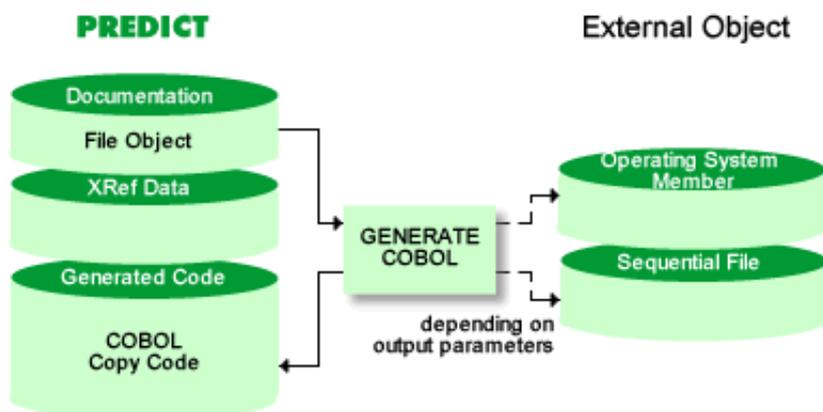


COBOL Copy Code

The function Generate COBOL Copy Code generates a record buffer for use in COBOL programs based on a Predict file object.

In addition, an Adabas format buffer can also be generated if required for files of the following types:

- Adabas file (file type A) with parameter Adabas SQL usage = N
- Adabas userview (file type U)



This section contains:

- Calling the Function
 - Generate COBOL Copy Code in Batch Mode
 - Names in COBOL Copy Code
 - Field Format and COBOL Copy Code
 - COBOL Copy Code for DB2 Tables/Views
 - Sample Output
-

Calling the Function

The Generate COBOL Copy Code screen is displayed with function code G and object code CO in a Predict main menu, or with the command GENERATE COBOL.

```

10:03:05          ***** P R E D I C T 4.3.1 *****          2003-05-31
Plan    0          - Generate COBOL Copy Code -

File ID .....*
Save as member .....          Save in library .... COBLIB
Overwrite option ..... Y (Y,N) Op. system member ..
Punch / output .....* N          List offsets .....* N
List generated code ..... Y (Y,N) Adabas version ....* I7
Generate format buffer .....* N          Field name prefix ..
Check field name .....* A          Field name suffix ..
Start level ..... 1 (0-40) Validate ..... -
Level number increment ..... 1 (1-40) Truncation .....* R
Level shift increment ..... 3 (0-9) With Cond. names ... N (Y,N)
Nr. of abstract lines ..... 3 (0-16) Indexed by .....* N
Generate initial value .....* N          Literal delimiter .* S
Synchronized .....* Y          Decimal character .* P
Depending on ..... N (Y,N) Redefinition name .* S
Record buffer name .....
Format buffer name .....

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help Next Stop Last LnKEl Flip Print Impl AdmFi SelFi Prof Main
    
```

Values for input fields which have been locked by your data dictionary administrator cannot be overwritten. These fields are skipped when positioning the cursor with the TAB key. See Generation Defaults.

Presettings	
The parameters below can be changed in the Modify COBOL Defaults screen. See Generation Defaults.	
Library system	Library system for which the generated code is punched. Determines which additional cards need to be punched. An operating system member must be entered for the additional cards to be generated. See Parameters Specifying the Form of Output for more information.
Compiler	Copy code can be generated for the following compilers: 7 Standard COBOL 74 compiler. 8 COBOL 85 compiler. W COBOL compiler for WANG.
Parameters	
The parameters Save as member, Save in library, Overwrite option, Op. system member, Punch/output, Workfile name and List generated code are described in the section Parameters Specifying the Form of Output.	
File ID	ID of the Predict file object from which the definitions are to be generated. Enter an asterisk to display a selection screen. Alternatively, use an asterisk as a wildcard. For example, enter A* to display IDs beginning with A only.

