ABAP Part I

Lesson 06: String Operations

Lesson Objectives

- In this lesson, you will learn about:
  - The various commands for handling strings

Using the Shift command

- This command is used to shift the contents of a string.
- It shifts a string by a given number of places
- Syntax:
  - *shift s_field [by n places] <mode>.*
- Following points apply:
  - If the by n places clause is omitted, the string is shifted by 1 place.
  - The <mode> defines the direction of movement the options being
    - right
    - left (default)
    - circular: shifts n positions to the left so that leftmost characters appear on the right.
- One can also specify shift…circular right

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

Shift a string up to a given string

- Syntax:
  - ***shift str_1 up to str_2 <mode>***
- Searches str_1 for str_2 and the string up to the field margin.
- If str_2 is not found sy-subrc is set to 4.
- <mode> specification is as given previously
- Eg:

```
data : alpha(10) value 'abcdef'.
Shift alpha up to 'cd'.
Write alpha.
Shift alpha up to 'cd' right.
Write / alpha.
```

- The output is

Cdef
Cd

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

Shifting the string up to the first or last character

- Shift str_1 left deleting leading str2.
  Eg: *Shift '000100100500' left deleting leading '0'.*
  Will output
  100100500

- Shift str_1 right deleting trailing str2.
  Eg: *shift '100100500' right deleting trailing '0'.*
  Will output
  1001005

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

Demo

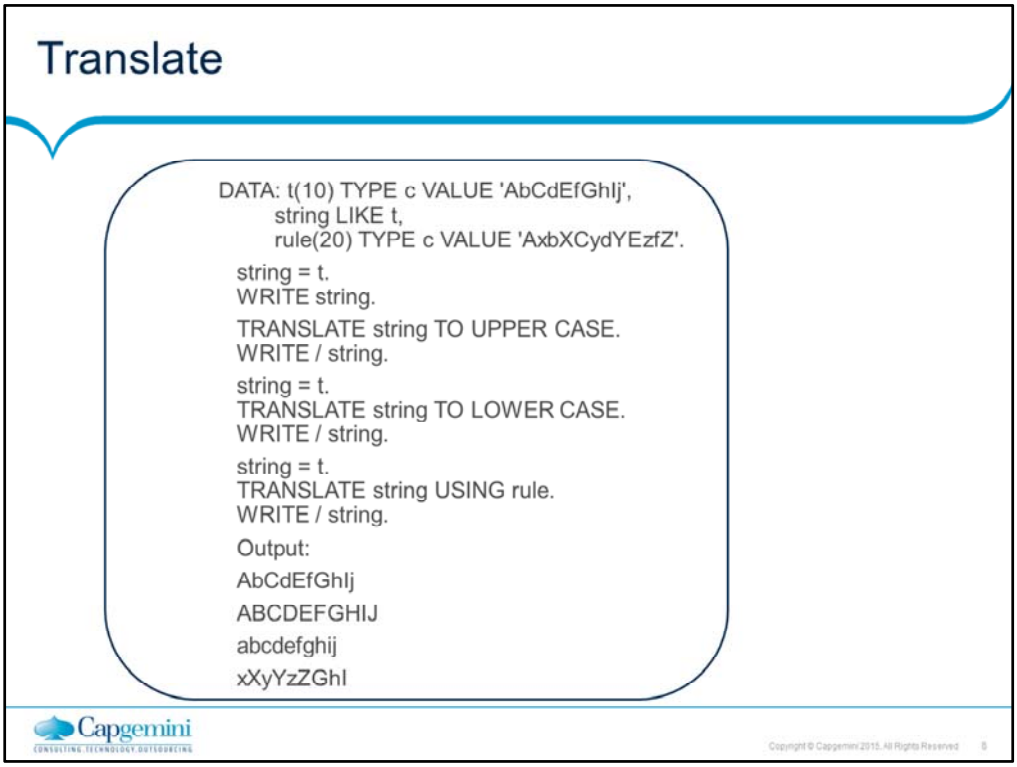
- Program on shift statement

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Translate

- The TRANSLATE statement converts characters into upper or lower case, or uses substitution rules to convert all occurrences of one character to another character.
- Substituting Characters
  - TRANSLATE text USING pattern.
    - This statement replaces all characters in the text field according to the substitution rule stored in pattern.
    - Pattern contains letter pairs where the first letter of each pair is replaced by the second.
    - Pattern can be a variable.
- Syntax:
  - *replace str_1 with str_2 into str_3 [length n].*

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING


Translate
DATA: t(10) TYPE c VALUE 'AbCdEfGhIj',
string LIKE t,
rule(20) TYPE c VALUE 'AxbXCydYEzfZ'.
string = t.
WRITE string.
TRANSLATE string TO UPPER CASE.
WRITE / string.
string = t.
TRANSLATE string TO LOWER CASE.
WRITE / string.
string = t.
TRANSLATE string USING rule.
WRITE / string.
Output:
AbCdEfGhIj
ABCDEFGHIJ
abcdefghij
xXyYzZGhI
Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING
Copyright © Capgemini 2015. All Rights Reserved

Translate

- translate … to upper case.
  - Converts the contents of the field to uppercase.
  - Eg:

    data : alpha(10) value 'abcd12'.

