AREA	PIC S9(5) COMP.
03	NO-SENSITIVE-SEGMENTS	PIC S9(5) COMP.
03	KEY-FEEDBACK-AREA	PIC X(17).

I/O area for each segment to be accessed is defined in Working Storage. It is usual for all SSAs and Function Codes to also be defined in Working Storage, rather than being passed as literals to DL/1.

```
01 QUAL-COURSE-SSA.
  03 Q-COURSE-SEGMENT-NAME PIC X(8) VALUE 'COURSE '.
  03 COURSE-LEFT-PAREN     PIC X   VALUE '('.
  03 COURSE-FIELD-NAME     PIC X(8) VALUE 'CNUMBER '.
  03 COURSE-REL-OP         PIC XX  VALUE 'EQ'.
  03 COURSE-FIELD-VALUE    PIC 99.
  03 COURSE-RIGHT-PAREN    PIC X   VALUE ')'.

01 FUNCTION-CODES.
  03 FUNCTION-CODE         PIC X(4).
  03 GET-NEXT              PIC X(4) VALUE 'GN '.
  03 GET-NEXT-WITHIN-PARENT PIC X(4) VALUE 'GNP '.
  03 GET-HOLD-NEXT-PARENT  PIC X(4) VALUE 'GHNP'.
  03 GET-HOLD-UNIQUE       PIC X(4) VALUE 'GHU '.
  03 DLI-REPLACE           PIC X(4) VALUE 'REPL'.
  03 DLI-INSERT            PIC X(4) VALUE 'ISRT'.
  03 DLI-DELETE            PIC X(4) VALUE 'DLET'.
```

To retrieve COURSE number 43 for update, the code will look something like this :

```
MOVE GET-HOLD-UNIQUE TO FUNCTION-CODE.
MOVE 43 TO COURSE-FIELD-VALUE.
CALL 'CBLTDLI' USING FUNCTION-CODE, PCB-MASK, COURSE-IO-AREA,
                                QUAL-COURSE-SSA.
```

Finally there must be a 'GOBACK' statement at the end of the program, to send control back to IMS.

DL/1 PROGRAMMING EXERCISES

Note : Before running these exercises, run a job to restore the database to its "pre-DL1-course" state. This job can be found in DA1.DL1RESTR.

DA1.DL1SKEL contains a skeleton COBOL program with the required JCL which can be used as a basis for the exercises.

1. Using the Curriculum Database, write a program to list all OFFERINGS of all COURSEs in a report format. The report should show the Course Number, Name, Location, Start Date and Duration.
2. Write a program to list all STUDENTs on all COURSEs (one COURSE per page), showing the Course Number and Name, the Student Name and Status, and the Start Date of the COURSE.
3. STUDENTs with Status fields unset (ie. spaces), are to be reclassified according to the following criteria. Those who are enrolled on COURSEs running in 1995 or later are to be classified as status "F" for future, those on COURSEs which ran in 1994 should have status "A" for active, whereas those on courses which ran before 1994 should be marked as "H" for historical. Those STUDENTs who are marked as cancelled ("C") should be left unchanged. Write a program to update the database accordingly.
4. COURSE 45 is no longer being run. Write a program to delete this COURSE from the database and replace it with course number 35, entitled "Spanish 1". There is to be only one OFFERING of this COURSE as yet, on the 1st November 1994, in Barcelona. The duration of this new COURSE should be 8 days.
5. It has been decided that COURSE number 35 should in fact be a Portugese COURSE instead. Write a program to alter the title of the COURSE and the location to Lisbon. The Spanish COURSE described above should now be COURSE number 46.

