**INC518933 - Thesauras Investigation**

**Description**

I have presented the main thesaurus logic to the team, but there is a bit more detail needed on a couple of aspects of how the current system does comparisons. Would it be possible to learn the following:

• How are search terms altered prior to comparison?

o E.g. Double vowels are removed (sometimes? Always?)

o 'Ks' are replaced with 'Cs'?

o Numerals are transposed to letters (1 becomes One)

o Other?

**Analysis**

**CONE3055 – Load the word array**

**COBRP.THESAURS.DATA.\* or COBRP.QMFP.THESAURS.\*  
2555 records**

For each word in the search string (I)

DB2PARM\_IO.SEARCH\_RELATWRD = WORD(I)

DO WHILE (VERIFY(SEARCH\_RELATWRD, ' %') > 0)

UNTIL (DB2PARM\_IO.DB2SQLCODE = NOT\_FOUND | THE\_WORD (I, 2) ¬= '');

FETCH THESAURAS WORDID

USING FETCHED WORDID REFETCH ALL SYNONYMS. UP TO 40.

THREE DIFFERENT PATHS

**CONE3040 - FETCH\_NAME**

NAME\_INDEX = NAME\_INDEX + 1;

RETRIEVED\_NAME(NAME\_INDEX) = (10)' ' || WORD1\_VAR || ' ' ||WORD2\_VAR;

**NE\_AREA.SEARCH\_KEY = WORD1\_VAR || WORD2\_VAR || (140)'%';**

L = INDEX (NE\_AREA.SEARCH\_KEY, ' ');

DO WHILE (L > 0);

**SUBSTR(NE\_AREA.SEARCH\_KEY, L) = SUBSTR(NE\_AREA.SEARCH\_KEY, L+1)**

**|| '%';**

L = INDEX (NE\_AREA.SEARCH\_KEY, ' ');

END;

COMPNAME 968639 NO SPACES  
COMPNAM1 970543 SPACES ALLOWED



CONE40 Subroutine does the insert into this table. COMPRESSED NAME is

1/\* CO##0001: PROC,COMPRESSES COMPANY NAME TO THE COBR COMPRESSED KEY \*/

/\* STATUS: VERSION=1.0, ORIGINAL, DATE=870112, WBP \*/

0CO##0001: PROC(COMPANY\_NAME) RETURNS(CHAR(30));

0/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* THIS PROCEDURE ACCEPTS AS INPUT THE CO. NAME AND RETURNS \*/

/\* THE COMPRESSED KEY (30 BYTE CO. NAME) \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

0 DCL COMPANY\_NAME CHAR(150),

BUFFER1 CHAR(150) INIT(' '),

BUFFER2 CHAR(150) INIT(' ');

0 DCL BUFFER1\_ARRAY(150) CHAR(1) DEF BUFFER1,

BUFFER2\_ARRAY(150) CHAR(1) DEF BUFFER2;

0 DCL COMPRESSED\_KEY CHAR(30) DEF BUFFER1;

0 DCL I FIXED BIN(15),

POS1 FIXED BIN(15),

POS2 FIXED BIN(15);

0 DCL PICZ PIC'Z',

PICZ\_DEF CHAR(1) BASED (ADDR (PICZ));

0 DCL NBRTAB(0:9,5) CHAR(1) STATIC INIT(

'Z','E','R','O',' ',

'O','N','E',' ',' ',

'T','W','O',' ',' ',

'T','H','R','E','E',

'F','O','U','R',' ',

'F','I','V','E',' ',

'S','I','X',' ',' ',

'S','E','V','E','N',

'E','I','G','H','T',

'N','I','N','E',' ');

0 DCL NDIGIT(0:9) FIXED BIN(15,0) STATIC INIT

(4,3,3,5,4,4,3,5,5,4);

0 DCL ALPHA CHAR(26) STATIC INIT

('ABCDEFGHIJKLMNOPQRSTUVWXYZ'),

NUM CHAR(10) STATIC INIT ('1234567890'),

ALPHANUM CHAR(38) STATIC INIT

('&#ABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890');

0 DCL SUBSTR BUILTIN,

VERIFY BUILTIN;

1 BUFFER1 = COMPANY\_NAME;

DO I = 1 TO 150;

IF VERIFY (BUFFER1\_ARRAY(1), ALPHANUM) > 0

THEN BUFFER1 = SUBSTR (BUFFER1, 2);

ELSE LEAVE;

END;

IF SUBSTR (BUFFER1, 1, 4) = 'THE '

THEN BUFFER1 = SUBSTR (BUFFER1, 5);

ELSE;

POS2 = 1;

DO POS1 = 1 TO 150;

IF BUFFER1\_ARRAY (POS1) = ''

THEN;

ELSE DO;

BUFFER2\_ARRAY (POS2) = BUFFER1\_ARRAY (POS1);

POS2 = POS2 + 1;

END;

END;

IF SUBSTR (BUFFER2, 1, 15) = 'BRITISHCOLUMBIA'

THEN DO;

BUFFER1 = 'BC';

POS1 = 3;