    Translate alpha to upper case.

    Write alpha.

    Will output – ABCD12
- translate … lower case
  - Converts the contents of the field to lower case.

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

Translate

- translate X using Y.
  - Modifies the contents of X using the values given in Y.
  - Replaces each occurrence of the odd numbered characters of Y in X with the even numbered character in Y.
  - Eg:

    data : alpha(10) value 'abcd12'.

    * a->A : B->c : 1->2

    Translate alpha using 'aABc12'.

    Write alpha.

    Will output Abcd22.

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

Demo

- Program on Translate statement

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

Find

- FIND statement is used for finding occurrences of a pattern within a string.
- It can also be used for finding the string pattern within an internal table.
  - *Find <str> for <pattern> <option>.*
    - *<str>* is the string that is being searched.
    - *<pattern>* is the sequence of characters we're looking for.
  - While specifying the search patterns the following forms are available .
    - Find <first occurrence> <all occurrences> of pattern in <str> <respecting|Ignoring case> <match count mcnt> <match length mlen>
    - <match offset moff> <results result_tab|result_wa>

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

Demo

- Program on find statement

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 13

Strlen

Used to find the length of the string.

```
Eg:     data :  len type I,
        string(12) value 'avbsok'.
 len = strlen( string ).
 write len.
```

The output will be 6.

Demo

- Program on strlen

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Condense

- This command is used to remove superfluous spaces from a string.
- After using the *condense* statement, all the words in the string are separated by 1 blank space only.

Eg: *data : alpha(20) value 'lets get on-line whatsay??'.*
condense alpha.
write alpha.

Will display

lets get on-line whatsay??

Condense no-gaps

- With the no-gaps option all the blank spaces from the string are removed and the string is left-justified.

```
Alpha = '      lets get on-line whatsay??'.
Condense alpha no-gaps.
Write alpha.
```

This will display

letsgeton-linewhatsay??

Demo

- Program on condense and condense no-gaps

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Concatenate

- Is used to combine separate strings into a single string.

Eg:
```
data :  f1(20) value 'A',
        F2(20) value 'B',
        F3(20).
concatenate f1 f2 into f3.
write f3.
```
Gives the following output

AB

Concatenate

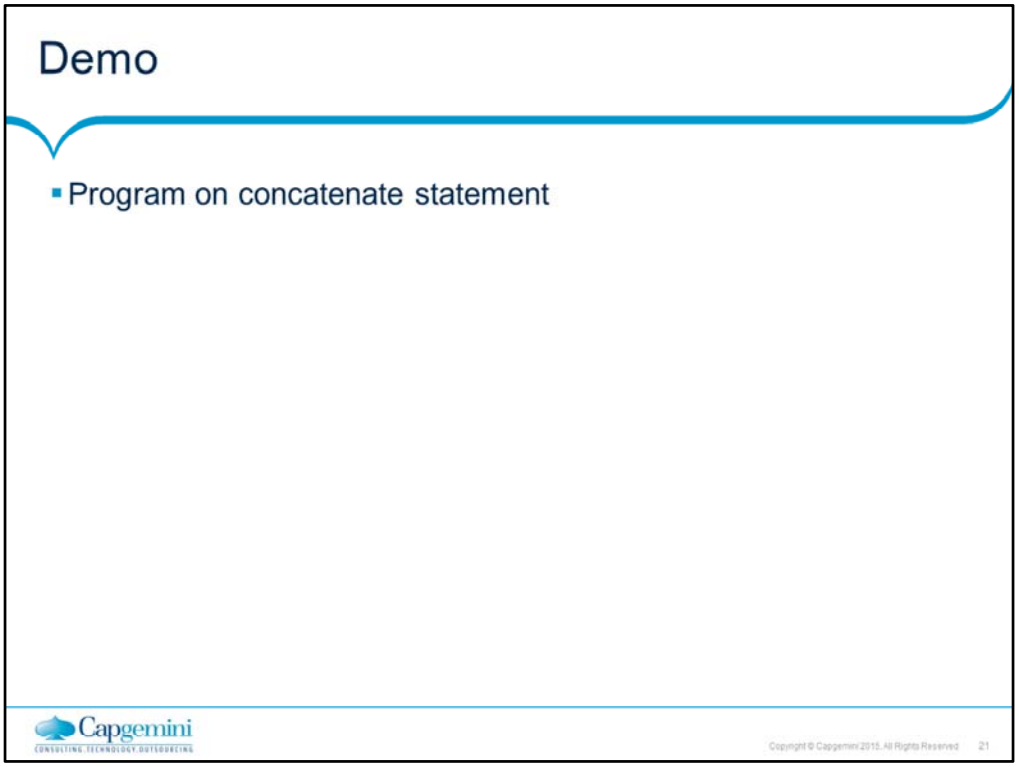
The separated by option is used as follows

