IBM Systems and Technology Group

As Cool as Ice: Analyzing Your RACF® Data Using DFSORT™ and ICETOOL

RACF-2014
Session RAA1
June 2014

Mark Nelson, CISSP®, CSSLP®
RACF Design and Development
IBM® Poughkeepsie
markan us.ibm.com

Trademarks

Agenda

The Shortest Overview of IRRDBU00 and IRRADU00 Ever

Processing IRRDBU00 and IRRADU00 Output using DFSORT

- Introduction to DFSORT
- Ordering records
- Selecting records
- Writing Records to Multiple Files
- Using ICETOOL
- Selecting Records using Relative Dates

The Shortest Overview of the RACF Data Base Unload Utility (IRRDBU00) and RACF SMF Unload Utility (IRRADU00) That You Will Ever Hear From Me

The Shortest Overview of IRRxxU00 Ever

The RACF Database Unload Utility (IRRDBU00) and the RACF SMF Unload Utility (IRRADU00) convert security information from its terse and cryptic format to a “flat file” format

- Unloaded format is easy to read and import into data-analysis tools such as DB[illegible]®, Microsoft® Excel®, SAS®, DFSORT, and even ISPF edit or browse
- Everything is unloaded, with the exception of security-sensitive fields (such as passwords and keys) and information which is “reserved for IBM use
- The output of these utilities is documented in “RACF Macros and Interface
- RACF ships default [illegible]+ sample reports in‘ SYS1.SAMPLIB(IRRICE)’

An Introduction to DFSORT

An Introduction to DFSORT: A Few Notes

All of the DFSORT information contained within this presentation is derived from:

- ***DFSORT: Getting Started (SC26-7527)***
- ***DFSORT Application Programming Guide (SC26-7523)***
  - *http://www.ibm.com/servers/storage/support/software/sort/mvs/srtmpub.html* contains these book and may more DFSORT publications

This focus of this session is that subset of DFSORT's functionality which is most used when doing basic analysis of RACF information. This is only a small subset of the many functions available with DFSORT and DFSORT's ICETOOL.

These examples may or may not work if you are using a sort product other than DFSORT. Consult your OEM sort product vendor for questions on other sort products.

An Introduction to DFSORT: The Big Picture

IBM’s DFSORT is a high-performance product that you can use to sort data, merge data, copy data, select data, reformat data, and create reports.

DFSORT works on information that is contained within data sets.

A *record* is a collection of related information that is managed as a unit, such as an employee record.

A *field* is a specific portion of a record that defines a particular category of data , such as an employee’s name.

- In DFSORT, each field is identified by a starting position, a length, and a data type (format).
- DFSORT supports a large number of data types, such as CH, ZD, PD, BI, FS, UFF, SFF, Y2K, etc.).
- DFSORT can also parse delimited fields, such as CSV data, into fixed fields

An Introduction to DFSORT: Sorting Data

As its name implies, sorting is one of the main functions of DFSORT, and is invoked with the SORT statement:

- SORT FIELDS=(startingPosition,length,dataType,sortOrder)
  - sortOrder ‘ A’ for ascending,‘ D’ for descending
  - dataType ‘ CH’ for character (EBCDIC) data

For example, to sort the output of the RACF SMF Unload Utility by the type of record

- `SORT FIELDS=(5,8,CH,A)`

An Introduction to DFSORT: Sorting Data...

You can sort on multiple fields by specifying multiple sort fields.

For example, to sort on the type of record, the date, and the time, code:

```
  SORT FIELDS=(5,8,CH,A,32,10,CH,A,23,8,CH,A)
```

Tip: If all of the fields have the same data type, you can code the FORMAT= keyword, which applies the specified data type to all of the fields in the statement:

```
  SORT FORMAT=CH,FIELDS=(5,8,A,32,10,A,23,8,A)
```

An Introduction to DFSORT: Selecting Records

DFSORT can be used to select records using the INCLUDE and OMIT statements.

- INCLUDE COND=(startingPosition,length,dataType,testType,value)
- OMIT COND=(startingPosition,length,dataType,testType,value)

You can specify either a constant value (C'YES ') or another field in the record (39,4,CH) for the value.

