CA-Easytrieve to Data-Miner: Compare, Convert, Replace

Data-Miner

Version 7 Release 1

Comprehensive Information and Data-Management System

CSI
INTERNATIONAL

CA-EASYTRIEVE TO DATA-MINER: COMPARE, CONVERT, REPLACE
Version 7 Release 1C
June 2011

Published by Connectivity Systems, Inc.
8120 State Route 138, Williamsport OH 43164
Phone: 800-795-4914
Fax: 740-986-6022

E-Mail: *info@csi-international.com*

Internet: *http://www.csi-international.com*

Documentation comments: *documentation@csi-international.com*

Table of Contents

CA-Easytrieve to Data-Miner: Compare, Convert, Replace

CA-Easytrieve to Data-Miner: Compare, Convert, Replace

Introduction

Replacing CA-Easytrieve with DATA-MINER can be as simple as following the few steps listed below:

1. Install the batch component of DATA-MINER. Refer to either the z/OS or the z/VSE installation guide.

   At the time of this document's publication, the installation guides were at these locations:

   - For z/OS:
     http://www.csi-international.com/support/doc/zOS/Data-Miner/7.1/Data-Miner_for_z-OS_7.1x_Installation_20110331.pdf

   - For z/VSE:
     http://www.csi-international.com/support/doc/zVSE/Data-Miner/7.1/Data-Miner_for_z-VSE_7.1x_Installation_20110331.pdf

2. Choose an existing CA-Easytrieve program and make the following changes.
   - Change EZTPA00 to CSIDMBMD on your EXEC statement.
   - Change your STEPLIB or LIBDEF to reference the DATA-MINER install load library.
3. Make any other changes necessary. There may be no changes required. Refer to the list of differences between CA-Easytrieve and DATA-MINER below.
4. Execute the job.

It is that easy to replace CA-Easytrieve with DATA-MINER for all your data-mining needs.

Many documented DATA-MINER control commands are common to both DATA-MINER and CA-Easytrieve. There are also some CA-Easytrieve commands that are not common with the documented DATA-MINER commands. Refer to the *DATA-MINER User Reference Guide* to discover just how many commands are common between both products. At the time of this document's publication, the product pages and the user guides were at these locations:

For z/OS:

- Product information page
  http://www.csi-international.com/products/zOS/Data-Miner/Data-Miner.htm

- *User Reference Guide*
  http://www.csi-international.com/support/doc/zOS/Data-Miner/7.1/Data-Miner_for_z-OS_7.1C_User_Ref_20110331.pdf

For z/VSE:

- Product information page
  http://www.csi-international.com/products/zVSE/Data-Miner/Data-Miner.htm

- *User Reference Guide*
  http://www.csi-international.com/support/doc/zVSE/Data-Miner/7.1/Data-Miner_for_z-VSE_7.1C_User_Ref_20110331.pdf

Generally speaking, DATA-MINER can process existing CA-Easytrieve programs with few or no alterations. Input/Output files and the corresponding record fieldnames can be defined the same way. Program flow is processed with similar or identical command syntax. DATA-MINER supports all of CA-Easytrieve's conditional operators and more. The powerful DATA-MINER report writer handles defining and generating reports the same as CA-Easytrieve.

Same Program/Script Structure

Introduction

What CA-Easytrieve calls a program, DATA-MINER calls a script. The structure of a DATA-MINER script is the same as a CA-Easytrieve program. CA-Easytrieve calls the three sections

- Environment
- Library, and
- Activity

Let's look at all three, and again you will see how similar DATA-MINER and CA-Easytrieve truly are.

Environment Section

The PARM statement is not supported directly by DATA-MINER. Each execution of the DATA-MINER job parses, compiles (in storage), and executes the script. There is no mechanism to compile a DATA-MINER script for later execution. The DEBUG function is accomplished with the DATA-MINER TRACE command.

Library Section

The FILE Statement. DATA-MINER uses the INPUT and OUTPUT commands to define files to the script; however, the CA-Easytrieve FILE command syntax is supported, excluding the IS, VIRTUAL, DLI, and PASSWORD operands. The RECORD-LENGTH, RECORD-COUNT, and FILE-STATUS system variables you are familiar with are supported for each file. The TABLE operand is also supported along with DATA-MINER's native TABLE command.

The DEFINE Statement. DATA-MINER uses the DEFINE command the same as CA-Easytrieve. DATA-MINER fieldnames can be 32 characters long and must not start with a number, whereas CA-Easytrieve fieldnames can be 40 characters long and can start with a number. That is the only difference in field definition. Fields can be redefined in the same way CA-Easytrieve does, by referencing a previous fieldname and optionally adding an offset. DATA-MINER detects the subtle difference between record fields and user variables (work fields) but also allows for the CA-Easytrieve W|S location indicators. All location parameters and more are supported. The attribute parameters A (alphabetic), N (zoned-decimal), P (packed), B (binary), and more are supported. Optional parameters MASK, VALUE, and HEADING are supported.

Activity Section

What CA-Easytrieve calls a statement, DATA-MINER calls a command. DATA-MINER commands are free flowing and do not require a continuation character when they extend onto more than one input card.

DATA-MINER supports the CA-Easytrieve statement structure and syntax rules. Terminating statements with a period, statement continuation and commenting conventions are all supported. Literals and numerics are specified in the same fashion.

CA-Easytrieve has JOB and SORT activities. Any number of either is allowed in a CA-Easytrieve program. DATA-MINER supports the JOB and SORT

statements with one difference: only one JOB and/or SORT statement can be coded in any single DATA-MINER script. If more or both are needed, each must run in a separate jobstep. If multiple input files must be processed in the same script, you must use task-control I/O (using READ and WRITE).

The CA-Easytrieve JOB statement is equivalent to the DATA-MINER AUTO command. Each causes the program/script to function in an automatic read mode. Scripts can alternatively process the input file(s) themselves with the READ command. As stated, DATA-MINER supports one JOB statement per script, excluding the KEY parameter.