POS2 = 16;

END;

ELSE DO;

BUFFER1 = '';

POS1, POS2 = 1;

END;

DO UNTIL (POS1 > 30 | SUBSTR (BUFFER2, POS2) = '');

SELECT;

WHEN (VERIFY (BUFFER2\_ARRAY (POS2), ALPHA) = 0) DO;

BUFFER1\_ARRAY (POS1) = BUFFER2\_ARRAY (POS2);

POS1 = POS1 + 1;

END;

WHEN (BUFFER2\_ARRAY (POS2) = '#') DO;

SUBSTR (BUFFER1, POS1) = 'NUMBER';

POS1 = POS1 + 6;

END;

WHEN (BUFFER2\_ARRAY (POS2) = '&') DO;

SUBSTR (BUFFER1, POS1) = 'AND';

POS1 = POS1 + 3;

END;

WHEN (VERIFY(BUFFER2\_ARRAY(POS2), NUM) = 0) DO;

PICZ\_DEF = BUFFER2\_ARRAY(POS2);

DO I = 1 TO (NDIGIT(PICZ));

BUFFER1\_ARRAY (POS1) = NBRTAB (PICZ, I);

POS1 = POS1 + 1;

END;

END;

OTHER;

END;

POS2 = POS2 + 1;

END;

RETURN (COMPRESSED\_KEY);

END CO##0001;

PROGRAMS THAT INSERT INTO COMPNAME

COBRPRS: RESTORE A COMPANY

COBRPSN: CHANGE COMPANY NAME

CONE40

DCL 1 KEY\_ARRAY UNAL BASED (KEY\_ARRAY\_PT

3 #\_OF\_ENTRIES FIXED BIN(15),

3 WORD(50) CHAR(30),

3 LENGTH(50) FIXED BIN(15);

CO$$03 routine

1/\* CO$$0310: INCLUDED, MISCELLANEOUS DECLARES FOR CO$$03 SUBROUTINE \*/

/\* STATUS: VERSION=1.0, ORIGINAL, DATE=921126, WBP \*/

0 DCL PERCENT CHAR(1);

0 DCL COMPANY\_NAME CHAR(\*);

0 DCL (BUFFER1,

BUFFER2) CHAR(300) INIT(' ');

0 DCL BUFFER1\_ARRAY(300) CHAR(1) DEF BUFFER1,

BUFFER2\_ARRAY(300) CHAR(1) DEF BUFFER2;

0 DCL CHAR\_VAR CHAR(30) VARYING;

0 DCL (I, J,

POS1,

POS2) FIXED BIN(15);

0 DCL VALID\_CHAR CHAR(42) STATIC INIT

('AABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890#$¢+'),

NUM CHAR(10) STATIC INIT ('1234567890');

0 DCL PICZ PIC'Z',

PICZ\_DEF CHAR(1) BASED (ADDR (PICZ));

0 DCL NBRTAB(0:9,5) CHAR(1) STATIC INIT(

'Z','E','R','O',' ',

'O','N','E',' ',' ',

'T','W','O',' ',' ',

'T','H','R','E','E',

'F','O','U','R',' ',

'F','I','V','E',' ',

'S','I','X',' ',' ',

'S','E','V','E','N',

'E','I','G','H','T',

'N','I','N','E',' ');

0 DCL NDIGIT(0:9) FIXED BIN(15,0) STATIC INIT

(4,3,3,5,4,4,3,5,5,4);

0 DCL ADDR BUILTIN,

INDEX BUILTIN,

LENGTH BUILTIN,

SUBSTR BUILTIN,

TRANSLATE BUILTIN,

VERIFY BUILTIN;

1/\* CO$$0320: INCLUDED, MAINLINE FOR CO$$03 \*/

/\* STATUS: VERSION=1.0, ORIGINAL, DATE=921126, WBP \*/

0 BUFFER1 = ' ' || COMPANY\_NAME;

0 IF PERCENT = '%'

THEN SUBSTR(VALID\_CHAR, 1, 1) = PERCENT;

ELSE;

0 IF PERCENT = '\*'

THEN DO;

PERCENT = '%';

SUBSTR(VALID\_CHAR, 1, 1) = PERCENT;

BUFFER1 = TRANSLATE (BUFFER1, '%', '\*');

END;

ELSE;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* REMOVE INVALID '(' AND ')' \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

0 DO I = 2 TO 151;

IF BUFFER1\_ARRAY (I) = '(' | BUFFER1\_ARRAY (I) = ')'

THEN BUFFER1\_ARRAY (I) = '';

ELSE;

END;

0/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* CHANGE WORDS CONTAINING SPECIAL CHARACTERS \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

0 I = INDEX (BUFFER1, ' B.C.''S');

DO WHILE (I > 0);

SUBSTR (BUFFER1, I) = ' BC' || SUBSTR (BUFFER1, I + 7);

I = INDEX (BUFFER1, ' B.C.''S');

END;

0 I = INDEX (BUFFER1, ' B. C.''S');

DO WHILE (I > 0);