You can code multiple selection criteria, joined together with the Boolean AND and OR operators

- INCLUDE COND=(startingPosition,length,dataType,testType,value,AND OR, startingPosition,length,dataType,testType…)
- OMIT COND=(startingPosition,length,dataType,testType,value,AND OR, startingPosition,length,dataType,testType…)

You can code only one INCLUDE or OMIT statement per sort operation

An Introduction to DFSORT: Selecting Records…

You must code a SORT, MERGE, or COPY statement with your INCLUDE or OMIT statement

- SORT FIELD=COPY or OPTION COPY can be used instead of SORT FIELDS=( ) if you don’t want to or need to sort the input data

Example: Select all of the RACF Database Unload records (‘0200’ in columns 5-8) which define users who have the SPECIAL, OPERATIONS, or AUDITOR attribute (‘YES ‘ in columns 44-46, 49-51, or 390-392)

```
SORT    FIELDS=(10,8,CH,A)
INCLUDE COND=(5,4,CH,EQ,C'0200',AND,
                 (44,3,CH,EQ,C'YES',OR,
                  49,3,CH,EQ,C'YES',OR,
                 390,3,CH,EQ,C'YES'))
```

An Introduction to DFSORT: The Substring Operator

You can use the DFSORT substring ("SS") comparison test to find a specified character string anywhere in a field

```
INCLUDE COND=(10,44,CH,SS,C'*')
```

selects any record in which the character‘ *’ appears within columns 1[illegible] to

Consider this example:

```
INCLUDE COND=(5,4,CH,EQ,C'0500',AND,
              266,4,CH,EQ,C'NO  ',AND,
                 (10,249,SS,EQ,C'*',OR,
                  10,249,SS,EQ,C'%',OR,
                 10,249,SS,EQ,C'&'))
```

- Which finds all general resource profiles (record type '[illegible]') which are not generic (record offset [illegible] contains 'NO') but have a generic character in the name (the "SS" operands)

An Introduction to DFSORT: Reformatting Records

t

An Introduction to DFSORT: Reformatting Records…

One very powerful feature of OUTREC is the ability to insert text into a record during the reformatting processing

- Can be very useful in transforming data into more usable forms

Example: To copy the user ID, programmer name, and owner of the IRRDBU00 User Basic Data record, with a blank between each field, you would code

```
SORT    FIELDS=COPY
INCLUDE COND=(5,4,CH,EQ,C'0200')
OUTREC FIELDS=(1,4,C'The owner of ID ',10,8,C' is ',
       30,8,C' and is assigned to ',79,20)
OPTION   VLSHRT
```

An Introduction to DFSORT: Reformatting Records…

```
The owner of ID AUDTR01  is MARKN    and is assigned to EXTERNAL AUDITOR 01
The owner of ID AUDTR02  is MARKN    and is assigned to EXTERNAL AUDITOR 02
The owner of ID A6       is MARKN    and is assigned to ##################
The owner of ID BBB4     is MARKN    and is assigned to ##################
The owner of ID FRED     is MARKN    and is assigned to FRED THE OPERATOR
```

Some other powerful features of OUTREC are the ability to:

- **Reformat different records in different ways using IFTHEN clauses**
- **Overlay selected bytes of a record using OVERLAY fields**
- **Extract delimited fields (such as CSV) into fixed fields**
- **Justify and squeeze data**

An Introduction to DFSORT: OUTFIL

The OUTFIL statement allows you to create one or more output data sets from a single pass of your input data

- The output data sets can be complete copies of the input data set or may contain only selected records
  - You can select the records go into each output data set
    - Based on values within the record
    - Based on relative record number
    - “Round robin or “one rotation assignment
    - Every “nth record
  - You can reformat the selected records in various ways
- Very useful if you want to partition your RACF data base information or SMF
- OUTFIL can also be used to create reports

An Introduction to DFSORT: OUTFIL…

To put all of the IRRADU00 ACCESS records in one data set, all of the JOBINIT records into a second data set, and the remaining records into a third data set, you could code:

```
//SELECT    EXEC PGM=SORT
//SYSOUT    DD SYSOUT=*
//SORTIN    DD DISP=SHR,DSN=MARKN.TEST.IRRADU00
//ACCESS    DD DISP=SHR,DSN=MARKN.ACCESS.IRRADU00
//JOBINIT   DD DISP=SHR,DSN=MARKN.JOBINIT.IRRADU00
//OTHERS    DD DISP=SHR,DSN=MARKN.OTHERS.IRRADU00
//SYSIN   DD *
 SORT    FIELDS=COPY
 OUTFIL  FNAMES=ACCESS,INCLUDE=(5,4,CH,EQ,C'ACCESS')
 OUTFIL  FNAMES=JOBINIT,INCLUDE=(5,4,CH,EQ,C'JOBINIT')
 OUTFIL  FNAMES=OTHERS,SAVE
/*
```