DATA-MINER has its own SORT command syntax but also supports the CA-Easytrieve SORT statement syntax. The SORT command must run in its own script (in a separate jobstep). CA-Easytrieve uses the BEFORE *procname* operand to screen and/or modify input records to the SORT. DATA-MINER does not support this operand. It does, however, provide several timing commands as a method to accomplish this.

Below is a sample DATA-MINER script that looks and performs very similarly to a CA-Easytrieve program. The only DATA-MINER–specific differences are that the two report subcommands SUM and AVG were added to calculate totals and averages. By default, CA-Easytrieve calculates and prints totals and requires a report procedure to calculate the averages.

This script uses some of the more common CA-Easytrieve statements, and they show how closely DATA-MINER's and CA-Easytrieve's syntaxes match.

```
    CONNECTIVITY SYSTEMS INC.
DATAMINER INPUT SCRIPT REPORT - VERSION 7.1C
=============================================

    1
    2 *
    3 * DEFINE THE INPUT EMPLOYEE VSAM FILE AND CORRESPONDING RECORD FIELDS
    4 *
    5
    6 FILE EMPVSAM VS
    7   EMPNO         1  6  C
    8   FIRSTNME      9 10  A
    9   MIDINIT      19  1  A
   10   LASTNAME     22 13  A
   11   WORKDEPT     35  3  C HEADING('DEPT')
   12   PHONENO      38  4  N
   13   HIREDATE     42 10  C
   14     HIRE-YEAR  HIREDATE   4  N
   15     HIRE-MONTH HIREDATE+5 2  N
   16     HIRE-DAY   HIREDATE+8 2  N
   17   JOBTITLE     52  8  C
   18   SALARY       73  5  P 2
   19   BONUS        78  5  P 2
   20   COMM         83  5  P 2
   21
   22 *
   23 * DEFINE WORK FIELDS USED IN THE SCRIPT
   24 *
   25
   26   YEARS-WORKED           W   2  P
   27   RAISE                  W   2  P
   28   CURR-DATE              S  12  C
   29     CURR-MONTH  CURR-DATE    2  N
```

```
30     CURR-DAY    CURR-DATE+3  2  N
31     CURR-YEAR   CURR-DATE+6  4  N
32
33 *
34 * DEFINE OUTPUT LOGFILE
35 *
36
37 FILE LOGFILE PRINTER
38
39 *
40 * THIS SCRIPT AUTO READS THE EMPLOYEE FILE AND CALCULATES RAISES
41 *
42
43 JOB INPUT EMPVSAM
44
45  IF WORKDEPT = 'E11'
46    DISPLAY LOGFILE 'NO RAISE FOR' FIRSTNME LASTNAME
47    GO TO JOB
48  END-IF
49
50  PERFORM SET-RAISE
51
52  CURR-DATE = SYSDATE-LONG.              * GET CURRENT DATE (MM/DD/YYYY)
53  YEARS-WORKED = CURR-YEAR - HIRE-YEAR.  * CALC NUMBER OF YEARS WORKED
54
55  IF YEARS-WORKED > 50
56    RAISE = RAISE + 5
57  END-IF
58  DISPLAY LOGFILE 'RAISE FOR' FIRSTNME LASTNAME 'IS' +1 RAISE +0 '%'
59
60  PRINT DETAIL-REPORT
61  PRINT SUMMARY-REPORT
62
63 *
64 * SET RAISE PERCENTAGE
65 *
66
67 PROC SET-RAISE
68   IF JOBTITLE = 'MANAGER'
69     RAISE = 15
70     GOBACK
71   END-IF
72   IF JOBTITLE = 'DESIGNER'
73     RAISE = 10
74     GOBACK
75   END-IF
76   RAISE = 5
77 END-PROC.
78
79 *
80 * DEFINE EMPLOYEE DETAIL REPORT
81 *
82
83 REPORT DETAIL-REPORT
84 SEQUENCE WORKDEPT LASTNAME
85 CONTROL WORKDEPT
86 TITLE 1 'DETAILED EMPLOYEE REPORT'
87 LINE WORKDEPT LASTNAME SALARY BONUS COMM YEARS-WORKED RAISE
88 HEADING LASTNAME 'NAME'
89 HEADING YEARS-WORKED 'YRS'
90 SUM SALARY BONUS COMM
91 AVG YEARS-WORKED RAISE
92
93 *
94 * DEFINE EMPLOYEE SUMMARY REPORT
```

```
 95 *
 96
 97 REPORT SUMMARY-REPORT SUMMARY
 98 SEQUENCE WORKDEPT
 99 CONTROL WORKDEPT
100 TITLE 1 'SUMMARY DEPARTMENT REPORT'
101 LINE WORKDEPT TALLY SALARY BONUS COMM YEARS-WORKED RAISE
102 HEADING WORKDEPT 'DEPT' 'CODE'
103 HEADING TALLY    'EMPLOYEE' 'COUNT'
104 HEADING YEARS-WORKED 'AVG' 'YRS'
105 HEADING RAISE 'AVG' 'RAISE'
106 SUM SALARY BONUS COMM
107 AVG YEARS-WORKED RAISE
```

The script above reads an employee VSAM dataset named EMPVSAM, which contains various employee-related fields. Record fields and work fields are defined. Raises are calculated for each employee, and a logfile is written out. A detailed report and a summary report are also created.