SUBSTR (BUFFER1, I) = ' BC' || SUBSTR (BUFFER1, I + 8);

I = INDEX (BUFFER1, ' B. C.''S');

END;

0 I = INDEX (BUFFER1, ' B.C. ''S');

DO WHILE (I > 0);

SUBSTR (BUFFER1, I) = ' BC' || SUBSTR (BUFFER1, I + 8);

I = INDEX (BUFFER1, ' B.C. ''S');

END;

0 I = INDEX (BUFFER1, ' B C''S');

POS1 = 1;

DO WHILE (I > 0);

I = I + POS1 - 1;

IF I > 2 &

BUFFER1\_ARRAY(I-2) = ' ' &

BUFFER1\_ARRAY(I-1) ¬= ' ' THEN POS1 = I+6;

ELSE

SUBSTR (BUFFER1, I) = ' BC' || SUBSTR (BUFFER1, I + 6);

I = INDEX (SUBSTR(BUFFER1,POS1),' B C''S');

END;

0 I = INDEX (BUFFER1, ' B C ''S');

POS1 = 1;

DO WHILE (I > 0);

I = I + POS1 - 1;

IF I > 2 &

BUFFER1\_ARRAY(I-2) = ' ' &

BUFFER1\_ARRAY(I-1) ¬= ' ' THEN POS1 = I + 7;

ELSE

SUBSTR (BUFFER1, I) = ' BC' || SUBSTR (BUFFER1, I + 7);

I = INDEX (SUBSTR(BUFFER1,POS1), ' B C ''S');

END;

0 I = INDEX (BUFFER1, ' BC''S');

DO WHILE (I > 0);

SUBSTR (BUFFER1, I) = ' BC' || SUBSTR (BUFFER1, I + 5);

I = INDEX (BUFFER1, ' BC''S');

END;

0 I = INDEX (BUFFER1, ' BC ''S');

DO WHILE (I > 0);

SUBSTR (BUFFER1, I) = ' BC' || SUBSTR (BUFFER1, I + 6);

I = INDEX (BUFFER1, ' BC ''S');

END;

0 I = INDEX (BUFFER1, '''S ');

DO WHILE (I > 0);

SUBSTR (BUFFER1, I) = SUBSTR (BUFFER1, I + 1);

I = INDEX (BUFFER1, '''S ');

END;

0 I = INDEX (BUFFER1, 'O''');

DO WHILE (I > 0);

SUBSTR (BUFFER1, I + 1) = SUBSTR (BUFFER1, I + 2);

I = INDEX (BUFFER1, 'O''');

END;

0 I = INDEX (BUFFER1, 'CO-OPERATIVE ');

DO WHILE (I > 0);

SUBSTR (BUFFER1, I + 2) = SUBSTR (BUFFER1, I + 3);

I = INDEX (BUFFER1, 'CO-OPERATIVE ');

END;

0 I = INDEX (BUFFER1, ' N'' ');

DO WHILE (I > 0);

SUBSTR (BUFFER1, I) = SUBSTR (BUFFER1, I + 3);

I = INDEX (BUFFER1, ' N'' ');

END;

0 I = INDEX (BUFFER1, ' ''N ');

DO WHILE (I > 0);

SUBSTR (BUFFER1, I) = SUBSTR (BUFFER1, I + 3);

I = INDEX (BUFFER1, ' ''N ');

END;

0 I = INDEX (BUFFER1, ' ''N'' ');

DO WHILE (I > 0);

SUBSTR (BUFFER1, I) = SUBSTR (BUFFER1, I + 4);

I = INDEX (BUFFER1, ' ''N'' ');

END;

0/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* REMOVE INVALID CHARACTERS \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

0 BUFFER1 = TRANSLATE (BUFFER1, ' ', '-/');

I = 1;

DO UNTIL (SUBSTR (BUFFER1, I) = '' | I > 151);

IF VERIFY (BUFFER1\_ARRAY (I), VALID\_CHAR) > 0

THEN SUBSTR (BUFFER1, I) = SUBSTR (BUFFER1, I + 1);

ELSE I = I + 1;

END;

1/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* SUBSTITUTIONS FOR #,$,¢,AND NUMBERS 0 TO 9 \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

0 POS1, POS2 = 1;

DO UNTIL (SUBSTR (BUFFER1, POS1) = '');

SELECT;

WHEN (BUFFER1\_ARRAY (POS1) = '$')

IF BUFFER1\_ARRAY (POS1 - 1) = ' '

& BUFFER1\_ARRAY (POS1 + 1) = ' '

THEN DO;

SUBSTR (BUFFER2, POS2) = ' DOLLAR ';

POS2 = POS2 + 8;

END;

ELSE DO;

BUFFER2\_ARRAY (POS2) = 'S';

POS2 = POS2 + 1;

END;

0 WHEN (BUFFER1\_ARRAY (POS1) = '¢')

IF BUFFER1\_ARRAY (POS1 - 1) = ' '

& BUFFER1\_ARRAY (POS1 + 1) = ' '

THEN DO;