List offsets	<p>Y The offset of each item in the record buffer structure (relative to the beginning of the structure) in decimal and hexadecimal formats is to be included as a comment. The total length of each buffer is also included.</p> <p>P As above, but the absolute position (offset+1) is included as a comment.</p> <p>L The total lengths of the record buffer and the format buffer are to be included as a comment.</p> <p>V The file number and the calculated lengths of the record buffer and the format buffer are to be generated as constants in the copy code. The name of the file number constant is the record buffer name prefixed by N-. The name of each length constant is the appropriate buffer name prefixed by L-. Each name is prefixed, suffixed, validated and truncated in the same way as any other field name.</p> <p>N No offset.</p>
Adabas version	The version of Adabas for which the copy code is to be generated. Enter an asterisk for valid values or see table in the section Adabas Version for more information.
Generate format buffer	<p>The contents of the format buffer will correspond exactly to the contents of the record buffer. Only valid for files of type A (with parameter Adabas SQL usage set to N) or for files of type U.</p> <p>Valid values:</p> <p>Y Adabas format buffer is to be generated. Adabas groups, standard formats and lengths are used whenever possible. The resulting format buffers are then as short as possible.</p> <p>F Full format buffer is to be generated. Length and format of Adabas fields are included.</p> <p>N No format buffer is to be generated.</p> <p>Note: If you are generating for a WANG environment, set this parameter to F or N.</p>
Field name prefix	The prefix to be used for each field name generated.
Check field name	<p>A COBOL field names are checked for uniqueness within the whole structure.</p> <p>Y Structure levels are included in the validation check of the field names: if two fields have the same name, they must be separated by at least one field with a different name and a smaller level number.</p> <p>N No check for duplicate field names is performed.</p>
Field name suffix	The suffix to be used for each field name generated.
Start level	The starting level number to be used for the generated statements.

<p>Validate</p>	<p>Determines how invalid characters are handled.</p> <p>blank Invalid characters in a field name will result in an error message but will not be modified.</p> <p>rep.char Invalid characters in a field name are replaced by this character. Valid replace characters: letters A-Z, digits 0-9 and hyphen (-).</p> <p>*</p> <p>Invalid characters in a field name are deleted.</p>												
<p>Level number increment</p>	<p>This parameter is evaluated with the parameter Start level. The level numbers of the generated statements are incremented as follows</p> <table border="1" data-bbox="391 734 1142 936"> <thead> <tr> <th>Start Level</th> <th>Level number increment (example)</th> <th>Level numbers</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>5</td> <td>1, 6, 11...</td> </tr> <tr> <td>1</td> <td>5</td> <td>1, 5, 10 ...</td> </tr> <tr> <td>2 - 40</td> <td>5</td> <td>n, n+5, n+10...</td> </tr> </tbody> </table>	Start Level	Level number increment (example)	Level numbers	0	5	1, 6, 11...	1	5	1, 5, 10 ...	2 - 40	5	n, n+5, n+10...
Start Level	Level number increment (example)	Level numbers											
0	5	1, 6, 11...											
1	5	1, 5, 10 ...											
2 - 40	5	n, n+5, n+10...											
<p>Truncation</p>	<p>Specifies which characters are deleted if a generated field name is too long:</p> <p>L truncate from the left</p> <p>R truncate from the right</p> <p>M truncate from the middle</p> <p>A warning is given if field names are truncated.</p>												
<p>Level shift increment</p>	<p>The number of positions to be shifted right when a level higher than the current level number is found.</p>												
<p>With Cond. names</p>	<p>Y Any condition names defined in the Predict field objects are to be generated on level 88, provided that the respective field objects have one of the following formats:</p> <ul style="list-style-type: none"> ● A All lengths ● N or P Less than 19 digits ● I or B 2, 4 or 8 digits ● L A FALSE-condition will always be generated. The Condition name is then generated by concatenating the field name to the prefix N- (if not specified explicitly). <p>These names are prefixed, suffixed, validated and truncated in the same way as field names.</p>												
<p>Nr. of abstract lines</p>	<p>The number of Predict abstract lines per field to be included in the generated code.</p>												