Here is the first resultant report from the above script.

```
05/19/2011                DETAILED EMPLOYEE REPORT                PAGE     1

DEPT       NAME          SALARY        BONUS        COMM       YRS   RAISE
A00    HAAS            305634.48      9102.00      4220.00     46      5
       LUCCHESI        269424.05      4202.00      3720.00     53     10
       O'CONNELL       169515.15      3902.00      2340.00     48      5
====   =============   ===========   ===========   ===========  ====   ====
A00                    744573.68     17206.00     10280.00     49      6

B01    THOMPSON         43417.50      4102.00      3300.00     38     15
====   =============   ===========   ===========   ===========  ====   ====
B01                     43417.50      4102.00      3300.00     38     15

C01    KWAN             40267.50      4102.00      3060.00     36     15
       NICHOLLS         29946.00      3902.00      2274.00     35      5
       QUINTANA         25095.00      3802.00      1904.00     40      5
====   =============   ===========   ===========   ===========  ====   ====
C01                     95308.50     11806.00      7238.00     37      8

D11    ADAMSON          47857.78      3802.00      2022.00     39     10
       BROWN            52496.45      3902.00      2217.00     45     10
       JONES            34639.40      3702.00      1462.00     32     10
       LUTZ             56456.32      3902.00      2387.00     43     10
       PIANKA           42144.29      3702.00      1780.00     34     10
       SCOUTTEN         40428.29      3802.00      1707.00     38     10
       STERN            61000.78      3902.00      2580.00     38     15
       WALKER           38750.10      3702.00      1636.00     37     10
       YOSHIMURA        46726.40      3802.00      1974.00     33     10
====   =============   ===========   ===========   ===========  ====   ====
D11                    420499.81     34218.00     17765.00     37     10

D21    JEFFERSON        88544.85      3702.00      1774.00     45      5
       JOHNSON           6983.12      3602.00      1380.00     36      5
       MARINO          114781.75      3902.00      2301.00     32      5
       PEREZ           109279.19      3802.00      2190.00     31      5
       PULASKI         144328.18      4002.00      2893.00     31     15
       SMITH            76582.75      8702.00      1534.00     42      5
====   =============   ===========   ===========   ===========  ====   ====
D21                    540499.84     27712.00     12072.00     36      6

E01    GEYER            42288.75      4102.00      3214.00     62     20
====   =============   ===========   ===========   ===========  ====   ====
E01                     42288.75      4102.00      3214.00     62     20
```

E21	GOUNOT	25137.00	3802.00	1907.00	64	10
	LEE	26743.50	3802.00	2030.00	35	5
	MEHTA	21052.50	3702.00	1596.00	46	5
	SPENSER	27562.50	3802.00	2092.00	31	15
E21		100495.50	15108.00	7625.00	44	8
		1987083.58	114254.00	61494.00	40	9

Here is the resulting summary report:

05/19/2011 SUMMARY DEPARTMENT REPORT PAGE 1

DEPT CODE	EMPLOYEE COUNT	SALARY	BONUS	COMM	AVG YRS	AVG RAISE
A00	3	744573.68	17206.00	10280.00	49	6
B01	1	43417.50	4102.00	3300.00	38	15
C01	3	95308.50	11806.00	7238.00	37	8
D11	9	420499.81	34218.00	17765.00	37	10
D21	6	540499.84	27712.00	12072.00	36	6
E01	1	42288.75	4102.00	3214.00	62	20
E21	4	100495.50	15108.00	7625.00	44	8
	27	1987083.58	114254.00	61494.00	40	9

Here is the resulting logfile:

```
RAISE FOR    CHRISTINE     HAAS              IS   5 %
RAISE FOR    MICHAEL       THOMPSON          IS  15 %
RAISE FOR    SALLY         KWAN              IS  15 %
RAISE FOR    JOHN          GEYER             IS  20 %
RAISE FOR    IRVING        STERN             IS  15 %
RAISE FOR    EVA           PULASKI           IS  15 %
NO RAISE FOR    EILEEN        HENDERSON
RAISE FOR    THEODORE      SPENSER           IS  15 %
RAISE FOR    VINCENZO      LUCCHESI          IS  10 %
RAISE FOR    SEAN          O'CONNELL         IS   5 %
RAISE FOR    DOLORES       QUINTANA          IS   5 %
RAISE FOR    HEATHER       NICHOLLS          IS   5 %
RAISE FOR    BRUCE         ADAMSON           IS  10 %
RAISE FOR    ELIZABETH     PIANKA            IS  10 %
RAISE FOR    MASATOSHI     YOSHIMURA         IS  10 %
RAISE FOR    MARILYN       SCOUTTEN          IS  10 %
RAISE FOR    JAMES         WALKER            IS  10 %
RAISE FOR    DAVID         BROWN             IS  10 %
RAISE FOR    WILLIAM       JONES             IS  10 %
RAISE FOR    JENNIFER      LUTZ              IS  10 %
RAISE FOR    JAMES         JEFFERSON         IS   5 %
RAISE FOR    SALVATORE     MARINO            IS   5 %
RAISE FOR    DANIEL        SMITH             IS   5 %
RAISE FOR    SYBIL         JOHNSON           IS   5 %
RAISE FOR    MARIA         PEREZ             IS   5 %
```

```
NO RAISE FOR    ETHEL         SCHNEIDER
NO RAISE FOR    JOHN          PARKER
NO RAISE FOR    PHILIP        SMITH
NO RAISE FOR    MAUDE         SETRIGHT
RAISE FOR    RAMLAL        MEHTA           IS   5 %
RAISE FOR    WING          LEE             IS   5 %
RAISE FOR    JASON         GOUNOT            IS  10 %
```

CA-Easytrieve and DATA-MINER Have Many Keywords/Functions in Common

Review the list below to see a sample of functions/keywords common to both DATA-MINER and CA-Easytrieve. Many of the commands are discussed further in this document. For more detail, refer to the *User Reference Guide*.

CALL	Call an external load module.
DEFINE	Define record fields and user variables.
DISPLAY	Write data to system printer or a specified file.
DO WHILE	Handle loop processing.
ELSE	Control conditional flow.
END-IF	Control conditional flow.
END-DO	Control conditional flow.
GET	Read the next sequential record.
GOTO	Allow the execution to branch to a label.
IF	Control conditional flow.
PARM-DATA	Pass data into DATA-MINER via the EXECUTE statement PARM operand.
PERFORM	Call a procedure somewhere else in the script.
POINT	Position a VSAM file.
PRINT	Used in conjunction with REPORT command to create reports.
PUT	Write a record to a sequential or VSAM file.
READ	Read a record from a sequential or VSAM file.
REPORT	Define a report (used in conjunction with PRINT)
SET	Assign a record field or user variable a value.
SORT	Sort an input file into an output file.
STOP	Stop execution of script.
WRITE	Write a record to a sequential or VSAM file.