SUBSTR (BUFFER2, POS2) = ' CENT ';

POS2 = POS2 + 6;

END;

ELSE DO;

BUFFER2\_ARRAY (POS2) = 'C';

POS2 = POS2 + 1;

END;

0 WHEN (BUFFER1\_ARRAY (POS1) = '+') DO;

SUBSTR (BUFFER2, POS2) = ' PLUS ';

POS2 = POS2 + 6;

END;

0 WHEN (BUFFER1\_ARRAY (POS1) = '#') DO;

SUBSTR (BUFFER2, POS2) = ' NUMBER ';

POS2 = POS2 + 8;

END;

0 WHEN (VERIFY(BUFFER1\_ARRAY(POS1), NUM) = 0) DO;

PICZ\_DEF = BUFFER1\_ARRAY(POS1);

DO I = 1 TO (NDIGIT(PICZ));

BUFFER2\_ARRAY (POS2) = NBRTAB (PICZ, I);

POS2 = POS2 + 1;

END;

END;

0 WHEN (BUFFER1\_ARRAY (POS1) = '%' & PERCENT = '%') DO;

SUBSTR (BUFFER2, POS2) = ' % ';

POS2 = POS2 + 3;

END;

0 OTHER DO;

BUFFER2\_ARRAY (POS2) = BUFFER1\_ARRAY (POS1);

POS2 = POS2 + 1;

END;

END;

POS1 = POS1 + 1;

END;

1/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* REMOVE DOUBLE BLANKS \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

0 BUFFER1 = '';

POS1 = 1;

0 DO I = 1 TO 300 UNTIL (SUBSTR (BUFFER2, I) = '');

IF BUFFER2\_ARRAY (I) = '' & BUFFER2\_ARRAY (I + 1) = ''

THEN;

ELSE DO;

BUFFER1\_ARRAY (POS1) = BUFFER2\_ARRAY(I);

POS1 = POS1 + 1;

END;

END;

-/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* REMOVE OR MODIFY SPECIAL PHRASES \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

0 I = INDEX (BUFFER1, ' BRITISH COLUMBIAS ');

DO WHILE (I > 0);

SUBSTR (BUFFER1, I) = ' BC ' || SUBSTR (BUFFER1, I + 19);

I = INDEX (BUFFER1, ' BRITISH COLUMBIAS ');

END;

0 I = INDEX (BUFFER1, ' BRITISH COLUMBIA ');

DO WHILE (I > 0);

SUBSTR (BUFFER1, I) = ' BC ' || SUBSTR (BUFFER1, I + 18);

I = INDEX (BUFFER1, ' BRITISH COLUMBIA ');

END;

0 I = INDEX (BUFFER1, ' BRITISH COLUMBIAN ');

DO WHILE (I > 0);

SUBSTR (BUFFER1, I) = ' BC ' || SUBSTR (BUFFER1, I + 19);

I = INDEX (BUFFER1, ' BRITISH COLUMBIAN ');

END;

0 I = INDEX (BUFFER1, ' BRITISH COLUMBIANS ');

DO WHILE (I > 0);

SUBSTR (BUFFER1, I) = ' BC ' || SUBSTR (BUFFER1, I + 20);

I = INDEX (BUFFER1, ' BRITISH COLUMBIANS ');

END;

0 I = INDEX (BUFFER1, ' NON PERSONAL LIABILITY ');

IF I > 0

THEN SUBSTR (BUFFER1, I) = SUBSTR (BUFFER1, I + 24);

ELSE;

0 I = INDEX (BUFFER1, ' N P L ');

IF I > 0

THEN SUBSTR (BUFFER1, I) = SUBSTR (BUFFER1, I + 7);

ELSE;

0 I = INDEX (BUFFER1, ' NPL ');

IF I > 0

THEN SUBSTR (BUFFER1, I) = SUBSTR (BUFFER1, I + 5);

ELSE;

0 I = INDEX (BUFFER1, ' IN VOLUNTARY LIQUIDATION ');

IF I > 0

THEN SUBSTR (BUFFER1, I) = SUBSTR (BUFFER1, I + 26);

ELSE;

0 I = INDEX (BUFFER1, ' OF CANADA ');

IF I > 0 & SUBSTR (BUFFER1, I + 11) = ''

THEN SUBSTR (BUFFER1, I) = '';

ELSE;

0 I = INDEX (BUFFER1, ' OF BC ');

IF I > 0 & SUBSTR (BUFFER1, I + 7) = ''

THEN SUBSTR (BUFFER1, I) = '';

ELSE;

1/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* BUILD WORD ARRAY \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

0 IF BUFFER1\_ARRAY(1) = ''

THEN BUFFER1 = SUBSTR (BUFFER1, 2);

ELSE;

0 POS1 = INDEX (BUFFER1, ' ');

0 DO I = 1 TO 50;

0 KEY\_ARRAY.WORD(I) = SUBSTR(BUFFER1, 1, POS1);

IF POS1 > 30

THEN KEY\_ARRAY.LENGTH(I) = 30;

