Module 5

Temporary Fields

In this module you will learn:

- The difference between the COMPUTE and DEFINE commands
- How to create temporary fields using the COMPUTE command
- How to create temporary fields using the DEFINE command
- How to store defined fields outside of the report request
- Types of expressions

Supporting Documents/Files:

- FOCUS Documentation
- CIRS Data Element Dictionary
- CIRS Web Site (Sample Requests)
- CIRS Common Library

Overview

A variety of temporary fields can be created from existing fields in the database by using the COMPUTE and DEFINE commands. The maximum number of fields, both real and defined, that can be referenced in a report request is 256.

The major difference between COMPUTE and DEFINE is the point of calculation.

- Computed fields are calculated on the results of a SUM, PRINT, or COUNT command after all records have been selected, sorted, and summed.

- Defined fields are calculated after records are selected per the screening criteria, but before the sorts and display commands are applied.

Compute Command

Temporary fields created by the COMPUTE command, are calculated on the **table results** after all records have been selected, sorted, and summed. The syntax is:

```
TABLE FILE filename
display command
COMPUTE newfield [/format] = expression;
END
```

filename	Any database available in CIRS.
display command	Is PRINT, SUM OR COUNT followed by fieldname(s). Also referred to as the verb phrase.
newfield	New name for the field being created. The name can contain up to 12 characters without blanks. Special characters allowed are the dollar sign ($), colon (:), underscore (_) and pound sign (#).
/format	Optional command to specify the format type, length and edit options for the values of the new field.
= expression;	Expression establishing the value of the new field.

- Specify the COMPUTE command after the display command inside the TABLE request.
- Computed fields are only available for the specified report request.
- Optionally, a field can be referred to by its report column position (C1 for the first column, C2 for the second, etc). BY fields are not counted.
- Only 256 fields may be defined for a single file and the total length of all defined fields and real fields cannot exceed 12,288 characters.
- The computed field can be used in subsequent screening statements, but cannot be used as sort fields.

Compute Example

The report example below uses the COMPUTE command to determine the new salary rate for employees after receiving a 3 percent increase.

Report Request:

```
-* EMPLOYEE SALARY INFO
EX AC
TABLE FILE AC
PRINT AC:CLASS AC:BASEPAY
COMPUTE NEWSAL = AC:BASEPAY * 1.03;
BY AC:WNAME
END
```

Report Generated:

```
AC:WNAME            AC:CLASS    AC:BASEPAY      NEWSAL
--------            --------    ----------      ------
LION-JUGUAR, ANN    2355           $800.00      824.00
COUGAR, CHRIS       2358         $6,278.00    6,569.34
NEWT, KATHERINE     2358         $4,110.00    4,233.30
MONKEY, GORDON S    2360         $6,868.00    7,074.04
FROG, NANCY L       3306         $9,375.00    9,656.25
```

Compute Practice

Write a report that determines the years of service an employee has based on their months of employment. Use the COMPUTE command to determine the years.

Report Generated:

```
AC:WNAME                    AC:EMPMTHS     YEARS
--------                    ----------     -----
ANT, SYLVIA                          0       .00
ANTEATER, C DONALD                  38      3.17
CAMEL, TOM  N                      246     20.50
GIRAFFE, MARY L                    282     23.50
GOPHER, GARY G                     270     22.50
```

Report Request:

Define Command

Temporary fields can also be created using the DEFINE command. The calculation is performed on each record retrieved per the screening statements. The result of the expression is treated as though it is a real field stored in the database. The syntax is:

```
DEFINE FILE filename ADD
newfield/format = expression;
END
```

filename	Any database available in CIRS.
newfield	New name for the field being created. The name can contain up to 12 characters without blanks. Special characters allowed are the dollar sign ($), colon (:), underscore (_) and pound sign (#).
/format	The type, length and edit options for the new field values. Valid types are A (alphanumeric), D (floating-point double-precision), F (floating-point single-precision), I (integer), P (packed decimal), and D, W, M, Q, or Y used in a valid combination (date). The length of the format cannot exceed 8 characters.
= expression;	Expression establishing the value of the new fieldname.
END	Required to end the DEFINE.

- Specify the DEFINE command after executing the standard defines for the file and before beginning the TABLE request.
- ADD must be part of the syntax to add the new FIELDNAME(s) to the automated DEFINES for the database.
- Once a new FIELDNAME is defined it is available for the entire FOCUS session and can be used in verb phrases, sort phrases, screening statements or in another define expression.
- Only 256 fields may be defined for a single file and the total length of all defined fields and real fields cannot exceed 12,288 characters.
- Alpha (A) values and Smart Dates must be enclosed in single quotes.

Storing Defines

Defines can be incorporated into the report request or stored by themselves. Defines are often stored by themselves to simplify maintenance and to share with other users.