An Introduction to DFSORT: Symbols

DFSORT allows you to define symbols that can be used to replace fields, constants, and output columns in DFSORT and ICETOOL statements with easy-to-read labels

- **USBD_OPER could be used as a symbol for 44,1,CH**

The RACFICE package on the “Downloads” section of the RACF web site (http://www.ibm.com/servers/eserver/zseries/zos/racf/) contains DFSORT symbols for all of the IRRADU00 and IRRDBU00 fields. Using these symbols you can specify:

```
SORT FIELDS=(USBD_NAME,A)
INCLUDE COND=(GRBD_RECORD_TYPE,EQ,C'0500',AND,
              GRBD_GENERIC,EQ,C'NO  ',AND,
                 (GRBD_NAME,SS,EQ,C'*',OR,
            GRBD_NAME,SS,EQ,C'%',OR,
            GRBD_NAME,SS,EQ,C'&'))
```

An Introduction to DFSORT: Variable Length Records

In z/OS, there are two data set structures:

- *Fixed-length* records, in which all records in the data set have the same length (RECFM=F, or FB) and
- *Variable-length* records, in which the records in the data seta do not have to have the same size records (RECFM=V,VB, VS, or VBS). Each record must have a length with is less than the maximum record length specified for the data set.
  - Each variable length record is prefixed with a four-byte *record descriptor word* (RDW) that contains the length of the record in the first two bytes.

When specifying the starting position, the RDW must by included.

- which means that you must add 4 to all of the starting positions documented for the IRRADU00 and IRRDBU00 output in *RACF Macros and Interfaces.* The DFSORT Symbols mentioned previously already have the RDW built into their starting positions.

DFSORT's ICETOOL Utility

An Introduction to ICETOOL: The Big Picture

ICETOOL is a multipurpose DFSORT utility that extends and simplifies the use of DFSORT

ICETOOL allows the processing of multiple input and output data sets within the same job step

ICETOOL processing is controlled by thirteen ICETOOL operators.

All of the power of DFSORT, as well as additional functions, are available to you through ICETOOL!

An Introduction to ICETOOL: ICETOOL Operators

ICETOOL Operator	Description
COPY	Copies a data set to one or more output data sets
COUNT	Counts the records in a data set or sets a return code based on the count
DEFAULTS	Prints the DFSORT installation defaults
DISPLAY	Print fields values with titles, headings, sections, etc
MODE	Sets the ICETOOL error option STOP, CONTINUE or SCAN STOP is the default
OCCURS	Print field values and their counts with titles, headings, etc
RANGE	Prints the count of values in a specified range
SELECT	Selects records based on value counts
SORT	Sorts a data set to one or more output data sets
SPLICE	Performs various join and match operations
STATS	Prints minimum, maximum, total, and average values
UNIQUE	Prints a count of unique values
VERIFY	Identifies invalid decimal values

An Introduction to ICETOOL: ICETOOL Operators...

All of the sample ICETOOL reports that are shipped in member IRRICE of ‘SYS1.SAMPLIB’ were written using just these ICETOOL operators:

- **COPY**
- **SORT**
- **DISPLAY**
- **OCCURS**

An Introduction to ICETOOL: SORT

The SORT operator allows you to sort data, select the records which are of interest, reformat the records, etc.

The syntax is:

```
SORT FROM(indd) USING(cntl) TO(outdd)
```

…where

- indd is the input data set DD name
- outdd is the output data set DD name
- cntl is the one to four character prefix for the DD name that contains the DFSORT control statements
  - The DD name is formed by appending “CNTL” to the USING value

An Introduction to ICETOOL: SORT…

Let’s create a realistic example. To create a report that lists all of the users who have either SPECIAL, OPERATIONS, or AUDITOR, we have to find all of the user basic data records (which have ‘0200’ in columns 5 to 9) and which have a ‘Y’ in column 44, column 49, or column 390.

