UNIT 10

COBOL Intrinsic Functions

Instrinsic Functions

- **Instrinsic functions allow you to access certain values that are derived at run time**

- **Examples**
  - **Current-date**
  - **Length**
  - **Lower-case**
  - **Date-of-integer**

- **Coded as part of statements in the Procedure Division**

- **FUNCTION is now a reserved word**

Figure 10-1 Instrinsic Functions

Notes:

Instrinsic functions are coded in statements in the procedure division. The function is evaluated and the value participates in the statement execution. Functions may not stand alone they must be coded as part of another statement.

Notice that the word FUNCTION is now a reserved word in COBOL. But the names of the functions are not reserved.

Instrinsic Functions – Syntax

- **Specify the reserve word "FUNCTION" followed by the name of the function**
- **Instrinsic Functions may not be used as a receiving operand**
- **Examples**

```
Move  function current-date to d-string

If function date-of-integer(base-date)……

When function day-of-integer(base-date)……
```

Figure 10-2 Instrinsic Functions - Syntax

Notes:

Intrinsic Functions – Arguments and Values

- **The number and format of the arguments depend on the function**
- **The resulting value is an elementary data item implicitly defined by COBOL**
- **Numeric and integer functions may only be used where arithmetic expressions may be used**
- **If fuction value is a character(alphanumeric) string, the reference may be followed by a reference modification**

```
Move function current-date (1:8)  to rpt-string
```

Figure 10-3 Instrinsic Functions – Arguments and Values

Notes:

- The number and format of the arguments depend on the function
- The resulting value is an elementary data item implicitly defined by COBOL
  - The value is a character string, a numeric value, or an integer
  - The length of the result depends on the function and the length of the argument(s)
- Numeric and integer functions may only be used where arithmetic expressions may be used

If a function value is a character(alphanumeric) string, the reference may be followed by a reference modification

COBOL Intrinsic Functions – Date Formats

- **Range: January 1, 1601 to December 31, 9999**
- **Gregorian Date – YYYYMMDD**
- **Integer Date - 1 to 3,067,671 number of days since December 31, 1600**
- **Julian Date – YYYYDDD**

Figure 10-4 COBOL Instrinsic Functions – Date Formats

Notes:

Before we examine date type intrinsic functions, we need to define threee basic date formats that COBOL can work with

Gregorian date, or Standard date

- An eight digit date of the form YYYYMMDD
- In the range of January 1, 1601 throug December 31, 9999
- With MM being from 01 through 12 and DD being from 01 through 31, dependent upon the month

Integer date

- An integer in the range 1 to 3,067,671
- Represents the number of days since December 31, 1600
- For example, January 1, 1994 is 143908 as an integer date

Julian date

- A seven digit integer of the form YYYYDD
- DDD is between 1 and 366, must be valid for the year(that is, leap year are taken into account)

COBOL Instrinsic Functions- CURRENT-DATE

MOVE FUNCTION CURRENT-DATE(1:8) TO DATE-ON-FILE

Returns a character string – length 21
YYYYMMDDHHmmsshhShhmm
System Gregorian date
Current time in 24 hour clock
Difference from GMT

Figure 10-5 COBOL Instrinsic Functions – CURRENT-DATE

Notes:

- Current-date returns a character string of length 21, as follows:
  YYYYMMDDHHmmsshhShhmm

Representing

- A Gregorian date(YYYYMMDD)
- Current time in hours (24-hour clock), minutes, seconds, and hundredths of a second(HHmmsshh)
- Difference of local time zone from Greenwich Mean Time as a '+' or '-' followed by the hours and minutes difference (Shhmm)

COBOL Intrinsic Functions – DATE-OF-INTEGER

COMPUTE RPT-DATE = FUNCTION DATE-OF-INTEGER(INT-DATE)
* If INT-DATE is integer 144337 then RPT-DATE is integer 19960317

Returns a Gregorian date using an inputted integer date

*** Based on number of days since Dec. 31, 1600**
*** Integer could have been output of date intrinsic function modified by arithmetic**

Figure 10-6 COBOL Instrinsic Functions –DATE-OF-INTEGER

Notes:

- Date-of-integer converts an integer date to a Gregorian date

  Function date-of-integer(argument)

- The argument must be a valid integer date
- The function reference is an integer representing YYYYMMDD

COBOL Intrinsic Functions – INTEGER-OF-DAY

COMPUTE INT-DATE = FUNCTION INTEGER-OF-DAY(JUL-DATE)
* If JUL-DATE is integer 1996107 then INT-DATE is integer 144337

Returns an integer using an inputeed Julian date

- **Number of days since Dec. 31, 1600**
- **Integer can be used for date arithmetic or date comparisons.**

Figure 10-7 COBOL Instrinsic Functions –INTEGER-OF-DAY

Notes:

Integer-of-day converts a Julian date to an integer date

Function integer-of-day(argument)

The argument must be a valid Julian date(YYYYDDD)

The function reference is an integer date

COBOL Intrinsic Functions – DAY-OF-INTEGER

COMPUTE JUL-DATE = FUNCTION DAY-OF-INTEGER(INT-DATE)
* If INT-DATE is integer 144337 then JUL-DATE is integer 1996107

Returns an Julian date using an inputeed Julian date

- **Based on number of days since Dec. 31, 1600**
- **Integer could have been output of date intrinsic function of Gregorian date**

Figure 10-8 COBOL Instrinsic Functions –DAY-OF-INTEGER

Notes:

- Day-of-integer converts a Integer date to an Julian date

  Function integer-of-day(argument)

  - The argument must be a valid integer date
  - The function reference is an integer representing YYYYDDD

New Intrinsic Functions Overview

- **New Intrinsic Functions**
  - **DATE-To-YYYYMMDD**
  - **DAY-TO-YYYYDDD**
  - **YEAR-TO-YYYY**

- **First function argument is date with two position year (yymmdd, yyddd, or yy)**

- **Second function argument is an optional integer that is used in determination of 100-year range used in YY to YYYY conversion. Default is 50**

- **Returned value is a date of the same type as the first argument but with a four digit year**

Figure 10-9 New Intrinsic Functions Overview

Notes:

- New Intrinsic Functions
  - The second argument to these three functions is called the sliding window, and it works this way:
    - Add the second argument to the current (run-time) year (as a four digit year), giving an ending year.
      - For example, if a program is running in 1998 and the sliding window is 20, then the result of the add is 2018.
    - Subtract 99 from the ending year to get a 100-year range.
      - For example, 1919-2018
    - For two digit years in the range of 00 to **last-two-digits-of-end**, assign the century from the ending date; for two digit years in the range of **last-two-digts-of-start** to 99, assign the century from the starting date.
      - For example, the ranges are 00-18 and 19-99, so given a year of 82, assign a century of 19; given a year of 17, assign a century of 20.

Intrinsic Function: DATE-TO-YYYYMMDD

- **Syntax**

  FUNCTION DATE-TO-YYYYMMDD (YYMMDD[SW])

- **Examples**

COMPUTE FUNCTION DATE-TO-YYYYMMDD (IN-HIRE-DATE) TO OUT-HIRE-DATE

IF FUNCTION DATE-TO-YYYYMMDD (BIRTH-DATE-20) > QUERY-DATE THEN ….

- **VALUE EXAMPLES:**

RUN-TIME YEAR	INPUT VALUE	SW Argument	FUNCTION VALUE
1998	890315	-10	18890315
1998	770122	-10	19770122
1998	890315	-1	19890315
1998	770122	-1	19770122
1998	890315	0	19890315
1998	770122	0	19770122
1998	890315	85	19890315
1998	770315	85	20770315
1998	890315	-120	
1998	890315	120	

Figure 10-10 Intrinsic Function: DATE-TO-YYYYMMDD

Notes:

Intrinsic Function: DATE-TO-YYYYDD

- **Syntax**

FUNCTION DATE-TO-YYYYDD (YYDD[SW])

- **Examples**

COMPUTE FUNCTION DATE-TO-YYYYDD (IST-LOGON-DAY) TO OUT-LOGON-DATE

IF FUNCTION DATE-TO-YYYYDD(ID-DATE-20) > QUERY-DATE THEN ….