- If included in the reports, defines are written after executing the standard defines for the file and before the table request. For example:

  REPORT REQUEST: AGERPT

```
-* COUNT BY AGE GROUP
EX AC
DEFINE FILE AC ADD
GROUP/A5 = IF AC:AGE GE 55 THEN 'YES' ELSE 'NO';
END
-*
TABLE FILE AC
COUNT AC:SSA
BY GROUP
END
```

- If stored by itself, it can be called into a report using the –INCLUDE command after the standard defines are executed. For example:

  REPORT REQUEST: AGETEST

```
DEFINE FILE AC ADD
GROUP/A5 = IF AC:AGE GE 55 THEN 'YES' ELSE 'NO';
END
```

  REPORT REQUEST: AGERPT

```
-* COUNT BY AGE GROUP
EX AC
-*
-INCLUDE AGETEST
-*
TABLE FILE AC
COUNT AC:SSA
BY GROUP
END
```

Types of Expressions

Fields used in an expression can be real data fields, or a field created in a previous DEFINE expression. The expression is terminated with a semi-colon. An expression can be one of the following types:

Date	A date expression manipulates the display, and returns a date or an integer that represents the number of days, months, quarters, or years between two dates.
Numeric	A numeric expression returns a numeric value.
Alphanumeric	An alphanumeric expression returns an alphanumeric value.
Conditional	A conditional expression (IF ... THEN ... ELSE) returns a numeric or alphanumeric value.

Date Format

Most dates in CIRS are defined as Smart Dates. A smart date is internally stored as the number of days lapsed since December 31, 1900. Dates in this format can easily be manipulated. For instance, you can extract date components such as year, quarter, month, and day from existing dates; you can rearrange the order of the date components, and you can refer to dates in a natural way, such as JAN 1 60.

Syntax: newfield/dateformat = date;

Example: effdate/mtdy = ac:effdate;

Date Format Display Options

The date format does not specify type or length; instead, it specifies date component display options. Some of these options are shown below. Refer to your FOCUS documentation for additional display formats. Note: FOCUS does not support formats of month and/or day without year.

Display	**Meaning**	**Effect**
D	Day	Prints a value from 1 to 31 for the day.
M	Month	Prints a value from 1 to 12 for the month.
Y	Year	Prints a two-digit year.
YY	Four-digit year	Prints a four-digit year.
T	Translate month	Prints a three-letter abbreviation for month if "M" is included in the FORMAT specification.
TR	Translate month	Functions the same as T (described above), except that the entire month name is printed instead of an abbreviation.
Q	Quarter	Prints the quarter (1 - 4 if Q is specified by itself, or Q1 - Q4 if it is specified together with other date format items such as Y).
JUL	Julian format	Prints date in Julian format.
YYJUL	Julian format	Prints a Julian format date in the format YYYYDDD.

Date Format Practice

Write defines using the date expression for the following criteria. For each exercise use the field AC:BRTHDATE.

Rearrange the order of the date components to display as year, month and day.

Extract the year component of the date field.

Display the date in a natural way, such as JUNE 1 1960.

Numeric Expression

A numeric expression returns a numeric value. This type of expression is used to perform arithmetic calculations on existing field values, or to arithmetically join values from two or more fields.

Syntax: **newfield/format=numeric expression;**

Example: ANNSAL/P12.2M = XX:SALARY * 12;

- Can only be used with numeric values.
- If the resulting value is too large or small for the format assigned, asterisks (*****) will appear.
- Available arithmetic operators:
  Addition (+)
  Subtraction (-)
  Multiplication (*)
  Division (/)
- Arithmetic operations are executed in the following order:

First:	( ) Parentheses
Second:	* / Multiplication and division
Third:	+ - Addition and subtraction
Fourth:	Left to right

Numeric Edit Options

Edit options can be used to display numeric formats in various ways. They are for display purposes only and do not affect how the data is stored. The most frequently used options are displayed below. Refer to your FOCUS documentation for more options.

Edit Option	**Meaning**	**Effect**
B	Bracket negative	Encloses negative numbers in parentheses.
C	Comma edit	Inserts a comma after every third significant digit, or a period instead of a comma if continental decimal notation is in use.
L	Leading zeroes	Adds leading zeroes.
M	Floating $	Places a floating dollar sign ($) to the left of the highest significant digit.
N	Fixed $	Places a dollar sign ($) to the left of the field.
R	Credit (CR) negative	Places CR after negative numbers.
S	Zero suppress	If the data value is zero, prints a blank in its place.

Numeric Expression Practice

Write defines using the arithmetic expression for the following:

Amount of a 3% Performance Based Salary Increase.

Time and One-Half Overtime Rate (divide base pay by 173.333 to get hourly rate). Display the amount with a floating dollar sign.

Years of employee service based on Employment Date.

Alphanumeric Expression

An alphanumeric expression returns an alphanumeric value. Use an alphanumeric expression to manipulate alphanumeric constants or fields. An alphanumeric expression can be created in various ways:

- An alphanumeric constant (character string) enclosed in single quotation marks:

  Syntax: newfield/format = ‘character string’;

  Example: TEXT/A10 = ‘COMMENTS’;

- Two or more alphanumeric fields or constants joined by the concatenation bar.

