

Reusable TRANS I-descriptor dictionary items

Ross Morrissey

Article for the Geac October 1994 Service Plus Newsletter

Have you ever created an I-descriptor for a "one-off" report only to find that you've now created a demand for this "new" piece of information? With a little extra work, these I-descriptors can be made flexible enough to work in more than one dictionary. A sample approach might go like this:

Suppose we are working on a checkout statistics problem. We may need to relate three types of data: demographic, bibliographic, and transaction information. These data are related to PATRON, B.MASTER, and CHECKOUT files, and are keyed by the patron barcode, LCN, and pieces barcode respectively. If we create an I-descriptor, say L.B.MASTER.KEY, in each file that we run reports from, we can use any I-descriptor that performs a TRANS based on B.MASTER.KEY.

We can create keys involving complex inter-relationships between files by building them from simpler keys. Begin by building keys for the three types of data, in each file that may be used to originate a report. Note that each set of I-descriptors will be identical for any file with the same key (@ID).

To review, here is a typical I-descriptor:

DICT PATRON L.PIECES.KEY	- L. prefix denotes local item
0001: I	- Indicates I-descriptor
0002: TRANS ("BLOCKS", PATRON.KEY, 9, "X")	- Field definition
0003:	- Conversion code (optional)
0004: Item bar code	- Column heading (optional)
0005: 14L	- Output format
0006: S	- Single/Multivalued indicator

To save room only the item name and attribute 2 are shown below:

```
File:  PATRON (or BLOCKS).
L.PATRON.KEY      @ID
L.PIECES.KEY      TRANS ("BLOCKS",L.PATRON.KEY,9,"X")
L.B.MASTER.KEY    TRANS ("PIECES",L.PIECES.KEY,1,"X")
```

```
File:  CHECKOUT (or PIECES):
L.PIECES.KEY      @ID
L.B.MASTER.KEY    TRANS ("PIECES",L.PIECES.KEY,1,"X")
L.PATRON.KEY      TRANS ("CHECKOUT",L.PIECES.KEY,5,"X")
```

```
File:  B.MASTER (or B.MARC)
L.B.MASTER.KEY    @ID
L.PIECES.KEY      TRANS ("B.MASTER",L.B.MASTER.KEY,3,"X")
L.PATRON.KEY      TRANS ("CHECKOUT",L.PIECES.KEY,5,"X")
```

Now we can create some "universal" I-descriptors containing information in any of these files, that will be identical in each file, for example:

```
L.PATRON.CODE      TRANS ("PATRON",L.PATRON.KEY,14,"X")
L.COLL.CODE        TRANS ("PIECES",L.PIECES.KEY,8,"X")
L.CHKOUT.SUBLOC    TRANS ("CHECKOUT",L.PATRON.KEY,14,"X")
L.PATRON.NAME      TRANS ("PATRON",L.PATRON.KEY,1,"X")
L.TITLE
      OCONVS (TRANS ("B.MASTER",L.B.MASTER.KEY,4,"X"), "TA.MASTER;X1; ;1")
```

```
L.AUTHOR
  OCONVS (TRANS ("B.MASTER", L.B.MASTER.KEY, 5, "X"), "TA.MASTER;X1;;1")
L.SUBJECT
  OCONVS (TRANS ("B.MASTER", L.B.MASTER.KEY, 6, "X"), "TA.MASTER;X1;;1")
```

Note that the last three use OCONV to extract only the first subvalue of the authority.

Now, to list titles checked out to each patron with checkout sub-location:

```
SELECT CHECKOUT SAVING UNIQUE L.PATRON.KEY
SORT PATRON BY L.PATRON.NAME L.PATRON.NAME L.TITLE L.CHKOUT.SUBLOC
```

or alternatively:

```
SORT CHECKOUT BY L.PATRON.NAME L.PATRON.NAME L.TITLE L.CHKOUT.SUBLOC
```

or:

```
SELECT CHECKOUT
SORT PIECES BY L.PATRON.NAME L.PATRON.NAME L.TITLE L.CHKOUT.SUBLOC
```

A useful way to output these translations is in a "spreadsheet" format. To count checkouts by sub-location and patron group, create additional I-descriptors for each collection breakdown you're interested in:

```
L.COLL.RBR   IF (L.COLL.CODE EQ "RBR") THEN "1" ELSE "0"
L.COLL.A.V   IF (L.COLL.CODE EQ "A/V") THEN "1" ELSE "0"
L.COLL.OTHER
  IF (L.COLL.CODE NE "RBR") AND (L.COLL.CODE NE "A/V") THEN "1" ELSE
"0"
```

Now, to get a "spreadsheet" of collection vs. sub-location:

```
SORT CHECKOUT BY L.CHKOUT.SUBLOC BREAK-ON L.CHKOUT.SUBLOC TOTAL
L.COLL.RBR TOTAL L.COLL.A.V TOTAL L.COLL.OTHER ID-SUPP DET-SUPP
HEADING "Circ stats spreadsheet'L'"
```