ELSE KEY\_ARRAY.LENGTH(I) = POS1 - 1;

0 BUFFER1 = SUBSTR (BUFFER1, POS1 + 1);

0 IF BUFFER1 = ''

THEN LEAVE;

ELSE POS1 = INDEX (BUFFER1, ' ');

END;

0 IF I > 50

THEN KEY\_ARRAY.#\_OF\_ENTRIES = 50;

ELSE KEY\_ARRAY.#\_OF\_ENTRIES = I;

0/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* EDIT LAST WORD OR PHRASE \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

0 DO UNTIL (KEY\_ARRAY.#\_OF\_ENTRIES = I);

0 I = KEY\_ARRAY.#\_OF\_ENTRIES;

0 SELECT (KEY\_ARRAY.WORD(I));

WHEN ('ASSOCIATION', 'ASSOC', 'ASSN',

'COMPANY', 'CO',

'CORPORATION', 'CORP',

'INCORPORATED', 'INC', 'INCORPOREE',

'LIABILITY',

'LIMITED', 'LTD', 'LIMITEE', 'LTEE',

'SOCIETY', 'SOC') DO;

KEY\_ARRAY.WORD(I) = '';

KEY\_ARRAY.LENGTH(I) = 0;

KEY\_ARRAY.#\_OF\_ENTRIES = KEY\_ARRAY.#\_OF\_ENTRIES - 1;

END;

OTHER;

END;

END;

1/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* REMOVE SOME PREPOSITIONS AND DUPLICATE WORDS \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

0 I = 1;

DO WHILE (I <= KEY\_ARRAY.#\_OF\_ENTRIES);

IF (KEY\_ARRAY.WORD(I) = 'PLUS' & I > 1)

| KEY\_ARRAY.WORD(I) = 'AMPERSAND'

| KEY\_ARRAY.WORD(I) = 'AND'

| KEY\_ARRAY.WORD(I) = 'OF'

| KEY\_ARRAY.WORD(I) = 'THE'

| KEY\_ARRAY.WORD(I) = 'TO'

THEN DO;

DO J = I TO KEY\_ARRAY.#\_OF\_ENTRIES;

KEY\_ARRAY.WORD(J) = KEY\_ARRAY.WORD(J + 1);

KEY\_ARRAY.LENGTH(J) = KEY\_ARRAY.LENGTH(J + 1);

END;

KEY\_ARRAY.#\_OF\_ENTRIES = KEY\_ARRAY.#\_OF\_ENTRIES - 1;

END;

ELSE I = I + 1;

END;

0/\*I = 1;

DO WHILE (I <= KEY\_ARRAY.#\_OF\_ENTRIES);

IF KEY\_ARRAY.WORD(I) = KEY\_ARRAY.WORD(I + 1)

THEN DO;

DO J = I TO KEY\_ARRAY.#\_OF\_ENTRIES;

KEY\_ARRAY.WORD(J) = KEY\_ARRAY.WORD(J + 1);

KEY\_ARRAY.LENGTH(J) = KEY\_ARRAY.LENGTH(J + 1);

END;

KEY\_ARRAY.#\_OF\_ENTRIES = KEY\_ARRAY.#\_OF\_ENTRIES - 1;

END;

ELSE I = I + 1;

END;

\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* CONCATENATE SINGLE LETTER WORDS \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

0 DO I = 1 TO 50 WHILE (I < KEY\_ARRAY.#\_OF\_ENTRIES);

IF KEY\_ARRAY.LENGTH(I) = 1 & KEY\_ARRAY.LENGTH(I + 1) = 1

& KEY\_ARRAY.WORD(I) ¬= '%' & KEY\_ARRAY.WORD(I + 1) ¬= '%'

THEN DO;

CHAR\_VAR = SUBSTR (KEY\_ARRAY.WORD(I), 1, 1);

0 DO WHILE (KEY\_ARRAY.LENGTH(I + 1) = 1 &

KEY\_ARRAY.WORD(I + 1) ¬= '%');

CHAR\_VAR = CHAR\_VAR || SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 1);

DO J = (I + 1) TO KEY\_ARRAY.#\_OF\_ENTRIES;

KEY\_ARRAY.WORD(J) = KEY\_ARRAY.WORD(J + 1);

KEY\_ARRAY.LENGTH(J) = KEY\_ARRAY.LENGTH(J + 1);

END;

KEY\_ARRAY.#\_OF\_ENTRIES = KEY\_ARRAY.#\_OF\_ENTRIES - 1;

END;

0 KEY\_ARRAY.WORD(I) = CHAR\_VAR;

KEY\_ARRAY.LENGTH(I) = LENGTH (CHAR\_VAR);

END;

ELSE;

END;

1/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* CONCATENATE SPECIAL PHRASES \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