Sorting Files

As stated previously, DATA-MINER has its own SORT command syntax; however, the example below shows how you could use DATA-MINER to sort a file using CA-Easytrieve syntax. The VSAM input EMPVSAM employee file is sorted into the sequential output EMPSORT file. The records are sorted into ascending order by salary. This example works for either DATA-MINER or CA-Easytrieve.

```
FILE EMPVSAM VS
 SALARY       73  5  P 2
FILE EMPSORT FB(100 1000)
SORT EMPVSAM TO EMPSORT USING SALARY
```

A second example shows how to instruct DATA-MINER to select only certain records to include in the output SORT file. In this example, only employees with a salary greater than or equal to 50,000 are included into the SORT output file. This differs from the CA-Easytrieve method of using the "BEFORE *procname*" operands to call a procedure to include only certain records in the output sort file.

```
SORT EMPVSAM TO EMPSORT USING SALARY
DURING INPUT
 SKIP SALARY < 50000
```

Data Manipulation

Field Assignment

DATA-MINER and CA-Easytrieve assign values to fields in the same way. The DATA-MINER [SET] command is equivalent to the CA-Easytrieve assignment statement. For example, each allows and processes the following statements in the same fashion:

```
SALARY = 50000 /* Set salary to 50000 */
JOBTITLE = 'MANAGER' /* Set job title to MANAGER */
```

Arithmetic Expression

DATA-MINER and CA-Easytrieve handle arithmetic expressions in the same way. For example, each allows and processes the following statements in the same fashion:

```
* Calculate total compensation
TOT-COMPENSATION = SALARY + BONUS + COMMISSION

* Increase compensation by 10%
NEW-COMPENSATION = TOT-COMPENSATION * 1.10
```

Decision and Branching Logic

IF, ELSE, and END-IF Statements

The IF, ELSE, and END-IF statements are supported same as in CA-Easytrieve, including multiple conditions using AND or OR connectors and optional parentheses. DATA-MINER supports all relational conditions you are familiar with in CA-Easytrieve, including EQ, =, NE, !=, LT, <, LE, <=, GT, >, GE, and >=.

```
IF SALARY < 50000
  BONUS = 5000
ELSE
  BONUS = 10000
END-IF
```

DO and END-DO Statements

The DO and END-DO statements are supported. The example below produces the same output for either DATA-MINER or CA-Easytrieve. The employee file is read until EOF is detected, writing out a few details about each employee. Note that DATA-MINER will handle the JOB and STOP statements, but in this example does not require them.

```
JOB INPUT NULL

 GET EMPVSAM
 DO WHILE (FILE-STATUS != 'EOF')
    DISPLAY EMPNO WORKDEPT LASTNAME SALARY
   GET EMPVSAM
 END-DO
 STOP
```

GOTO Statement and Labels

The GOTO statement and corresponding labels are supported. The example below produces the same results as the previous example, but this time using GOTO and a label.

```
GET EMPVSAM
 READLOOP
  IF (FILE-STATUS != 'EOF')
    DISPLAY EMPNO WORKDEPT LASTNAME SALARY
    GET EMPVSAM
    GOTO READLOOP
  END-IF
```

Procedures

DATA-MINER procedures are defined and called identically to CA-Easytrieve. The procedures are defined with the PROC and END-PROC commands and called with the PERFORM command. DATA-MINER does not currently support REPORT procedures.

Input and Output

DATA-MINER provides for both automatic and script-controlled I/O. DATA-MINER short-cut commands place the script into automatic I/O mode where the script itself does not need to read each input record. The DATA-MINER AUTO command also places the script into automatic I/O mode. The simulated JOB statement (excluding the INPUT NULL option) also places a script into automatic I/O mode. The script can optionally perform its own I/O using, for example, the READ and WRITE commands.

TABLE Processing

DATA-MINER has its own TABLE definition command and corresponding FIND command, but DATA-MINER supports the CA-Easytrieve method of defining a table with the FILE statement and accessing the table through the SEARCH statement. DATA-MINER supports loading a table either from a file or directly in the script input cards.

Simple Reports

The DISPLAY command works identically as in CA-Easytrieve. It is used to write output to the system printer or other sequential file. The HEX operand is not supported; however, DATA-MINER provides the SHOW command to write hexadecimal fields or records.

The PRINT command can be used in conjunction with the SELECT command to create non-sorted reports. The PRINT command is also used in conjunction with the REPORT command to generate more complex reports that can be sorted, summarized, and controlled very similarly to CA-Easytrieve reports.

Report Processing

DATA-MINER handles report processing in the same way that CA-Easytrieve does. Reports are defined with the REPORT command, and almost all common operands and subcommands are supported. The 'PRINT *reportname'* command in conjunction with the 'REPORT *reportname*' command are used in the same fashion. DATA-MINER can also produce non-sorted reports with the PRINT and SELECT commands. Many DATA-MINER operands and subcommands differ in name from CA-Easytrieve, but in most cases the CA-Easytrieve syntax is also supported.

All the following report operands and subcommands for both DATA-MINER and CA-Easytrieve are listed below. For several of these, DATA-MINER has its own syntax, but in most cases the corresponding CA-Easytrieve syntax is also supported. As you can see, DATA-MINER can handle all your existing CA-Easytrieve reporting needs.

For further information, see the chapter “Reports” in the *DATA-MINER User Reference Guide*.

REPORT Operands

SUMMARY	Produce summary report.
LABELS	Produce label format.
ACROSS	Specify number labels to print across page.
WIDTH	Specify number of columns for each label. Easytrieve SIZE operand also supported.
SIZE	Supported Easytrieve synonym for DATA-MINER WIDTH operand.
HEIGHT	Specify number of lines for each label.
DOWN	Supported CA-Easytrieve synonym for DATA-MINER HEIGHT operand.
PRINTER	Optionally specify the printer dataset (default is system printer).
LENGTH	Specify number of lines per page.
PAGESIZE	Supported CA-Easytrieve synonym for DATA-MINER LENGTH operand.
LINESIZE	Specify the minimum width of the report line.
SPREAD	Not supported.
NOSPREAD	Not supported; however, DATA-MINER defaults to this method.
NOADJUST	Not supported; DATA-MINER always left justifies the report.
NOHDRS	Do not print column headers.
NOHEADING	Supported CA-Easytrieve synonym for DATA-MINER NOHDRS operand.
NODATE	Suppress printing of current date in leftmost 10 columns in first title line.

	NOPAGE	Suppress printing of page number in rightmost 10 columns in first title line.
	LIMIT	Testing aid to limit the maximum number of report lines printed.
	EVERY	Testing aid to skip all but every n^{th} PRINT command for this report.
	SPACING	Specify single, double, or triple line spacing (DATA-MINER only).
	GAP	Specify number of blank columns between each field in report (DATA-MINER only).
REPORT Subcommands	ORDER	Specify order in which you want your report sorted. Any number of fields in ascending or descending order is allowed.
	SEQUENCE	Supported CA-Easytrieve synonym for DATA-MINER ORDER subcommand.
	BREAK	Specify the fieldname(s) to use as control breaks in your report.
	CONTROL	Supported CA-Easytrieve synonym for DATA-MINER BREAK subcommand.
	NEWPAGE	Skip to new page on control breaks.
	RENUM	Skip to new page and reset page number to 1 on control breaks.
	NOPRINT	Not supported.
	TITLE	Create up to eight title lines for the report. The syntax is identical to CA-Easytrieve's.
	HEADING	Define up to three alternate column headings for specified fieldname.
	LINE	Define up to 255 lines of data to print for each execution of the PRINT *reportname*. The syntax is identical to CA-Easytrieve.
	SUM	Specify fieldname(s) in which you want summary totals calculated at the break level. CA-Easytrieve by default calculates and displays this for all numeric fields.
	AVG	Specify fieldname(s) in which you want summary averages calculated at the break level. CA-Easytrieve requires you to create a BEFORE-BREAK report procedure to accomplish this.
REPORT System Variable	TALLY	A tally is kept at each break level to count the number of detail lines.

Steps to Generate a Report

The following steps are necessary to create custom reports with DATA-MINER.

1. Determine input required for the report.
2. Define fields used by the report.
3. Instruct the script to read the input file.
4. Perform any record selection.
5. Perform any data manipulation.
6. Write a line to the report

Each step is explained in more detail in the following sections.

Step 1: Determine Input Required for the Report

Input to the report is designated with the INPUT command. One or more INPUT commands are used to tell DATA-MINER which data source to read for the report. You can use the MAX=*numrecs* operand of the INPUT command to limit the input to a small number of records while testing your report.

Are You Replacing CA-Easytrieve with DATA-MINER? DATA-MINER processes CA-Easytrieve's FILE statements as though they were native DATA-MINER INPUT statements.

Step 2: Define Fields Used by the Report

You can define all record fields for each data source (INPUT command) or only the fields in which the report is interested.

Any user variables must be defined as well. User variables are used to contain any information that needs manipulation before writing a line to the report.

Are You Replacing CA-Easytrieve with DATA-MINER? DATA-MINER processes all record fields and user variables defined in CA-Easytrieve as though they had been defined in DATA-MINER.

Step 3: Instruct the Script to Read the Input File

If only one input file is required, you can use the AUTO command to put DATA-MINER into auto-read mode. An alternative method is to manage the reads within your script with the READ command and to construct a loop to perform each read.

Are You Replacing CA-Easytrieve with DATA-MINER? DATA-MINER processes CA-Easytrieve's JOB statements as though they were native DATA-MINER AUTO statements.

DATA-MINER processes its own and CA-Easytrieve's READ statements exactly the same.

Step 4: Perform Any Record Selection

Many methods are provided to limit what data is written to each report. A common method used is to wrap the PRINT command with IF statements to control which records are input to the report.

Are You Replacing CA-Easytrieve with DATA-MINER? DATA-MINER processes its own and CA-Easytrieve's IF statements exactly the same.

Step 5: Perform Any Data Manipulation

Any number of computations can be performed before writing a line to the report. User variables are defined to hold these values and then passed to the report.

Are You Replacing CA-Easytrieve with DATA-MINER? DATA-MINER processes its own and CA-Easytrieve's variables and arithmetic statements exactly the same.

Step 6: Write a Line to the Report

The PRINT command specifies which report to write a line to. All record fieldnames and user variables are available to the report. The PRINT statement writes a line to the report only if the PRINT statement is executed. If it is bypassed because of some selection logic, no write is performed.

Are You Replacing CA-Easytrieve with DATA-MINER? DATA-MINER processes its own and CA-Easytrieve's PRINT statements exactly the same.

Designing a Report

A report can be constructed very simply. It can be designed to print one line for each input record in the order that the record is read. Or it can be sorted and summarized by particular fieldnames. Many reports can be generated from a single run of the script and/or many input files can be read to generate one report.

The examples below start with a simple report and build upon it in subsequent examples. New operands added to the code since the previous code example are double underscored as this sentence is double underscored.

Notice MAX=15 on the INPUT command. These examples limit these reports to only 15 input records. Also the INPUT command and fieldname definitions are not included in all the subsequent examples.

The complete output from each report may not be included in these examples.

Report Example 1

This is a simple AUTO-mode script that writes each record read in from an employee VSAM file to the report. The report is defined with one line for each employee, printed in the order in which each record was read.

```
INPUT=VSAM FILENAME=EMPVSAM MAX=15
EMPNO 1,6,C
LASTNAME 22,13,C
DEPT   35,3,C
SALARY 73,5,P,2
BONUS  78,5,P,2
COMM   83,5,P,2

AUTO EMPVSAM /* This is an auto-read script */
PRINT EMPRPT /* Write line to report        */
* DEFINE THE REPORT
REPORT EMPRPT
LINE 1 DEPT EMPNO LASTNAME SALARY BONUS COMM
```

Here is the resulting report:

DEPT	EMPNO	LASTNAME	SALARY	BONUS	COMM
D11	000170	YOSHIMURA	46726.40	3802.00	1974.00
D11	000160	PIANKA	42144.29	3702.00	1780.00
D11	000150	ADAMSON	47857.78	3802.00	2022.00
C01	000140	NICHOLLS	29946.00	3902.00	2274.00
C01	000130	QUINTANA	25095.00	3802.00	1904.00
A00	000120	O'CONNELL	169515.15	3902.00	2340.00
A00	000110	LUCCHESI	269424.05	4202.00	3720.00
E21	000100	SPENSER	27562.50	3802.00	2092.00
E11	000090	HENDERSON	31342.50	3902.00	2380.00
D21	000070	PULASKI	144328.18	4002.00	2893.00
E01	000050	GEYER	42288.75	4102.00	3214.00
D11	000060	STERN	61000.78	3902.00	2580.00
C01	000030	KWAN	40267.50	4102.00	3060.00
B01	000020	THOMPSON	43417.50	4102.00	3300.00
A00	000010	HAAS	305634.48	9102.00	4220.00

Report Example 2

This script generates the same report as Report Example 1. It uses TASK mode rather than AUTO mode; therefore, the script must create a loop to read the input records.

```
READ EMPVSAM
DO WHILE (EMPVSAM:IO-RESULT = 'OK ')
  PRINT EMPRPT
  READ EMPVSAM
ENDDO
* DEFINE THE REPORT
REPORT EMPRPT
LINE 1 DEPT EMPNO LASTNAME SALARY BONUS COMM
```

The resulting report is the same as in Report Example 1.

Report Example 3

This and the remaining scripts use AUTO mode.

This example includes a title, and it sorts the report by department number. It also includes a summary report.

Report subcommands can be entered in any order in the report.

> **Are You Replacing CA-Easytrieve with DATA-MINER?** DATA-MINER's ORDER subcommand is used in this and the following examples; however, DATA-MINER processes SEQUENCE statements in existing CA-Easytrieve programs as though they were native DATA-MINER ORDER statements.

```
AUTO EMPVSAM /* This is an auto-read script */
PRINT EMPRPT /* Write line to report        */
* DEFINE THE REPORT
REPORT EMPRPT
TITLE 1 'EMPLOYEE LIST'
ORDER DEPT /* Order by department         */
LINE 1 DEPT EMPNO LASTNAME SALARY BONUS COMM
SUM SALARY BONUS COMM /* Create summary totals */
```

Here is the resulting report:

```
05/03/2011                  EMPLOYEE LIST                PAGE     1


 DEPT  EMPNO      LASTNAME        SALARY         BONUS         COMM
 A00   000110   LUCCHESI         269424.05       4202.00       3720.00
 A00   000120   O'CONNELL        169515.15       3902.00       2340.00
 A00   000010   HAAS             305634.48       9102.00       4220.00
 B01   000020   THOMPSON          43417.50       4102.00       3300.00
 C01   000130   QUINTANA          25095.00       3802.00       1904.00
 C01   000140   NICHOLLS          29946.00       3902.00       2274.00
 C01   000030   KWAN              40267.50       4102.00       3060.00
 D11   000170   YOSHIMURA         46726.40       3802.00       1974.00
 D11   000160   PIANKA            42144.29       3702.00       1780.00
 D11   000060   STERN             61000.78       3902.00       2580.00
 D11   000150   ADAMSON           47857.78       3802.00       2022.00
 D21   000070   PULASKI          144328.18       4002.00       2893.00
 E01   000050   GEYER             42288.75       4102.00       3214.00
 E11   000090   HENDERSON         31342.50       3902.00       2380.00
 E21   000100   SPENSER           27562.50       3802.00       2092.00
```

```
===   ======   ============   ==========   ==========   ==========
                              1326550.86     64130.00     39753.00
```

Report Example 4

The addition of the BREAK subcommand in the following example organizes the report by department. Each department now displays a sub-total.

> **Are You Replacing CA-Easytrieve with DATA-MINER?** DATA-MINER's BREAK subcommand is used in this and the following examples; however, DATA-MINER processes CONTROL statements in existing CA-Easytrieve programs as though they were native DATA-MINER BREAK statements.

```
AUTO EMPVSAM * This is an auto-read script */
PRINT EMPRPT * Write line to report        */
* DEFINE THE REPORT
REPORT EMPRPT
TITLE 2 'EMPLOYEE BY DEPARTMENT'
ORDER DEPT /* Order by department         */
BREAK DEPT /* Organize by department      */
LINE 1 DEPT EMPNO LASTNAME SALARY BONUS COMM
SUM SALARY BONUS COMM /* Create summary totals */
```

Here is the resulting report:

```
05/03/2011                  EMPLOYEE LIST                  PAGE    1

DEPT  EMPNO     LASTNAME       SALARY        BONUS         COMM
A00   000110   LUCCHESI        269424.05      4202.00       3720.00
      000120   O'CONNELL       169515.15      3902.00       2340.00
      000010   HAAS            305634.48      9102.00       4220.00
===   ======   =============   ============   ============  ============
A00                            744573.68     17206.00      10280.00

B01   000020   THOMPSON         43417.50      4102.00       3300.00
===   ======   =============   ============   ============  ============
B01                             43417.50      4102.00       3300.00

C01   000130   QUINTANA         25095.00      3802.00       1904.00
      000140   NICHOLLS         29946.00      3902.00       2274.00
      000030   KWAN             40267.50      4102.00       3060.00
===   ======   =============   ============   ============  ============
C01                             95308.50     11806.00       7238.00

D11   000170   YOSHIMURA        46726.40      3802.00       1974.00
      000160   PIANKA           42144.29      3702.00       1780.00
      000060   STERN            61000.78      3902.00       2580.00
      000150   ADAMSON          47857.78      3802.00       2022.00
===   ======   =============   ============   ============  ============
D11                            197729.25     15208.00       8356.00
```

Report Example 5

The addition of the NEWPAGE operand in the following example forces a page break after each department summary is printed. The department is added to a TITLE subcommand, and the employees within each department are now sorted in descending order.

```
AUTO EMPVSAM /* This is an auto-read script */
PRINT EMPRPT /* Write line to report        */
* DEFINE THE REPORT
REPORT EMPRPT
TITLE 1 'EMPLOYEE LIST'
```

```
TITLE 2 'FOR DEPARTMENT' DEPT
ORDER DEPT -SALARY /* Order by dept,then salary   */
BREAK DEPT NEWPAGE /* Organize by department     */
LINE 1 DEPT EMPNO LASTNAME SALARY BONUS COMM
SUM SALARY BONUS COMM /* Create summary totals      */
```

Here is the resulting report:

```
05/02/2011                  EMPLOYEE LIST                  PAGE     1
                       FOR DEPARTMENT   A00

DEPT  EMPNO      LASTNAME        SALARY         BONUS         COMM
A00   000010   HAAS             305634.48       9102.00       4220.00
      000110   LUCCHESI         269424.05       4202.00       3720.00
      000120   O'CONNELL        169515.15       3902.00       2340.00
===   ======   =============   ===========   ===========   ===========
A00                            744573.68      17206.00      10280.00
05/02/2011                  EMPLOYEE LIST                  PAGE     2
                       FOR DEPARTMENT   B01

DEPT  EMPNO      LASTNAME        SALARY         BONUS         COMM
B01   000020   THOMPSON          43417.50       4102.00       3300.00
===   ======   =============   ===========   ===========   ===========
B01                             43417.50       4102.00       3300.00
05/02/2011                  EMPLOYEE LIST                  PAGE     3
                       FOR DEPARTMENT   C01

DEPT  EMPNO      LASTNAME        SALARY         BONUS         COMM
C01   000030   KWAN              40267.50       4102.00       3060.00
      000140   NICHOLLS          29946.00       3902.00       2274.00
      000130   QUINTANA          25095.00       3802.00       1904.00
===   ======   =============   ===========   ===========   ===========
C01                             95308.50      11806.00       7238.00
```

Report Example 6

A user variable is added to calculate total compensation. First TOTAL_COMP is defined. Then within the AUTO-read loop, a total compensation value is calculated by adding salary, bonus, and commisions. This new variable is included in the report.

A second report is also defined and generated. It is a summary report by department and also includes the count of employees in each department.

Here is the first resulting report:

```
DEFINE TOTAL_COMP 5,P,2 HEADING('TOTAL' 'COMPENSATION')  /* define total var */

AUTO EMPVSAM                                 /* This is an auto-read script  */
TOTAL_COMP = SALARY + BONUS + COMM           /* Calc total compensation      */
PRINT EMPRPT                                 /* Write line to detail report  */
PRINT EMPSUM                                 /* Write line to summary report */

* DEFINE THE DETAIL REPORT
REPORT EMPRPT
TITLE 1 'EMPLOYEE LIST'
TITLE 2 'FOR DEPARTMENT' DEPT
ORDER DEPT -SALARY                           /* Order by dept,then salary     */
BREAK DEPT NEWPAGE                    /* Organize by department      */

LINE 1 DEPT EMPNO LASTNAME TOTAL_COMP SALARY BONUS COMM
SUM SALARY BONUS COMM TOTAL_COMP             /* Create summary totals         */

* DEFINE THE SUMMARY REPORT
```

```
REPORT EMPSUM
TITLE 2 'DEPARTMENT SUMMARY'
ORDER DEPT                                  /* Order by dept,then salary     */
BREAK DEPT                                  /* Organize by department        */
LINE 1 DEPT TOTAL_COMP SALARY BONUS COMM TALLY
HEADING TALLY 'EMPLOYEE' 'COUNT'
SUM SALARY BONUS COMM TOTAL_COMP            /* Create summary totals         */
SUMMARY                                     /* Print summary lines only      */
```

```
05/02/2011                          EMPLOYEE LIST                          PAGE     1
                                  FOR DEPARTMENT   A00

DEPT  EMPNO      LASTNAME         TOTAL          SALARY         BONUS         COMM
                                COMPENSATION
A00   000010   HAAS              318956.48      305634.48       9102.00       4220.00
      000110   LUCCHESI          277346.05      269424.05       4202.00       3720.00
      000120   O'CONNELL         175757.15      169515.15       3902.00       2340.00
===   ======   =============   ============   ============   ============   ============
A00                             772059.68      744573.68      17206.00      10280.00

05/02/2011                          EMPLOYEE LIST                          PAGE     2
                                  FOR DEPARTMENT   B01

DEPT  EMPNO      LASTNAME         TOTAL          SALARY         BONUS         COMM
                                COMPENSATION
B01   000020   THOMPSON          50819.50       43417.50       4102.00       3300.00
===   ======   =============   ============   ============   ============   ============
B01                             50819.50       43417.50       4102.00       3300.00

05/03/2011                          EMPLOYEE LIST                          PAGE     3
                                  FOR DEPARTMENT   C01

DEPT  EMPNO      LASTNAME         TOTAL          SALARY         BONUS         COMM
                                COMPENSATION
C01   000030   KWAN              47429.50       40267.50       4102.00       3060.00
      000140   NICHOLLS          36122.00       29946.00       3902.00       2274.00
      000130   QUINTANA          30801.00       25095.00       3802.00       1904.00
===   ======   =============   ============   ============   ============   ============
C01                             114352.50      95308.50      11806.00       7238.00
```

Here is the resulting summary report:

```
05/02/2011                                                          PAGE     1
                                DEPARTMENT SUMMARY

DEPT     TOTAL          SALARY         BONUS         COMM         EMPLOYEE
       COMPENSATION                                                 COUNT
A00      772059.68      744573.68      17206.00      10280.00              3
B01       50819.50       43417.50       4102.00       3300.00              1
C01      114352.50       95308.50      11806.00       7238.00              3
D11      221293.25      197729.25      15208.00       8356.00              4
D21      151223.18      144328.18       4002.00       2893.00              1
E01       49604.75       42288.75       4102.00       3214.00              1
E11       37624.50       31342.50       3902.00       2380.00              1
E21       33456.50       27562.50       3802.00       2092.00              1
===   ============   ============   ============   ============   =============
       1430433.86     1326550.86      64130.00      39753.00             15
```

Report Example 7

Record selection is added next to restrict the report only to employees earning a salary of $50,0000 and greater.