<p>Indexed by</p>	<p>Y COBOL clause INDEXED BY is generated for all repetitive fields (MC, MU, PC and PE). For a field which has no INDEXED BY name in its Predict object, I- followed by the field name is used as index name. These names are prefixed, suffixed, validated and truncated in the same way as field names.</p> <p>N COBOL clause INDEXED BY will not be generated for any field.</p> <p>S COBOL clause INDEXED BY is generated only for repetitive fields which have an INDEXED BY name in their Predict objects.</p>																		
<p>Generate initial value</p>	<p>Y Statements are generated to initialize the structure with the value for Init value defined for the field objects in Predict. Fields with no value defined for Init value are initialized as follows</p> <table border="1" data-bbox="391 801 903 1102"> <thead> <tr> <th>Format</th> <th>Length</th> <th>Initialized with</th> </tr> </thead> <tbody> <tr> <td>A</td> <td>all lengths</td> <td>spaces</td> </tr> <tr> <td>N, NS, P, PS</td> <td>< 19 digits</td> <td>zeros</td> </tr> <tr> <td>I, B</td> <td>2,4,or 8 digits</td> <td>zeros</td> </tr> <tr> <td>D, T</td> <td>-</td> <td>zeros</td> </tr> <tr> <td>Logical</td> <td>-</td> <td>low value</td> </tr> </tbody> </table> <p>No members of redefine groups are initialized.</p> <p>N No initialization.</p> <p>S Only fields which have a value defined for Init value in the corresponding Predict object are initialized.</p>	Format	Length	Initialized with	A	all lengths	spaces	N, NS, P, PS	< 19 digits	zeros	I, B	2,4,or 8 digits	zeros	D, T	-	zeros	Logical	-	low value
Format	Length	Initialized with																	
A	all lengths	spaces																	
N, NS, P, PS	< 19 digits	zeros																	
I, B	2,4,or 8 digits	zeros																	
D, T	-	zeros																	
Logical	-	low value																	
<p>Literal delimiter</p>	<p>Literal delimiter character:</p> <p>D double quotation marks (")</p> <p>S single quotation marks (').</p>																		
<p>Synchronized</p>	<p>Determines which fields are to be aligned. Fields are aligned in the record buffer and in the format buffer on the boundary shown in the table in the section Field Format and COBOL Copy Code.</p> <p>Y All fields with COBOL attribute COMP, COMP-1 or COMP-2 are aligned.</p> <p>N No fields are aligned.</p> <p>S Fields are aligned only if the 3GL specification Synchronized of the corresponding Predict object is set to S.</p>																		

Decimal character	<p>P decimal point</p> <p>C decimal comma</p>
Depending on	<p>Y COBOL attribute OCCURS DEPENDING ON is generated for a field or field group in a file if it has type PE or MU and a DEPENDING ON name is specified for this field.</p> <p>These names are prefixed, suffixed, validated and truncated in the same way as field names.</p> <p>Note: This option is not allowed for files of type A or U. This option is ignored when using a WANG COBOL compiler.</p>
Redefinition name	<p>Determines how COBOL field names for Predict fields of type RE are generated:</p> <p>F The string FILLER is used as redefinition name.</p> <p>S The suffix REGR is added to the Predict field name. If a field name is redefined more than once, the suffix will have the form REGRn, where n is an integer incremented by 1 for each field of type RE.</p>
Record buffer name	Specifies the name of the record buffer in the generated structure. If omitted, the file ID is used.
Format buffer name	Specifies the name of the format buffer in the generated structure. If omitted, the file ID prefixed by FORMAT-BUFFER- is used.

Generate COBOL Copy Code in Batch Mode

Command: GENERATE COBOL

Enter parameters on next line in positional or keyword form. File ID is obligatory, all other parameters are optional. If a parameter is not specified, the default value is taken.

Field	Keyword	Position
File ID	FILE-ID	1
Save as member	MEM	2
Save in library	LIB	3
Overwrite option	REPLACE	4
Op. system member	OS-MEMBER	5
Start level	START-LEVEL	6
Field name prefix	PREFIX	7
Level number increment	LEVEL-INCREMENT	8
Field name suffix	SUFFIX	9
Level shift increment	LEVEL-SHIFT	10
Nr. of abstract lines	NR	11

Validate	VALIDATION	12
Generate initial value	INIT	13
Truncation	TRUNCATION	14
Generate format buffer	FORMAT-BUFFER	15
List offsets	OFFSET	16
List generated code	LIST	17
Punch / output	PUNCH	18
Record buffer name	RECORD-BUFFER-NAME	19
Format buffer name	FORMAT-BUFFER-NAME	20
With Cond. names	COND-NAME	21
Synchronized	SYNC	22
Indexed by	INDEXED	23
Depending on	DEPENDING	24
Adabas version	ADA-VER	25
Check field name	CHECK-NAME	26
Literal delimiter	DELIMITER	27
Decimal character	DECIMAL-CHAR	28
Redefinition name	REDEFINE-NAME	29
Workfile name (see note below)	WORKFILE-NAME	30
If Entire System Server is used		
- DB-ID	NP-DBID	31
- Dataset	NP-DSNAME	32
- Volume	NP-VOLSER	33
- Library	NP-LIB	34
- Sublibrary	NP-SUBLIB	35
- Member type	NP-MEMTYPE	36
- VSAM catalog	NP-VSAMCAT	37

Note:

You can not specify a value other than blank (' ') for WORKFILE-NAME on mainframes.