0 DO I = 1 TO 50 WHILE (I < KEY\_ARRAY.#\_OF\_ENTRIES);

0 IF (KEY\_ARRAY.WORD(I) = 'AIR' &

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 9) = 'CONDITION')

| (KEY\_ARRAY.WORD(I) = 'AUTO' &

(SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 6) = 'BROKER' |

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 5) = 'COURT' |

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 4) = 'SALE'))

| (KEY\_ARRAY.WORD(I) = 'BED' &

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 9) = 'BREAKFAST')

| (KEY\_ARRAY.WORD(I) = 'BOWEN' & KEY\_ARRAY.WORD(I+1) ='ISLAND')

| (KEY\_ARRAY.WORD(I) = 'CAMPBELL' & KEY\_ARRAY.WORD(I+1) ='RIVER' )

| (KEY\_ARRAY.WORD(I) = 'CAN' &

(KEY\_ARRAY.WORD(I + 1) = 'AM' |

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 7) = 'AMERICA'))

| (KEY\_ARRAY.WORD(I) = 'CANADIAN' &

KEY\_ARRAY.WORD(I + 1) = 'AMERICAN')

| (KEY\_ARRAY.WORD(I) = 'CAR' &

(SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 6) = 'CENTER' |

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 6) = 'CENTRE' |

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 4) = 'SALE'))

| (KEY\_ARRAY.WORD(I) = 'COFFEE' & KEY\_ARRAY.WORD(I+1) ='SHOP' )

| (KEY\_ARRAY.WORD(I) = 'COLD' &

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 6) = 'STORAG')

| (KEY\_ARRAY.WORD(I) = 'DAWSON' & KEY\_ARRAY.WORD(I+1) ='CREEK' )

| (KEY\_ARRAY.WORD(I) = 'FOOD' &

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 6) = 'LOCKER')

| (KEY\_ARRAY.WORD(I) = 'FORT' & KEY\_ARRAY.WORD(I+1) ='ST' )

| (KEY\_ARRAY.WORD(I) = 'FOREST' &

(SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 5) = 'MANAG' |

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 4) = 'PROD'))

| (KEY\_ARRAY.WORD(I) = 'FRASER' & KEY\_ARRAY.WORD(I+1) ='VALLEY' )

| (KEY\_ARRAY.WORD(I) = 'GOLD' & KEY\_ARRAY.WORD(I+1) ='RIVER' )

| (KEY\_ARRAY.WORD(I) = 'HEAT' &

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 5) = 'PUMPS')

| (KEY\_ARRAY.WORD(I) = 'LIMITED' &

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 7) = 'EDITION')

| (KEY\_ARRAY.WORD(I) = 'JET' &

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 4) = 'WASH')

| (KEY\_ARRAY.WORD(I) = 'MANUFACTURED' &

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 4) = 'HOME')

| (KEY\_ARRAY.WORD(I) = 'MAPLE' & KEY\_ARRAY.WORD(I+1) ='RIDGE' )

| (KEY\_ARRAY.WORD(I) = 'MILL' & KEY\_ARRAY.WORD(I+1) ='BAY' )

| (KEY\_ARRAY.WORD(I) = 'MOBILE' &

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 4) = 'HOME')

| (KEY\_ARRAY.WORD(I) = 'MODULAR' &

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 4) = 'HOME')

| (KEY\_ARRAY.WORD(I) = 'MOTOR' &

(SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 5) = 'COURT' |

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 4) = 'HOME' |

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 5) = 'HOTEL' |

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 3) = 'INN'))

| (KEY\_ARRAY.WORD(I) = 'MUTUAL' &

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 4) = 'FUND')

THEN DO;

CHAR\_VAR = SUBSTR (KEY\_ARRAY.WORD(I), 1, KEY\_ARRAY.LENGTH(I))||

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1,

KEY\_ARRAY.LENGTH(I + 1));

DO J = (I + 1) TO KEY\_ARRAY.#\_OF\_ENTRIES;

KEY\_ARRAY.WORD(J) = KEY\_ARRAY.WORD(J + 1);

KEY\_ARRAY.LENGTH(J) = KEY\_ARRAY.LENGTH(J + 1);

END;

0 KEY\_ARRAY.#\_OF\_ENTRIES = KEY\_ARRAY.#\_OF\_ENTRIES - 1;

0 KEY\_ARRAY.WORD(I) = CHAR\_VAR;

KEY\_ARRAY.LENGTH(I) = LENGTH (CHAR\_VAR);

END;

ELSE IF

(KEY\_ARRAY.WORD(I) = 'NEW' & KEY\_ARRAY.WORD(I+1) ='WEST' )

| (KEY\_ARRAY.WORD(I) = 'NEW' & KEY\_ARRAY.WORD(I+1) ='WESTMINSTER')

| (KEY\_ARRAY.WORD(I) = 'NORTH' &

(KEY\_ARRAY.WORD(I + 1) = 'AMERICA' |

KEY\_ARRAY.WORD(I + 1) = 'AMERICAN' |

KEY\_ARRAY.WORD(I + 1) = 'EAST' |

KEY\_ARRAY.WORD(I + 1) = 'VAN' |

KEY\_ARRAY.WORD(I + 1) = 'VANCOUVER' |

KEY\_ARRAY.WORD(I + 1) = 'WEST'))

| (KEY\_ARRAY.WORD(I) = 'OAK' & KEY\_ARRAY.WORD(I+1) = 'BAY' )

| (KEY\_ARRAY.WORD(I) = 'PACIFIC' &

(KEY\_ARRAY.WORD(I+1) = 'COAST' | KEY\_ARRAY.WORD(I+1) = 'RIM'))

| (KEY\_ARRAY.WORD(I) = 'PEE' & KEY\_ARRAY.WORD(I+1) ='GEE' )

| (KEY\_ARRAY.WORD(I) = 'PITT' & KEY\_ARRAY.WORD(I+1) ='MEADOWS')

| (KEY\_ARRAY.WORD(I) = 'PORT' &

(KEY\_ARRAY.WORD(I+1) = 'ALBERNI' |

KEY\_ARRAY.WORD(I+1) = 'CALL' |

KEY\_ARRAY.WORD(I+1) = 'COQUITLAM' |

KEY\_ARRAY.WORD(I+1) = 'MOODY' |

KEY\_ARRAY.WORD(I+1) = 'O'))

| (KEY\_ARRAY.WORD(I) = 'POWELL' & KEY\_ARRAY.WORD(I+1) ='RIVER' )

| (KEY\_ARRAY.WORD(I) = 'POWER' &

(SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 5) = 'CLEAN' |

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 4) = 'WASH'))

| (KEY\_ARRAY.WORD(I) = 'PRESSURE' & KEY\_ARRAY.WORD(I+1) ='WASH' )

| (KEY\_ARRAY.WORD(I) = 'PRINCE' &

(KEY\_ARRAY.WORD(I + 1) = 'GEORGE' |

KEY\_ARRAY.WORD(I + 1) = 'RUPERT'))

| (KEY\_ARRAY.WORD(I) = 'QUEEN' &

KEY\_ARRAY.WORD(I + 1) = 'CHARLOTTE')

| (KEY\_ARRAY.WORD(I) = 'READY' & KEY\_ARRAY.WORD(I+1) = 'MIX' )

| (KEY\_ARRAY.WORD(I) = 'REAL' & KEY\_ARRAY.WORD(I+1) = 'ESTATE' )

| (KEY\_ARRAY.WORD(I) = 'RECREATION' &

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 7) = 'VEHICLE')

| (KEY\_ARRAY.WORD(I) = 'RED' & KEY\_ARRAY.WORD(I+1) = 'D' )

| (KEY\_ARRAY.WORD(I) = 'SALMON' & KEY\_ARRAY.WORD(I+1) = 'ARM' )

| (KEY\_ARRAY.WORD(I) = 'SERVICE' &

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 7) = 'STATION')

| (KEY\_ARRAY.WORD(I) = 'SIMON' & KEY\_ARRAY.WORD(I+1) = 'FRASER' )

| (KEY\_ARRAY.WORD(I) = 'SOUTH' &

(KEY\_ARRAY.WORD(I+1) = 'WEST' | KEY\_ARRAY.WORD(I+1) = 'EAST'))

| (KEY\_ARRAY.WORD(I) = 'TIMBER' &

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 5) = 'MANAG')

| (KEY\_ARRAY.WORD(I) = 'TREE' &

(SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 5) = 'PLANT' |

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 4) = 'PRUN' |

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 7) = 'SERVICE' |

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 7) = 'SURGEON' |

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 3) = 'TOP'))

| (KEY\_ARRAY.WORD(I) = 'WEST' & KEY\_ARRAY.WORD(I + 1) = 'COAST')

| (KEY\_ARRAY.WORD(I) = 'WHITE' & KEY\_ARRAY.WORD(I + 1) = 'ROCK')

| (KEY\_ARRAY.WORD(I) = 'UNIVERSITY' &

KEY\_ARRAY.WORD(I + 1) = 'VICTORIA')

| (KEY\_ARRAY.WORD(I) = 'VANCOUVER' &

KEY\_ARRAY.WORD(I + 1) = 'ISLAND' )

| (KEY\_ARRAY.WORD(I) = 'VAN' & KEY\_ARRAY.WORD(I + 1) = 'ISLE' )

| (KEY\_ARRAY.WORD(I) = 'WALL' &

(SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 5) = 'COVER' |

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 5) = 'PAPER'))

| (KEY\_ARRAY.WORD(I) = 'WATER' &

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 6) = 'SYSTEM')

| (KEY\_ARRAY.WORD(I) = 'WILLIAMS' &

KEY\_ARRAY.WORD(I + 1) = 'LAKE')

| (KEY\_ARRAY.WORD(I) = 'WINDOW' &

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 7) = 'FASHION')

| (KEY\_ARRAY.WORD(I) = 'WOOD' &

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1, 7) = 'PRODUCT')

THEN DO;

CHAR\_VAR = SUBSTR (KEY\_ARRAY.WORD(I), 1, KEY\_ARRAY.LENGTH(I))||

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1,

KEY\_ARRAY.LENGTH(I + 1));

DO J = (I + 1) TO KEY\_ARRAY.#\_OF\_ENTRIES;