The M5 mask is added to the total compensation variable in order to print the leading dollar sign. Finally, the summary line is now replaced with average values, using the AVG subcommand.

```
DEFINE TOTAL_COMP 5,P,2 HEADING('TOTAL' 'COMPENSATION') M5 /* define total var */

AUTO EMPVSAM                                /* This is an auto-read script  */
IF SALARY >= 50000.00                       /* Include only 50K+ employees  */
  TOTAL_COMP = SALARY + BONUS + COMM        /* Calc total compensation      */
  PRINT EMPRPT                              /* Write line to detail report  */
  PRINT EMPSUM                              /* Write line to summary report */
ENDIF

* DEFINE THE DETAIL REPORT
REPORT EMPRPT
TITLE 1 '50K+ EMPLOYEE LIST'
TITLE 2 'FOR DEPARTMENT' DEPT
ORDER DEPT -SALARY                          /* Order by dept,then salary     */
BREAK DEPT NEWPAGE                   /* Organize by department      */

LINE 1 DEPT EMPNO LASTNAME TOTAL_COMP SALARY BONUS COMM
AVG SALARY BONUS COMM TOTAL_COMP            /* Create summary averages       */

* DEFINE THE SUMMARY REPORT
REPORT EMPSUM
TITLE 1 '50K+ EMPLOYEE LIST'
TITLE 2 'DEPARTMENT SUMMARY'
ORDER DEPT                                  /* Order by dept,then salary     */
BREAK DEPT                             /* Organize by department      */

LINE 1 DEPT TOTAL_COMP SALARY BONUS COMM TALLY
HEADING TALLY 'EMPLOYEE' 'COUNT'
AVG SALARY BONUS COMM TOTAL_COMP            /* Create summary averages       */
SUMMARY                                     /* Print summary lines only      */
```

Here is the resulting first report:

```
05/03/2011                      50K+ EMPLOYEE LIST                      PAGE     1
                               FOR DEPARTMENT   A00

DEPT  EMPNO      LASTNAME         TOTAL          SALARY        BONUS        COMM
                                COMPENSATION
A00   000010   HAAS             $318,956.48     305634.48      9102.00      4220.00
      000110   LUCCHESI         $277,346.05     269424.05      4202.00      3720.00
      000120   O'CONNELL        $175,757.15     169515.15      3902.00      2340.00
===   ======   =============   ==============   ===========   ===========   ===========

A00                           $257,353.22     248191.22      5735.33      3426.66
05/03/2011                      50K+ EMPLOYEE LIST                      PAGE     2
                               FOR DEPARTMENT   D11

DEPT  EMPNO      LASTNAME         TOTAL          SALARY        BONUS        COMM
                                COMPENSATION
D11   000060   STERN             $67,482.78      61000.78      3902.00      2580.00
===   ======   =============   ==============   ===========   ===========   ===========

D11                            $67,482.78      61000.78      3902.00      2580.00
05/03/2011                      50K+ EMPLOYEE LIST                      PAGE     3
                               FOR DEPARTMENT   D21

DEPT  EMPNO      LASTNAME         TOTAL          SALARY        BONUS        COMM
                                COMPENSATION
D21   000070   PULASKI           $151,223.18     144328.18      4002.00      2893.00
===   ======   =============   ==============   ===========   ===========   ===========

D21                           $151,223.18     144328.18      4002.00      2893.00
```

Here is the resulting summary report:

05/03/2011 50K+ EMPLOYEE LIST PAGE 1
DEPARTMENT SUMMARY

DEPT	TOTAL COMPENSATION	SALARY	BONUS	COMM	EMPLOYEE COUNT
A00	$257,353.22	248191.22	5735.33	3426.66	3
D11	$67,482.78	61000.78	3902.00	2580.00	1
D21	$151,223.18	144328.18	4002.00	2893.00	1
===	==========	==========	==========	==========	==========
	$198,153.12	189980.52	5022.00	3150.60	5

Differences between DATA-MINER and CA-Easytrieve

- In DATA-MINER, the SORT statement must be in its own script; that is, in its own separate jobstep.

- In DATA-MINER, only one JOB statement per script is allowed. If more than one input file needed to be processed, either create multiple jobsteps or process the input files manually with script-level I/Os (using the READ command).

- DATA-MINER does not currently support IMS or DL/I. Supported data sources are VSAM and sequential datasets and the DB2 database.

- DATA-MINER does not support REPORT level procedures (for example BEFORE-BREAK). A common use of the BEFORE-BREAK report procedure is to calculate averages. The DATA-MINER report subcommand AVG can be used to accomplish the same results.

- DATA-MINER requires the REPORT subcommand SUM to calculate and print summary totals by report and control break.

- CA-Easytrieve programs are referred to by DATA-MINER as scripts.

- The DATA-MINER SHOW command replaces the debugging format of the CA-Easytrieve DISPLAY statement.

- The DATA-MINER TRACE command replaces the CA-Easytrieve PARM DEBUG statement.

- In DATA-MINER, the POINT command currently does not issue a VSAM POINT. Rather, it saves the position information and uses it on the READ command. This is different from CA-Easytrieve, and therefore you may need to alter existing CA-Easytrieve program logic if the return code from the POINT command is being checked and acted on.

- CA-Easytrieve commonly uses the %*name* syntax to include file field definitions or any other program source code. DATA-MINER both uses the INCLUDE command and supports the %*name* syntax.

- Each execution of the DATA-MINER job parses, compiles (in storage), and executes the script. There is no mechanism to compile a DATA-MINER script for later execution.