concatenate f1 f2 into f3 separated by ''.*

write f3.

Gives the following output

A*B.


Demo
Program on concatenate statement
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING
Copyright © Capgemini 2015. All Rights Reserved
21

Split command

- The *split* command is used to split a string at a given delimiter. The format is
  - ***split <string> at <delimiter> into <f1> <f2> <f3>….***

Eg: *data : string(20) value 'hello what was that??',*
F1(10),
F2(10).
split string at ' ' into f1 f2.
F1 and f2 will contain 'hello' and 'what' respectively.

Operators for Character Strings

- These operators can be used in any comparison expression. The CS, NS, CP, and NP operators ignore trailing blanks and are not case sensitive

Operator	Means	True When	Case Sensitive?	Trailing Blanks Ignored?
v1 CO v2	Contains Only	v1 is composed solely of characters in v2	Yes	No
v1 CN v2	not v1 CO v2	v1 contains characters that are not in v2	Yes	No
v1 CA v2	Contains Any	v1 contains at least one character in v2	Yes	No
v1 NA v2	not v1 CA v2	v1 does not contain any character in v2	Yes	No
v1 CS v2	Contains String	v1 contains the character string v2	No	Yes
v1 NS v2	not v1 CS v2	v1 does not contain the character string v2	No	Yes
v1 CP v2	Contains Pattern	v1 contains the pattern in v2	No	Yes
v1 NP v2	not v1 NP v2	v1 does not contain the pattern in v2	No	Yes

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

Pattern Matching Characters

- The *CP* (contains pattern) and *NP* (no pattern) operators perform a string search that allows pattern-matching characters.
- The expression *v1 CP v2* is true when *v1* contains a string that matches the pattern in *v2*.
- The expression *v1 NP v2* is true when *v1* does not contain a string that matches the pattern in *v2*.
- The pattern matching characters allowed in *v2* are given in table below.

Character	Used to
*	Match any sequence of characters
+	Match any single character
#	Interpret the next character literally

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

.

Example of Wild card characters used for Pattern Matching

Statement	True When
v1 CP 'A+C'	v1 contains "a" in the first position and "c" in the third. Either character can be in upper- or lowercase. Any character can appear in the second position
v1 CP '*Ab*'	The string can appear anywhere within v1. Either character can be in upper- or lowercase.
v1 CP '*#A#b*'	v1 contains a capital A followed by lowercase b
v1 CP '*##*'	v1 contains a #

- # is the escape character.
- A single character following it is interpreted exactly.
- Special meaning, if it exists, is lost.
- # can also be used to make a search case sensitive or to search for the *, +, or # character.
- The escape character is needed when you want to perform a case-sensitive search using *CS, NS, CP,* or *NP*.
- You also need it if you want to perform a pattern search (*CP* or *NP*) for a string containing *, +, or #.

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

Values of sy-fdpos with string comparisons

Comparison	if True sy-fdpos =	if False sy-fdpos =
v1 CO v2	length (v1)	1st char (v1) not in v2
v1 CN v2	1st char (v1) in v2	length (v1)
v1 CA v2	1st char (v1) in v2	length (v1)
v1 NA v2	length (v1)	1st char (v1) in v2
v1 CS v2	1st char (v2) in v1	length (v1)
v1 NS v2	length (v1)	1st char (v1) in v2
v1 CP v2	1st char (v2) in v1	length (v1)

- Use of these operators always sets the system variables *sy-fdpos*.
- If the result of the comparison is true, *sy-fdpos* contains the zero-based offset of the first matching or non-matching character.
- Otherwise, *sy-fdpos* contains the length of *v1*.
- The value assigned to *sy-fdpos* by each operator is described in the above table

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

Demo

- Program operators for Character Strings

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Summary

- In this Lesson you learnt:
  - Shift, replace, translate, find, strlen, condense, concatenate, and split command
  - Operators for Handling Strings

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

Review Question

- Question 1: _____ is used to find the length of the string.
- Question 2: Condense and Concatenate command perform the same function.
  - True/False