KEY\_ARRAY.WORD(J) = KEY\_ARRAY.WORD(J + 1);

KEY\_ARRAY.LENGTH(J) = KEY\_ARRAY.LENGTH(J + 1);

END;

0 KEY\_ARRAY.#\_OF\_ENTRIES = KEY\_ARRAY.#\_OF\_ENTRIES - 1;

0 KEY\_ARRAY.WORD(I) = CHAR\_VAR;

KEY\_ARRAY.LENGTH(I) = LENGTH (CHAR\_VAR);

END;

ELSE;

END;

0 DO I = 1 TO 50 WHILE (I < KEY\_ARRAY.#\_OF\_ENTRIES);

IF (KEY\_ARRAY.WORD(I) = 'FORTST' &

(KEY\_ARRAY.WORD(I + 1) = 'JOHN' |

KEY\_ARRAY.WORD(I + 1) = 'JAMES'))

| (KEY\_ARRAY.WORD(I) = 'PORTO' & KEY\_ARRAY.WORD(I + 1) = 'CALL' )

| (KEY\_ARRAY.WORD(I) = 'QUEENCHARLOTTE' &

KEY\_ARRAY.WORD(I + 1) = 'ISLANDS' )

| (KEY\_ARRAY.WORD(I) = 'REDD' & KEY\_ARRAY.WORD(I+1) = 'MIX' )

| (KEY\_ARRAY.WORD(I) = 'SIMONFRASER' &

KEY\_ARRAY.WORD(I + 1) = 'UNIVERSITY' )

THEN DO;

CHAR\_VAR = SUBSTR (KEY\_ARRAY.WORD(I), 1, KEY\_ARRAY.LENGTH(I))||

SUBSTR (KEY\_ARRAY.WORD(I + 1), 1,

KEY\_ARRAY.LENGTH(I + 1));

DO J = (I + 1) TO KEY\_ARRAY.#\_OF\_ENTRIES;

KEY\_ARRAY.WORD(J) = KEY\_ARRAY.WORD(J + 1);

KEY\_ARRAY.LENGTH(J) = KEY\_ARRAY.LENGTH(J + 1);

END;

0 KEY\_ARRAY.#\_OF\_ENTRIES = KEY\_ARRAY.#\_OF\_ENTRIES - 1;

0 KEY\_ARRAY.WORD(I) = CHAR\_VAR;

KEY\_ARRAY.LENGTH(I) = LENGTH (CHAR\_VAR);

END;

ELSE;

END;

1/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* REMOVE BC IF LAST WORD \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

0 IF KEY\_ARRAY.WORD(KEY\_ARRAY.#\_OF\_ENTRIES) = 'BC'

THEN DO;

KEY\_ARRAY.WORD(I) = '';

KEY\_ARRAY.LENGTH(I) = 0;

KEY\_ARRAY.#\_OF\_ENTRIES = KEY\_ARRAY.#\_OF\_ENTRIES - 1;

END;

ELSE;

0/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* MANIPULATE SPECIAL PHRASES \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

0 DO I = 1 TO KEY\_ARRAY.#\_OF\_ENTRIES;

0 J = INDEX (KEY\_ARRAY.WORD(I), 'EX');

DO WHILE (J > 0);

SUBSTR(KEY\_ARRAY.WORD(I), J) = SUBSTR(KEY\_ARRAY.WORD(I), J + 1);

KEY\_ARRAY.LENGTH(I) = KEY\_ARRAY.LENGTH(I) - 1;

J = INDEX (KEY\_ARRAY.WORD(I), 'EX');

END;

0 KEY\_ARRAY.WORD(I) = TRANSLATE(KEY\_ARRAY.WORD(I), 'C', 'K');

0 J = INDEX (KEY\_ARRAY.WORD(I), 'MAC');

DO WHILE (J > 0);

SUBSTR(KEY\_ARRAY.WORD(I), J+1) = SUBSTR(KEY\_ARRAY.WORD(I), J+2);

KEY\_ARRAY.LENGTH(I) = KEY\_ARRAY.LENGTH(I) - 1;

J = INDEX (KEY\_ARRAY.WORD(I), 'MAC');

END;

0 IF KEY\_ARRAY.WORD(I) = '' |

KEY\_ARRAY.WORD(I) = 'NEWWEST' |

KEY\_ARRAY.WORD(I) = 'NEWWESTMINSTER' |

KEY\_ARRAY.WORD(I) = 'PACIFICCOAST'

THEN;

ELSE DO;

J = 1;

DO WHILE (J < KEY\_ARRAY.LENGTH(I));

IF SUBSTR(KEY\_ARRAY.WORD(I), J, 1) =

SUBSTR(KEY\_ARRAY.WORD(I), J + 1, 1)

THEN DO;

SUBSTR (KEY\_ARRAY.WORD(I), J) =

SUBSTR(KEY\_ARRAY.WORD(I), J + 1);

KEY\_ARRAY.LENGTH(I) = KEY\_ARRAY.LENGTH(I) - 1;

END;

ELSE J = J + 1;

END;

END;

END;