The DFSORT INCLUDE statement which does this is:

```
INCLUDE COND=((44,1,CH,EQ,C'Y',OR,
               49,1,CH,EQ,C'Y',OR,
               390,1,CH,EQ,C'Y'),AND,
               5,4,CH,EQ,C'0200')
```

An Introduction to ICETOOL: DISPLAY

The DISPLAY operator allows you to:

- Select the information that is going to be in your report
- Specify single line or multi-line column headings
- Specify the title elements

The syntax is:

```
DISPLAY FROM(indd) ON(field,<parms>) LIST(listdd) options
```

…where

- indd is the DD name of the input data set
- field is the starting position, length and data type of the field that is to be in displayed the report
- listdd is the DD name of the report data set
- ON(field) can be (and usually is) repeated, once for each field in the report, up to 20 times

An Introduction to ICETOOL: DISPLAY…

For processing RACF data, the options that are most of interest are:

- ON(starting position,length,datatype), which identifies a field to be processed
- HEADER('string1','string2','string3'), which specifies a one, two, or three line column heading for each ON field
- TITLE('string'), which puts a title on the report
- PAGE, which puts the page number on the report
- DATE(format), which puts the current date, in various forms, on the report
- TIME(format), which puts the current time, in various forms, on the report
- BLANK, which suppresses leading zeros in numeric fields
- WIDTH, which sets the line length (width) of the report as 121 to 2048 bytes

There are many other DISPLAY operands… see "DFSORT: Getting Started" or "DFSORT Application Programming Guide" for details.

An Introduction to ICETOOL: DISPLAY…

Continuing our example: To create a report which lists all of the users who have either SPECIAL, OPERATIONS, or AUDITOR, along with their name (from columns 79 to 98) and their user ID (from columns 10 to 17), you code this ICETOOL DISPLAY operator:

```
DISPLAY FROM(TEMP0001) LIST(PRINT) -
        PAGE -
        TITLE('User IDs With Extraordinary Global Authorities')-
        DATE(YMD/) -
        TIME(12:)  -
        BLANK -
        ON(10,8,CH)  HEADER('User ID') -
        ON(79,20,CH) HEADER('User Name') -
        ON(44,4,CH)  HEADER('Special') -
        ON(49,4,CH)  HEADER('Operations') -
        ON(390,4,CH) HEADER('Auditor')
```

An Introduction to ICETOOL: JCL

```
//REPORT    EXEC PGM=ICETOOL
//TOOLMSG   DD SYSOUT=*
//DFSMSG    DD SYSOUT=*
//DBUDATA   DD DISP=SHR,DSN=USER01.IRRDBU00
//TEMP0001  DD DISP=(NEW,DELETE),SPACE=(CYL,(5,1,0)),UNIT=SYSALLDA
//PRINT     DD SYSOUT=*
//TOOLIN    DD *
 SORT    FROM(DBUDATA) TO(TEMP0001) USING(RACF)
 DISPLAY FROM(TEMP0001) LIST(PRINT) -
        PAGE -
        TITLE('User IDs With Extraordinary Global Authorities') -
        DATE(YMD/) -
        TIME(12:)  -
        BLANK -
        ON(10,8,CH)  HEADER('User ID') -
        ON(79,20,CH) HEADER('User Name') -
        ON(44,4,CH)  HEADER('Special') -
        ON(49,4,CH)  HEADER('Operations') -
        ON(390,4,CH) HEADER('Auditor')


/*
//RACFCNTL  DD *
 SORT    FIELDS=(10,8,CH,A)
 INCLUDE COND=((44,1,CH,EQ,C'Y',OR,
                49,1,CH,EQ,C'Y',OR,
               390,1,CH,EQ,C'Y'),AND,
               5,4,CH,EQ,C'0200')
/*
```

Sample RACFICE Report: Output

```
- 1 -        User IDs With Extraordinary Global Authorities        12/01/19

User ID     User Name               Special    Operations    Auditor
--------    --------------------    -------    ----------    -------
AUDTR01     EXTERNAL AUDITOR 01     NO         NO            YES
AUDTR02     EXTERNAL AUDITOR 02     NO         NO            YES
FRED        FRED THE OPERATOR       NO         YES           NO
IBMUSER                             YES        YES           YES
MARKN       ####################    YES        YES           YES
SYSPRG1     SYSTEMS PROGRAMMER 1    YES        NO            NO
```

An Introduction to ICETOOL: OCCURS

The OCCURS (or OCCUR) operator prints each unique value and its count (how many records it appears in) for the field you specify. You can use various options (title elements, headings, etc.) as for DISPLAY.

The syntax is:

```
OCCURS    FROM(indd) ON(field<,parms>)
           ON(VALCNT<,parms>) options
```

…where

- **indd is the DD name of the input data set**
- **field is the starting position, length and data type of the field to be displayed in the report**
  - **VALCNT is a special keyword for the count field**
- **ON(field) can be (and usually is) repeated, once for each field up to 10 times**

An Introduction to ICETOOL: OCCURS…

Example: Create a report which counts all of the RACF events by hour

- **The time is in hh:mm:ss format starting in column 23**

The ICETOOL OCCURS statement for this is:

```
OCCURS  FROM(IRRADU00) LIST(PRINT) -
    PAGE -
    TITLE('RACF Activity by Hour') -
    DATE(YMD/) -
    TIME(12:)  -
    BLANK -
    ON(23,2,CH)                    HEADER('Hour') -
    ON(VALCNT)                     HEADER('RACF Events Logged')
```

Sample RACFICE Report: OCCURS Output

```
- 1 -        RACF Activity by Hour        05/02/06        08:46:51 pm

Hour   RACF Events Logged
----   ------------------
00                    756
01                    160
02                    170
03                    214
04                    220
05                    206
06                    228
07                    244
08                    320
09                    390
10                    398
11                    448
12                    378
13                    704
14                   1308
15                    762
16                    762
17                    516
18                    260
19                    296
20                    160
```

Selecting Records Using Relative Dates

DFSORT allows you to select records based on a relative date, such as:

- [illegible] days ago
- [illegible] days from now

For IRRDBU00 and IRRADU00 output, the format of a DFSORT relative date is:

- DATE1(-)-d for past dates or DATE1(-)+d for future dates
  - DATE1 is the DFSORT date format for YYYYxMMxDD dates
  - (-) is the separator
    - IRRDBU00 and IRRADU00 dates are in the format yyy-mm-dd
  - "-d is the number of days prior to the current date "+d is the number of dates after the current date (not particularly useful for IRRADU00 and IRRDBU00 output)

Selecting Records Using Relative Dates…

Example: Find all of the user IDs which have been defined in the past 90 days

- In the past, this was usually done by writing a program which generated the DFSORT selection statements which would have hard-coded date values
  - See the “ULAST” example in ‘SYS1.SAMPLIB(IRRICE)’

Selecting Records Using Relative Dates...

DFSORT INCLUDE Statement:

```
SORT     FIELDS=COPY
INCLUDE COND=(5,4,CH,EQ,C'0200',AND,
19,10,CH,GE,DATE1(-)-90)
  OPTION   VLSHRT
```

Output:

```
1- 1 -        UL90: User IDs defined in the past 90 days        06/05/25        06:30:06 pm

 Date          User ID    Owner      Special    Operations   Auditor    Last Date    Last Time
 ----------    --------   --------   -------    ----------   -------    ----------   ---------
 2006-05-25    MARKN      IBMUSER    YES        YES          YES        2006-05-25   15:13:58
 2006-03-28    OMVSKERN   IBMUSER    NO         NO           NO         2006-03-28   10:32:45
 2006-04-05    ZOS17      IBMUSER    NO         NO           NO         2006-04-05
```

What RACF DB Reports Are in RACFICE?

Users who have extraordinary global/group RACF attributes
Discrete data set/general resource profiles which contain generic characters
Users who have more than 20 group connections
Count of user/group/data set/general resource (by class) profiles
User IDs with group privileges above USE
Data set standard and general resources with a UACC of other than NONE
Data set standard and conditional access lists with ID(*) of other than NONE
General resource standard and conditional access lists with ID(*) of other than NONE
Users who have explicit RRSF associations defined
User IDs with an OMVS segment
OS/390 UNIX super users (UID of zero)
OS/390 UNIX UIDs which are used more than once
HLQs with excessive generic profiles
HLQs with excessive fully-qualified generic profiles
User profiles defined in the past 90 days

What SMF RACFICE Reports are in RACFICE?

Events associated with a specific user
User IDs with excessive incorrect passwords
Terminals with excessive incorrect passwords
Accesses allowed due to WARNING mode profiles
Accesses allowed because the user has OPERATIONS
Users who are using Automatic Command Direction
Users who are directing command explicitly
User who log on with LOGON BY
RACLINK audit records
Users who are using password synchronization
Access violations

An Introduction to DFSORT: Sources of Information

Sources of Information

- The DFSORT website at **www.ibm.com/storage/dfsort** which contains samples, tricks, Q&A, papers, etc.
- The DFSORT library at **www.ibm.com/servers/storage/support/software/sort/mvs/srtmpub.html**, which has links to all of the DFSORT books and papers

DFSORT: Getting Started is an excellent tutorial on DFSORT, DFSORT's ICETOOL, and DFSORT Symbols

For complete information on the newer (April 2006) DFSORT/ICETOOL functions (such as relative dates), see:
www.ibm.com/servers/storage/support/software/sort/mvs/peug/

For complete information on the December 2004 DFSORT/ICETOOL functions (such as IFTHEN and OVERLAY), see:
www.ibm.com/servers/storage/support/software/sort/mvs/pdug/

The Joy of JOINKEYs

The Joy of JOINKEYS

- **JOINKEYS was introduced to DFSORT in November, 2009:**
  - UK 1[illegible] for z OS DFSORT V1R PTF
  - UK 1[illegible] for z OS DFSORT V1R1[illegible]
- **JOINKEYS allows you easily to create joined records in a variety of ways including inner join, full outer join, left outer join, right outer join, and unpaired combinations.**
- **The data for the JOINKEYS is in two input DD names**
  - The two input DD names can be of different types (fixed, variable, VSAM, and so on)
  - The keys (common fields) can be in different locations in the record
  - The two DD names can point to the same data set
- **There are three control statements for a JOINKEYS operation:**
  - **JOINKEYS** You must specify **two** JOINKEYS statements, one for each input file, specifying
    - The DD name of the file
    - The length and sequence of the keys in the file
    - Indicate whether the file is already sorted by those keys,
  - **JOIN** (optional) Defines the type of join Defaults to “inner
  - **REFORMAT** (optional for JOIN ONLY) **:** Defines the fields that you want in the joined records You can also request an indicator of where the key was found ('B' for both files, '1' for file 1 only or '[illegible]' for file [illegible] only) and a fill character for missing bytes

The Joy of JOINKEYS

- **Consider two tables**
  - One which contains baseball players names and a team ID
  - One which contains the team ID and the name of the team

Player	**Team ID**
Kranepool, Ed	NYM
Berra, Yogi	NYY
Gaedel, Eddie	SLB

Team ID	**Name**
NYM	NY Mets
NYY	NY Yankees
SFG	SF Giants

- **JOINKEYS allows you to create these JOINs**
  - **Inner join:** (Default) Only the paired records from (Kranepool, Berra)
  - **Left outer join:** The player records (Kranepool, Berra, Gaedel)
    NAME will be blank for Gaedel
  - **Right outer join** The team ID records
    Player will be blank for SF Giants
  - **Full outer join:** All records
    Player will be blank for SF Giants
    NAME will be blank for Gaedel
  - **Unpaired players** (Gaedel)
  - **Unpaired teams** (SFG)
  - **All unpaired** (Gaedel, SFG)

Sample JOINKEYS Job

```
//STEP0100 EXEC PGM=SORT
//SYSOUT   DD SYSOUT=*
//INA      DD *
----+----1----+----2----+----3----+----4
KRANEPOOL, ED                 NYM
BERRA, YOGI                   NYY
GAEDEL, EDDIE                 SLB
//INB      DD *
NYM NY METS
NYY NY YANKEES
SFG SF GIANTS
//SORTOUT  DD SYSOUT=*
//SYSIN    DD *
  OPTION COPY
  JOINKEYS F1=INA,FIELDS=(31,3,A)
  JOINKEYS F2=INB,FIELDS=(01,3,A)
  REFORMAT FIELDS=(F1:1,35,F2:5,15)
//*

The output would be:


----+----1----+----2----+----3----+----4----+----5
KRANEPOOL, ED                 NYM  NY METS
BERRA, YOGI                   NYY  NY YANKEES
```

Real-World Example of JOINKEYS

- Imagine a RACF database with a group (we'll call it “BIGGROUP”) into which almost every user is placed
- Imagine how easy it would be to merely put BIGGROUP on access lists to “get things to work”
- Imagine an auditor finding that profiles which had BIGGROUP on the access list were flagged as violating the installations “need to know” policy
- What would you do?