Parameters NP-LIB, NP-SUBLIB and NP-MEMTYPE must be specified if the generated code is written to workfile 1 (Punch/output=Y) and Library system=3.

Names in COBOL Copy Code

The following rules apply to the generation of names for COBOL copy code.

- Field names are derived from the name of the corresponding Predict field object if no COBOL field name synonym has been specified.
- Alphabetic characters of field and file names are converted to upper-case. The parameters Field name prefix,

Field name suffix, Validate and Truncate apply to the generation of field names (see description of parameters above).

- In cases when supplementary fields have to be generated the names of these extra fields are derived from the Predict field names by generating prefixes and/or suffixes.
- If a resulting field name, indexed by name or depending on name is not unique, an error message may be displayed if Check field name is either set to A or Y.

The rules applying when deriving the names of extra fields when generating COBOL copy code are described in the sections below.

COBOL Names for Counter Fields - MC or PC

When generating COBOL fields for Predict fields of type MC or PC, a counter field will automatically be generated in the copy code. The name of this field is derived from the name of the MC or PC field by adding the prefix C-.

COBOL Names for Multiple Fields - MU - Contained in Periodic Groups - PE

When generating COBOL fields for a Predict field of type PE that contains a field of type MU and the parameter Gr.structur is set to N, a dummy field will automatically be generated preceding the MU field. The name of this field is derived from the name of the MU field by adding the prefix G-.

The MU field can then be used as a two-dimensional table and its real occurrence need not be computed.

COBOL Names for Redefined Fields - RE in Periodic Groups - PE

When generating COBOL fields for a Predict field of type PE that contains a redefinition and the parameter Gr.structur is set to N, a dummy field will automatically be generated preceding the RE field. The name of this field is derived from the name of the RE field by adding the prefix R-.

COBOL Names for Logical Fields - L

When generating COBOL fields for a Predict logical field (field format L) with the parameter With cond. names set to Y and no condition name is defined for the field, a condition name is generated by adding the prefix N- to the field ID.

COBOL Names for Record Buffer and Format Buffer

The record buffer name and the format buffer name are generated without prefix and suffix. The value(s) supplied must not contain invalid characters. If the file ID is taken as default record buffer name or, with the prefix FORMAT-BUFFER-, as default format buffer name, it is validated and, if necessary, truncated.

COBOL Names for Default Indexed by Name

When COBOL copy code is generated with the option Indexed by set to Y, and no INDEXED BY NAME has been specified, this name is generated by adding the prefix I- to the Predict name.

COBOL Names for Additionally Generated Indicator Fields

An additional indicator field prefixed with S- is generated in the following cases:

- a field is defined with Suppression option set to U (null allowed)
- a field of a file of type A, U or B is defined with Suppression option set to R (not null)

Note:

For Adabas fields, the additional indicator field is only generated for the following Adabas versions:

- I3 or above
- U1 or above
- V4 or above.

Examples for the Generation of COBOL Field Names

The GENERATE COBOL function is applied on a field HOURS_DAY. This field has the following attributes:

- A field name synonym for COBOL is not defined.
- The field type is MC.
- The field occurs in a periodic group.
- The parameter Gr.structur is set to N.

Three fields are generated in the COBOL copy code

```
C-HOURS_DAY
G-HOURS_DAY
HOURS_DAY
```

If the parameter Field name prefix had been set to Adabas, the parameter Field name suffix to *PERSONNELOFFICE, Validate to -, Truncation to R and Indexed by to Y when generating COBOL copy code from this field, the following COBOL field names are generated:

```
ADABAS-C-HOURS-DAY-PERSONNELOF indexed by ADABAS-I-C-HOURS-DAY-PERSONNEL
ADABAS-G-HOURS-DAY-PERSONNELOF indexed by ADABAS-I-G-HOURS-DAY-PERSONNEL
ADABAS-HOURS-DAY-PERSONNELOFFI indexed by ADABAS-I-HOURS-DAY-PERSONNELOF
```

Field Format and COBOL Copy Code

Fields in the COBOL copy code have a COBOL clause determined by the following factors:

- length and format of the corresponding Predict field object
- file type of the file containing the field
- the compiler for which you are generating.

File type A(SQL) in the table below represents files of type A with parameter Adabas SQL usage set of Y.

Predict Format	Predict Length	File Type or Compiler	COBOL Clause	Note
B	<i>l</i> =1,3,5,6,7		PIC X(<i>l</i>)	
I	1	File Type: JT, JV	PIC S9(2) COMP	
		other file types	PIC X(1)	

B/I	2	File Type: BT, BV or Compiler: WANG/VS (and file type not JT, JV)	PIC S9(4) BINARY	aligned on half word boundary (if word length=4)
		File Type: JT, JV or Compiler: 74,85 (and file type not BT, BV)	PIC S9(4) COMP	
	4	File Type: BT, BV or Compiler: WANG/VS (and file type not JT, JV)	PIC S9(9) BINARY	aligned on word boundary (if word length=4)
		File Type: JT, JV or Compiler: 74,85 (and file type not BT, BV)	PIC S9(9) COMP	
	8	Compiler: WANG/VS	PIC X(8)	
		other compiler	PIC S9(18) COMP	aligned on word boundary (if word length=4)
B	$l \geq 9$		PIC X(<i>l</i>)	
F	$l=4,8$	File Type: not BT, BV, JT, JV Compiler: WANG/VS	PIC X(<i>l</i>)	
		File Type: JT, JV	PIC S9(10)V9(8) COMP-3	
	4	File Type: BT, BV or Compiler: not WANG/VS	COMP-1	aligned on word boundary (if word length=4)
	8	File Type: BT, BV or Compiler: not WANG/VS	COMP-2	aligned on double word boundary (if word length=4)
N/U	$nn.m$ ($nn+m < 19$)		PIC 9(<i>nn</i>)V9(<i>m</i>)	if $m=0$, V9(<i>m</i>) is not applicable
	$nn.m$ ($nn+m > 18$)		PIC X($nn+m$)	
NS/US	$nn.m$ ($nn+m > 15$)	File Type: B, A(SQL), AT	PIC X($nn+m$)	
	$nn.m$ ($nn+m < 19$)		PIC S9(<i>nn</i>)V9(<i>m</i>)	if $m=0$, V9(<i>m</i>) is not applicable
	$nn.m$ ($nn+m > 18$)		PIC X($nn+m$)	

P	nn.m (nn+m<19)	Compiler: 74	PIC 9(nn)V9(m)COMP-3	if m=0, V9(m) is not applicable
		Compiler: 85	PIC 9(nn)V9(m)PACKED DECIMAL	
		Compiler: WANG	PIC 9(nn)V9(m)COMP	
	nn.m (nn+m>18)		PIC X(l)	$l=(nn+m+2)/2$
PS	nn.m (nn+m>15)	File Type: B, A(SQL) AT	PIC X(l)	$l=(nn+m+2)/2$
		File Type: BT, BV or Compiler: 85	PIC S9(nn)V9(m) PACKED-DECIMAL	if m=0, V9(m) is not applicable
		File Type: not BT, BV or Compiler: 74	PIC S9(nn)V9(m)COMP-3	
	File Type: not BT, BV or Compiler: WANG	PIC S9(nn)V9(m)COMP		
	nn.m (nn+m>18)		PIC X(l)	$l=(nn+m+2)/2$
D		File Type: D, E, BT, BV	PIC X(10)	
		File Type: XT, XV	see format I, length 4	
		other file types	see Format P, length 6	
T		File Type: D, E, BT, BV	PIC X(8)	
		other file types	see Format P, length 12	
L		File Type: BT, BV	PIC S9(4) BINARY	
		other file types	PIC X(1)	
BT		File Type: YT, YV	PIC X(1)	
DT		File Type: JT, JV	PIC X(25)	
		File Type: OT, OV	PIC X(9)	
		File Type: XT, XV	PIC X(23)	
		File Type: YT, YV	PIC X(32)	
DS		File Type: YT, YV	PIC X(28)	
G, GV, GL	n	File Type: D, E	PIC X(m)	$m=2*n$
IV	7 / 17	File Type: XT, XV	PIC X (n)	
S		File Type: XT, XV	see format I, length 4	
TK		File Type: JT, JV	PIC X(8)	

TS		File Type: BT, BV, D, E	PIC X(26)	
		File Type: YT, YV	PIC X(8)	
OK		File Type: JT, JV	PIC X(16)	
MO		File Type: JT, JV, YT, YV	see format F, length 8	
	n.m	File Type: XT, XV	PIC S9(n)V9(m)	
MS		File Type: YT, YV	see format F, length 4	

A numeric or binary format field with a length not included in the table above is treated in COBOL copy code as an alphanumeric format field and cannot have an initial value generated for it. A warning message is issued.

File Type in Predict	Predict Format	Predict Char Set	Length	COBOL Clause	Note
YT, YV	A	S	n	PIC X(n)	
		B	n	PIC X(m)	m=2*n
		D	n	PIC X(m)	m=2*n
	AV	S	n	Two-level group is generated: 02 Fieldname	m=n
B, D			03 Fieldname-LEN corresponding to I2 03 Fieldname-TXT Char of length m	m=2*n	
JT, JV	A, AV	any	n	PIC X(n)	
	AL	B	n	PIC X(n)	Field is skipped if no length is specified
	BL		n	PIC X(n)	Field is skipped if no length is specified
	BV		n	PIC X(n)	
XT, XV, OT, OV,	A, AL	any	n	PIC X(n)	
	AV	there is only one Char Set	n	Two-level group is generated: 02 Fieldname 03 Fieldname-LEN corresponding to I2 03 Fieldname-TXT Char of length n	

D, E	A, AL	any	n	PIC X(n)	
	AV	any	n	Two-level group is generated: 02 Fieldname 03 Fieldname-LEN corresponding to I2 03 Fieldname-TXT Char of length n	
A, U	A	there is only one Char Set	n	PIC X(n)	
	AV		n	Two-level group is generated: 02 Fieldname 03 Fieldname-LEN corresponding to I2 03 Fieldname-TXT Char of length n	
A(SQL), AT, B	A, AV	there is only one Char Set	n	PIC X(n)	
BT, BV	A	any	n	PIC X(n)	
	AL	any	n	Two-level group is generated: 02 Fieldname 03 Fieldname-LEN corresponding to I2 03 Fieldname-TXT Char of length n	Field is skipped if no length is specified
	AV	any	n	Two-level group is generated: 02 Fieldname 03 Fieldname-LEN corresponding to I2 03 Fieldname-TXT Char of length n	

Automatically Generated Counter Fields

An automatically generated counter field has the following clause, depending on the COBOL compiler used:

Compiler	Clause
COBOL 74, COBOL 85	PIC S9(4) COMP
WANG	PIC S9(4) BINARY

COBOL Copy Code for DB2 Tables/Views

If COBOL copy code for DB2 Tables/Views is generated, the format NS or US is converted to the following COBOL format depending on the COBOL compiler used:

Compiler	Clause
COBOL 74,	PIC S9(nn)V9(m) COMP-3
COBOL 85	PIC S9(nn)V9(m) COMP
WANG	PIC S9(nn)V9(m) PACKED-DECIMAL

Any redefinitions that exist for this field are skipped.

A DB2 large object is generated as locator like an I4 field.

Sample Output

```

13:27:53          ***** P R E D I C T 4.3.1 *****          2003-05-31
                   - Generate COBOL Copy Code -                   Page: 1

File ID .. GENERATION-EXAMPLE

*****
*      THIS RECORD-BUFFER LAYOUT WAS GENERATED BY PREDICT
*      FOR FILE: GENERATION-EXAMPLE
*      ON: 2003-05-31 STARTING AT 13:27:53
*      FILE-COMMENTS: Example file for the
*                      PREDICT generation subsystem.
*
*****

01  GENERATION-EXAMPLE.
   02  GROUP-1.
      03  ELE-N-9V5          PIC 9(9)V9(5).
      03  GR-IN-GROUP.
         04  ELE-B-4          PIC S9(9) COMP SYNC.
            88  INIT-ELE-B-4  VALUE IS 4096.
            88  COND-ELE-B-4-V-1
               VALUE IS -1.
         04  ELE-PS-5V2      PIC S9(5)V9(2) COMP-3.
         04  R-MU-B-4.
            05  MU-B-4          OCCURS 5 TIMES PIC S9(9) COMP
                               SYNC.
         04  R-MU-B-4-REGR   REDEFINES R-MU-B-4.
            05  MU-RED-B-7    PIC X(7).
* >>> DIC1818 WARNING: FORMAT(S) CHANGED ( ' B ' --> ' A ' )
         05  MU-RED-US-13    PIC S9(13).
         04  R-MU-B-4-REGR1  REDEFINES R-MU-B-4.
            05  MU-RED-B-8    PIC S9(18) COMP.
            88  MU-RED-B-8-COND1
               VALUE IS 1118481.
         03  ELE-A-42        PIC X(42).
            88  COND-NAME-START VALUE IS 'COND-START' THRU
               'COND-ZEND'.
            88  FILL-WITH-STRING VALUE IS ALL 'A'.
         03  ELE-F-8         COMP-2 SYNC.
         03  ELE-B-3        PIC X(3).
    
```

```

13:27:53          ***** P R E D I C T 4.3.1 *****          2003-05-31
                  - Generate COBOL Copy Code -                  Page: 2

File ID .. GENERATION-EXAMPLE

* >>> DIC1818 WARNING: FORMAT(S) CHANGED ( ' B ' --> ' A ' )

      02 C-PC-OCC-7          PIC S9(4) COMP SYNC.
      02 PC-OCC-7          OCCURS 7 TIMES.
      03 PC-ELE-DE-NS-7V3  PIC S9(7)V9(3).
      03 C-PC-MC-PS-6V1    PIC S9(4) COMP SYNC.
      03 PC-MC-PS-6V1      OCCURS 11 TIMES PIC S9(6)V9(1)
                          COMP-3.

      03 PC-GR.
      04 PC-ELE-I-2        PIC S9(4) COMP SYNC.
      04 PC-PS-20V7        PIC X(14).
* >>> DIC1818 WARNING: FORMAT(S) CHANGED ( ' PS ' --> ' A ' )
      03 PC-ELE-F-4        COMP-1 SYNC.
      02 ELE-D            PIC 9(6) COMP-3.
      02 ELE-T            PIC 9(12) COMP-3.
      02 ELE-L            PIC X(1).
      88 N-ELE-L          VALUE IS LOW-VALUE.
*****
*          THIS FORMAT-BUFFER WAS GENERATED BY PREDICT
*          FOR FILE: GENERATION-EXAMPLE
*          ON: 2003-05-31 STARTING AT 13:27:53
*****
01          FORMAT-BUFFER-GENERATION-EXAMP.
* >>> DIC1809 WARNING: FIELDNAME(S) TRUNCATED
      02 FILLER          PIC X(54) VALUE IS
      'AB,2X,AD,AE,AF1-5,AG,2X,AH,AI,1X,ALC,2,AM1,AN1C,2,AN1(' .
      02 FILLER          PIC X(54) VALUE IS
      '1-11),AO1,2X,AR1,2X,AM2,AN2C,2,AN2(1-11),AO2,2X,AR2,2X' .
      02 FILLER          PIC X(54) VALUE IS
      ',AM3,AN3C,2,AN3(1-11),AO3,2X,AR3,2X,AM4,AN4C,2,AN4(1-1' .
      02 FILLER          PIC X(54) VALUE IS
      '1),AO4,2X,AR4,2X,AM5,AN5C,2,AN5(1-11),AO5,2X,AR5,2X,AM' .
      02 FILLER          PIC X(54) VALUE IS
      '6,AN6C,2,AN6(1-11),AO6,2X,AR6,2X,AM7,AN7C,2,AN7(1-11),' .
      02 FILLER          PIC X(23) VALUE IS
      'AO7,2X,AR7,2X,AS,AT,AU.' .
* *****

DIC1800 SUMMARY:      25 FIELD(S) PROCESSED
DIC1809 WARNING:      1 FIELDNAME(S) TRUNCATED
DIC1818 WARNING:      3 FORMAT(S) CHANGED
DIC1819 MESSAGE:      33 SLACK BYTE(S) GENERATED
DIC1847 MESSAGE:      3 FIELD(S) SKIPPED FOR RECORDBUFFER STRUCTURE
    
```