  Syntax: newfield/format=fieldname | fieldname;

  Example: XX:AGYUNIT/A6 = XX:AGENCY|XX:UNIT;

  A single concatenating bar (|) is used for weak concatenation. Field lengths are preserved, including trailing blanks.
  Double concatenating bars (| |) are used for strong concatenation. Trailing blanks are suppressed.

- Extracting characters from existing alphanumeric values.

  Syntax: newfield/format=EDIT (fieldname, 'mask');

  Example: REP/A1 = EDIT (XX:CBID, ‘9$$);

  For the mask, use a ‘$’ to ignore a corresponding character, and a ‘9’ to select a corresponding character. Any other character used will be inserted into the values.

- To change alphanumeric fields to numeric fields and vice versa.

  Syntax: newfield/format=EDIT (fieldname);

  Example: XX:NUMER/I3 = EDIT (XX:NUM);

Alphanumeric Expression Practice

For each of the exercises below, use the Active Current Status (AC) file.

Create a new field named SIGNATURE that displays as a dashed line.

Create a new field named ADDRESS that consists of an employee’s City, State and Zip Code.

Create a new field named AREA that selects the first 2 characters of an employee’s Work Location.

Conditional Expression

A conditional expression (IF ... THEN ... ELSE) returns a numeric or alphanumeric value. This expression is used to create a new value to be generated using specific criteria. It allows you to compare fieldnames and values, create "OR" screening conditions, create test conditions, and create new values from existing values.

Syntax for a single IF statement:

```
newfield/format=
    IF (fieldname logical operand value/fieldname)
    THEN value/fieldname
    ELSE value/fieldname;
```

Syntax for multiple IF statements:

```
newfield/format=
    IF (fieldname logical operand value/fieldname)
    THEN value/fieldname
    ELSE
        IF (fieldname logical value/fieldname)
        THEN value/fieldname
        ELSE value/filename;
```

- Each expression must have an IF, THEN and ELSE. The expressions following THEN and ELSE must result in a format that is compatible with the assigned format.
- Conditional expressions can have up to 16 IF phrases.
- Parentheses are used for clarification in compound IFs. Place the parentheses before the fieldname and after the values for that fieldname. Expressions in parentheses are evaluated before any other expression.
- The commands, AND/OR are used to connect values/fieldnames. Double parentheses may be required when AND and OR are both used in one expression.
- Operations are processed after any arithmetic operations in the following order:

  First: EQ NE LE LT GE GT
  Second: AND
  Third: OR

Conditional Expression Practice

For each of the exercises below, use the fields from the Active Current Status (AC) file.

Create a new field named **GROUP,** whose values are **FULLTIME** and **PARTTIME**. Evaluate the decimal time base (full time equivalent) of a record to determine if the position is full-time or part-time.

Create a new field named **FACULTY,** whose values are **YES** and **NO.** A faculty position is identified as any record where the collective bargaining id is R03, or the class code is 2353 or 2354 or 2363.

Create a new field named **GROUP,** whose values are **FACULTY, STAFF** and **MGMT.** Identify faculty positions per the criteria in the previous exercise. Identify Management positions as any record with a CBID value of M80 or M98. Identify all other CBID's as staff.

Retrieving Preceding Values

When using conditional expressions, you can use the LAST function to retrieve a value from the preceding record or line. The syntax is:

LAST fieldname

The effect of the keyword LAST depends on whether it appears in a DEFINE or COMPUTE.

- In a DEFINE, the LAST value is that of the previous record retrieved from the file before sorting takes place.
- In a COMPUTE, the LAST value is that of the record in the previous line in the report.

The following example produces a running total (RUN_TOT) of the salary field within departments. It uses LAST to determine whether the previously retrieved value of department equals the current value. If the values are equal, salary is added to RUN_TOT. If the values are different, RUN_TOT starts with the value of the first salary in the new department.

Report Request

```
-* TOTAL SALARY BY DEPT
EX AC
TABLE FILE AC
PRINT AC:SALARY AND COMPUTE
RUN_TOT/P12.2M =
   IF DEPTNAME EQ LAST DEPTNAME
   THEN (RUN_TOT + AC:SALARY)
   ELSE AC:SALARY;
BY DEPTNAME SKIP-LINE
BY AC:LASTNAME
END
```

Report Generated

```
DEPTNAME  LASTNAME          AC:SALARY         RUN_TOT
--------  ---------         ---------         -------
ENGLISH   GRASSHOPPER       $2,401.20       $2,401.20
          JAGUAR            $2,498.70       $4,896.90

MUSIC     DINGO               $643.03         $643.03
          LION                $216.87         $859.90
          SNAKE             $3,822.00       $4,681.90
```

Practice

Choose at least 3 temporary fields used as examples or created for practice in this module and write a report request using those fields.

- Be sure to test conditional expressions before using in the report request.
- If using defines to create temporary fields, you can choose to include the code in your report request or to store them separately.

What You Have Learned

In this module you have learned:

- The difference between the COMPUTE and DEFINE
- How to create temporary fields using the COMPUTE command
- How to create temporary fields using the DEFINE command
- How to store defined fields outside of the report request
- Types of expressions you can write