Real-World Example of JOINKEYS…

- **What this installation decided to do was to segment BIGGROUP into a set of organizational groups (we'll call them SMLGRP01, SMLGRP02, SMLGRP03.... etc.)**

- **Considerations: The client had:**
  - Only three months to get this done
  - A major application and an unmovable project deadline that depended on the BIGGROUP entries
- **Question: How would the client:**
  - Find all of the references to BIGGROUP
  - Notify the profile owners that they needed to move from BIGGROUP to one or more SMLGRPxx access list entries?
  - Provide a backout plan in to ensure that there were application outages caused by this migration?

Real-World Example of JOINKEYS: Approach

1. **Create a list/report which identified every BIGGROUP access list entry, showing the:**
   - Profile name and class
   - Access level
   - Profile owner
2. **Survey all of the application owners and users and create and populate the SMLGRPs**
3. **Work with the application owners and profile owners to add the SMLGRPs to the access list**
   - BIGGROUP would remain on the access list
4. **De-populate BIGGROUP**
   - In the event of a production problem, users could be re-connected to BIGGROUP on an emergency basis

Real-World Example of JOINKEYS: Approach..

- **We used an IRRDBU00-unload of the RACF Data Base to create the reports and data needed to:**
  - **Find all of the BIGGROUP references and the associated profile owners**
  - **Map the contents of the BIGGROUP and the SMLGRPs to identify:**

    All of the users in BIGGROUP who were not in any SMLGRP

    - These were the users who would no longer have access once BIGGROUP was “drained

    The set of SMLGRP members who were not in BIGGROUP

    - These were the users who may have gotten more authority than they had before

JOINKEYS Joining Access List Entry to Profile Owner

```
//GROUPREF  EXEC PGM=ICETOOL
//TOOLMSG   DD SYSOUT=*
//PRINT     DD SYSOUT=*
//DFSMSG    DD SYSOUT=*
//DBU1      DD DISP=SHR,DSN=USER01.IRRDBU00
//DBU2      DD DISP=SHR,DSN=USER01.IRRDBU00
//TEMP0001  DD UNIT=SYSALLDA,SPACE=(TRK,(10,10,0))
//TOOLIN    DD *
COPY JKFROM         TO(TEMP0001) USING(JOIN)

DISPLAY  FROM(TEMP0001) LIST(PRINT) -
         PAGE -
         TITLE('Data Set Profiles with References to BIGGROUP') -
         DATE(YMD/) -
         TIME(12:)  -
         BLANK -
         ON(01,44,CH)                HEADER('Data Set Name')     -
         ON(46,06,CH)                HEADER('VOLSER')            -
         ON(53,08,CH)                HEADER('Owner')


//JOINCNTL  DD *
 OPTION VLSCMP
 JOINKEYS F1=DBU1,FIELDS=(10,44,A,55,6,A),
    INCLUDE=(5,4,CH,EQ,C'0400')
 JOINKEYS F2=DBU2,FIELDS=(10,44,A,55,6,A),
    INCLUDE=(5,4,CH,EQ,C'0404',AND,62,8,CH,EQ,C'BIGGROUP')
 REFORMAT FIELDS=(F1:10,45,55,7,78,9,
                 F2:71,9)
/*
```

JOINKEYS Joining Access List Entry to Profile Owner…

```
- 89 -        Data Set Profiles with References to BIGGROUP        13/03/10
11:03:37 pm

 Data Set Name                                  VOLSER   Owner
 --------------------------------------------   ------   --------
 SYS1.MACS                                               PPP
 SYS1.TOOL*                                              PPP
 SYS1.TOOL.TSCENV                                        MVSSPT
 SYS1.TOOL.TSCUSER                                       MVSSPT
```

Disclaimer

The information contained in this document is distributed on as "as is" basis, without any warranty either express or implied. The customer is responsible for use of this information and or implementation of any techniques mentioned. IBM has reviewed the information for accuracy, but there is no guarantee that a customer using the information or techniques will obtain the same or similar results in its own operational environment.

In this document, any references made to an IBM licensed program are not intended to state or imply that only IBM's licensed program may be used. Functionally equivalent programs that do not infringe IBM's intellectual property rights may be used instead. Any performance data contained in this document was determined in a controlled environment and therefore, the results which may be obtained in other operating environments may vary significantly. Users of this document should verify the applicable data for their spec [illegible]