- **VALUE EXAMPLES:**

RUN-TIME YEAR	INPUT VALUE	SW Argument	FUNCTION VALUE
1998	89315	-10	1889315
1998	77122	-10	1977122
1998	89315	-1	1989315
1998	77122	-1	1977122
1998	89315	0	1989315
1998	77122	0	1977122
1998	89315	85	1989315
1998	77315	85	2077315
1998	89315	-120	
1998	890315	120	

Figure 10-11 Intrinsic Function: DATE-TO-YYYYDD

Notes:

Intrinsic Function: YEAR-TO-YYYY

- **Syntax**

FUNCTION YEAR-TO-YYYY (YY[SW])

- **Examples:**

MOVE FUNCTION YEAR-TO-YYYY(START-YEAR) TO OUT-START-YEAR

IF FUNCTION DATE-TO-YYYYMMDD(B-DATE-20) > QUERY-DATE THEN ….

- **VALUE EXAMPLES:**

RUN-TIME YEAR	INPUT VALUE	SW Argument	FUNCTION VALUE
1998	89	-10	1889
1998	77	-10	1977
1998	89	-1	1989
1998	77	-1	1977
1998	89	0	1989
1998	77	0	1977
1998	89	85	1989
1998	77	85	2077
1998	89	-120	
1998	89	120	

Figure 10-12 Intrinsic Function: YEAR-TO-YYYY

Notes:

COBOL Intrinsic Functions- Nesting functions

```
COMPUTE NEW-DUE-DATE =
        FUNCTION DATE-OF-INTEGER(
        FUNCTION INTEGER-OF-DATE(DATE-OF-ORDER) + 30)
```

IF DATE-OF-ORDER IS 19960317 then
FUNCTION INTEGER-OF-DATE(DATE-OF-ORDER) +30 is 14337+30 = 144367
FUCNTION DATE-OF-INTEGER(144367) gives NEW-DUE-DATE of 19960416

After Converting a Gregorian due to an integer date, and adding 30 days to the integer date, the newly calculated Gregorian date is displayed

Figure 10-13 COBOL Intrinsic Function: Nesting Functions

Notes:

Length Intrinsic Functions

MOVE IN-REC(1:FUNCTION LENGTH(OUT-AREA)) TO OUT-AREA
* If the length of OUT-AREA is 10, only positions 1 to 10 of IN-REC are move to OUT-AREA

Returns a nine-digit integer specifying the number of bytes the argument takes in storage.

- **The LENGTH OF special register and LENGTH intrinsic function work similarly. LENGTH intrinsic function is more robust because it can have a literal operand and it works with null-terminated strings.**
- **The syntax is different.**

Figure 10-14 Length Intrinsic Function

Notes:

The length intrinsic function takes a single argument (a non-numeric literal, a data element, a structure, or an array) and returns a nine-digit integer specifying the numbe of bytes the argument takes in storage.

Move in-rec(1:function length (out-area)) to out-area.

LOWER-CASE and UPPER-CASE Intrinsic Functions

MOVE FUNCTION UPPER-CASE(ANSWER) TO UPPER-ANSWER
* IF ANSWER contains ‘y’ then UPPER-ANSWER contains ‘Y’

Returns a string that is either all upper case or lower case.

- **Alphanumeric data items of same length all in required case returned**
- **Very useful in comparing two strings**

Figure 10-15 LOWER-CASE and UPPER-CASE Intrinsic Functions

Notes:

These two intrinsic functions return a character string that contains all lower-case or all upper-case characters

Function lower-case(argument)

Function upper-case(argument)

The argument is an alphanumeric data item

The function reference is a string of the same length as the argument but with all letters forced to lower-case or upper-case, respectively

REVERSE Intrinsic Functions

```
IF FUNCTIO REVERSE(ASTRING) = ASTRING
        PERFORM FOUND-PALINDROME
```

* If ASTRING contained ‘OTTO’ the paragraph FOUND-PALINDROME would be performed

Returns a string containing the characters of the argument in reverse order

- **Could be used to look for first non-blank character**
- **In languages that are written from right to left**

Figure 10-16 Reverse Intrinsic Function

Notes:

CHAR and ORD Intrinsic Functions

IF FUNCTION ORD('1') < FUNCTION ORD('A') PERFORM FOUND-ASCII

* If this IF is true then system is running using ASCII character set

CHAR(n) returns the character that is the 'n' the character in coding sequence

ORD(char) returns the position that character belongs in the collating sequence

* **Used where coding scheme, ASCII or EBCDIC are not know until compile time.**

Figure 10-17 CHAR and ORD Intrinsic Functions

Notes:

Arithmetic, Business, and Mathematical Intrinsic Functions

Trigonometric and Logarithmic Intrinsic Functions

Function name	Value returned
ACOS	Arc-cosine of numeric item
ASIN	Arc-sine of numeric item
ATAN	Arc-tangent of numeric item
COS	Cosins of numeric item
LOG	Natural logarithm of numeric item
LOG10	Logarithm to base 10 of numeric item
SIN	Sine of numeric item
TAN	Tangent of numeric item

Figure 10-18 Arithmetic, Business, and Mathematical Functions

Notes:

Arithmetic, Business, and Mathematical Intrinsic Function 2

Statistical and other Mathematical Intrinsic Functions

Function name	Value returned
FACTORIAL	Factorial value of “integer”. Item
INTEGER	Greates integer not greater than “numeric” item
INTEGER-PART	Value of “numeric” item truncated at decimal point
MAX	Largest value in a list of values; all items in the list are of the same type, one of: “alphabetic”, “integer”, “numeric”, or “alphanumeric”
MEAN	Arithmetic mean of list of “numeric” items
MEDIAN	Median of list of “numeric” items
MIDRANGE	Mean of the minimum and maximu values in a list of “numeric” items
MIN	Smallest value in a list of values; see MAX

Figure 10-19 Arithmetic, Business, and Mathematical Functions, 2

Notes:

Arithmetic, Business, and Mathematical Intrinsic Function 3

Statistical and other Mathematical Intrinsic Functions, 2

Function name	Value returned
MOD	Modulo value of “integer” item to “integer” base
ORD-MAX	Position of maximum item in a list
ORD-MIN	Position of miniumum item in a list
RANDOM	Random number based on supplied or default “integer” seed number
RANGE	Value of maximum argument minus value of miniumum argument; all arguments either “integer” or all “numeric”
REM	Remainder of dividing one “numeric” item by another “numeric” item
SQRT	Square root of a “numeric” item
SUM	Sum of list of items, all items are “numeric” or all are “integer”

Figure 10-20 Arithmetic, Business, and Mathematical Functions, 3

Notes:

Arithmetic, Business, and Mathematical Intrinsic Function, 4

Conversion Type Intrinsic Functions

Function name	Value returned
NUMVAL	Numeric value of numeric edited item
NUMVAL-C	Numeric value of numeric edited item with currency symbol

Investment / Depreciation Statistical Type Intrinsic Functions

Function name	Value returned
ANUITY	Ratio of annuity paid for “integer” periods at “numeric” interest, on initial investment of 1
PRESENT-VALUE	Present value using “numeric” discount rate for 1 or more periods, the value in each period specified as a “numeric” item
STANDARD-DEVIATION	Standard deviatio of list of “numeric” items
VARIANCE	Variance of list of “numeric” items

Figure 10-21 Arithmetic, Business, and Mathematical Functions, 4

Notes:

MIN and MAX Intrinsic Functions

COMPUTE MAX-HOLD = FUNCTION MAX(EMP1SALES, EMP2SALES, EMP3SALES, EMP4SALES,EMP5SALES)

- MAX-HOLD would contain the highest value of all elements in the list.

The MAX function returns the value of the largest item in a list of items, the MIN function returns the value of the smallest item.

The argument in example are assumed to be numeric so compute had to be used

Figure 10-22 MIN and MAX Intrinsic Functions

Notes:

The ALL Subscript

COMPUTE TOTAL-IN = FUNCTION SUM(STORE-SALES(ALL)

- TOTAL-IN will be equal to the summation of all elements of the STORE-SALES table array

When an intrinsic function may have a variable number of arguments, you may reference a table as one or more of the arguments.

*** If a multidimensional table, ALL may be used in place of one or more of the subscripts**

Figure 10-23 The ALL Subscript

Notes: