OCCASION

This publication has been made available to the public on the occasion of the 50th anniversary of the United Nations Industrial Development Organisation.

DISCLAIMER

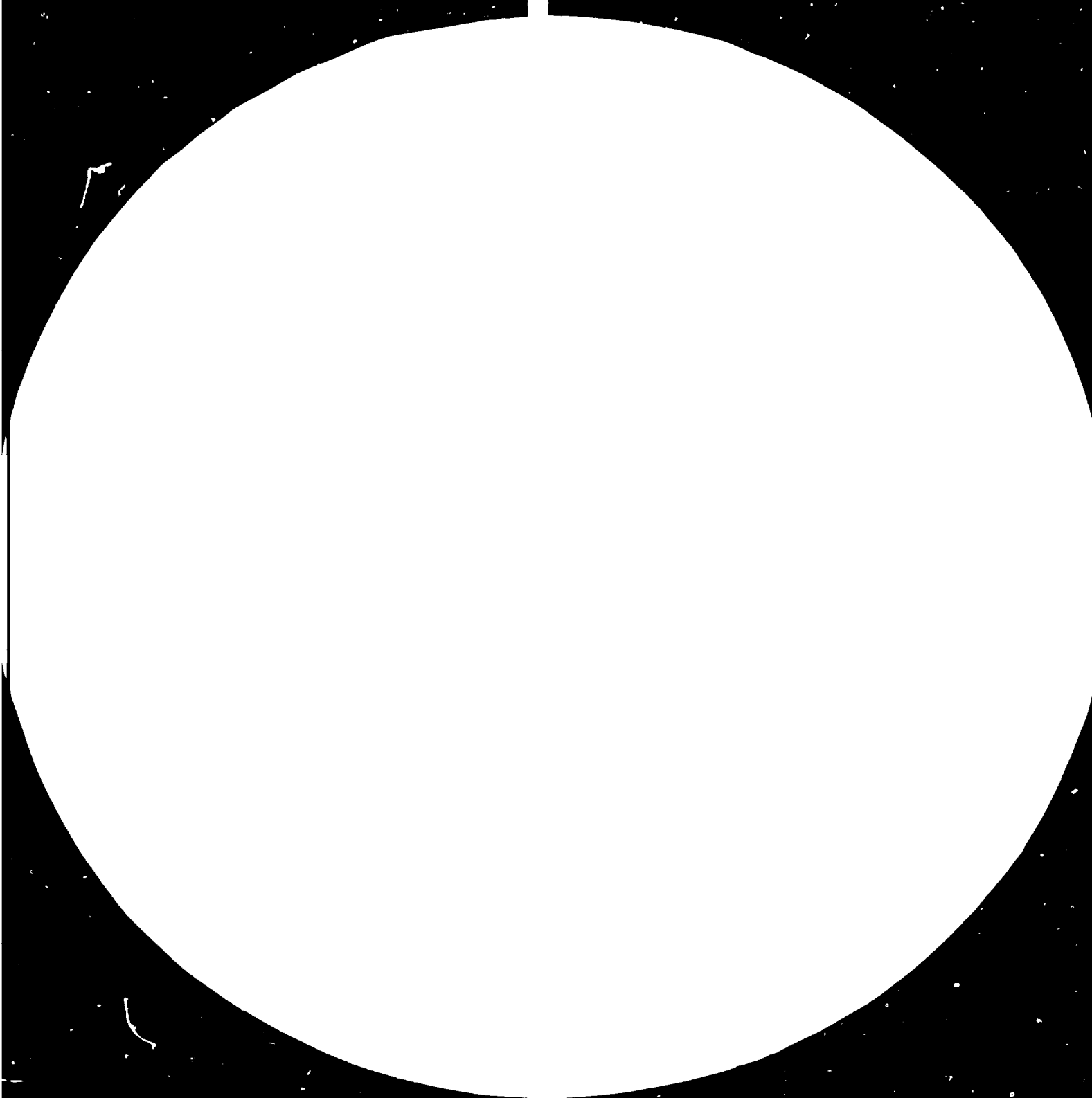
FAIR USE POLICY

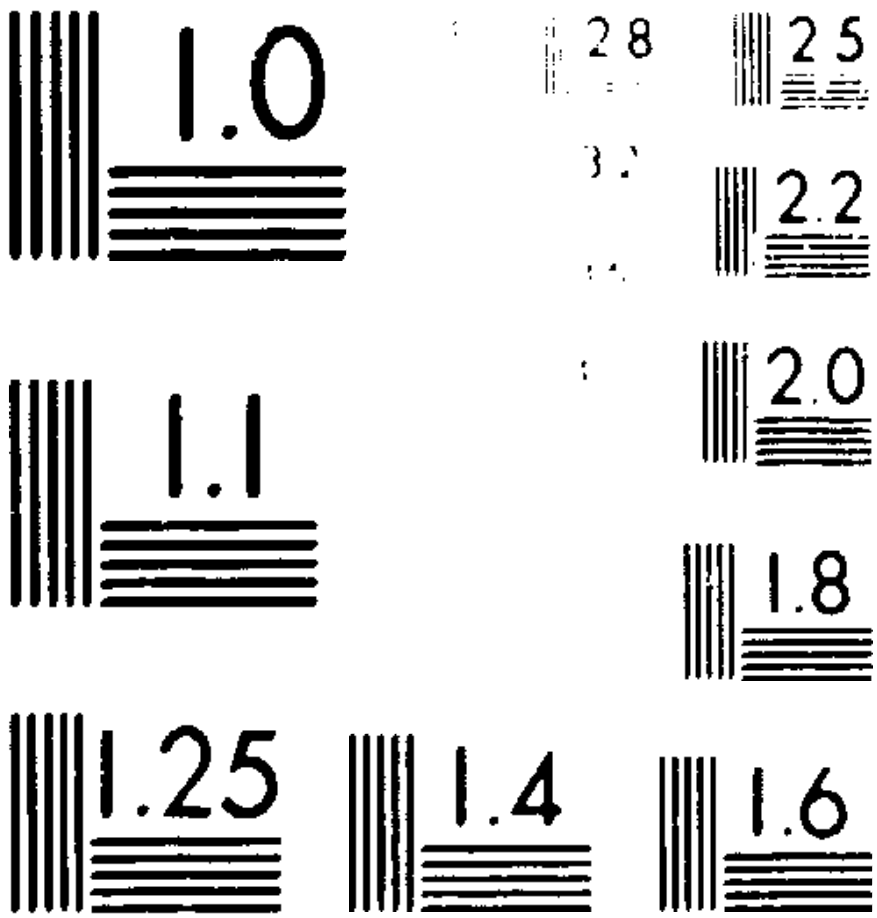
CONTACT

Please contact publications@unido.org for further information concerning UNIDO publications.

For more information about UNIDO, please visit us at www.unido.org

UNITED NATIONS INDUSTRIAL DEVELOPMENT ORGANIZATION
Vienna International Centre, P.O. Box 300, 1400 Vienna, Austria
Tel: (+43-1) 26026-0 · www.unido.org · unido@unido.org




1.0
2.8
2.5
2.2
1.1
2.0
1.8
1.25
1.4
1.6

11630

Distr.
LIMITED

UNIDO/IS.328
9 July 1982

UNITED NATIONS
INDUSTRIAL DEVELOPMENT ORGANIZATION

ENGLISH

THE UNIDO INPUT-OUTPUT DATABANK *

B. Dissmann

Prepared by the
Global and Conceptual Studies Branch

World Modelling Working Paper

* This document has been reproduced without formal editing.

V.82-28704

CONTENTS

FOREWORD

For some years UNIDO has been collecting input-output data, which of its nature is especially valuable for use in inter-industry studies and in the construction of economic models.

Input-output tables show the transaction between the sectors of the economy, and are thus of value in providing estimates of the linkages that operate not only between the subsectors of manufacturing but more generally between manufacturing and the primary and tertiary sectors of the economy as a whole. The tables collected have been of national economies. They vary widely in size, detail, and statistical concepts employed in their compilation. Before any use is made of an input-output table, or a set of tables for comparative purposes, it is therefore necessary to classify them according to the concepts that have been employed in their construction. To assist in this, a detailed standard description of each table has been prepared, according to a scheme known as NUDID (Numerical Data's Index and Description), which is described in Section 2.2 of the present report.

The completion of this data description and its analysis forms an essential preliminary step in the use of the input-output tables. The tables themselves are in some cases quite large, and in order to facilitate their manipulation it has been found necessary to construct a database system specifically directed towards input-output computations, which is the subject of the present report. The system allows for the storage, checking, editing, and analysis of the input-output data collection at present in UNIDO. It has been constructed by B. Dissmann, who wrote this manual.

1. Introduction

With time input - output tables become more and more important in our work. Input - output techniques become indispensable not only as a supplement and improvement of industrial statistics, but also for structural analysis, modelling and forecasting of national and regional economies.

More and more countries compile input - output tables more or less regularly. For some countries already some time series of input - output tables exists so that in the nearer future dynamic analysis of the structural change would be possible.

When collecting the individual tables it soon became clear that a real data bank would be necessary, but the diversity of the tables seems to be a barrier. Amongst others the following reasons are responsible for the differences:

- There exists no standard method of compiling individual tables.
- Even though the U.N. - SNA - system includes the input - output tables in the accounting and the OECD had developed guiding principles for compiling input - output tables, a lot of problems remain unsolved.
- The great difference between the developed and developing economies.
- Different accounting systems between planned and market economies.
- Different needs of individual countries.
- Different booking methods for handling imports and taxations.

With the development of the "Numerical Data's Index and Description", (called NUDID), a new step in the development of a data bank was reached. The main idea behind NUDID was to have a standardized description and documentations of the input - output - tables that is also suitable for computerization. Even though there is a wide range

of very heterogeneous tables, a standardized input - output table format was developed. Chapter 2 discusses this format in more detail.

Very often published and computerized tables diverge or contain errors. Often rows, columns or simple elements are missing, totals are inconsistent or are missing, single figures are wrongly punched and there are often other problems. So an input - output databank must include a feature for editing and checking the tables. Some desirable functions are:

- inserting
- deleting
- modifying existing
- copying
- summing
- subtracting

} of rows, columns and single elements

- division and multiplication by scalars
- multiplication by vectors and submatrices
- matrix transposition
- inverting matrices and calculation of determinants.

Another aspect of a databank will be to carry out some standard calculations for input - output analysis such as:

- computation of coefficients
- aggregation
- calculation of Leontief inverse
- calculation of eigenvalues and -vectors
- diagonalization and triangularization
- calculation of "most important" coefficients
- maximum and minimum vectors and elements in a given matrix and sub matrix
- backward and foreward linkages

Different updating techniques for input - output tables such as the different RAS-methods and updating by the Leontief inverse should also be possible.

In any case correct bookkeeping and documentation (via the NUDID) should be carried out by the databank-system.

Thus it could be possible that different versions of an original input - output table will be created by the user. So the administration of the whole databank will be another aspect, and features like the following must be supplied:

- creating copies of existing tables
- deleting members of the databank
- modifying existing members
- renaming members
- selecting members according to search criterias like U.N. codes, years and so on.

Another aspect of the databank must be to perform utilities for the user, such as:

- listing the Directory and a short description of the individual databank members
- displaying the tables and the NUDID at the terminal
- printing and plotting of the individual tables
- "punching" members of the databank in the UNIDO standard input - output-table format and other formats
- creating interfaces for the use of the tables in other programs
- listing the creation - history of a databank member.

Other demands on the databank are an easy handling by the user and a wide range of flexibility;

- a simple command structure
- making all functions independent, so that each function could be done in any order
- getting as much information as possible by the system
- using default values for the most common functions.

A particular program controls all these functions. (The UNIOP Program in Figure 1.1.) It must be noted that an input - output

table databank is a rather complex affair and that the UNIOP program is far from completion. But in the current version the basic structure and the most important and basic commands are implemented. Chapters 4 - 8 discuss the whole system in detail. The structure of the UNIDO input - output table databank is summarized in Figure. 1.1.

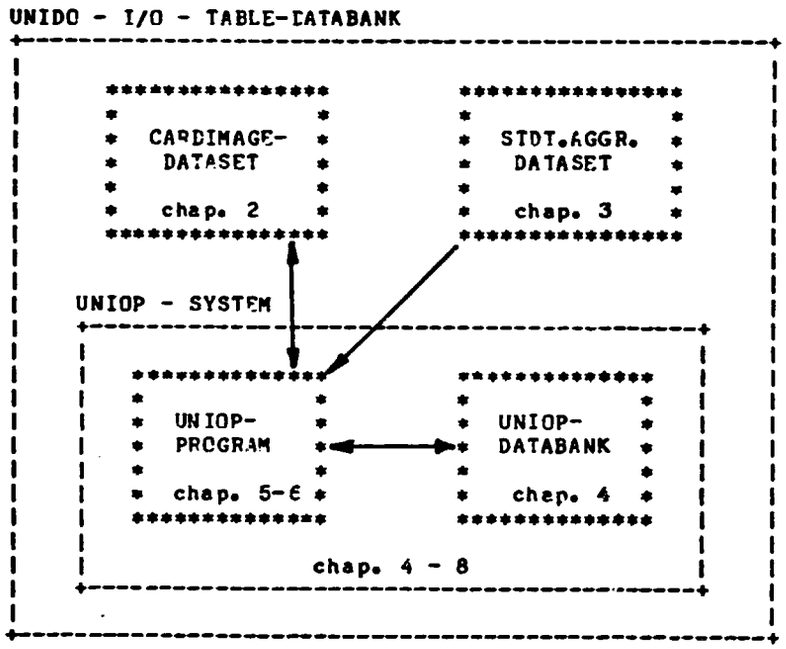


Fig. 1.1: Structure of the UNIDO - Input - Output - Databank

Since it was felt from the beginning that the most important input - output tables should be readily usable for other purposes, in the current version there is a distinction between the UNIOP system (consisting of the UNIOP program and the UNIOP databank) and two datasets (the master cardimage dataset and the standard aggregation scheme dataset) both together creating the UNIDO input - output table databank.

There are technical programming reasons why there is no permanent direct link between the UNIOP system and the master cardimage dataset. (The new FORTRAN V compiler will overcome this disadvantage. In any case there exists a link between the UNIOP system and the master cardimage file). Chapter 2 discusses the UNIDO standard input - output

table format, which is the way the master cardimage dataset is organized. Chapter 3 discusses in full detail the organization of the standard aggregation scheme dataset, necessary to perform aggregations on input - output tables. Chapter 4 and Chapter 5 describes the UNIOP system. Chapter 6 discusses the available commands in detail. Chapter 7 shows how to start an UNICP session. Chapter 8 gives some idea of the implementation of the UNIOP system.

2. The UNIDO standard input - output table format.

To handle the very different input - output tables by the computer and to make documentation possible a standardized format for the input - output tables was developed. For processing by the UNIOP system, the tables must be in this standard format. It is a very flexible instrument for describing and processing the tables.

In this chapter the following topics will be discussed:

2.1. Organization of input - output tables
2.2. Numerical data's index and description (NUDID)
2.3. Data formats

2.1. Organization of input - output tables

You can distinguish between two kinds of input - output table organization:

a) single tables (or 2 dimensional tables)
b) set of tables (or 3 dimensional tables)

The structure of a single table is shown in Figure 2.1.

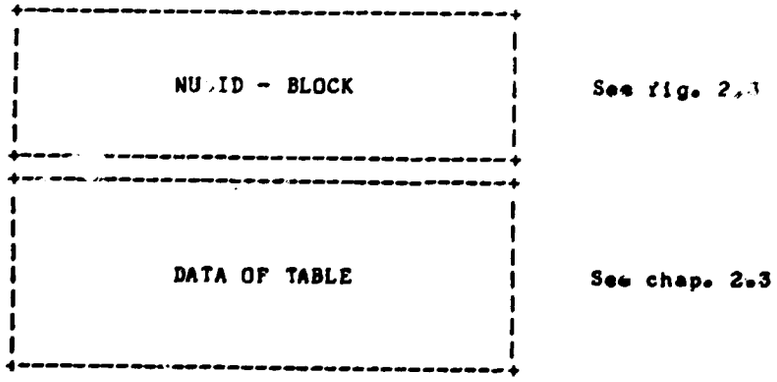


Fig. 2.1: Single Input - Output Table (2-dimensional Table)

It consists of two parts:

- the Numerical Data's Index and Description (NUDID) where all information about the table is stored (see also Chapter 2.2.) and
- the particular data which could be stored in different ways (see also Chapter 2.3.)

Input - output tables with the same structure (e.g. standardized tables) could be described by a set of tables. Figure 2.2. shows the structure of the latter.

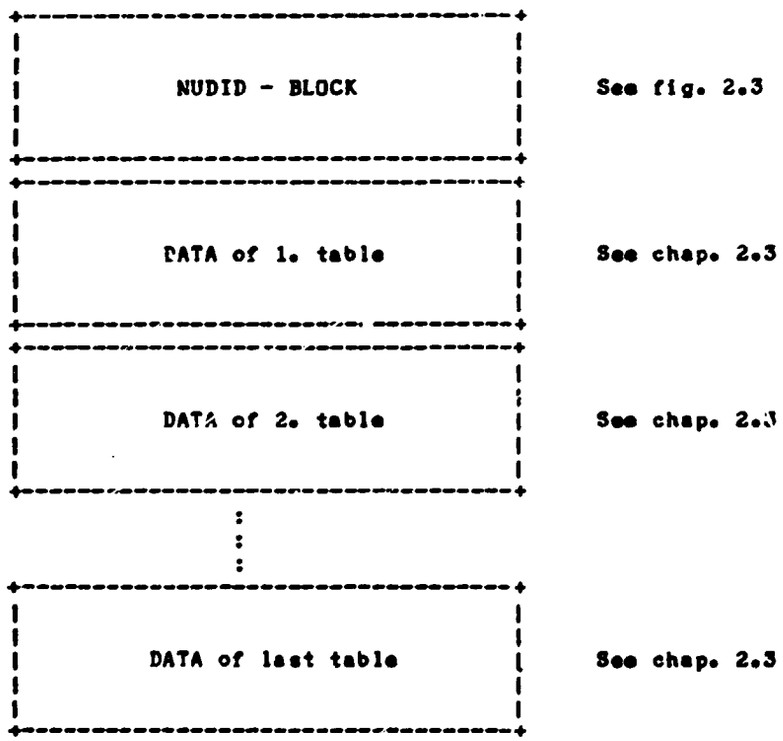


Fig. 2.2: Set of Input - Output Tables (3-dimensional Table)

In the current version of the UNIOP system a maximum of 50 tables could be collected with one NUDID. This feature saves a considerable amount of storage requirements.

The Numerical Data's Index and Description (NUDID) has some slight modifications compared to that of a single table. (See also Chapter 2.2.)

Each member (single table or set of tables) could be divided in one or more subtables. The most common case is a division in domestic, import and total transactions (or any permutation of the three). But also cases such as capital tables etc. could be described and processed.

The proper distinction between the subtables will be done in the NUDID (See Chapter 2.2.). In any case they will be stored one after the other.

2.2. Numerical Data's Index and Description (NUDID)

The current structure of NUDID is very flexible and makes use of our previous experience (UN statistical office, ECE, Dr. J. Skolka, DIW-Berlin, WU-Vienna, etc.) NUDID serves for documentary, administration and computing purposes. You can distinguish three main parts. (See figure 2.3.)

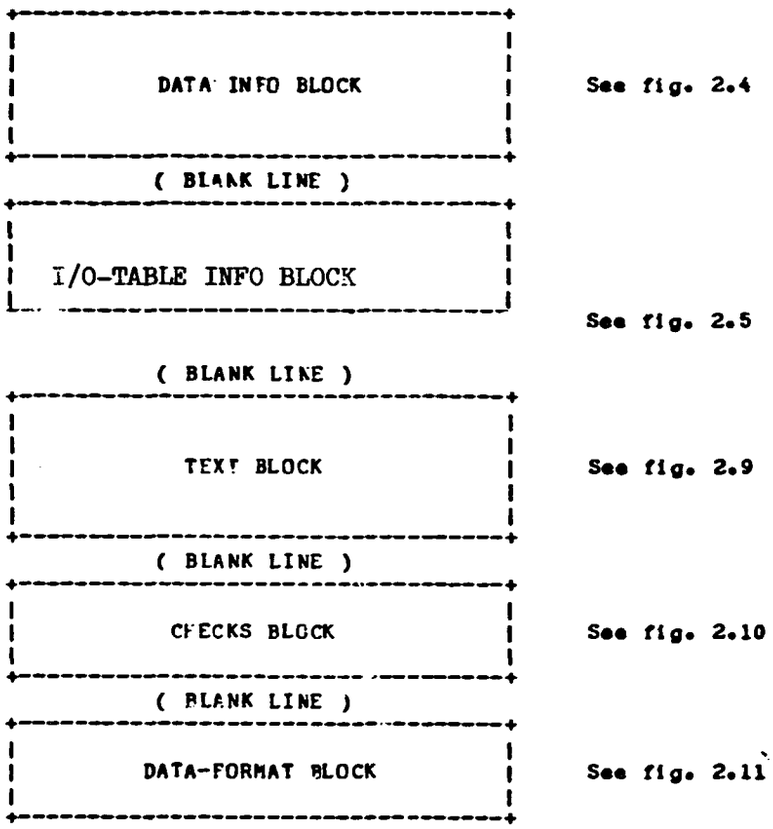


Fig. 2.3: Mainparts of Numerical Data's Index and Description (NUDID)

The next paragraph discusses these parts. Note that the blocks have to be separated from each other by a blanck line. The Endline (4 stars '*') separates the NUDID block from the data.

The UNIOP program processes the information from NUDID and makes it accessible to the program. Each modification of an input - output table will be updated by the program so that there always exists an updated NUDID for each table. Once the NUDID for the original table is filled in, the user will need no further work to get the correct NUDID for any table created out of the original table. This is one of the advantages of the UNIOP system.

The following rules are valid for the following figures:

- Dots (...) means the field of any alphanumeric character string. If you do not want to specify such an alphanumeric entry you fill the dots with blanks.
- Underscores (_____) means the field of any numeric string. If you do not want to specify such a numeric entry fill it with zeros (Ø) or with blanks.
- Numbers must be right justified in the field.
- Alphanumeric information must be left justified (except alphanumeric paragraphs).
- Ranges of numbers are indicated by a colon (:). entries which show such ranges could be specified by ranges.

  e.g. ----: ---- ----:---- ----:----

  could be specified like:

  1: 13 17: 19 25

  In any case a blank or zero field will terminate the number sequence.

2.2.1. Data info block of NUDID

Figure 2.4. describes the format of the data information block.

```
Rec      =cols   1         2         3         4         5         6         7         8
no.      ----+----0----+----0----+----0----+----0----+----0----+----0----+----0----+----0
 01      N U M E R I C A L   D A T A ' S   I N D E X   &   D E S C R I P T I O N  (NUDID)
 02      ------------------------------------------------------------------------------
 03      TITLE - ..................................................................
 04      ------------------------------------------------------------------------------
 05      INVENTORY NUMBER OF DATASET:                 .....-....
 06      TYPE OF DATASET:                             INPUT-OUTPUT-MATRIX
 07      REGION(S):                                   ........................................
 08      U.N. REGION CODE(S):                           __
 09      COUNTRY(S):                                  ........................................
 10      U.N. COUNTRY CODE(S):                        ___: ___  ___: ___  ___: ___  ___: ___
 11      AREA(S):                                     ........................................
 12      AREACODE(S):                                   __:  __ ____:  __   __:  __   __:  __
 13      YEAR(S):                                     19__ ____ ____ ____ ____ ____ ____
 14      MONTH:                                       ____ ____
 15      DAY(S):                                      ____ ____
 16      UNIT: (CURRENCY/COEFFICIENTS)                ..........................  OF Y 19__ __
 17      SCALE FACTOR                                 10E0
 18      DECIMALS:                                      __
 19      MISSING VALUE INDICATOR:                   _._____
 20      EXCHANGE RATE: 1 U.S.DOLLAR =              _._____
 21      DATA ORIGIN - ............................................................
 22      PUBLISHED :
 23      CONTACT - ................................................................
 24      ***
 25      METHOD OF COMPILATION - ..................................................
 26      DATA COMPUTERIZED BY - ...................................................
 27      DATA CHECKED BY - ........................................................
 28      DATA CORRECTED BY - ......................................................
 29      LEVEL OF PROCESSING:                         ........................................
 30      DATA ASSESSMENT:                             ........................................
 31      VERSION NUMBER OF DATASET                      __
 32      DIMENSIONS OF DATASET                          -
 33      TOTAL NUMBER OF ROWS (OBSERVATIONS):         ____
 34      TOTAL NUMBER OF COLUMNS (VARIABLES):         ____
 35 (a)  TOTAL NUMBER OF TABLES (REGIONS):            ____
 36      COMMENTS ON DATA :
 37      ***

Note:    (a) ... only for a set of tables
```

Fig. 2.4: Data - Info Block of NUDID

Rec. Nr. 3 : Optional title of the table (or set of tables).

Rec. Nr. 5 : Serial inventory number of the dataset.

Versions of the original dataset are indicated by an alpha numeric postfix string.

Rec. Nr. 6 : Is fixed for our purpose (NUDID serves also for other purposes).

Rec. Nr. 22 : Here you can include a maximum of 20 comment lines. "CONTACT-" is a preset comment line for I/O tables.

Rec. Nr. 24 : Ends the comment block.

Rec. Nr. 32 : If you want to describe a single table you must type 2 dimensions. If you want to describe a set of tables you must type 3 dimensions.

Rec. Nr. 35 : Is only for a set of tables (i.e. a 3 dimensional dataset).
There you type the number of tables that will be described by the single NUDID.

Rec. Nr. 36 : Here you can include a maximum of 20 comment lines.

Rec. Nr. 37 : Ends the comment block.

2.2.2. Input - Output Table - Information Block of NUDID

The input - output table Info Block has the following structure.
(See figure 2.5.)

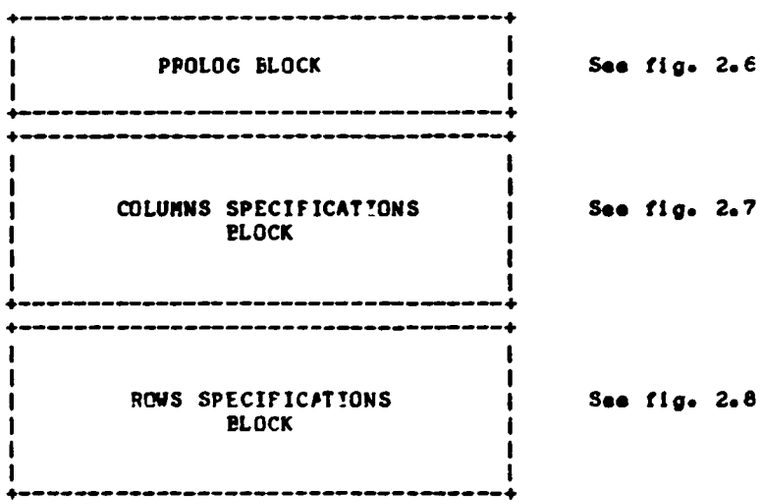


Fig. 2.5: Input - Output Table Info Block of NUDID

Figure 2.6 describes the format of the prolog block

```
Rec  =cols    1         2         3         4         5         6         7         8
no.  ----+----0----+----0----+----0----+----0----+----0----+----0----+----0----+----0
 1   SPECIAL INFORMATIONS FOR INPUT-OUTPUT-MATRICES :
 2   TABLE/MATRIX TYPE:                          I/O
 3   ENTRIES IN TABLE/MATRIX:                    FLOWS
 4   PRICING SYSTEM:                             PRODUCERS  CURRENT MARKET PRICES
 5   ACCOUNT SYSTEM:
 6   PRINCIPAL DIAGONAL:                         PRESENT
 7   ++++++++++++++++++++++++++++++++++++++++++++
```

Fig. 2.6: Prolog Block for i/o-table Info Block of NUDID

Rec. Nr. 2 : Input - output MAKE or ABSORPTION

Rec. Nr. 3 : Following types of entries exists

FLOWS or COEFFICIENTS

Rec. Nr. 4 : Describes the pricing system, e.g.

PRODUCERS CURRENT MARKET PRICES
PURCHASERS
BASIC VALUES

Rec. Nr. 5 : Describes the accounting system, e.g.

SNA (i.e. System of National Accounts)

MPS (i.e. System of Material Product Balances)

Rec. Nr. 6 : Says - if the principal diagonal is available in the tables (s). (PRESENT or NOT PRESENT)

Rec. Nr. 7 : Is a delimeter line.

Figure 2.7. describes the format of the columns - specification block.

```
Rec  =cols   1         2         3         4         5         6         7         8
no.  ----+----0----+----0----+----0----+----0----+----0----+----0----+----0----+----0
 01  STATISTICAL UNIT OF QUADRANT 1 COLUMNS: INDUSTRY
 02  TRANSACTION COLUMNS                     ____:____ ____:____ ____:____ ____:____
 03  SUBTOTAL COLUMNS IN TRANSACTIONS        ____:____ ____:____ ____:____ ____:____
 04  SUBDISAGGREGATION COLUMNS IN TRANSACT.  ____:____ ____:____ ____:____ ____:____
 05  UNALLOCATED/DUMMY COLUMN                ____
 06  TOTAL INTERMEDIATE DEMAND SUM   COLUMN  ____
 07  DOMESTIC FINAL DEMAND COLUMNS           ____:____ ____:____ ____:____ ____:____
 08  PRIVATE CONSUMPTION COLUMNS             ____:____ ____:____ ____:____ ____:____
 09  (TOTAL) PRIVATE CONSUMPTION COLUMN      ____
 10  GOVERNMENT CONSUMPTION COLUMNS          ____:____ ____:____ ____:____ ____:____
 11  (TOTAL) GOVERNMENT CONSUMPTION COLUMN   ____
 12  (TOTAL) CONSUMPTION COLUMN              ____
 13  GROSS FIXED CAPITAL FORMATION COLUMNS    ____:____ ____:____ ____:____ ____:____
 14  PRIVATE GROSS FIXED CAPITAL F. COLUMN    ____
 15  GOVERN. GROSS FIXED CAPITAL F. COLUMN    ____
 16  (TOTAL) GROSS FIXED CAPITAL F. COLUMN    ____
 17  CHANGES IN STOCKS COLUMNS                ____:____ ____:____ ____:____ ____:____
 18  CHANGES IN STOCKS COLUMN                 ____
 19  (TOTAL) INVESTMENT COLUMN                ____
 20  TAXES LESS SUBSIDIES COLUMN (IF -SIGN N  ____   +1
 21  DOMESTIC FINAL DEMAND COLUMN             ____
 22  EXPORT COLUMNS                           ____:____ ____:____ ____:____ ____:____
 23  (TOTAL) EXPORT COLUMN                    ____
 24  TOTAL FINAL DEMAND      COLUMN           ____
 25  UNALL./STATISTICAL DIFFERENCES           ____
 26  TOTAL DEMAND      COLUMN                 ____
 27  IMPORT & USES FROM STOCKS COLUMNS        ____:____ ____:____ ____:____ ____:____
 28  IMPORT COLUMN (IF - SIGN NOT GIVEN: -1)  ____   +1
 29  CUSTOM DUTIES ON IMPORTS COLUMN (IF - S  ____   +1
 30  INDIRECT TAXES ON IMPORTS COLUMN (IF - S____   +1
 31  DUTIES & TAXES ON IMPORTS COLUMN (IF - S____   +1
 32  TRANSPORT MARGIN COLUMN (IF _ SIGN NOT G____   +1
 33  TRADE MARGIN COLUMN (IF - SIGN NOT GIVEN____   +1
 34  TRANSP.&TRADE MARGIN COLUMN (IF - SIGN N____   +1
 35  TOTAL IMPORT COLUMN (IF - SIGN NOT GIVEN____   +1
 36  USES FROM STOCKS (IF -SIGN NOT GIVEN -1)____   +1
 37  STATISTICAL DIFFERENCES COLUMN           ____
 38  TOTAL OUTPUT COLUMN                      ____
 39  ****************************************
```

Fig. 2.7: Columns Specifications Block for i/o-table Info Block of NUDID

Rec. Nr. 1 : The following statistical units are possible
INDUSTRY or COMMODITY

Rec. Nr. 2 : Describes the range of transaction column of the table (s), including all totals and sub totals.

Rec. Nr. 39 : Is the delimiter line of the column specification block.

Figure 2.8. describes the format of the rows specification block

There, all subtables will be described and also the value added quadrant of the table (s).

Rec no.	=cols 1 2 3 4 5 6 7 8
01	STATISTICAL UNIT OF QUADRANT 1 ROWS: INDUSTRY
02	DOMESTIC TRANSACTION ROWS
03	SUBTOTAL ROWS IN DOMESTIC TRANSACTIONS
04	SUBDISAGGREGATION ROWS IN DOM. TRANSACT.
05	UNALLOCATED/DUMMY DOMESTIC TRANSACTIONS
06	DOMESTIC INTERMEDIATE INPUT SUMM. ROW
07	IMPORT TBL 0=NO/1=TOTAL/2=COMPET/3=SIMIL
08	IMPORTS ARE 1=FOB/2=CIF/3=INC.TRADE MAR
09	(T/C/S) IMPORT TRANSACTION ROWS
10	SUBTOTAL ROWS IN ( ) IMPORT TRANSACTIONS
11	SUBDISAGGREGATION ROWS IN ( )IMP. TRANSA.
12	UNALLOCATED/DUMMY (T/C/S) IMPORT ROW
13	(T/C/S) IMPORT SUMMATION ROW
14	NONCOMPETITIVE IMPORT TRANSACTION ROWS
15	SUBTOTAL ROWS IN NONCOMP. IMPORT TRANS.
16	SUBDISAGGREGATION ROWS IN NONC. IMP. TR.
17	UNALLOCATED NONCOMPETITIVE IMPORT ROW
18	NONCOMPETITIVE IMPORT SUMMATION ROW
19	(DIRECTLY ALLOCATED) TOTAL IMPORTS ROW
20	(TOTAL) TRANSFERRED IMPORTS ROW
21	TOTAL TRANSACTION ROWS
22	SUBTOTAL ROWS IN TOTAL TRANSACTIONS
23	SUBDISAGGREGATION ROWS IN TOTAL TRANSAC.
24	UNALLOCATED/DUMMY TOTAL TRANSACTIONS ROW
25	TOTAL INTERMEDIATE INPUT SUMMATION ROW
26	GROSS FIXED CAPITAL F. TRANSACTION ROWS
27	SUBTOTAL ROWS IN G.F.C.F. TRANSACTIONS
28	SUBDISAGGREGATION ROWS G.F.C.F. TRANSAC.
29	UNALLOCATED G. F. C. F. TRANSACTION ROW
30	GROSS FIXED CAPITAL F. SUMMATION ROW
31	SALES BY FINAL CONSUMERS
32	TYPE OF OTHER TRANSACTIONS
33	OTHER TRANSACTION ROWS
34	SUBTOTAL ROWS IN OTHER TRANSACTIONS
35	SUBDISAGGREGATION ROWS IN OTHER TRANSAC.
36	UNALLOCATED/DUMMY OTHER TRANSACTION ROW
37	OTHER TRANSACTION SUMMATION ROW

Fig. 2.8: Rows Specifications Block for i/o-table Info Block of NUDID

(cnt.)

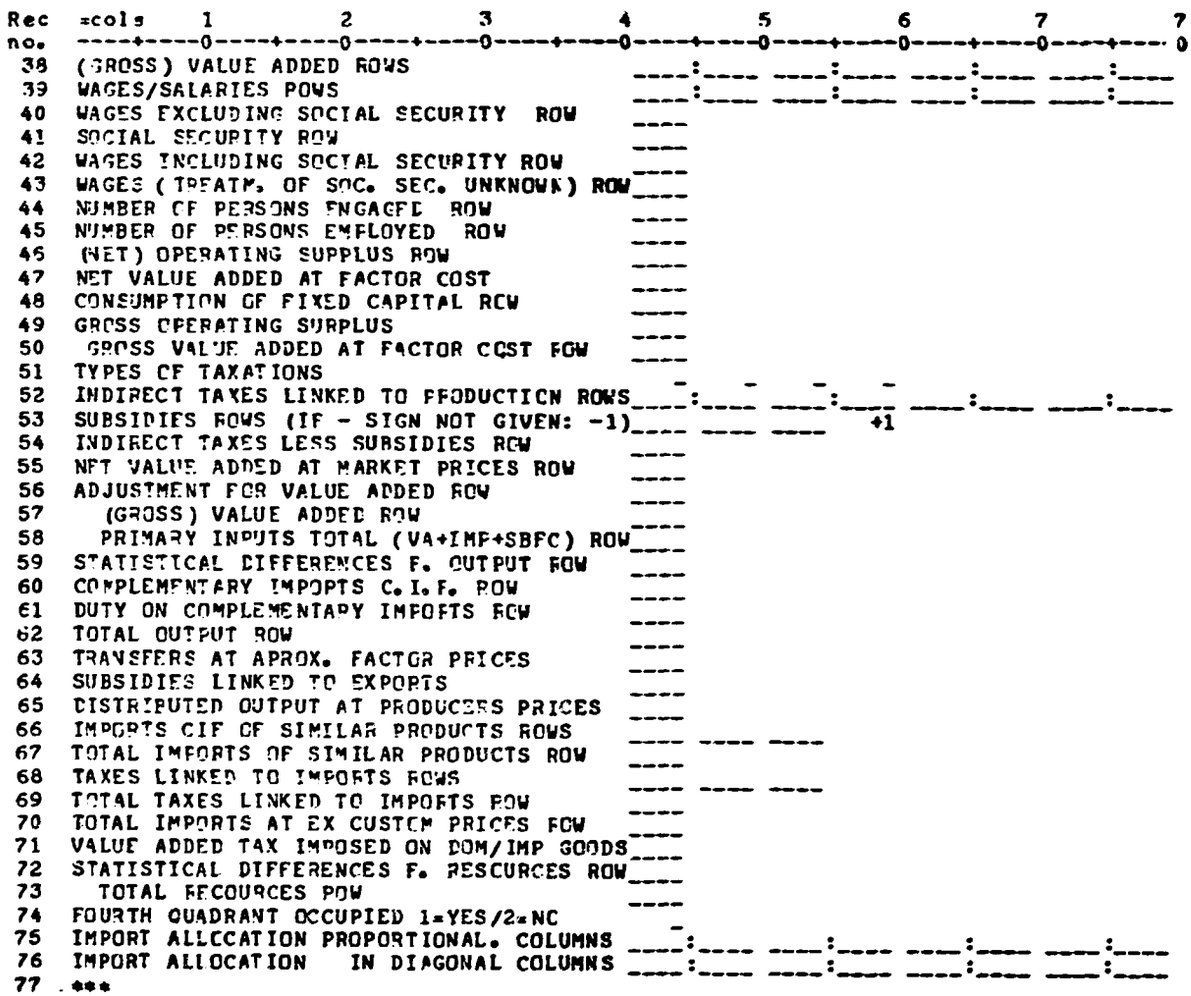


Fig. 2.8: Rows-Specification Block for i/o-table Info Block of NUDID

Rec. Nr. 1 : The following statistical units are possible.

INDUSTRY or COMMODITY

Rec. Nr. 2 - 6 : Description of domestic transactions.

Rec. Nr. 2 : Range of domestic transaction rows.
(Must be specified if given)

Rec. Nr. 7 - 20 : Description of imported transaction.

Rec. Nr. 7 : = 0 if none or only a "degenerated" table is given.

> 0 if a full input table is given.

Rec. Nr. 8 : Type of inports.

Rec. Nr. 9 : Range of input transactions.
(Must be specified if given.)

Rec. Nr. 21 - 25 : Description of total transaction.

Rec. Nr. 21 : Range of total transactions.
(Must be specified if given.)

Rec. Nr. 26 - 30 : Description of gross fixed capital formation.

Rec. Nr. 32 - 37 : Description of transaction not defined.

Rec. Nr. 32 : Integer indicator for describing type of other transactions. (Not defined until now)

Rec. Nr. 74 : = 1 if fourth quadrant occupied

= 2 if not.

2.2.3. TEXT Blocks

Figure 2.9. describes the format of the text block

```
Rec      #cols   1        2        3        4        5        6        7        8
no.      ----+----0----+----0----+----0----+----0----+----0----+----0----+----0----+----0
 01      TEXTS OF ROWS:
 02      ....... . ..............................  ........................................
 03                ISIC          ----,----,----
 04
 05      TEXTS OF COLUMNS:
 06      ....... . ...............  ..............  ........................................
 07                ISIC          ----,----,----
 08 (a)
 09 (a)  TEXTS OF TABLES:
 10 (a)  CCCYYYY . ..............................  ........................................
 11 (a)            REGION(S)     ----,----,----

Note:    (a) ... only for a set of tables
```

Fig. 2.9: Text Block for i/o-table Info Block of NUDID

Here, all rows, columns and eventually tables will be named.

Rec. Nr. 1 : (If the texts of the rows are stored in the Data block, you should fill in the reference specification.)

Rec. Nr. 2-3 : Repeat these lines according to the number of rows.

Rec. Nr. 3 : Is an optional comment and will not be handled in the current version of UNIOP.

Rec. Nr. 2 : Col. 1-7 : Specifies a row sequence number (will be handled as alpha numeric characters.

Col. 9 : Character which specifies an optional delimiter line in the outlay of the table (PRINT, BROWSE - command) '.' and '' (blank): no delimiter line.

Others (e.g. '-', '='): a delimiter line of the specified character will be created before the row is printed.

Rec. Nr. 4 : Delimiter line (blanck line) for TEXTS OF ROWS.

Rec. Nr. 5-7 : Analogous to Rec. Nr. 1-3 except that column texts have to be prepared in 2-line-format (2x15 characters).

Rec. Nr. 8-11 : Must only be specified if a set of tables will be described.

Repeat Rec. Nr. 10-11 according to the number of tables specified.

Rec. Nr. 10 : Col. 1-3 : U.N. Country Code

Col. 4-7 : Year (e.g. 1965)

Col. 9 : Will be ignored by the programme.

Col. 11-32: Short header for table.

Rec. Nr. 11 : Will be ignored in the current version.

2.2.4. CHECKSUM Block

Figure 2. 10. describes the format of the checks block.

```
Rec  =cols   1         2         3         4         5         6         7         8
no.  ----+----0----+----0----+----0----+----0----+----0----+----0----+----0----+----0
 01  CHECKSUMS TO CALCULATE   1          ROW  ____=____:____+____:____+____:____+____
 02  CHECKSUMS TO CALCULATE   2          ROW  ____=____:____+____:____+____:____+____
 03  CHECKSUMS TO CALCULATE   3          ROW  ____=____:____+____:____+____:____+____
 04  CHECKSUMS TO CALCULATE   4          ROW  ____=____:____+____:____+____:____+____
 05  CHECKSUMS TO CALCULATE   1          COL  ____=____:____+____:____+____:____+____
 06  CHECKSUMS TO CALCULATE   2          COL  ____=____:____+____:____+____:____+____
 07  CHECKSUMS TO CALCULATE   3          COL  ____=____:____+____:____+____:____+____
 08  CHECKSUMS TO CALCULATE   4          COL  ____=____:____+____:____+____:____+____
 09
```

Fig. 2.10: Checks Block for i/o-table Info Block of NUDID

The checks will not be handled in the current version of UNIOP.

Rec. Nr. 1 - 4 : Specifies checks for rows.

Rec. Nr. 5 - 8 : Specifies checks for columns.

Rec. Nr. 9 : Delimiter line.

2.2.5. Computerization Block

Figure 2.11. describes the format of the data format block.

```
Rec  =cols   1         2         3         4         5         6         7         8
no.  ----+----0----+----0----+----0----+----0----+----0----+----0----+----0----+----0
 1   COMMENTS ON COMPUTERIZATION OF DATA :
 2   ***
 3   DATAFORMAT :
 4   FIGURES ARE PUNCHED: ..............
 5        __  RECORDS PER ......
 6   ALFA INTG  INDE INDE INDE  RECO INDE INDE ITEM # OF  ALFA INTG
 7   IDEN SEQU  XING XING XING  RDSE XING XING QLTY ITEM  IDEN SEQU FORMATSTRING
 8   TIF. INFO  WHAT WHAT WHAT  QUEN WHAT WHAT MARK /REC  TIF. INFO
 9   ___________(..........................................................)
```

Fig. 2.11: Data-Format Block for i/o-table Info Block of NUDID

This block gives information related to the computerization of the data which follow the NUDID block. For a more detailed discussion of the possible formats see also Chapter 2.3.

Rec. Nr. 1 : You can include after this line maximum 20 comment lines.

Rec. Nr. 2 : Ends this comment block.

The following notation will be used in the next paragraphs:

A record can consist of 3 parts;

- the record prefix,
- the record infix,
- the record suffix.

The proper data will be described in the record infix. Record-prefix and suffix describe optional alpha indentifiers and index specifications.

Rec. Nr. 2 : The permitted entries are:
ROWWISE if data is stored row wise
COLUMNWISE if data is stored column wise
INDEXED ITEMS if data is stored in indexed format

Rec. Nr. 3 : Specifies the number of record per column or row.

Rec. Nr. 4 - 6 : Only for comments of Rec. Nr. 7.

Rec. Nr. 7 : Specifies contents and format of records.

Col. 1 : Number of words of record prefix - Alpha identification.
Ø no alphanumeric identifier

Col. 2 : Prefix - Integer sequence for records:
Ø ..... not present
1 ..... present

Col. 3 : 1. Index specification in record prefix.
Ø ..... no index
1 ..... row index
2 ..... column index
3 ..... table index

Col. 4 : 2. Index specification in record prefix.
Ø ..... no index
1 ..... row index
2 ..... column index
3 ..... table index

Col. 5 : 3. Index specification in Record prefix
Ø ..... no index
1 ..... row index
2 ..... column index
3 ..... table index

Col. 6 : Record sequence number in record prefix

Ø ..... not present

1 ..... present

Col. 7 : 1. Index specification in record infix.

Ø ..... no index

1 ..... row index

2 ..... column index

3 ..... table index

Col. 8 : 2. Index specification in record infix

Ø ..... no index

1 ..... row index

2 ..... column index

3 ..... table index

Col. 9 : Item quality mark

Ø ..... no quality mark

This feature will not be processed in the current version of UNIOP was introduced for general use of NUDID.

Col. 10 - 11 : Number of items per record. 2 digit number. Must be specified.

Col. 12 : Number of words of record - suffix - alpha identifier.

Ø ..... not present

Col. 14 - 80 : FORTRAN - Format string. The format string describes the single record analogue to the specification in Col. 1 - 13.

The format string must start with an opening bracket '(' and end with a closing bracket ')'. The rules are the same as in FORTRAN.

2.3. The Dataformat

The following three types of data can be processed by the UNIOP system:

i) Row wise stored tables.

ii) Column wise stored tables.

iii) Tables in indexed format.

Even the record specification (see rec. No. 7 in Figure 2.11.) allows a mixture of data of different tables by specifying a table index (3rd index). The current version of the UNIOP program does not handle such a case. To overcome this restriction, store the single tables sequentially one after the other.

2.3.1. Row wise and column wise stored tables

In this mode all data must be punched. You can specify a row or column index but do not have to. Also a record sequence number is optional. The mode (if row - or column wise will be determined by the indicator in Rec. Nr. 2. of the data format block (See figure 2.11.). If you do not specify the number of records per row or column (See figure 2.11.), the system will compute and check it.

In any case the number of item/record in the record specification (See Rec. Nr. 7 in figure 2.11.) should fit the replicator in the format string. Row wise storing of data will be more efficient in handling with the UNIOP program for row wise storing should be favoured above column wise storing. Since the system knows the number of records needed for reading a table no separating line between or at the end of tables is permitted.

2.3.2. Tables stored in indexed format

This mode has the advantage that only non zero entries in the table need to be stored (for large and spare tables). Here you must specify the table indices. Since the number of record per row or column is variable you need not specify rec. no. 3 in figure 2.11. (i.e. data

format block). You can store the data row wise or column wise (but not a mixture of both) by specifying the corresponding "indexing what" entries in the record specification of the data format block (see rec. Nr. 7. of figure 2.11.). Here row wise storing should also be favoured.

The "number of items/record" entry in the record specification of the data format block (see Rec. Nr. 7 of figure 2.11.) must be specified and shows the maximal number of data (and index) items in a record. In any case a blank or zero index in a record terminates the record.

It follows that at the end of each single table you must specify an empty record to mark the end of the table.

3. The Standard Aggregation Scheme File (SAF)

There are two different modes of aggregation possible:

- row aggregation called ROWS-MODE
- column aggregation called COLS-MODE

You can apply these modes in any order. For large tables it would be more efficient to start with row aggregation.

The following topics will be discussed in this chapter:

3.1. The Standard Aggregation Scheme File format (SAF Format).
3.2. How to create a new aggregation scheme file.
3.3. How to enter a new aggregation type and/or aggregation scheme.
3.4. List of available aggregation types and for each type a list of countries.

3.1. The Standard Aggregation Scheme File Format (SAF Format).

The SAF Format consists of two main parts (See figure 3.1.).

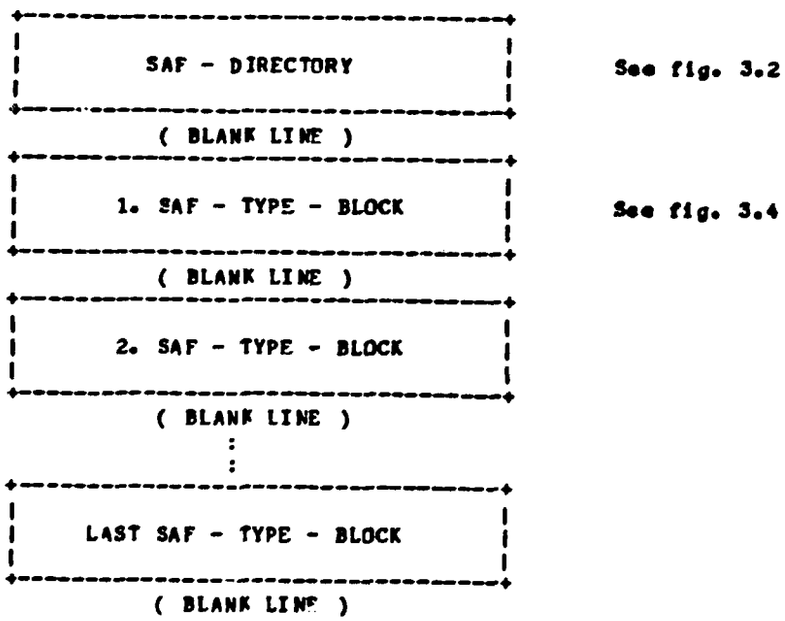


Fig. 3.1: Standard Aggregation Scheme File (SAF) Format

1. SAF Directory
2. SAF Type Block

After each main part you have to type in a blank line (separator).

For each standard aggregation scheme file (SAF) you must define:

a) only one SAF director
b) at least one SAF type block

3.1.1. The SAF - Directory

Format:

```
=cols    1         2         3         4         5         6         7         8
----+----0----+----0----+----0----+----0----+----0----+----0----+----0----+----0
COMMENT----------------------> AAAAAAAAAAAA NN AAAAAAAAAAAA NN AAAAAAAAAAAA NN
```

where:

```
COMMENT       ... OPTIONAL COMMENT: MAX. 32 CHARACTERS.
AAAAAAAAAAA   ... AGGREGATION-TYPE: MAX. 12 ALPHANUMERIC CHARACTES.
NN            ... TYPE-INDICATOR:   2 DIGITS.
```

Remarks:

1. AT LEAST ONE CARD MUST BE SPECIFIED.
2. AT LEAST ONE AGGREGATION-TYPE AND TYPE-INDICATOR MUST BE SPECIFIED.
3. ONE TO ONE CORRESPONDENCE BETWEEN AGGREGATION-TYPE AND TYPE-INDICATOR.

Example:

```
=cols    1         2         3         4         5         6         7         8
----+----0----+----0----+----0----+----0----+----0----+----0----+----0----+----0
DIRECTORY FOR AGGREGATIONTYPES:  UNITADR8     01 UNITADC8     02 UNIDOC16     03
                                 TECHNOLC43   04
```

Fig. 3.2: The SAF - Directory

The SAF directory is used for search purposes and must contain all aggregation types which are on the standard aggregation file. There must be a one to one correspondence between each aggregation type and each type indicator, because UNIOP tests the aggregation level of the tables. Only one SAF directory should be specified on the SAF. For a list of correspondence currently used see figure 3.3.

Correspondence Table for Standard Aggregation File:

Aggregation Type	Type Indicator
UNITADR8	01
UNITADC8	02
UNIDCC15	03
TECHNOLC43	04
RSUMCHCK	05
CSUMCHCK	06
IDIOMR7	07
IDIOMC7	08
CECDR	09
OECDC	10

Fig. 3.3: Currently used Correspondences

3.1.2. The SAF Type Block

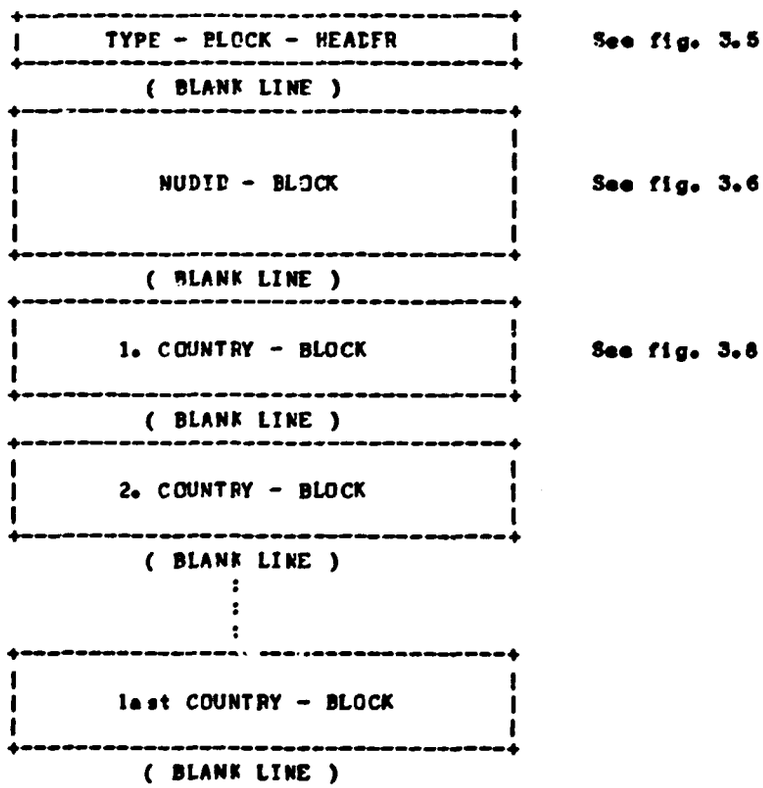


Fig. 3.4: The SAF Type Block

The SAF type block consists of a minimum of 3 segments (see figure 3.4.):

1. One type block leader.
2. One NUDID block corresponding to the type of aggregation (ROWS or COLUMNS).
3. At least one country block.

Each segment must be separated by a blank line (Separator).

3.1.2.1. The Type Block Header

Format:

```
=cols    1         2         3         4         5         6         7         8
----+----0----+----0----+----0----+----0----+----0----+----0----+----0----+----0
$AAAAAAAAAAA  NNN MMMM COMMENT-1 ------------------------------------------->

      where:
         AAAAAAAAAAA  ... AGGREGATIONTYPE:   12 ALPHANUMERIC CHARACTERS
         NNN          ... NEW SIZE OF TABLE: 3 DIGITS RIGHT BOUNDED
         MMMM         ... AGGREGATION MODE:  'ROWS' OR 'COLS'
         COMMENT-1    ... OPTIONAL COMMENT:  MAX. 56 CHARACTERS LONG

Remarks:
  1. A DOLLAR-SIGN ($) MUST BE PUNCHED IN THE FIRST COLUMN
  2. ONLY ONE CARD MUST BE SPECIFIED.
  3. COMMENT-1 MAY BE OMITTED.


Example:

  =cols    1         2         3         4         5         6         7         8
  ----+----0----+----0----+----0----+----0----+----0----+----0----+----0----+----0
  $UNITADR0      16 ROWS   9-SECTOR CLASSIFICATION
```

Fig. 3.5: The SAF - Type Block Header (one card)

This card must start with a dollar sign ($) in the first column. It opens the SAF type block.

"New size of table" means,

if ROWS MODE: New row number;

if COLS MODE: New column number.

"Aggregation mode" must be either the 4 characters 'ROWS' or COLS'.

3.1.2.2. The Standard Aggregation File (SAF) - NUDID Block

You have to take care of two different NUDID blocks, depending on the aggregation mode. In figure 3.6. all entries associated with:

a) Row aggregation (i.e. ROWS mode)
b) Column aggregation (i.e. COLS mode)

are shown.

Each entry signed with a mark ' ' must be filled in, according to the ROWS or COLS mode (e.g. entry 'Total number of ROWS' for ROWS mode and entry 'Total number of COLUMNS' for COLS mode.).

Each entry signed with a mark 'N' will not be updated in the NUDID for the aggregated table (viz. will be skipped) but must be in the SAF-NUDID block for the corresponding aggregation mode.

All other entries will be updated in the NUDID for the aggregated table.

Be aware that the SAF-NUDID block is slightly different to the standard NUDID block (See Chapter 2.). For example, you must not enter entries for a set of tables.

SAF - NUDID - BLOCK FOR ROWS- AND COLS-MODE

```
Rec        =cols    1           2           3           4           5           6           7           8
no.  Mark  ----+----0----+----0----+----0----+----0----+----0----+----0----+----0----+----0      Remarks

001   >    TITLE - ________________________________________________________________________         ***
002   N    INVENTORY NUMBER OF DATASET:                                                               |
003   N    TYPE OF DATASET:                                                                           |
004   N    REGION(S):                                                                                 |
005   N    U.N. REGION CODE(S):                                                                       |
006   N    COUNTRY(S):                                                                                |
007   N    U.N. COUNTRY CODE(S):                                                                      |
008   N    AREA(S):                                                                                   |
009   N    AREACODE(S):                                                                               |
010   N    YEAR(S):                                                                                   |
011   N    MONTH:                                                                                     |
012   N    DAY(S):                                                                                    |
013   N    UNIT: (CURRENCY/COEFFICIENTS)                                                              |
014   N    SCALE FACTOR                                                                               |
015   N    DECIMALS:                                                                                  |
016   N    MISSING VALUE INDICATOR:                                                                   |
017   N    EXCHANGE RATE:                                                                             |
018   N    DATA ORIGIN -                                                                              |
019        PUBLISHED :                                                                                |
020   N    ***                                                                                        |    FOR BOTH  ROWS- AND COLS MODE
021   N    METHOD OF COMPILATION -                                                                    |    (REC.NR. 1 - REC.NR.39)
022        DATA COMPUTERIZED BY -                                                                     |
023        DATA CHECKED BY -                                                                          |
024        DATA CORRECTED BY -                                                                        |
025   N    LEVEL OF PROCESSING:                                                                       |
026   N    DATA ASSESSMENT:                                                                           |
027   N    VERSION NUMBER OF DATASET                                                                  |
028   N    DIMENSIONS OF DATASET                                                                      |
029   >    TOTAL NUMBER OF ROWS (OBSERVATIONS):      ____                                             |<= ONLY FOR  ROWS-MODE UPDATED
030   >    TOTAL NUMBER OF COLUMNS (VARIABLES):      ____                                             |<= ONLY FOR  COLS-MODE UPDATED
031        COMMENTS ON DATA :                                                                         |
032   N    ***                                                                                        |
033   N                                                                                               |
034   N    SPECIAL INFORMATIONS FOR INPUT-OUTPUT-MATRICES :                                           |
035   N    TABLE/MATRIX TYPE:                                                                         |
036   N    ENTRIES IN TABLE/MATRIX:                                                                   |
037   N    PRICING SYSTEM:                                                                            |
038   N    ACCOUNT SYSTEM:                                                                            |
039   N    PRINCIPAL DIAGONAL:                       PRESENT                                         ***
```

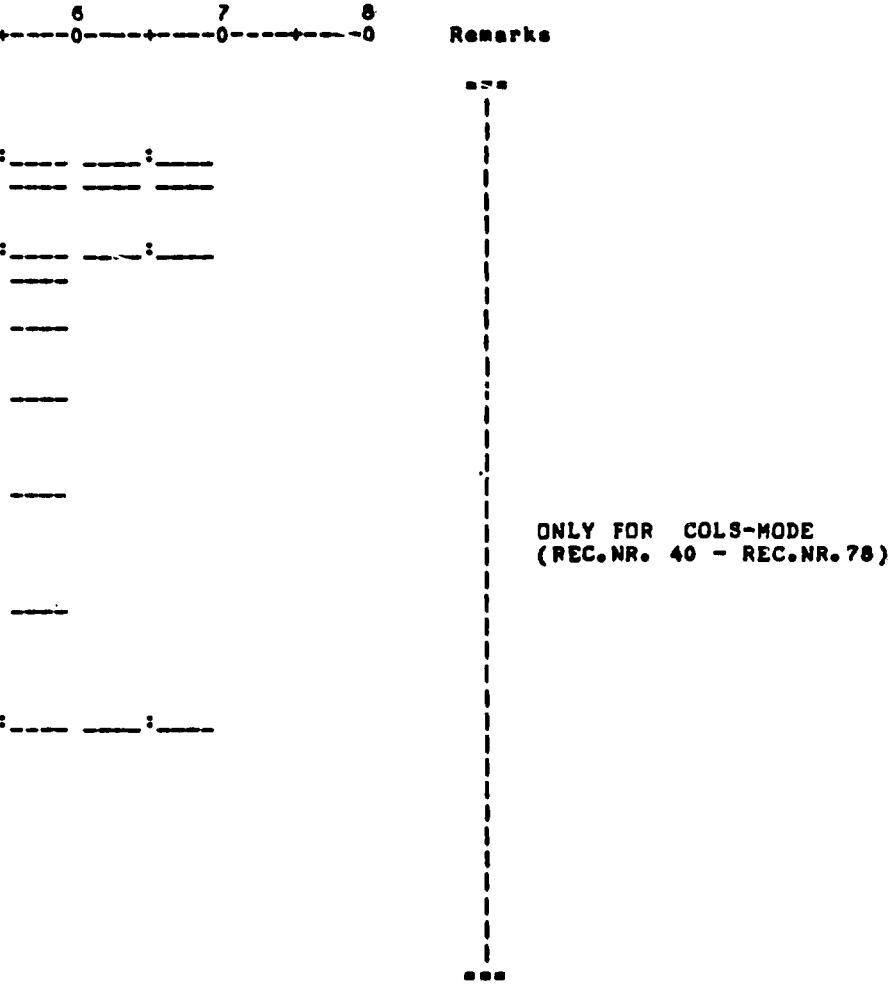
Fig. 3.6: The SAF - NUDID Block for ROWS- and COLS-mode

SAF - NUDID - BLOCK FOR ROWS- AND COLS-MODE

```
Rec          =cols    1         2         3         4         5
No.   Mark   ----+----0----+----0----+----0----+----0----+----0----
040    N     ++++++++++++++++++++++++++++++++++++++++
041    N     STATISTICAL UNIT OF QUADRANT 1 COLUMNS: INDUSTRY
042          TRANSACTION COLUMNS                     ____:____
043          SUBTOTAL COLUMNS IN TRANSACTIONS        ____:____ ____
044          SUBDISAGGREGATION COLUMNS IN TRANSACT.  ____ ____ ____
045          UNALLOCATED/DUMMY COLUMN                ____
046          TOTAL INTERMEDIATE DEMAND SUM   COLUMN  ____
047          DOMESTIC FINAL DEMAND COLUMNS           ____:____ ____
048          PRIVATE CONSUMPTION COLUMNS             ____ ____ ____
049          (TOTAL) PRIVATE CONSUMPTION COLUMN      ____
050          GOVERNMENT CONSUMPTION COLUMNS          ____ ____ ____
051          (TOTAL) GOVERNMENT CONSUMPTION COLUMN   ____
052          (TOTAL) CONSUMPTION COLUMN              ____
053          GROSS FIXED CAPITAL FORMATION COLUMNS   ____ ____ ____
054          PRIVATE GROSS FIXED CAPITAL F. COLUMN   ____
055          GOVERN. GROSS FIXED CAPITAL F. COLUMN   ____
056          (TOTAL) GROSS FIXED CAPITAL F. COLUMN   ____
057          CHANGES IN STOCKS COLUMNS               ____ ____ ____
058          CHANGES IN STOCKS COLUMN                ____
059          (TOTAL) INVESTMENT COLUMN               ____
060          TAXES LESS SUBSIDIES COLUMN (IF -SIGN N ____   +1
061          DOMESTIC FINAL DEMAND COLUMN            ____
062          EXPORT COLUMNS                          ____ ____ ____
063          (TOTAL) EXPORT COLUMN                   ____
064          TOTAL FINAL DEMAND      COLUMN          ____
065          UNALL./STATISTICAL DIFFERENCES          ____
066          TOTAL DEMAND       COLUMN               ____
067          IMPORT & USES FROM STOCKS COLUMNS       ____:____ ____
068          IMPORT COLUMN (IF - SIGN NOT GIVEN: -1) ____   +1
069          CUSTOM DUTIES ON IMPORTS COLUMN (IF - S ____   +1
070          INDIRECT TAXES ON IMPORTS COLUMN (IF - S____   +1
071          DUTIES & TAXES ON IMPORTS COLUMN (IF - S____   +1
072          TRANSPORT MARGIN COLUMN (IF _ SIGN NOT G____   +1
073          TRADE MARGIN COLUMN (IF - SIGN NOT GIVEN____   +1
074          TRANSP.&TRADE MARGIN COLUMN (IF - SIGN N____   +1
075          TOTAL IMPORT COLUMN (IF - SIGN NOT GIVEN____   +1
076          USES FROM STOCKS (IF -SIGN NOT GIVEN -1)____   +1
077          STATISTICAL DIFFERENCES COLUMN          ____
078          TOTAL OUTPUT COLUMN                     ____
```

Fig. 3.6: The SAF - NUDID Block for ROWS- a

(CNT.)

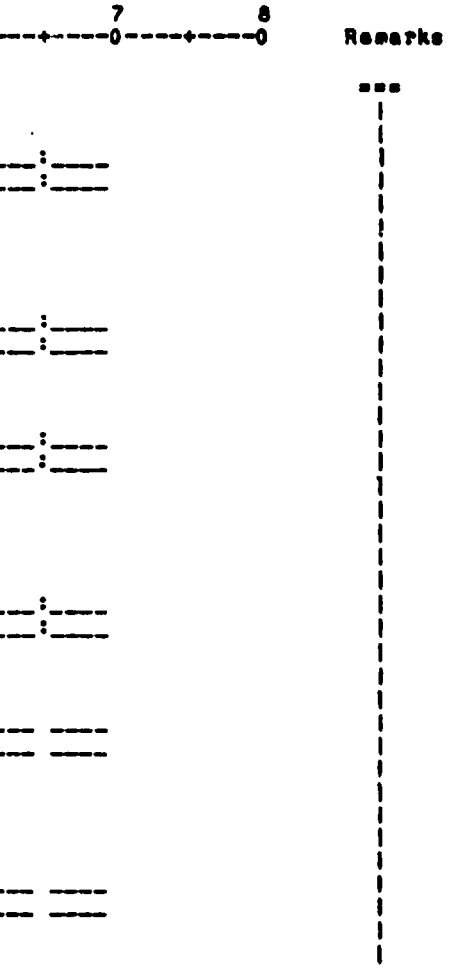


nd COLS-mode

SAF - NUDID - BLOCK FOR ROWS- AND COLS-MODE (CNT.)

```
Rec          =cols    1         2         3         4         5         6
no.   Mark  ----+----0----+----0----+----0----+----0----+----0----+----0--

079    N     ========================================
080    N     STATISTICAL UNIT OF QUADRANT 1 ROWS:     INDUSTRY
081          DOMESTIC TRANSACTION ROWS                ____:____
082          SUBTOTAL ROWS IN DOMESTIC TRANSACTIONS   ____:____ ____:____ --
083          SUBDISAGGREGATION ROWS IN DOM. TRANSACT. ____:____ ____:____ --
084          UNALLOCATED/DUMMY DOMESTIC TRANSACTIONS  ____
085           DOMESTIC INTERMEDIATE INPUT SUMM. ROW   ____
086          IMPORT TBL 0=NO/1=TOTAL/2=COMPET/3=SIMIL  _
087          IMPORTS ARE  1=FOB/2=CIF/3=INC.TRADE MAR  _
088          (T/C/S) IMPORT TRANSACTION ROWS          ____:____
089          SUBTOTAL ROWS IN ( ) IMPORT TRANSACTIONS ____:____ ____:____ --
090          SUBDISAGGREGATION ROWS IN ( )IMP. TRANSA.____:____ ____:____ --
091          UNALLOCATED/DUMMY (T/C/S) IMPORT ROW     ____
092           (T/C/S) IMPORT SUMMATION ROW            ____
093          NONCOMPETITIVE IMPORT TRANSACTION ROWS   ____:____
094          SUBTOTAL ROWS IN NONCOMP. IMPORT TRANS. ____:____ ____:____ --
095          SUBDISAGGREGATION ROWS IN NONC. IMP. TR.____:____ ____:____ --
096          UNALLOCATED NONCOMPETITIVE IMPORT ROW    ____
097           NONCOMPETITIVE IMPORT SUMMATION ROW     ____
098          (DIRECTLY ALLOCATED) TOTAL IMPORTS ROW   ____
099          (TOTAL) TRANSFERRED IMPORTS ROW          ____
100          TOTAL TRANSACTION ROWS                   ____:____
101          SUBTOTAL ROWS IN TOTAL TRANSACTIONS      ____:____ ____:____ --
102          SUBDISAGGREGATION ROWS IN TOTAL TRANSAC.____:____ ____:____ --
103          UNALLOCATED/DUMMY TOTAL TRANSACTIONS ROW____
104            TOTAL INTERMEDIATE INPUT SUMMATION ROW____
105          GROSS FIXED CAPITAL F. TRANSACTION ROWS ____:____
106          SUBTOTAL ROWS IN G.F.C.F. TRANSACTIONS  ____ ____ ____ ____ --
107          SUBDISAGGREGATION ROWS G.F.C.F. TRANSAC.____ ____ ____ ____ --
108          UNALLOCATED G. F. C. F. TRANSACTION ROW ____
109           GROSS FIXED CAPITAL F. SUMMATION ROW   ____
110          SALES BY FINAL CONSUMERS                 ____
111          TYPE OF OTHER TRANSACTIONS                _
112          OTHER TRANSACTION ROWS                   ____:____
113          SUBTOTAL ROWS IN OTHER TRANSACTIONS      ____ ____ ____ ____ --
114          SUBDISAGGREGATION ROWS IN OTHER TRANSAC.____ ____ ____ ____ --
115          UNALLOCATED/DUMMY OTHER TRANSACTION ROW  ____
116           OTHER TRANSACTION SUMMATION ROW         ____
```

Fig. 3.6: The SAF - NUDID Block for ROWS- and COLS-


7
8
0
0
Remarks
mode

SAF - NUDID - BLOCK FOR ROWS- AND COLS-MODE (CNT.)

```
Rec                 =cols    1         2         3         4         5         6         7         8
no.   Mark  ----+----0----+----0----+----0----+----0----+----0----+----0----+----0----+----0----+----0    Remarks

117           (GROSS) VALUE ADDED ROWS                    ____:____ ____:____ ____:____                 |  ONLY FOR  ROWS-MODE
118           WAGES/SALARIES ROWS                         ____:____ ____:____                           |  (REC.NR. 79 - REC.NR. 155)
119           WAGES EXCLUDING SOCIAL SECURITY  ROW        ____                                          |
120           SOCIAL SECURITY ROW                         ____                                          |
121           WAGES INCLUDING SOCIAL SECURITY ROW         ____                                          |
122           WAGES (TREATM. OF SOC. SEC. UNKNOWN) ROW    ____                                          |
123           NUMBER OF PERSONS ENGAGED  ROW              ____                                          |
124           NUMBER OF PERSONS EMPLOYED  ROW             ____                                          |
125           (NET) OPERATING SURPLUS ROW                 ____                                          |
126           NET VALUE ADDED AT FACTOR.COST              ____                                          |
127           CONSUMPTION OF FIXED CAPITAL ROW            ____                                          |
128           GROSS OPERATING SURPLUS                     ____                                          |
129            GROSS VALUE ADDED AT FACTOR COST ROW       ____                                          |
130           TYPES OF TAXATIONS                           -    -    -    -                            |
131           INDIRECT TAXES LINKED TO PRODUCTION ROWS    ____ ____ ____ ____                          |
132           SUBSIDIES ROWS (IF - SIGN NOT GIVEN: -1)    ____ ____ ____  +1                           |
133           INDIRECT TAXES LESS SUBSIDIES ROW           ____                                          |
134           NET VALUE ADDED AT MARKET PRICES ROW        ____                                          |
135           ADJUSTMENT FOR VALUE ADDED ROW              ____                                          |
136             (GROSS) VALUE ADDED ROW                   ____                                          |
137             PRIMARY INPUTS TOTAL (VA+IMP+SBFC) ROW    ____                                          |
138           STATISTICAL DIFFERENCES F. OUTPUT ROW       ____                                          |
139           COMPLEMENTARY IMPORTS C.I.F. ROW            ____                                          |
140           DUTY ON COMPLEMENTARY IMPORTS ROW           ____                                          |
141           TOTAL OUTPUT ROW                            ____                                          |
142           TRANSFERS AT APROX. FACTOR PRICES           ____                                          |
143           SUBSIDIES LINKED TO EXPORTS                 ____                                          |
144           DISTRIBUTED OUTPUT AT PRODUCERS PRICES      ____                                          |
145           IMPORTS CIF OF SIMILAR PRODUCTS ROWS        ____ ____ ____                                |
146           TOTAL IMPORTS OF SIMILAR PRODUCTS ROW       ____                                          |
147           TAXES LINKED TO IMPORTS ROWS                ____ ____ ____                                |
148           TOTAL TAXES LINKED TO IMPORTS ROW           ____                                          |
149           TOTAL IMPORTS AT EX CUSTOM PRICES ROW       ____                                          |
150           VALUE ADDED TAX IMPOSED ON DOM/IMP GOODS    ____                                          |
151           STATISTICAL DIFFERENCES F. RESOURCES ROW    ____                                          |
152             TOTAL RECOURCES ROW                       ____                                          |
153           FOURTH QUADRANT OCCUPIED 1=YES/2=NO          -                                            |
154           IMPORT ALLOCATION PROPORTIONAL. COLUMNS     ____:____ ____:____ ____:____ ____:____       |
155           IMPORT ALLOCATION   IN DIAGONAL COLUMNS     ____:____ ____:____ ____:____ ____:____       |***
```

Fig. 3.6: The SAF - NUDID Block for ROWS- and COLS-mode

SAF - NUDID - BLOCK FOR ROWS- AND COLS-MODE (CNT.)

```
Rec                 •cols    1         2         3         4         5         6         7         8
no.   Mark  ----+----0----+----0----+----0----+----0----+----0----+----0----+----0----+----0----+----0
156    N     •••
157    N
158    N     TEXTS OF ROWS:
159    >     0000001 • XXXXXXXXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
 :     >        :                   :                   :
 :     >        :                   :                   :
 :     >        :                   :                   :
160    N
161    N     TEXTS OF COLUMNS:
162    >     0000001 • XXXXXXXXXXXXXXX/XXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
 :     >        :                   :                   :
 :     >        :                   :                   :
 :     >        :                   :                   :
```

Remarks

```
 <== FOR BOTH  ROWS- AND COLS-MODE
•••
 |
 |   ONLY FOR  ROWS-MODE
 |   (REC.NR. 157 - 158+NROWS)
 |
•••
•••
 |
 |   ONLY FOR  COLS-MODE
 |   (REC.NR. 160 - 161+NCOLS)
 |
•••
```

Fig. 3.6: The SAF - NUDID Block for ROWS- and COLS-mode

3.1.2.3. Country Block

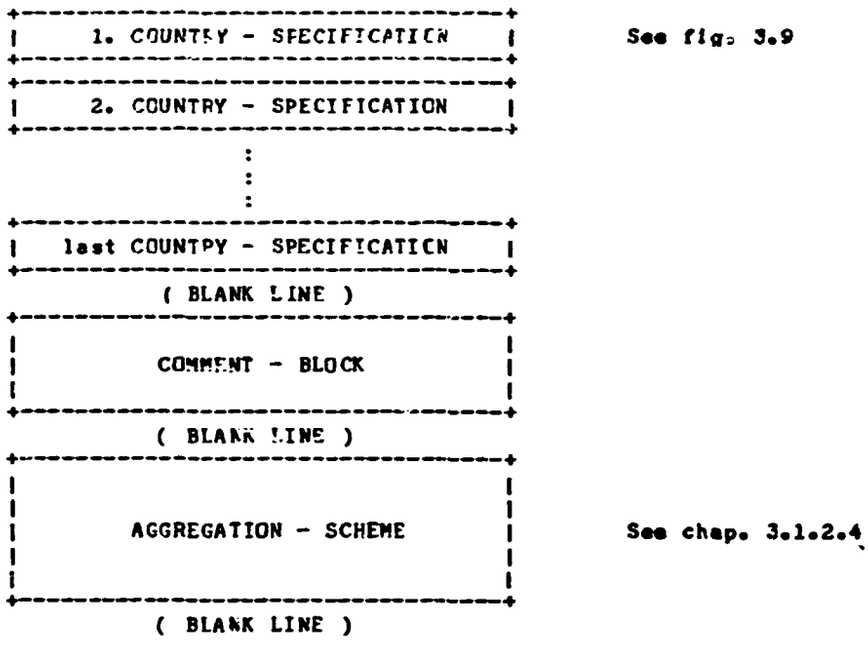


Fig. 3.8: The nonempty SAF - Country Block

Each country block is, either a non empty country block, OR, an empty country block, consisting only of a blank line (separator).

You must use at least one country block for one SAF type block.

If you use an empty country block it will close the SAF type block. Any other country block following this empty block will cause an error.

It follows, that each SAF block must have two blank lines (Separators) at the end. These two separators identify the end of the SAF block.

The country specification card is shown in figure 3.9.

Format:

```
=cols    1         2         3         4         5         6         7         8
----+----0----+----0----+----0----+----0----+----0----+----0----+----0----+----0
 CCC YYYY VV NNNNNNNNNNNN COMMENT-------------------------------------------->
```

where:

CCC	... U.N. COUNTRY CODE:	3 DIGITS, RIGHT ADJUSTED.
YYYY	... YEAR:	4 DIGITS, RIGHT ADJUSTED
VV	... VERSION:	2 DIGITS, RIGHT ADJUSTED.
NNNNNNNNNNNN	... COUNTRY-NAME:	MAX. 12 CHARACTERS, LEFT ADJUSTED.
COMMENT	... OPTIONAL COMMENT:	MAX. 53 CHARACTERS.

Remarks:
1. COMMENT IS OPTIONAL AND MAY BE OMITTED.
2. COUNTRY-NAME MUST NOT SPECIFIED IN U.N.-FORM.

EXAMPLES:

```
=cols    1         2         3         4         5         6         7         8
----+----0----+----0----+----0----+----0----+----0----+----0----+----0----+----0
  56 1966 00 BELGIUM
 250 1965 40 FRANCE       MAKEMATRIX
```

Fig. 3.9: The SAF - Country Specification Card (one card per country)

The structure of a non-empty country block is:

1. At least one country specification card must be there. (See figure 3.9.)
2. Following the last (or only) country specification card there follows a separator.
3. The comment block could be maximal 20 cards long and 80 characters each card. However, you must not specify any comment. In this case two separator lines follow the last (or only) country specification card.

4. The aggregation scheme (See Chapter 3.1.2.4.) follow) for all the countries specified by the country specification cards.

There is no limitation to add country blocks and country cards.

3.1.2.4. The Aggregation Scheme

You can add, subtract or distribute (i.e. multiply by a constant) old rows or columns to new aggregated tables.

The syntax of aggregation scheme is defined in the following way, using a grammar similar to the Backus-Naur-Form (BNF):

Using the conventions:

BNF Variable	.....	identifier enclosed in brackets <and>
A :: = B	.....	A is defined by B
A.B	.....	A is followed by B
A/B	.....	'A or B'
A*	.....	'Zero or more occurrences of A'

The grammar is as follows:

<aggregation scheme> :: = (<scheme line>.<continuation line>*)*
<scheme line> :: = <new nr.>.<specifications>*
<continuation line> :: = <space>.<specifications>*
<specifications> :: = <spec.> / <spec.>.<distribute>
<spec.> :: = <delim.>.<old numbers>/<sign>.<old numbers>
<distribute>:: = '*'.<constant>
<old numbers> :: =<old nr.>/<range>
<range>:: =<old nr.>. ':'.<old nr.>

<delim.> :: = ' '/'?'
<sign>:: = '+'/'-'
<constant> :: = unsigned 4 digit number with an implicit assumption of 3 decimal positions

<space> :: = 4 blanks ' '

<new nr.> :: = 4 digit number right adjusted, specifying the new row - or column number which will be created out of <specifications>

<old nr> :: = 4 digit number right adjusted, specifying the old row - or column number in <specifications>

An example of an aggregation scheme is shown in figure 3.10.

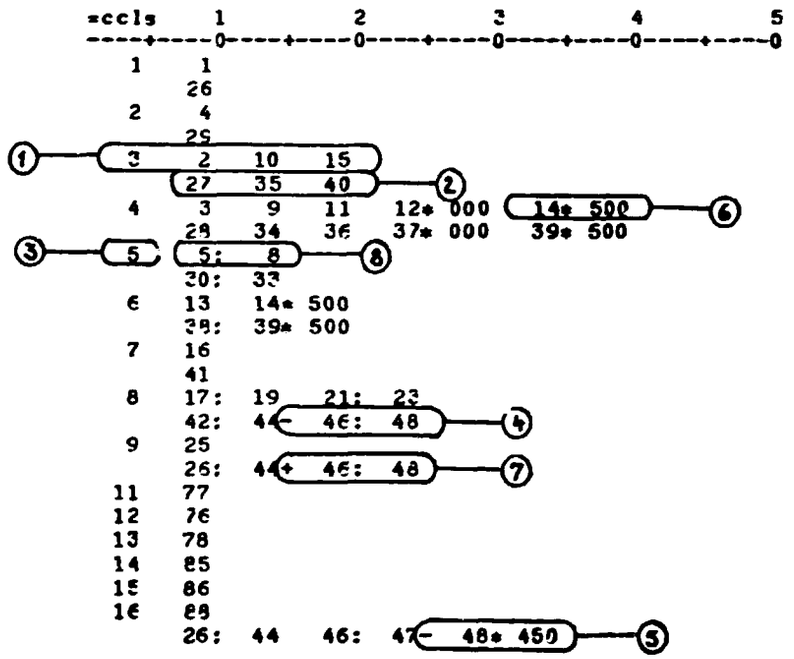


Fig. 3.10: Example of Aggregation Scheme

(1) is a <scheme line>

(2) is a <continuation line>

(3) is a <new nr>

(4) means that the old numbers 46 ÷ 48 will be subtracted from new number 8

(5) means that the old number 48 will be multiplied by the factor Ø.450 and will be then subtracted from new number 16

(6) means that old nr. 14 will be multiplied by a factor 0.500 and added to new row nr. 4.

(7) means that old numbers 46 ÷ 48 will be added to new number 9

(8) identifies a <range>

Remarks:

In this version there is a limitation of 15 numbers in one <scheme line> specification

3.2. How to Create a New Standard Aggregation Scheme File

1. Allocate a sequential fixed block dataset with a logical record size of 80 bytes. (For example SPF menue 3.2.)

The following steps could be completed in a text editor (for example SPF menue 2)

2. Enter the SAF director. (See Chapter 3.1.1.) Be aware that the type indicator is compatible with the previous defined aggregation types.

3. Enter a SAF type block (See Chapter 3.1.2.) by performing the following steps:

3.1. Enter type block header (See Chapter 3.1.2.1.).

3.2. Enter NUDID block (See Chapter 3.1.2.2.). Fill in all entries as discussed in the previous chapter.

3.3. Enter the country blocks as discussed in Chapter 3.1.2.3.

Now the new standard aggregation scheme file is ready for use with UNIOP.

For access to it in an UNIOP session either allocate it prior to the call of the UNIOP program or during the preparation phase of UNIOP (See Chapter 7: How to start an UNIOP SESSION).

3.3. How to Enter a New Aggregetion Type and/or Aggregation Scheme.

If you want to add a new aggregation type to the existing standard aggregation file do the following:

1. Add the new aggregation type in the SAF directory.

   Be aware that the type indicator is compatible with the previous defined aggregation types (See Chapter 3.1.1.)

2. Continue with topic 3 in Chapter 3.2.

If you want to add new aggregation schemes to an existing aggregation type act in the following way:

a) If you want to add a new table to an existing aggregation scheme (e.g. a standardized table or set of tables) do only the following step:

   Search the corresponding country block of the existing aggregation type and create a new country specification card. (See Chapter 3.1.2.3. figure 3.9.)

b) If you want to add a new aggregation scheme to an existing aggregation type do the following:

   1) Search the corresponding type block.

   2) Add a country block to the format type block as discussed in Chapter 3.1.2.3. and Chapter 3.1.2.4.

3.4. List of Available Aggregation Schemes and Types

On the original standard aggregation scheme file, 'UWM.ORIGINAL.IOTABLES.AGGREGA' there exists the following aggregation types with the corresponding indicators. (See figure 3.11.)

Correspondence Table for Standard Aggregation File:
'UWM.ORIGINAL.IOTABLES.AGGREGA'

Aggregation Type	Type Indicator	Meaning
UNITADR8	01	UNITAD 16 Rows aggregation (8 sectors)
UNITADC8	02	UNITAD 9 Columns aggregation (8 sectors)
UNIDOC15	03	UNIDO 15 Columns aggregation
TECHNOLC43	04	TECHNOLOGY 43 Columns aggregation
RSUMCHCK	05	Rowsum checks
CSUMCHCK	06	Columnsum checks
IDIOMR7	07	IDIOM 12 Rows aggregation (7 sectors)
IDIOMC7	08	IDIOM 8 Columns aggregation (7 sectors)
OECDR	09	OECD-ECE 88 Rows aggregation (3*25 sectors)
OECDC	10	CECD-ECE 36 Columns aggregation (25 sectors)

Fig. 3.11: Currently used Correspondences for SAF File 'UWM.ORIGINAL.IOTABLES.AGGREGA'

The following tables show the existing countries for each.
(See figure 3.12. - figure 3.15.)

UNITAIR8 - AGGREGATIONSCHEMES:

U.N. CODE	YEAR	TABLE VERS.	COUNTRY NAME	REMARKS
32	1963	0	ARGENTINA	
36	1974	0	AUSTRALIA	
40	1964	0	AUSTRIA	
50	1962	0	BANGLADESH	
56	1965	0	BELGIUM	
56	1970	0	BELGIUM	
68	1971	0	BOLIVIA	
76	1970	40	BRAZIL	ABSORPTION MATRIX
152	1962	0	CHILE	
170	1970	0	COLOMBIA	
178	1967	0	CONGO	
188	1966	0	COSTA RICA	
208	1970	0	DENMARK	
218	1963	0	ECUADOR	
250	1965	0	FRANCE	
250	1970	0	FRANCE	
280	1965	0	GERMANY	
280	1970	0	GERMANY	
288	1968	0	GHANA	
300	1970	0	GREECE	
320	1971	0	GUATEMALA	
356	1968	0	INDIA	
360	1971	0	INDONESIA	
364	1973	0	IRAN	PRELIMINARY
372	1965	0	IRELAND	
376	1972	0	ISRAEL	
380	1965	0	ITALY	
380	1970	0	ITALY	
384	1972	0	IVORY COAST	
384	1976	0	IVORY COAST	
392	1970	0	JAPAN	
404	1967	0	KENYA	
410	1970	0	KOREA	
450	1973	0	MADAGASCAR	
458	1965	0	WEST MALAYSIA	INCOMPLETE TABLE - DO NOT USE!!
466	1969	0	MALI	
484	1970	0	MEXICO	
528	1965	0	NETHERLANDS	
528	1970	0	NETHERLANDS	
554	1965	0	NEW ZEALAND	
588	1962	0	PAKISTAN	
598	1972	0	PAPUA NEW GUINEA	
604	1968	0	PERU	
608	1965	0	PHILIPPINES	
686	1959	0	SENEGAL	
716	1965	0	SOUTHERN RHODESIA	INCOMPLETE TABLE (FUELS MISSING)
702	1973	0	SINGAPORE	
724	1970	0	SPAIN	

Fig. 3.12: UNITADR8 Aggregation Schemes on SAF - Dataset 'UWM.ORIGINAL.IOTABLES.AGGREGA'

UNITADR8 - AGGREGATIONSCHEMES (CNT.):

U.N. CODE	YEAR	TABLE VERS.	COUNTRY NAME	REMARKS
158	1969	0	TAIWAN	
158	1971	0	TAIWAN	
834	1970	0	TANZANIA	
788	1968	0	TUNESIA	
826	1970	0	UNITED KINGDOM	
840	1967	0	UNITED STATES	
840	1972	40	UNITED STATES	ABSORPTION MATRIX
858	1961	0	URUGUAY	
894	1965	0	ZAMBIA	INCOMPLETE TABLE (DWELLING IS MISSING)

Fig. 3.12: UNITADR8 Aggregation Scheres on SAF - Dataset (CNT.) 'UWM.ORIGINAL.IOTABLES.AGGREGA'

UNITADC8 - AGGREGATIONSCHEMES:

U.N. CODE	YEAR	TABLE VERS.	COUNTRY NAME	REMARKS
32	1963	0	ARGENTINA	
36	1974	0	AUSTRALIA	
40	1964	0	AUSTRIA	
50	1962	0	BANGLADESH	
56	1965	0	BELGIUM	
56	1970	0	BELGIUM	
68	1971	0	BOLIVIA	
76	1970	40	BRAZIL	ABSORPTION MATRIX
152	1962	0	CHILE	
170	1970	0	COLOMBIA	
178	1967	0	CONGO	
188	1966	0	COSTA RICA	
218	1963	0	ECUADOR	
288	1968	0	GHANA	
300	1970	0	GREECE	
320	1971	0	GUATEMALA	
356	1968	0	INDIA	
360	1971	0	INDONESIA	
364	1973	0	IRAN PRELIMINARY	
376	1972	0	ISRAEL	
384	1972	0	IVORY COAST	
384	1976	0	IVORY COAST	
392	1970	0	JAPAN	
404	1967	0	KENYA	
458	1965	0	WEST MALAYSIA	INCOMPLETE TABLE DO NOT USE !
716	1965	0	SOUTHERN RHODESIA	INCOMPLETE TABLE (FUELS MISSING)
410	1970	0	KOREA	
466	1959	0	MALI	
450	1973	0	MADAGASCAR	
250	1965	0	FRANCE	
280	1965	0	GERMANY	
380	1965	0	ITALY	
528	1965	0	NETHERLANDS	
372	1969	0	IRELAND	
250	1970	0	FRANCE	
280	1970	0	GERMANY	
208	1970	0	DENMARK	
380	1970	0	ITALY	
528	1970	0	NETHERLANDS	
826	1970	0	UNITED KINGDOM	
484	1970	0	MEXICO	
554	1965	0	NEW ZEALAND	
586	1962	0	PAKISTAN	
598	1972	0	PAPUA NEW GUINEA	
604	1968	0	PERU	
608	1969	0	PHILIPPINES	
686	1959	0	SENEGAL	
702	1973	0	SINGAPORE	

Fig. 3.13: UNITADC8 Aggregation Schemes on SAF - Dataset 'UWM.ORIGINAL.IOTABLES.AGGREGA'

UNITADC8 - AGGREGATIONSCHEMES (CNT.):

U.N. CODE	YEAR	TABLE VERS.	COUNTRY NAME	REMARKS
724	1970	0	SPAIN	
158	1969	0	TAIWAN	
158	1971	0	TAIWAN	
834	1970	0	TANZANIA	
788	1968	0	TUNESIA	
840	1967	0	UNITED STATES	
840	1972	40	UNITED STATES	ABSORPTION MATRIX
858	1961	0	URUGUAY	
894	1965	0	ZAMBIA	INCOMPLETE TABLE (DWELLING MISSING)

Fig. 3.13: UNITADC8 Aggregation Schemes on SAF - Dataset (CNT.) 'UWM.ORIGINAL.IOTABLES.AGGREGA'

UNITADC15 - AGGREGATIONSCHEMES:

U.N. CODE	YEAR	TABLE VERS.	COUNTRY NAME	REMARKS

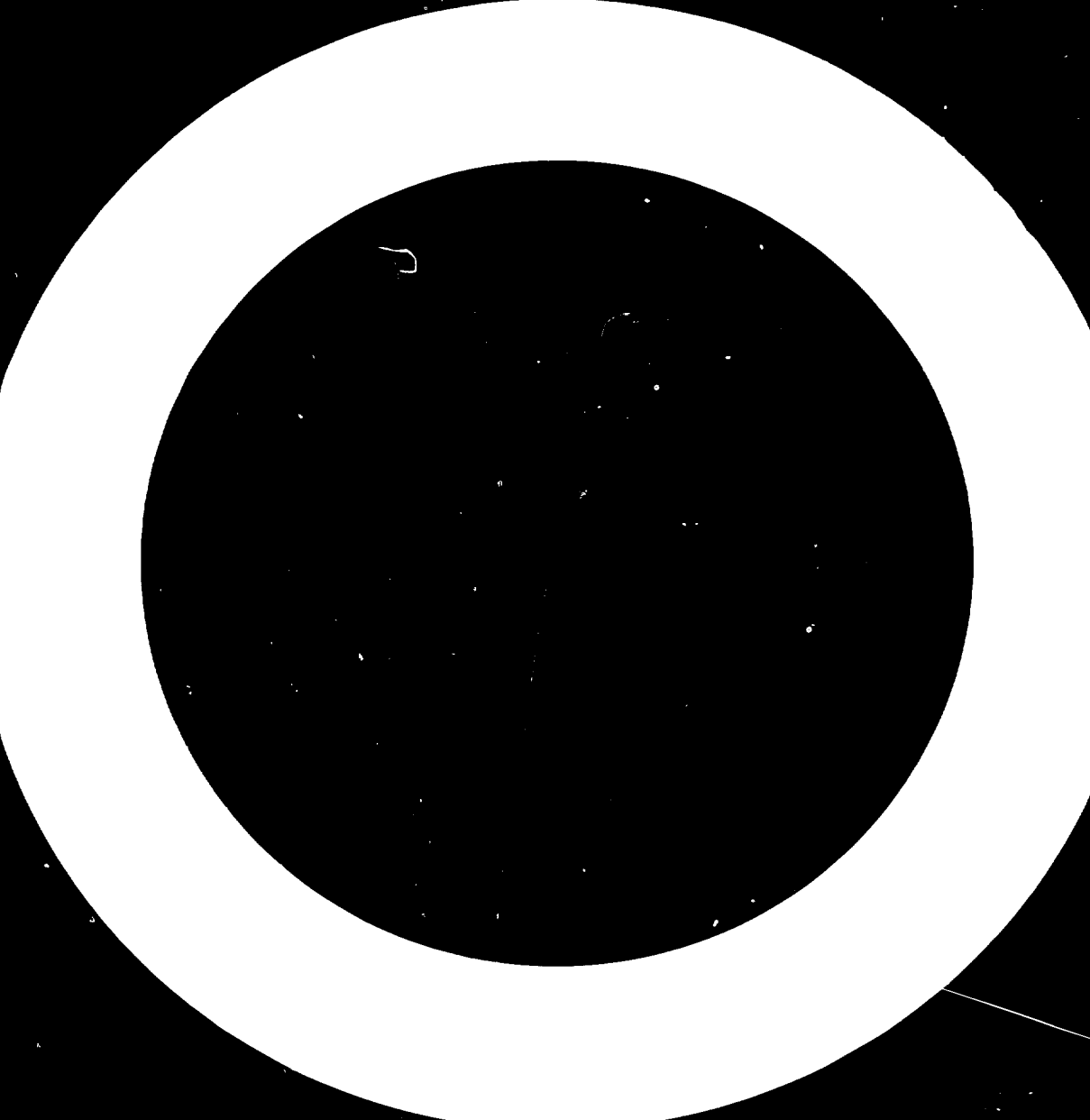
AT THE MOMENT NO AGGREGATIONSCHEMES IMPLEMENTED

Fig. 3.14: UNITADC15 Aggregation Schemes on SAF - Dataset 'UWM.ORIGINAL.IOTABLES.AGGREGA'

TECHNOLC43 - AGGRECATIONSCHEMES:

U.N. CODE	YEAR	TABLE VERS.	COUNTRY NAME	REMARKS
36	1974	0	AUSTRALIA	
40	1964	0	AUSTRIA	
50	1962	0	BANGLADESH	
56	1965	0	BELGIUM	
56	1970	0	BELGIUM	
76	1970	40	BRAZIL	ABSORPTION MATRIX
152	1962	0	CHILE	
188	1966	0	COSTA RICA	
208	1970	0	DENMARK	
250	1965	0	FRANCE	
250	1970	0	FRANCE	
280	1965	0	GERMANY	
280	1970	0	GERMANY	
300	1970	0	GREECE	
320	1971	0	GUATEMALA	
356	1968	0	INDIA	
360	1971	0	INDONESIA	
364	1973	0	IRAN	PRELIMINARY
372	1965	0	IRELAND	
380	1965	0	ITALY	
380	1970	0	ITALY	
404	1967	0	KENYA	
410	1970	0	KOREA	
463	1965	0	WEST MALAYSIA	
484	1970	0	MEXICO	
528	1965	0	NETHERLANDS	
528	1970	0	NETHERLANDS	
554	1965	0	NEW ZEALAND	
588	1962	0	PAKISTAN	
598	1972	0	PAPUA NEW GUINEA	
608	1969	0	PHILIPPINES	
716	1965	0	SOUTHERN RHODESIA	
724	1970	0	SPAIN	
788	1968	0	TUNESIA	
826	1970	0	UNITED KINGDOM	
840	1967	0	UNITED STATES	
894	1965	0	ZAMBIA	

Fig. 3.15: TECHNOLC43 Aggregation Schemes on SAF - Dataset 'UWM.ORIGINAL.IOTABLES.AGGREGA'



4. The UNIOP System

The UNIOP system is composed of two parts:

- the UNIOP program
- the UNIOP databank

The UNIOP system is part of the UNIDO input - output table databank. (See figure 4.1.)

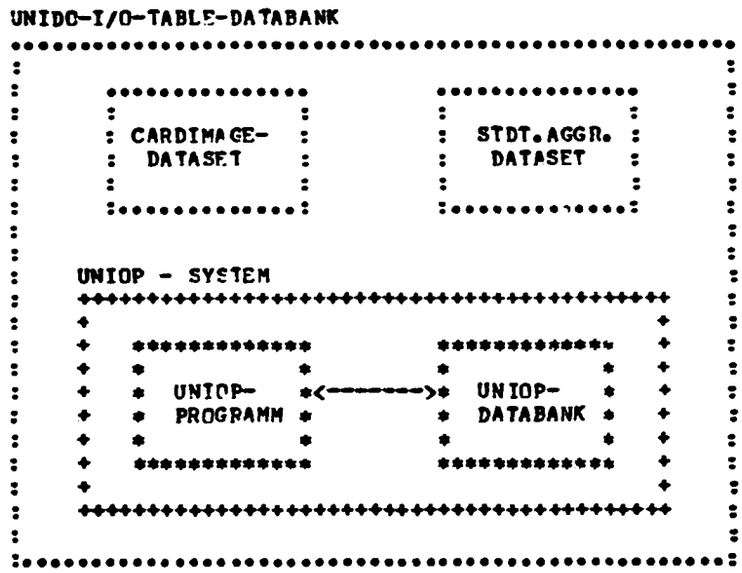


Fig. 4.1: Structure of the UNIOP - System

This chapter will discuss the following topics:

4.1. A short outline of the UNIOP program. A full description will be given in Chapter 5.

4.2. Description of the UNIOP databank.

- 4.2.1. Organization of UNIOP databank files
- 4.2.2. The MASTER file.
- 4.2.3. The TEXT file.
- 4.2.4. The DATA file.

4.3. Further improvements on the UNIOP databank.

4.1. Outline of the UNIOP Program

The UNIOP program has the following main purposes:

- Organization of the UNIOP databank.
- Editing and calculations of input - output tables.
  e.g. complete and check existing tables,

  create new versions of the original table,

  aggregate tables, compute inverse and coefficients,

  extract parts of existing tables and so on.
- Control the organization of the UNIDO input - output databank.
- Create links to make the input - output tables accessible for other purposes such as plotting, further use with other programs, etc.
- Perform utilities for the user, such as displaying desired tables on the terminal, testing the history of different tables, etc.
- Automatic updating of the "Numerical Data's Index and Description". (NUDID)

You can see the UNIOP programs as the core of the whole UNIDO input - output databank.

Chapter 5 will describe the UNIOP program in more detail.

4.2. Description of the UNIOP Databank

The UNIOP databank has two main uses:

i) As a work file for the UNIOP program,

ii) for collection and organization of original tables and versions of them.

The advantages of the UNIOP databank are:

1. It compresses all data and therefore saves a lot of storage,
2. easy access to single tables is possible,
3. deletion and overwriting of whole members or single tables in a set of tables could easily be made,
4. a correct bookkeeping on the tables is possible, so that you can trace all the computations you have made,
5. a number of generations can be made from the original published table.

A disadvantage of the UNIOP databank is that you can not have easy access to the tables from another program. To overcome this inconvenience, the UNIOP program has an interface which "punches" the members of the UNIOP databank in different formats on a standard PUNCH file (see WRITE command in chapter 6). The tables thus created could then easily be handled by another program.

If you decide to collect different tables on an external dataset, you can "punch" the wanted tables as described above, and afterwards delete it from the UNIOP databank.

4.2.1. Organization of UNIOP databank files

The UNIOP databank is composed of three direct access files, called:

a) MASTER file
b) TEXT file
c) DATA file

Figure 4.2. shows the links between these three UNIOP databank files.

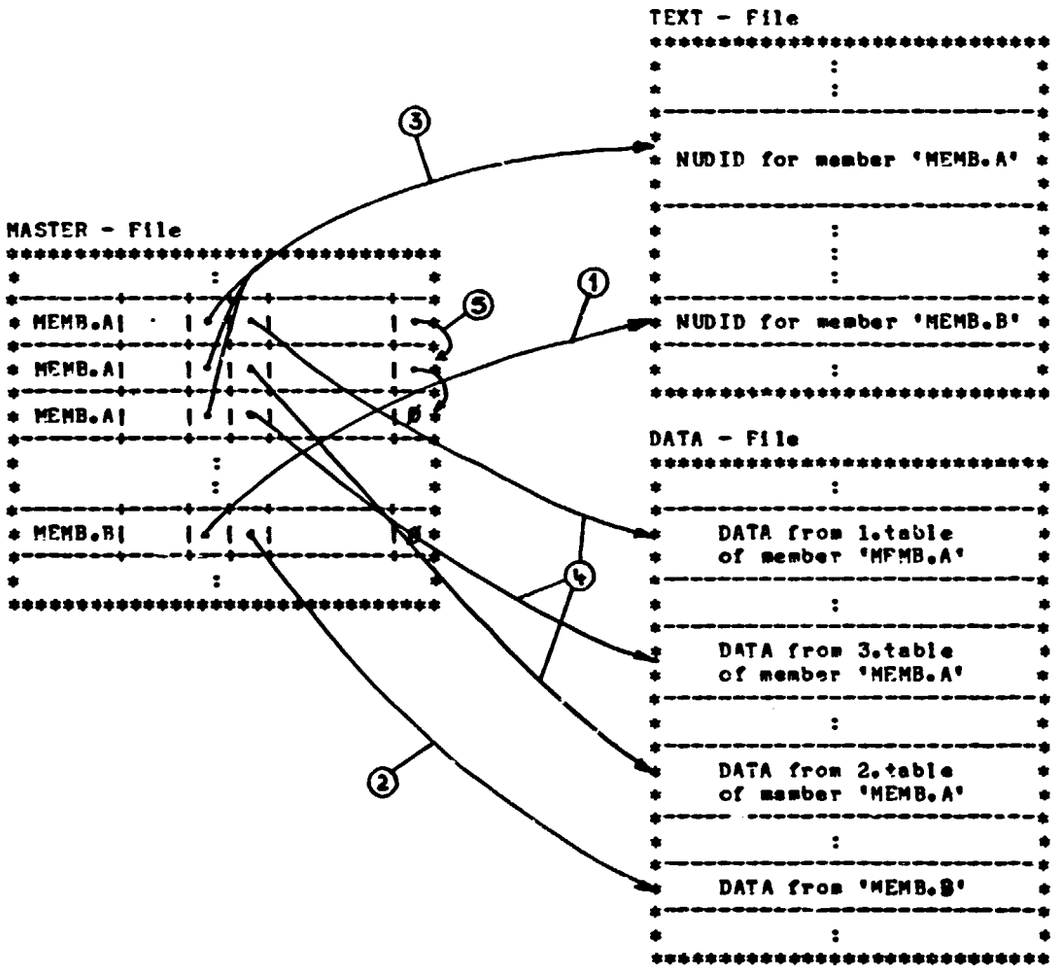


Fig. 4.2: Links between the UNIOP - Databankfiles

The main access to the members of the UNIOP databank is via the MASTER file. Here the most important information of the table is stored. Search information, system flags, trace information etc. are mainly found via this file.

Pointers in the MASTER file record link further information and data for each member:

a) Pointer to the TEXT file where the "Numerical data's Index and Description" (NUDID) is stored in a very compressed form. (See (1) in figure 4.2.)

b) Pointer to DATA file where the data is stored in a very packed form. (See (2) in figure 4.2.)

If a member is a set of tables, each single table has its own MASTER record in the MASTER file, so that you can access each single table.

In this case the pointer to the TEXT file for each single table of the set of tables is the same (See (3) in figure 4.2.). But since the data for each single table is different there is a different pointer to the DATA file in each single MASTER record. (See (4) in figure 4.2.)

The linking of the single MASTER records in a set of tables takes place via a continuation pointer (See (5) in figure 4.2.). So each single table is accessible. A member which is only a single table has only one MASTER record. (See MEMB. B in figure 4.2.).

In general the tables will have varying record numbers, not only on the DATA file but also on the TEXT file (because of packing of data).

Even the sequences of the blocks on the TEXT and the DATA files do not have to be the same as in the MASTER file (through deleting and overwriting of members).

These are two reasons amongst others to use direct access files.

4.2.2. The MASTER File

Access to the tables in the UNIOP databank is mainly through the MASTER file. Here a short description of the tables is found, so that you can search easily for members in the UNIOP databank. The table

name, pointers (links) to the other UNIOP databank files, system flags etc. can be found there.

For each member in the UNIOP databank there exists:

a) For members which are single tables a single record on the MASTER file,

b) for members which are sets of tables a linear list of single records on the MASTER file.

Record structure

The records have a record length of 120 bytes, are organized sequentially in this version and have a fixed structure. Figure 4.3. shows the record structure and gives a description of the contents of the record. All records are in binary format.

Byte No.	Meaning	Type
1 - 24	MEMBERNAME	HOLLERITH
25 - 26	U.N.-COUNTRY CODE	INTEGER*2
27 - 28	YEAR	INTEGER*2
29 - 38	INVENTORY NUMBER OF DATASET	HOLLERITH
39 - 40	VERSION NUMBER OF DATASET	INTEGER*2
41 - 44	HISTORY FLAG	INTEGER*4
45 - 48	SYSTEM FLAG	INTEGER*4
49 - 52	AGGREGATION-TYPE FLAG	INTEGER*4
53 - 60	TRANSACTION TYPE	INTEGER*2
61 - 68	UNUSED	
69 - 70	DIMENSION OF MEMBER	INTEGER*2
71 - 72	NUMBER OF ROWS	INTEGER*2
73 - 74	NUMBER OF COLUMNS	INTEGER*2
75 - 76	NUMBER OF TABLES	INTEGER*2
77 - 82	UNUSED	
83 - 84	MASTER-RECORD-NUMBER OF CONTINUATION	INTEGER*2
85 - 86	NUMBER OF RECORDS ON TEXT-FILE	INTEGER*2
87 - 88	NUMBER OF RECORDS ON DATA-FILE	INTEGER*2
89 - 96	DATE OF LAST USE	HOLLERITH
97 - 104	DATE OF IMPLEMENTATION	HOLLERITH
105 - 108	USER	HOLLERITH
109 - 112	POINTER TO TEXTFILE	INTEGER*4
113 - 116	POINTER TO DATAFILE	INTEGER*4
117 - 120	MASTER-RECORD-NUMBER OF PREDECESSOR	INTEGER*2

Fig. 4.3: Record Structure of MASTER-File

Limitations

In this version a maximum of 125 records can be stored on the MASTER file. However, this limitation could be relaxed in the future;

the existing FORTRAN compiler must know the record description in the OPEN statement of a D.A. file at compile time.

Further improvements

If we decide to store all tables in the UNIOP databank it would be useful to organize the MASTER file as a binary tree, to make searching more efficient.

Another possibility would be to organize the MASTER file as a sorted list. Other search criteria, such as region codes etc. could also be implemented.

4.2.3. The TEXT File

Here all significant NUDID information is stored. Since you can look at NUDID as a sparse occupied two dimensional array you can address each entry by an index.

Any entry which is a blank character string or zero number would not be stored on the TEXT file. In this way the space for storing a NUDID would be significantly reduced.

A pointer reference from the MASTER record makes each NUDID block addressable for the UNIOP program. Members which are sets of tables have only one NUDID block assigned.

Record structure

The records have a record length of 80 bytes. A NUDID block will vary in the number of records. Figure 4.4. gives you some information about the record structure. All records are in binary format.

No. of Bytes	Meaning	Type
4	NO. OF RECORDS FOR THIS NUDID-BLOCK	INTEGER*4

FOR ALL NON-BLANK CHARACTERSTRINGS

No. of Bytes	Meaning	Type
2	INDEX	INTEGER*2
2	LENGTH L OF CHARACTERSTRING	INTEGER*2
L	CHARACTERSTRING OF LENGTH L	HOLLERITH

FOR ALL NON-ZERO SINGLE NUMBERS

No. of Bytes	Meaning	Type
2	INDEX	INTEGER*2
2	NUMBER	INTEGER*2

FOR ALL NON-ZERO RANGES OF NUMBERS:

No. of Bytes	Meaning	Type
2	INDEX	INTEGER*2
2	NO. OF RANGES L	INTEGER*2
L times:	L times:	
2	FROM - NUMBER	INTEGER*2
2	TO - NUMBER	INTEGER*2

FOR ALL NON-ZERO LISTS OF SINGLE NUMBERS:

No. of Bytes	Meaning	Type
2	INDEX	INTEGER*2
2	NO. OF SINGLE NUMBERS L	INTEGER*2
L times:	L times:	
2	NUMBER	INTEGER*2

AT THE END OF THE RECORD:

No. of Bytes	Meaning	Type
8	SCALE	REAL*8
8	MISSING VALUES	REAL*8
8	EXPONENT	REAL*8

Fig. 4.4: Record Structure of TEXT-File

Limitations

In this version of UNIOP a maximum of 4750 records can be stored on the TEXT file. This limitation could be easily expanded. The limitation is due to the FORTRAN compiler which must know the record description of a D.A. file in the OPEN statement at compile time.

4.2.4. The DATA File

Here the data proper of the input - output tables is stored. The data will be stored in the most compressed form, according to the following criterias: (See also figure 4.5.)

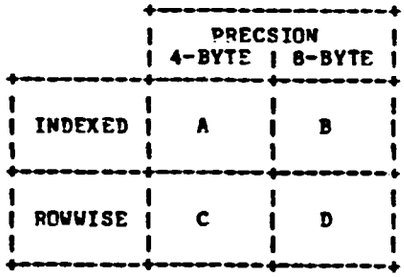


Fig. 4.5: Modes of storing Data on DATA-File

1. In single precision (i.e. 1 word packed decimal, FORTRAN REAL format) if the maximal and minimal figure is less than or equal to 7 significant digits.
2. In double precision (i.e. 2 words expanded packed decimal, FORTRAN DOUBLE PRECISION format) if the maximal and minimal figure is greater than 7 significant digits.
3. Indexed-wise storing of non zero elements if the number of necessary bytes would be less than the numbers of bytes necessary for row-wise storing of all figures.
4. Row-wise storing of all figures (if not 3).

For each single table there corresponds one data block (even for members which are set of tables). It means that each table will be stored separately on the DATA file.

Record structure

The records have a record length of 80 bytes. A data block (i.e. data of each single table) will vary in the number of records.

A flag in the MASTER record of the corresponding table will be set according to the storing modes. (A) ÷ (D) (see figure 4.5.).

Figure 4.6. gives information about the record structure. Each data block has a prolog of 20 bytes which is independent of the storing mode.

PROLOG OF DATA-BLOCK:

No. of Bytes	Meaning	Type
4	NO. OF PHYSICAL RECORDS FOR THE TABLE	INTEGER*4
8	MAXIMUM FIGURE IN TABLE	REAL*8
8	MINIMUM FIGURE IN TABLE	REAL*8

MODE A :

No. of Bytes	Meaning	Type
repeat:	repeat:	
| 2	| ROW INDEX-NO. OF NON-ZERO ROW	INTEGER*2
| repeat:	| repeat:	
| | 2	| | COL. INDEX-NO. OF NON-ZERO ELEMENT	
| |	| | IN ROW	INTEGER*2
| | 4	| | NON-ZERO ELEMENT IN ROW	REAL*4
| +-> 2	| +-> = 0 ... END-OF-ROW INDICATOR	INTEGER*2
+-----> 2	+-----> = 0 ... END-OF-TABLE INDICATOR	INTEGER*2

MODE B :

No. of Bytes	Meaning	Type
repeat:	repeat:	
| 2	| ROW INDEX-NO. OF NON-ZERO ROW	INTEGER*2
| repeat:	| repeat:	
| | 2	| | COL. INDEX-NO. OF NON-ZERO ELEMENT	
| |	| | IN ROW	INTEGER*2
| | 4	| | NON-ZERO ELEMENT IN ROW	REAL*8
| +-> 2	| +-> = 0 ... END-OF-ROW INDICATOR	INTEGER*2
+-----> 2	+-----> = 0 ... END-OF-TABLE INDICATOR	INTEGER*2

MODE C :

No. of Bytes	Meaning	Type
SIZE times:	SIZE times:	
4	ELEMENTS IN ROW-MAJOR FORM	REAL*4

SIZE = NO. OF ROWS * NO. OF COLUMNS

MODE D :

No. of Bytes	Meaning	Type
SIZE times:	SIZE times:	
8	ELEMENTS IN ROW-MAJOR FORM	REAL*8

SIZE = NO. OF ROWS * NO. OF COLUMNS

Fig. 4.6: Record Structure of DATA-File

Limitations

In this version of UNIOP a maximum of 5040 records could be stored on the DATA file. This limitation could easily be relaxed. (The limitation is a consequence of the FORTRAN compiler.)

4.3. Further improvements on the UNIOP databank

The current size of the databank files is not an optimal one. A better proportion between data set size and record numbers could be found.

There could be more detailed discussion about the links between the UNIOP databank and the cardimage master file.

5. The UNIOP Program

This chapter will discuss the following topics in more detail.

5.1. General features of the UNIOP program.
5.2. Structure.
5.3. Proposals for further improvements.

At the outset it must be noted that the current version of the UNIOP program is far from being complete. In particular a processor for input-output analyse is not programmed for the current version. (See specially Chapter 5.3.)

5.1. General features of the UNIOP program

The UNIOP program is an interactive FORTRAN-IV program. The current version is mainly for an interactive use at the terminal. Some of the commands, the most important, could also be used in a batch job.

The programm is part of the UNIOP system (See figure 5.1.)

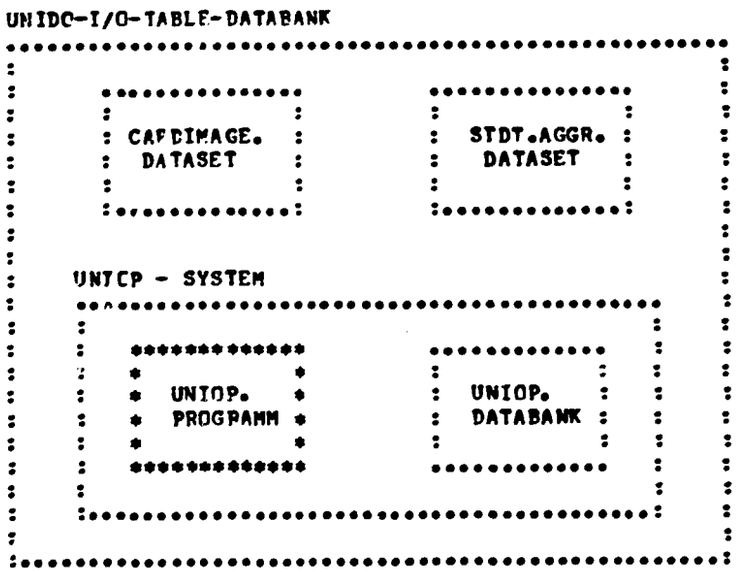


Fig. 5.1: Position of the UNIOP-Program in the UNIDO-Input-Output-Table Databank

The program is mainly orientated to handle single input - output tables (i.e. members consisting of single tables or set of tables).

Since it is often desirable to make use of the same command sequence to different members it could be useful' to implement a command where such 'looping' is possible. (See also Chapter 5.3.)

In the current version of the UNIOP program there exists a restrictive loop command (see also Chapter 6.) which handles the following 'loop'.

For all assigned card image file members:

a) Read the member.
b) Print the member.
c) Save the member in the UNIOP databank with a default name.

In the current version of the UNIOP system it is not possible to handle input - output models. For a discussion of the problem and possible solutions see Chapter 5.3.

The UNIOP program controls:

a) All data handling, organization and management
   . with the external environment
   . with the UNIOP databank
b) standard computation for input - output analysis
c) editing and changing of individual input - output tables.

The user controls all operations he wants to carry out with an own command language. (See Chapter 6.)

One of the aims was to make the working with the UNIOP program as easy as possible for the user. This could be realized by the following:

- In most cases abbreviations could be used and all input from the terminal is in a free format, so that the effort of typing is as little as possible.
- The syntax for the commands is very easy and the user can always get the necessary information about the use of the single commands by an HELP command (see Chapter 6.) so that he does not have to remember too much information.
- Self explanatory and extensive error messages prevents the user from misuse.
- A mixture of sequential proceeding (e.g. like TSO) and "full screen support" (e.g. like SPF) makes the working with the program easier.
- Utility commands give a lot of information about the single tables. (e.g. the LIST, BROWSE and PRINT command, see Chapter 6.)

The demand for variability of the UNIOP program could be realized by creating an interface between the UNIOP system external environment:

- The READ command to transfer input - output tables from the external environment to the UNIOP system. (See also Chapter 6.)
- The WRITE command to transfer an input - output table from the UNIOP system to the external environment, so that the input - output tables is accessible to other programs. (See also Chapter 6.)

For the current version there exists two formats for such a transfer:

a) the UNIDO standard input - output table format, (See Chapter 2.)

b) a more simpler rowwise storing of the whole single table. (See also WRITE command in Chapter 6.)

Further improvements could be done be creating e.g. an interface to SAS for handling tables in models or for transfer of single features (e.g. row or column vectors, submatrices or single elements). For more detail see Chapter 5.3.

Another important aspect of the requirements to the UNIOP program is to keep trace of all changes in the input - output tables.

- The program updates all entries within the "Numerical Data's Index Description" (NUDID) which were altered by any change of input - output table.

  The aim is, that no further editing is necessary by the user for further generations of the original tables. This feature makes working with the UNIOP program very efficient

- The program also remembers the predecessor of each member in the UNIOP databank and bears in mind the commands which modified the table. Therefore, illegal operation could be avoided. (e.g. To attempt to aggregate one table twice with the same aggregation type.) This feature helps the user to glance over the operations performed on the tables. (See also LIST command in Chapter 6.)

An important aspect of the UNIOP program is the administration of the UNIOP databank. You create copies of existing members or new members (in this version only by using the PUT command), delete (by the DELETE command) overwrite existing members (by the UPDATE command) or compress and re-organize the UNIOP databank (by the PACK command). You can also rename existing members in the UNIOP databank by the CHANGE command. (See Chapter 6 for information on how to use these features.)

The user has a mighty instrument at his disposal to create and save modified generations of an original input - output table in the UNIOP databank.

The next main aspect of the UNIOP program is to make standard calculations for input - output analysis possible.

The current version places two calculations at the users disposal, i.e.:

- Aggregation by means of the AGGREGATE command.

  You can aggregate row or column wise input - output tables according to a predefined standard aggregation scheme.

  Adding, subtracting and multiplying by a constant of rows or columns is possible.

  For further information see Chapter 6. (AGGREGATE command) and Chapter 3.

- Computing of input or output coefficients by means of the COEF command.

  You can calculate shares out of flows by either specifying the reference row or column or by taking 'total input use row' as default assumption. (See also COEF command in Chapter 6.)

In any case the NUDID of the member will be updated to the new specifications, so that no further editing is necessary.

It is easy to implement further calculations for input - output analysis. For example:

- Calculating backward and forward linkages,
- inverting (especially Leontief inverse),
- input - output updating techniques like simple RAS, modified RAS, updating by Leontief inverse,
- calculation of 'most important' coefficients,
- diagonalization and triangularization,
- eigen values and eigen vectors,
- calculation of determinant.

For more information see Chapter 5.3.

Since it is often necessary to modify an existing input - output table another important feature of the UNIOP program is an input - output table editor which makes the following handling possible.

a) insertion of rows, columns and elements
b) deletion of rows, columns and elements
c) overwriting of rows, columns and elements
d) adding or subtracting of rows, columns and elements
e) multiplication by scalars, vectors and sub matrices
f) elementwise division, etc.

The EDITOR - processor will be discussed in more detail in Chapter 6. (See EDITOR Command.)

5.2. Structure of the UNIOP program

Figure 5.2. gives a rough impression of the flow chart of the UNIOP program.

A main program decodes the command string and branches to the corresponding sub-programs. The FINISH command terminates the UNIOP program. A discussion of the function and use of the single commands can be found in Chapter 6.

The following are notes regarding storage requirements:

Since FORTRAN does not support a dynamic storage allocation and the table size may differ considerably, the following procedure will be performed by the UNIOP program to bring the tables into the core:

If the table size (i.e. number of rows * number of columns) does not exceed a defined size, then the table will be mapped into the core on a one dimensional array. For larger tables the program will use a direct access file to store the table in row major form. A buffer will make the working with this direct access file more efficient. By setting a parameter variable in the main program one can influence the decision of using the core array or the direct access file.

If the member is a set of tables, they will be processed one by one. Therefore only the table which has just been processed will be accessible to the program and brought into the core.

As discussed in Chapter 4 the access to the members of the UNIOP databank will be done via links in the MASTER file record of the corresponding table.

The GET command (see also Chapter 6) is used to get a member into the core. If the member is a set of tables only the first will be loaded. The others will be loaded whenever they will be used by

the corresponding subprogram. In consequence of what has been said, a set of tables will always create a new member if some modification (e.g. by the AGGREGATE command is carried out).

This procedure does not seem very elegant, since it violates the concept of storing only those members into the UNIOP databank, that the user wants. Therefore, a temporary storage of sets of tables would be useful. (See also Chapter 5.3.)

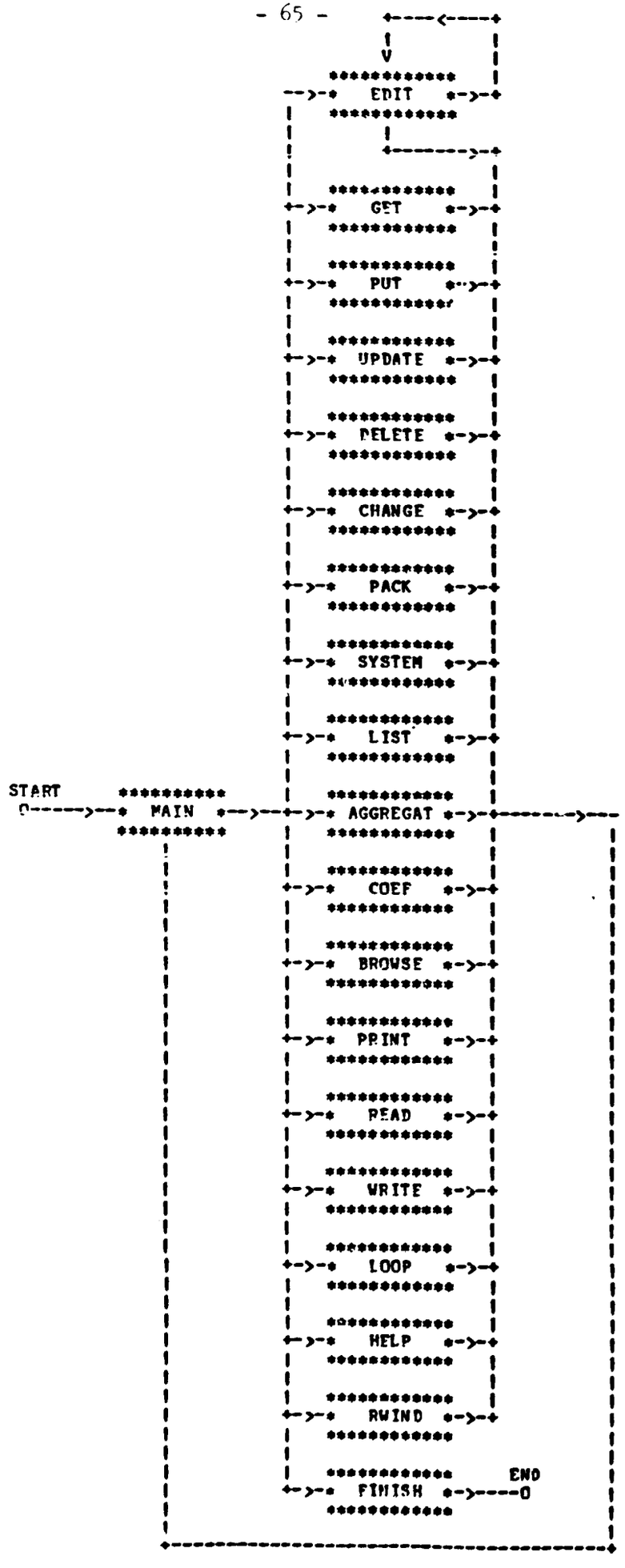


Fig. 5.2: Structure of the UNIOP-Programm

5.3. Proposals for further improvements of the UNIOP program

In this chapter mainly new features of the UNIOP program will be discussed. In Chapter 6 you can find improvements of existing features.

5.3.1. Editing of the NUDID by the UNIOP-EDITOR

Although the intention is, to update the NUDID automatically by the program, it is not possible to keep trace of all possible modifications by the EDITOR. (e.g. To insert command text, change title of the input - output table, etc..)

An explicit editing of NUDID would, therefore, be convenient for the user.

5.3.2. Overlay structure of the UNIOP program

Since the UNIOP program has an hierarchical structure it is easy to segmentate the program. To minimize the core size this improvement would be highly desirable. An efficient solution between unloading of segments and core size requirements must be found. Some effort is necessary to implement an efficient overlay structure.

5.3.3. Batch version

Since it would be desirable to use the UNIOP program, not only interactively, but in a batch job, a smaller batch version (not including the utility commands) would be useful especially since the most important commands of the current version work also in this mode.

5.3.4. Dynamic allocation of cardimage and standard aggregation (SAF) files

The structure of the UNIDO input - output databank would be improved and come into play, if a dynamic allocation of external datasets from the UNIOP program could be implemented.

Since FORTRAN does not have such a feature probably an assembler routine could solve the problem.

5.3.5. Improvement of the preparation phase of the UNIOP session

The preparation phase of an UNIOP session will be controlled by a CLIST procedure. An improvement especially in assigning the cardimage files and standard aggregation files could be made.

5.3.6. Plotter interface for plotting tables

For publishing reasons it would be useful to plot the tables at the plotter. Another command could be implemented to create a dataset which controls the plotting. This file could then be routed in the closing phase of an UNIOP session to the plotter.

5.3.7. Interface with SAS

Since the SAS package is a mighty instrument to handle models it would be useful to create an interface between SAS and the UNIOP system. An outline of such an interface should be discussed in more detail.

A possible way would be to create first a SAS readable version of the tables the user wants to handle with SAS, then leave the UNIOP program and enter SAS and after handling the model re-entering the UNIOP program if desired.

5.3.8. Models in the UNIOP system

The UNIOP system is mainly orientated to handle single tables. In this version there exists no possibility to access to the input - output tables exogenous variables e.g. price vectors, etc.

The possibility to do this should be discussed in more detail. (See topic 5.3.7. above.)

5.3.9. Implementation of a LOG file

This feature should write a log book of all manipulations done in an UNIOP session. It should help the user to identify the calculations and manipulations done during a session.

5.3.10. COPY command

The implementation of a COPY command would be useful especially to create sets of tables out of single tables (e.g. standardized tables) and vice versa. This feature could easily be incorporated in the existing structure.

5.3.11. LOOP command

The existing LOOP command is provisional. The idea is to handle the same command sequence for more than one table. To do this, a buffer must save the commands and a well defined end condition for the loop must be found. (e.g. Creating masks for members names, or searching of defined U.N. country codes, years, etc.)

Another way would be to define UNIOP-Macros which could be saved temporarily or permanently on a dataset. A reference to such a macro would then handle the defined command sequence.

5.3.12. Missing values handling

In the current version of UNIOP program there exists no handling of missing values. The definition of missing values happens by an entry in NUDID. The idea is to omit all values less equal to the defined one from calculation.

In any case a switching off of the missing values handling should be possible. (By setting an option in the corresponding commands.)

5.3.13. Implementation of further calculation commands

Since the structure of accessing members of the UNIOP databank is well defined it would be relatively easy to implement further commands for standard input - output analysis. For example:

- Inverting (especially creating Leontief inverse),
- input - output updating (by different updating techniques like simple RAS, updating with Leontief inverse),
- finding maximum and minimum rows, columns and or elements in an existing table,
- eigenvalue and -vector calculations,
- diagonalization and triagularization of input - output tables,
- calculation of forward - and backward linkages.

5.3.14. Temporary storage of set of tables (See also Chapter 5.2.)

As discussed above a temporary storage of members would be useful. This could be done in the concept of the existing UNIOP system, but some modifications must be carried out.

5.3.15. Adoption of UNIOP program to the FORTRAN V compiler

Since the new FORTRAN V compiler has a lot of advantages against the current compiler (especially a dynamic allocation of datasets) it would be useful to adapt the program to the new compiler.

6. The UNIOP Commands

The format of an UNIOP command is generally as follows:

command	option

- command - specifies the desired operation, could be abbreviated up to two characters
- option - supplementary specifications for the command. Not every command needs an option.

You can enter the UNIOP commands in a free format. This means you can separate each word by more than one blank and you need not start the command in the first column. In the most cases the options are not positional in the option string.

Common delimeters are blanks and comma (,). A semi colon (;) at the end of a line indicates that the option string will continue at the next line. You can type a maximum of 240 characters. The option string will be packed, so that all redundant blank characters will be omitted by the system.

The Syntax rules as given below apply to the following chapters as well:

- Optional options are indicated by a set of brackets (<) and (>) e.g.<LIST>.
- Exclusive choices are indicated by a slash (/).
- Mutually exclusive choices are indicated by 'OR'.
- User supplied values are shown in lower case letters.
- Words in upper case letters are key words and should be entered as shown. In most cases abbreviations are allowed.

You can divide the UNIOP commands into the following groups:

1. Commands for communication with the, to UNIOP external environment:

   READ Reads input - output table(s) from external card-image file.

WRITE	Writes input - output tables onto external dataset.
LOOP	Reads and prints all assigned input - output tables.

2. Commands for organizing the UNIOP databank.

GET	Get existing member from UNIOP databank.
PUT	Saves member into the UNIOP databank.
UPDATE	Overwrites an existing member in UNIOP databank.
CHANGE	Rename members in UNIOP databank.
DELETE	Delete members in UNIOP databank.
PACK	Compress and re-organize UNIOP databank.
SYSTEM	Handles system function (for databank manager only).

3. 'Computational' commands for input - output tables.

EDIT	Editor processor.
AGGREGAT	Aggregates input - output tables
COEF	Computes coefficients for input - output tables.

4. Utitilities.

BROWSE	Displays input - output tables and NUMID on terminal.
PRINT	Prints input - output tables
LIST	Gives different information about members in UNIOP databank.
HELP	Displays help information on the terminal.

5. Others.

REWIND	Rewinds specified dataset.
FINISH	Terminates UNIOP session.

In the following, all available UNIOP commands will be discussed in alphabetical order.

Each paragraph will be divided into the following topics:

i) Function of the command.
ii) Syntax of the command.
iii) Description of the operations.
iv) Supplementory explanations (if necessary).
v) Hints for the databank manager (not in all paragraphs).
vi) Proposals for further improvements.

6.1. AGGREGAT - Command

Function:

Aggregates input - output tables. The aggregations schemes must be defined on an external dataset.

Syntax:

```
AGGREGAT  member-name TYPE=agtyp <,LIST> <,PRINT> <,LUN=log-unit>
```

Alias	-	None
Required	-	'member-name' / '*' TYPE=agtyp
Defaults	-	LUN=25 NOLIST NOPRINT
Note:	-	NOLIST and NOPRINT should not be used as options.

Operands:

Member-name	-	User supplied membername for aggregated table in UNIOP databank or:
'*'	-	Uses currently active membername for aggregated table positional option.
TYPE	-	Keyword for specifying aggregation type.
agtyp	-	Aggregation scheme type.
LIST	-	Displays aggregation scheme on terminal.
PRINT	-	Prints aggregation scheme on standard printfile.
LUN	-	Keyword to specify another logical file number for another aggregation scheme dataset.
log-unit	-	Logical filenumber in the range of 25 ... 29.

Supplementary Explanations:

1. You can add, subtract and distribute (i.e. scalar multiplication by a real constant) either rows or columns in one step with the AGGREGAT command.

2. In this version the Aggregation schemes must be predefined on an external sequential dataset in a standard format. (See Chapter 3. Standard aggregation scheme file.)

3. In the preparation phase of an UNIOP session (see Chapter 7) you will be prompted by the system for allocation of a standard aggregation scheme file.

   By default this file will have the logical unit number LUN = 25 (and therefore the file name FT25F001)

4. The 'LUN = log-unit' Option:

   You can use alternative aggregation scheme files together with the standard aggregation scheme file. Those alternative files must have standard format.

   With the 'LUN=log-unit' option of the AGGREGAT command you can have access to the schemes of the alternative files in the UNIOP system.

Example: Additional aggregation schemes were written in standard format on two sequential datasets called 'UWM. AGG1' and 'UWM. AGG2'.

To use these schemes in an UNIOP session you have to type the following statements before you start an UNIOP session.

```
(1)  ALLOCATE DSN('UWM. AGG1') FILE(FT26FØØ1) SHR
(2)  ALLOCATE DSN('UWM. AGG2') FILE(FT27FØØ1) SHR
(3)  EXEC UNIOP
```

Remarks: Step (1) allocates the logical unit number LUN = 26 to the dataset 'UWM. AGG1'

Step (2) allocates the logical unit number LUN = 27 to the dataset 'UWM. AGG2'

Step (3) calls the UNIOP program starting with the preparation phase.

If you have entered the UNIOP program you can now use these by specifying the 'LUN = log-unit' option:

AGGREGATE tablename, LUN=26, TYPE=type

and

AGGREGATE tablename, TYPE=type, LUN=27

5. The 'TYPE=type' option:

This option must be specified. It defines the type of aggregation and must be in the directory of the aggregation scheme file. (See Chapter 3 for a list of types). A flag will be set for each aggregation type. If the table is already aggregated with the same type, the system will prompt you.

6. The aggregation scheme file will be searched in the following way:

i) Search directory of standard aggregation file (see Chapter 3) for desired aggregation type; if found then,

ii) search in the corresponding type block the scheme according to the following search criterias (See Chapter 3.).

a) U.N. country code

b) year

c) version

If no scheme was found, the system will prompt you with a message.

7. The 'LIST' option.

This option displays the aggregation scheme at your terminal (See figure 6.1.)

```
UNITADR8   .- Aggregationsceme for Table: KENYA.TEST.R8
              U.N.Code: 404, Year: 1967, Version:  0
Table aggregated cut of Databanktable: KENYA.TEST
Comment: ROWS    DOMESTIC AND IMPORT TRANSACTIONS INSERTED BLANK EXCEPT DIAGONAL
     New row     Nr.:   1 AGRICULTURE                      created out of
           1. old row    Nr.:   1 AGRICULTURE
           2. old row    Nr.:  26 AGRICULTURE
     New row     Nr.:   2 AGRO-FOOD                        created out of
           1. old row    Nr.:   4 FOOD, BEVERAGES & TOBACCO
           2. old row    Nr.:  29 FOOD, BEVERAGES & TOBACCO
     New row     Nr.:   3 ENGERGY AND PETROLEUM PROD.      created out of
           1. old row    Nr.:   2 COAL,CRUDE PETROLEUM,NAT.GAS
           2. old row    Nr.:  10 PETROLEUM AND COAL PRODUCTS
           3. old row    Nr.:  15 ELECTRICITY, GAS & STEAM
           4. old row    Nr.:  27 COAL,CRUDE PETROLEUM,NAT.GAS
           5. old row    Nr.:  35 PETROLEUM AND COAL PRODUCTS
           6. old row    Nr.:  40 ELECTRICITY, GAS & STEAM
     New row     Nr.:   4 BASIC PRODUCTS                   created out of
           1. old row    Nr.:   3 OTHER MINING AND QUARRYING
           2. old row    Nr.:   9 CHEMICALS
           3. old row    Nr.:  11 NON-METALLIC MINERAL PRODUCT
           4. old row    Nr.:  12         EMPTY
              Old row    distributed to  50.0%
```

Fig. 6.1: Example of LIST-Option Messages

8. The 'PRINT' option

   This option prints the aggregation schemes on the standard print file.

9. The UNIOP system will automatically update the Numerical Data's Index and Description (NUDID).

10. If you aggregate the same table row and columnwise, it would be more efficient to start with the ROW aggregation.

Further Improvements

1. For the user it will be convenient to know which aggregation types and aggregation schemes are defined on the external dataset. This could be easy implemented via the LIST command.

2. It will be useful to implement and feature, doing the following:

   a) change temporary existing aggregation schemes,
   b) add temporary or permanently aggregation schemes during an UNIOP session.

3. Some improvements in assigning the aggregation files in the preparation phase of the UNIOP sessions are possible.

6.2. BROWSE command

Function:

Displays currently active member on terminal:

- input - output table
- Numerical Data's Index and Description (NUDID) only

In both modes scrolling is possible.

Syntax:

```
BROWSE <NUDID>
```

Alias - None

Required - None

Defaults - Browsing of input - output table if no option is specified.

Operands:

NUDID - Browsing of Numberical Data's Index and Description only.

For HELP information about meaning of PF-keys if you are in BROWSE mode.

Supplementary Explanations

1. The BROWSE command displays the current active table on the terminal. The command makes use of the full screen support and should not be used in batch jobs.

There exists two modes:

Mode A Browsing of Numberical Data's Index and Description

Mode B Browsing of single or set of input - output table

Mode A could be entered:

i) directly via the option NUDID. In this case no access to the input - output table is possible

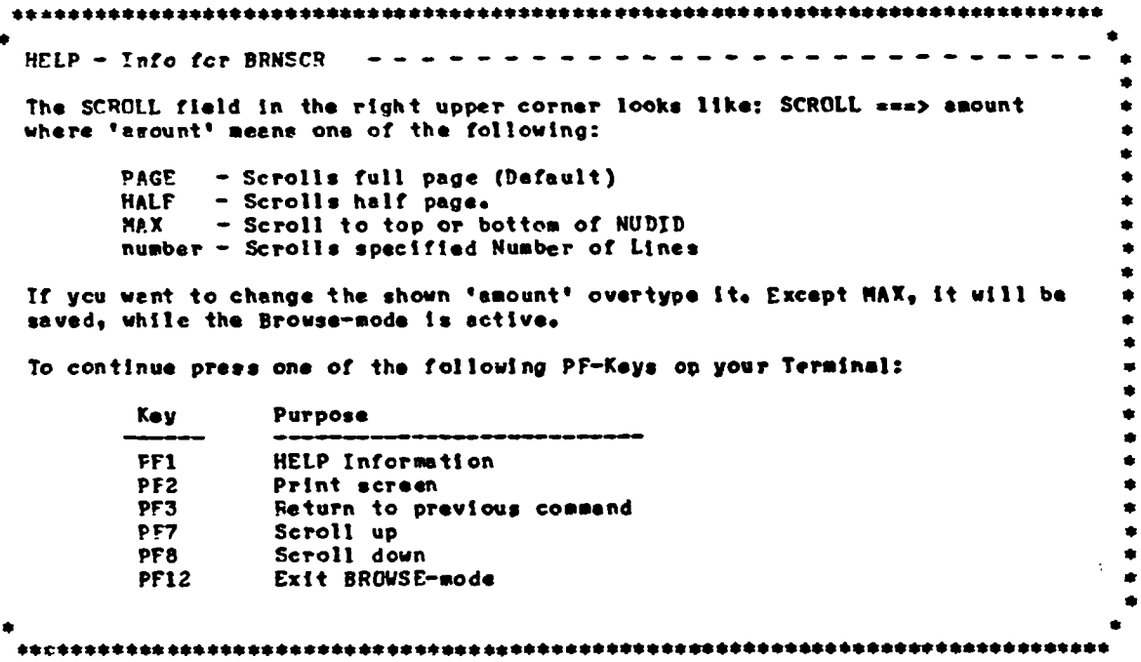
ii) via the "PF-9" key in Mode B

Mode B could only be entered by typing the BROWSE command without any option.

2. In both modes scrolling is possible. If you press the 'PF1' key you can get an help information on how to use the scrolling feature. (See figure 6.2. and 6.3.)

```
HELP - Info for BRTSCR  - - - - - - - - - - - - - - - - - - - - - - - - - - -

The SCROLL field in the right upper corner looks like: SCROLL ===> amount
where 'amount' means one of the following:

        PAGE   - Scrolls full page (Default)
        HALF   - Scrolls half page
        MAX    - Scroll to top, bottom, left or right margin
        number - Scrolls spec. number of rows or columns

If you want to change the shown 'amount' overtype it. Except MAX, it will be
saved, while the BROWSE-Mode is active.

To continue press one of the following PF-Keys on your Terminal:

     Key    Purpose                        Key    Purpose
    ------  -----------------------------  ------ ------------------------
     PF1    HELP Information               PF8    Scroll down
     PF2    Print  screen                  PF9    Browse NUDID
     PF3    Return to previous command     PF10   Scroll left
     PF4    Browse previous table          PF11   Scroll right
     PF5    Browse next table              PF12   Exit BROWSE-command
     PF7    Scroll up
```

Fig. 6.2: HELP Information you get for BROWSE i/o - table (pressing PF1-key)



```
HELP - Info for BRNSCR  - - - - - - - - - - - - - - - - - - - - - - - - -

The SCROLL field in the right upper corner looks like: SCROLL ===> amount
where 'amount' means one of the following:

        PAGE   - Scrolls full page (Default)
        HALF   - Scrolls half page.
        MAX    - Scroll to top or bottom of NUDID
        number - Scrolls specified Number of Lines

If you went to change the shown 'amount' overtype it. Except MAX, it will be
saved, while the Browse-mode is active.

To continue press one of the following PF-Keys on your Terminal:

         Key       Purpose
        -----     -------------------------
         PF1       HELP Information
         PF2       Print screen
         PF3       Return to previous command
         PF7       Scroll up
         PF8       Scroll down
         PF12      Exit BROWSE-mode
```

Fig. 6.3: HELP Information you get for BROWSE NUDID (pressing PF1-key)

In any case you can leave the BROWSE command by pressing the 'PF3' or the 'PF12' key.

3. If the current member is a set of input - output tables you can also look to other tables of the set by using the 'PF4' or 'PF5' key.

4. If you want to print the displayed screen press the "PF-2" key. A copy of the screen will then be printed on the standard print file.

6.3. CHANGE command

Function:

Renames members in UNIOP databank.

Syntax:

```
CHANGE old-member-name, new-member-name
```

Alias - CHG

Required - old-member-name
new-member-name

Defaults - none

Operands:

Old-member-name-membername in UNIOP databank you want to change.
Positional option.

New-member-name-New membername you want to give to OLD member
in UNIOP databank. Positional option.

Supplementary Explanations:

The CHANGE command does only change member names of the UNIOP databank. You cannot change in a set of tables any name of a single input - output table, because they will all be referenced by the same member name.

6.4. COEF command

Function:

Computes coefficients for input - output table.

Syntax:

```
COEF member-name <,ROW=rownr>/<,COLUMN=colnr>/<,STANDARD>
```

Alias - None
Required - 'member-name' / '*'
Defaults - STANDARD

Operands:

member-name	-	User supplied NEW membername for table in UNIOP databank
		or:
'*'	-	Uses currently active membername for table Positional option
ROW	-	Keyword for specifying computation of row-coefficients
rownr	-	Rownumber for which coefficients should be computed
COLUMN	-	Keyword for specifying computation of column-coefficients
colnr	-	Columnnumber for which coefficients should be computed.
STANDARD	-	Computes input-coefficients for TOTAL flows.

Supplementary Explanations:

1. You can compute coefficients of different types with the COEF command. Specifying the option 'COLUMN=colnr' you can, for example, compute output - coefficients for the specified column number.

2. The default option 'STANDARD' will compute input-coefficients for total flows. The corresponding row number of the total flows will be taken from the NUDID. If there is no row number specified, or there exists no total flows for the table the system will prompt you for specification.

3. If the specified number is beyond the table size the system will prompt you for continuation of the command and for correct ROW or COLUMN number if you decide to do so.

4. If the specified member is a set of tables the coefficient computation will be performed to all tables of the set.

5. The numerical data's index and description (NUDID) will be automatically updated. If, for example, the new table size is smaller than the original table, all entries in NUDID which are greater will be deleted.

Further Improvements

In this version of UNIOP a set of tables will not automatically be updated by the system if you specify '*' for the member name. Since there exists an updating routine this feature could be easy implemented. At the moment you should not use '*' for a set of tables.

6.5. DELETE command

Function:

Deletes member from UNIOP databank. The member will not be accessible any more. Physically it will be on the UNIOP databank until you compress the UNIOP databank with the PACK command.

Syntax:

```
DELETE  member-name
```

Alias - None

Required - Member-name

Defaults - None

Note - You will get a message displayed to confirm deletion.

Operands:

Member-name - Name of member in UNIOP databank you want to delete. (Syntax see HELP command)

Supplementary Explanations:

A message appears on the screen asking you for confirmation of delete request. An erroneously deletion could, therefore, be avoided.

Hints for the Databank Manager

The DELETE command sets a flag bit in the master record of the corresponding input - output table. The space of the deleted table will not, therefore, be free for storage of another table until you use the PACK command to re-organize the UNIOP databank.

The tracing reference will be updated by reference to the predecessor of the table, if such one exists. (See LIST command)

Further Improvements:

At the moment there is no possibility to delete a single input - output table in a set of tables. The implementation of such a feature could be easily incorporated.

6.6. EDIT command

Function:

Enters the UNIDP editor processor. It works in full screen mode.

Three types of commands controlls the editor:

a) PRIMARY commands

b) ROW commands

c) COLUMN commands

To leave the editor press the PF3- or PF12-key. A set of tables will be updated if a modification is made.

Syntax:

EDIT

Alias - None
Required - None

Operands:

None

6.6.1. General features of the UNIOP edtior.

6.6.1.1. Screen layout of the editor.
6.6.1.2. Valid PF-keys for the UNIOP editor.
6.6.1.3. Scrolling of the editor screen.
6.6.1.4. HELP processor for the editor.
6.6.1.5. Log book keeping of the editor.
6.6.1.6. The EDITOR buffer and calculations.
6.6.1.7. Matrix and Vector specifications.
6.6.1.8. Text mode.

6.6.2. Entering and leaving the UNIOP editor with a set of tables.

6.6.3. Modifying the input - output table figures.

6.6.4. PRIMARY commands

6.6.4.1. ADD command.
6.6.4.2. CANCEL command.
6.6.4.3. CLEAR command.
6.6.4.4. END command.
6.6.4.5. FIELDW command.
6.6.4.6. FRACTION command.
6.6.4.7. HELP command.
6.6.4.8. LOCATE command.
6.6.4.9. RESET command.
6.6.4.10. SDIVIDE command.
6.6.4.11. SMULTIPLE command.
6.6.4.12. SUBTRACT command.
6.6.4.13. TEXTMODE command.

6.6.5. ROW commands

6.6.5.1. D command (delete row)
6.6.5.2. IB command (insert buffer after row)
6.6.5.3. RB command (replace row by buffer)
6.6.5.4. SB command (show buffer)

6.6.6. COLUMN commands.

6.6.6.1. D command (delete column)

6.6.6.2. IB command (inset buffer after column)

6.6.6.3. RB command (replace column by buffer)

6.6.6.4. SB command (show buffer)

6.6.1. General Features of the UNIOP Editor

The main purpose of the UNIOP editor is to edit existing input - output tables by:

- inserting, deleting and replacing rows, columns and single elements
- making some simple calculations on the input - output table (or part of it)
- modifying the numerical Data's index and Description (NUDID) of a member

A screen, formatted during run time of the program ("dynamic screen") makes all figures of the input - output table directly modifiable.

Similar to the SPF editor, a set of primary - row - and column commands controlls the editor in an easy way. PF-keys permit additonal functions.

Principally the editor commands works on single input - output tables. If you edit a set of tables you have to make the single modifications for each table.

Exceptions are:

- The "delete" command (ROW and COLUMN command D) which deletes the specified rows or columns in the whole set (See also section 6.6.5. and 6.6.6.)
- The "inset buffer" command (ROW and COLUMN command IB) which insets a zero row or column for all the other tables, not displayed. (See also section 6.6.5 and 6.6.6)
- All commands which change the NUDID only (because the NUDID is the same for a whole set of tables.

The following will discuss this in more detail.

6.6.1.1. Screen Layout of the Editor

Figure 6.6.1. shows the significant fields of the editor screen.

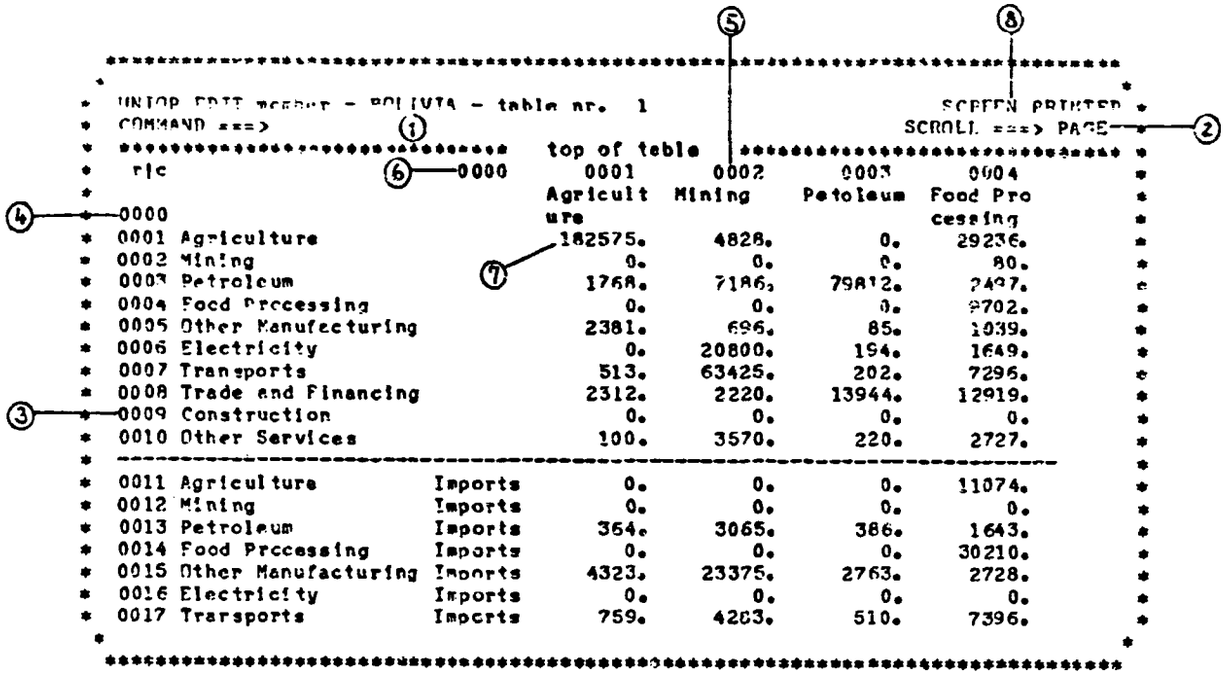


Fig. 6.6.1: EDITOR Screen

Field (1) is the PRIMARY command input field.

Field (2) is the scroll amount field.

Field (3) is the ROW command input field.

Field (4) is the "zero row field".

Field (5) is a COLUMN command input field.

Field (6) is the "zero column dield".

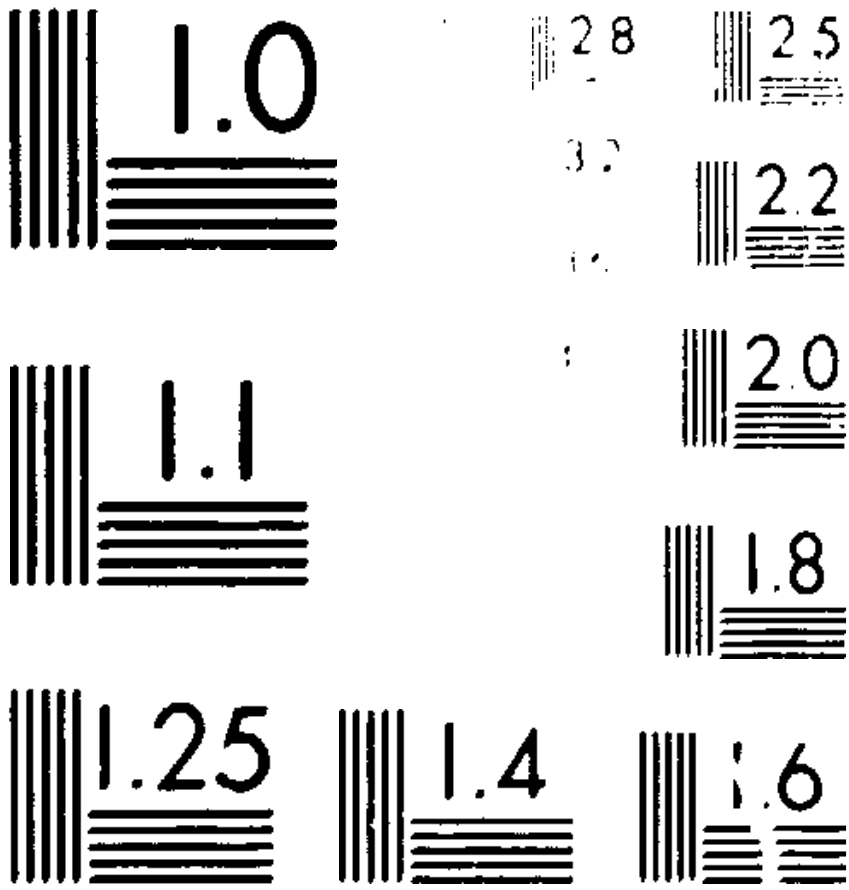
Field (7) is a proper input - output table figure field.

Field (8) is the (error -) message field.

Fields (1) - (7) are all modifiable. The cursor can be positioned at the beginning of such a field via the "TAB", "RETURN" or "TAB RESET" keys.

- The PRIMARY editor commands must be entered in the primary command field (See field (1) in figure 6.6.1.)

  A description of the primary commands will be given in section 6.6.4.


1.0
2.8
2.5
3.2
2.2
1.1
2.0
1.8
1.25
1.4
1.6

- The ROW editor commands must be entered in the row command fields. (See field (3) and (4) in figure 6.6.1.). A description of this type of command will be done in section 6.6.5.

  The "zero row field" (See field (4) in figure 6.6.1.) corresponds not to a proper row of the input - output table. It is a valid address only for some row editor commands (e.g. SB,IB).

- The COLUMN editor command must be entered in the colum command fields. (See field (5) and (6) in figure 6.6.1.). You can find a description of all valid column commands in section 6.6.6.

  The "zero column field" (See field (6) in figure 6.6.1.) is not a proper column of the input - output table. It is a valid address only for some column editor commands. (e.g. SB, IB).

- The proper figure fields (See field (7) in figure 6.6.1.) are all modifiable and can be overtyped. (See also section 6.6.3.).

- The (error) message field (See field (8) in figure 6.6.1.) blinks if an error occurs. The cursor is then positioned to the field in the screen, where the error occurred. If you press in such a code the "PF1 key" you will get displayed detailed information how to use it. (See section 6.6.1.4.).

6.6.1.2. Valid PF Keys for the UNIOP Editor.

The following program function keys (PF keys) are valid for the UNIOP editor. (See figure 6.6.2.)

HELP - Info for PFKEYS

Following PF-keys are valid in the EDITOR:

PF1 HELP	PF2 PRINT SCREEN	PF3 RETURN
PF4 PREV. TABLE	PF5 NEXT TABLE	PF6
PF7 UP	PF8 DOWN	PF9 NUDID
PF10 LEFT	PF11 RIGHT	PF12 RETURN

PF1 HELP Information
PF2 Print screen
PF3 Leave editor
(*) PF4 Display previous table
(*) PF5 Display next table
PF7 Scroll up
PF8 Scroll down
PF9 Edit NUDID
PF10 Scroll left
PF11 Scroll right
PF12 Leave editor

(*) only for set of tables

Fig. 6.6.2: Valid PF-Keys for the EDITOR

HELP (PF1-) Key:

If an error was detected during execution of an EDITOR command an error message will be displayed in the message field (field (8) in figure 6.6.1.). If you press the PF1 key you will get displayed a screen giving you the information that correct usage of the erroneous command. If you press the PF1 key, and no error ocurred you get displayed a list of all valid commands. (See also section 6.6.1.4.).

PRINT (PF2-) Key:

Pressing the PF2 key prints the displayed screen on the standard print file.

RETURN (PF3- and PF12) Keys:

Terminates the EDITOR (See also section 6.6.2.) and prompts you for the next UNIOP command.

NUDID (PF9-) Key:

Enters the numerical Data index and description (NUDID) for editing.

SCROLL (PF4-, PF5-, PF7-, PF8-, PF10-, PF11-) Keys:

Initializes scroll action (See section 6.6.1.3.)

6.6.1.3. Scrolling of the Editor Screen

You can see the editor screen as a "window" moving over a given input - output table (or set of tables) in any direction. This moving is called "scrolling" of the screen. If the data, for example, exceeds

the screen size, scrolling becomes necessary.

The UNIOP editor has the advantage of scrolling in different ways. You can scroll up and down, right or left and if you edit a set of tables, the previous or next input - output table. The amount of scrolling is controlled by the scroll amount field. (See field (2) in figure 6.6.1.)

You can scroll by a specified number of rows or columns or tables (if you edit a set of tables), by a half or a full page by using the cursor position to specify the row or column of the table. You can specify the scroll amount either permanently (i.e. during the whole EDITOR session) or temporarily (i.e. only for one scroll action).

The layout of this scrolling is very similar to SPF. The scroll action will be initialized by pressing the corresponding Program Function Key (PF-key) of the terminal. (See figure 6.6.3.)

```
HELP - Info for SCROLL  - - - - - - - - - - - - - - - - - - - - - - - -

The SCROLL field in the right upper corner, looks like:  SCROLL ===> amount
where  'amount'  means one of the following:
       MAX    - Scroll to top, bottom, left or right margin
       number - Scroll specified number of rows or columns
       PAGE   - Scrolls full page (default)
       HALF   - Scrolls half page
       CSR    - Scrolls to cursor position
  If you want to change the shown 'amount'
overtype it, except MAX, it will be saved
during this session. If you type 'amount'
into the PRIMARY-command field it will
work only temporary.
  To SCROLL press one of the following
PF-keys at your terminal:
   (*) PF4    Display previous table
   (*) PF5    Display next table
       PF7    Scroll up
       PF8    Scroll down
       PF10   Scroll left
       PF11   Scroll right
          (*) ... only for set of tables

+---------+---------+---------+
|  PF1    |  PF2    |  PF3    |
|  HELP   |  PRINT  |  RETURN |
|         |  SCREEN |         |
|---------|---------|---------|
|  PF4    |  PF5    |  PF6    |
|  PREV.  |  NEXT   |         |
|  TABLE  |  TABLE  |         |
|---------|---------|---------|
|  PF7    |  PF8    |  PF9    |
|  UP     |  DOWN   |  NUDID  |
|         |         |         |
|---------|---------|---------|
|  PF10   |  PF11   |  PF12   |
|  LEFT   |  RIGHT  |  RETURN |
|         |         |         |
+---------+---------+---------+
```

Fig. 6.6.3: Scrolling of the EDITOR Screen

The PF key assignments are:

PF4 - to scroll previous table (only if you edit a set of tables).

PF5 - to scroll next table (only if you edit a set of tables).

PF7 - to scroll up (towards the first row of the displayed table).

PF8 - to scroll down (towards the last row of the displayed table).

PF10 - to scroll left (towards the first column of the displayed table).

PF11 - to scroll right (towards the last column of the displayed table).

You can generally combine scrolling with other EDITOR actions. So you use these keys instead of the 'ENTER' key. The scrolling is orientated to row -, column - or table wise scrolling. You can change the scroll amount in two ways:

- permanently if you overtype the displayed amount in the scroll field (See field (2) in figure 6.6.1.). It will then remain until you change it the next time.
- temporarily if you type it into the primary command input field (see field (1) in figure 6.6.1.). It will then work only for the current scroll action.

Valid amounts are:

PAGE - to scroll one page (or decrement one table if the PF4 key was pressed, or increment one table if the PF5 key was pressed).

HALF - to scroll half a page.

CSR - to scroll the position of the cursor within the "table - window" shown. If the cursor is outside the "table - window", scroll by a page will occur.

Any number (in the range of 1 to 9999) means scroll by the specified number of rows or columns of the displayed input - output table, or by the number of tables (for a set of tables if PF4 or PF5 key was pressed).

MAX - to scroll to the top, bottom, left margin, right margin of the displayed table, or to the first or last table (if a set of tables will be edited).

MAX scrolling will be valid only temporarily.

Note: If you press the PF4 or PF5 key and you use the PAGE, HALF or CSR amount it will only increment or decrement the table by one.

6.6.1.4. The HELP processor for the editor.

There are two ways to access to the HELP processor:

(a) by pressing the HELP (PF1-) key (See section 6.6.1.2.)

(b) by typing the primary editor command HELP

ad. (a): - If no error occurred you will get displayed all valid PRIMARY, ROW or COLUMNS commands.

- If an error occurred (a blinking message in the message field (8) in figure 6.6.1.) and you press the PF1 key you will get displeyed a HELP info, telling you how to use the editor feature in a correct way.

ad. (b): - You can get HELP info by using the primary editor command HELP, at any time you are in the UNIOP editor.

For a more detailed usage of this feature see section 6.6.4.

In any case you will return to the editor screen by pressing one of the "action keys" (i.e. PF keys or 'ENTER' key). Some HELP info screens prompts you for continuation. If you want to continue the HELP info you must press the 'ENTER' key.

Pressing the PF keys immediately returns you to the editor screen. Pressing the PRINT (PF2-) key always prints HELP info on the standard print file.

6.6.1.5. Log Bookkeeping of the UNIOP Editor

All significant changes on the tables will be noted into the transaction log. You can document all changes on the tables in the closing phase of the UNIOP session. You can then route the log file to the printer.

6.6.1.6. The EDITOR Buffer and Calculations

The editor uses for some of the calculations a buffer. You add, for example, rows or columns to the buffer. Insertions of rows or columns will always be done via the buffer. You can set the buffer to zero via the primary command CLEAR.

For further details see sections 6.6.4., 6.6.5. and 6.6.6.

Note that calculations like adding, subtracting etc. will be done only by single tables. If you edit a set of tables you have to do the calculations with each table in the set.

6.6.1.7. Matrix and Vector Specifications

The following conventions hold for matrix, vector and element specifications:

Assume the input - output table (matrix) has M rows and N columns; A Matrix specification has then the following general form:

```
(row-from: row-to, col-from:col-to)
```

where

row-from	- index number specifying row begin	(in the range of 1:M)
row-to	- index number specifying row end	(in the range of 1:M)
col-from	- index number specifying column begin	(in the range of 1:N)
col-to	- index number specifying column end	(in the range of 1:N)

The examples below show some default assumptions: Values in square brackets [ ] are default assumptions.

MATRIX specifications:

(,)	whole matrix	i.e. rows [1] : [M] and columns [1] : [N]
(:5,)	submatrix	i.e. rows [1] : 5 and columns [1] : [N]
(,10:)	submatrix	i.e. rows [1] : [M] and columns 10 : [N]
(:5,10:)	submatrix	i.e. rows [1] : 5 and columns 10 : [N]
(1:5,10:15)	submatrix	i.e. rows 1 : 5 and columns 10 . 15

ROW-vector specifications:

(3,)	whole vector	i.e. row 3 and columns [1] : [N]
(2:2,)	whole vector	i.e. row 2 and columns [1] : [N]
(2.10:)	subvector	i.e. row 2 and columns 10 : [N]
(1,:15)	subvector	i.e. row 1 and columns [1] : 15
(2,10:15)	subvector	i.e. row 2 and columns 10 : 15

COLUMN-vector specifications:

(,3)	whole vector	i.e. column 3 and rows [1] : [M]
(,2:2)	whole vector	i.e. column 2 and rows [1] : [M]
(10:,2)	subvector	i.e. column 2 and rows 10 : [M]
(:15,1)	subvector	i.e. column 1 and rows [1] : 15
(10:15,2)	subvector	i.e. column 2 and rows 10 : 15

ELEMENT specifications:

(1,3)	element	i.e. row 1 and column 3
(5:5,2:2)	element	i.e. row 5 and column 2

Assume the vector has M elements: A Vector specification has then the following general form.

(elm-from:elm-to)

where

elm-from - index number specifying element begin (in the range of 1:M)
elm-to - index number specifying element end (in the range of 1:M)

The examples below show some default assumptions: Values in square brackets [ ] are default assumptions.

VECTOR specifications:

()	whole vector	i.e. elements [1] : [M]
(1:)	whole vector	i.e. elements 1 : [M]
(5:)	subvector	i.e. elements 5 : [M]
(:6)	subvector	i.e. elements [1] : 6

ELEMENT specifications:

(3)	single element	i.e. element 3
(5:5)	single element	i.e. element 5

6.6.1.8. TEXT MODE of the UNIOP Editor

In this mode of the editor you can modify the row and column texts. You can also insert and delete the delinatory lines in the layout of the member. (Feature in this version not implemented.)

6.6.2. Entering and Leaving the UNIOP Editor with a Set of Tables

If you edit a member which is a set of tables you will overwrite the current active table, if you did a modification on the data, by loading another table of the set. Therefore entering and leaving the editor is different to a member which is a single table.

If you enter the editor with the UNIOP command EDIT, a warning message will be displayed. (See figure 6.6.4.)

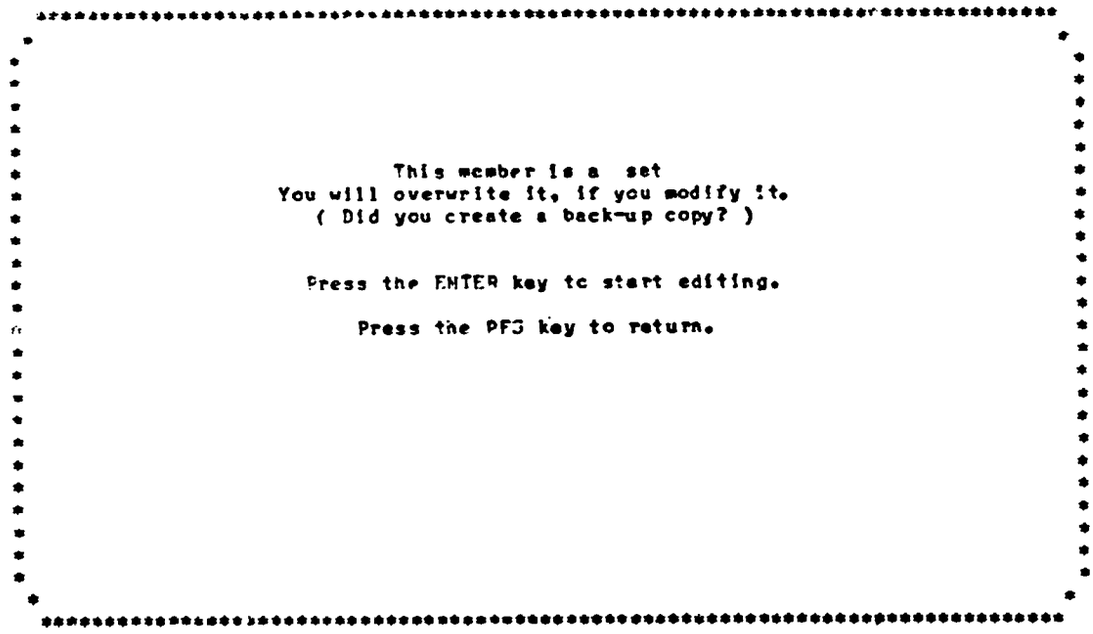


Fig. 6.6.4: Warning Message by entering the EDITOR by a Set of Tables

If you hit the 'ENTER' key you can start editing; in the other case you return to the UNIOP main program to make, e.g. a backup copy of the member.

If you leave the UNIOP editor (by pressing the PF3- or PF12 key or by using the PRIMARY command END), and you did some modification on the table data you will get a warning message, which tells you which modifications are probably open. (See figure 6.6.5.)

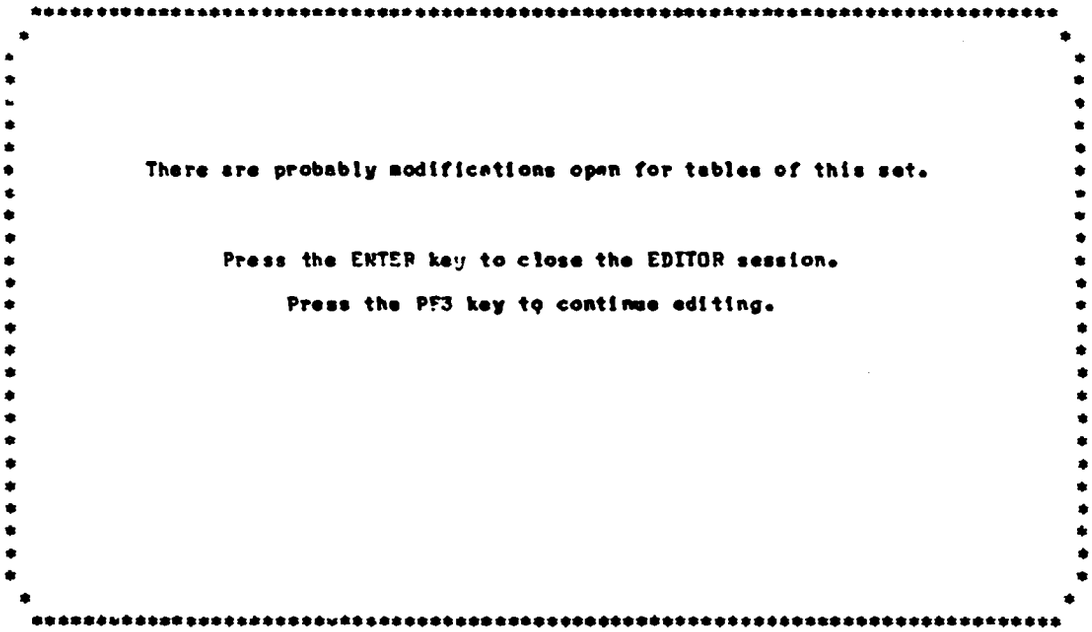


Fig. 6.6.5: Warning Message by leaving the EDITOR with a Set of Tables

6.6.3. Modifying Single Data Figures

If you want to change single elements in the input - output table you can just overwrite the corresponding figure field (See field (7) in figure 6.6.1.)

- In any case a blank character after the changed number will terminate the scanning in the field.

- If you erase the whole field (or type only blanks) it is equivalent to zero.

- If the field has a column delimiting character in the first position this character will be ignored by decoding the number. If you type an illegal character in the field an error message will be displayed.

- You can change more than one number before you hit a transmission key.

- Valid forms of numbers are for example:

  123
  1.23E2
  123.00

  } equivalent to 123.Ø

6.6.4. PRIMARY Editor Commands

The PRIMARY editor commands must be typed into the primary command input field (See field (1) in figure 6.6.1.) The following PRIMARY commands will be discussed in this section:

ADD	-	adds rows, columns or elements of input - output table to editor buffer
CANCEL	-	cancels editor session
CLEAR	-	set editor buffer to zero
END	-	leaves the editor (analog for pressing the PF3- or PF12 key).
FIELDW	-	change field-width of number fields in display
FRACTION	-	change fraction digits of input - output table
HELP	-	display HELP information
LOCATE	-	locates specified element of input - output table
RESET	-	delete display of editor buffer
SDIVIDE	-	divide input - output table scalar
SMULTIPL	-	multiply input - output table scalar
SUBTRACT	-	subtract rows, columns or elements of input - output table from buffer
TEXTMODE	-	switch TEXT mode of EDITOR on or off

In connection with SCROLL PF-keys you can use MAX, PAGE, HALF, CSR, number. (See also section 6.6.3.)

All PRIMARY commands can be abbreviated up to two characters.

6.6.4.1. ADD Primary Editor Command

Function:

"Vector-wise" addition into the editor buffer. This command handles only the displayed table.

Syntax:

ADD	spec <,spec ...> <,B(UFFER) (nr)>
Alias	- None
Required	- Spec
Defaults	- None
Note	- Must be typed into the PRIMARY command input field (See field (1) in figure 6.6.1.)

Operands:

Spec	-	type matrix-spec
Type	-	ROW Row-wise addition
	-	COLUMN Column-wise addition
Matrix-spec	-	Matrix specification indicating the range of operation: (row-from:row-to,col-from:col-to) (general form) For further information See Section 6.6.1.7.
BUFFER(nr)	-	Starts operation at editor buffer element number nr.
nr	-	Number of editor buffer element where operation starts.

6.6.4.2. CANCEL Primary Editor Command

Function:

Cancels the whole UNIOP EDITOR session. Returns without having access to any modification done on the input - output table in this EDITOR session.

Syntax:

CANCEL	
Alias	- None
Required	- None
Defaults	- None
Note	- Must be typed into the PRIMARY command input field (See field (1) in figure 6.6.1.)

Operands:

None

Supplementary Explanations:

If you edit a set of tables you will get displayed a warning message (See Figure 6.6.6), if an updating of one of the tables in the set occurred.

```
You will delete this member.

Press the ENTER key to continue the CANCEL request.
Press the PF3 key to continue editing.
```

Fig. 6.6.6: Warning Message by canceling the EDITOR with a Set of Tables

Closing the editor session via the CANCEL command will then delete the member from the UNIOP databank. In the other case you will not get this warning message and the table will not be accessible anymore, but will remain in the databank.

6.6.4.3. CLEAR Primary Editor Command

Function:

Set editor buffer to zero.

Syntax:

CLEAR	
Alias	- None
Required	- None
Defaults	- None
Note	- Must be typed into the PRIMARY command input field. (See field (1) in figure 6.6.1.)

Operands:

None

6.6.4.4. END Primary EDITOR Command

Function:

Ends the UNIOP editor processor. Has the same effect as pressing the PF3- or PF12 key. (See also section 6.6.2.)

Syntax:

END

Alias - None

Note - Must be typed in the PRIMARY command input field. (See field (1) in figure 6.6.1.)

Operands:

None

6.6.4.5. FIELDW Primary Editor Command

Function:

Change figure field-width of display.

If you omit the operand, the standard field width will be taken.

Syntax:

```
FIELDW  <number>
```

Alias - None

Required - None

Defaults - Standard field width.

Note - Must be typed into the PRIMARY command input field. (See field (1) in figure 6.6.1.)

Operands:

Number - Field-width of figures.

(= number of digits including decimal point and sign)

6.6.4.6. FRACTION Primary Editor Command

Function:

Change fraction digits of member. Command modifies NUDID.

Syntax:

FRACTION number

Alias - None

Required - Number

Defaults - None

Note - Must be typed into the PRIMARY command input field. (See field (1) in figure 6.6.1.)

Operands:

Number - Number of fraction digits.

6.6.4.7. HELP Primary Editor Command

Function:

The PRIMARY command HELP provides Function, Syntax and Operand infromation on EDITOR commands.

Use the following special HELP commands for special information:

HELP COLCMD	-	to display all valid COLUMN editor commands;
HELP MATSPEC	-	how a Matrix specification must look.
HELP PFKEYS	-	to display valid PF keys for the Editor.
HELP PRIMCMD	-	to display all valid PRIMARY editor commands.
HELP ROWCMD	-	to display all valid ROW editor commands
HELP SCROLL	-	for scrolling the screen.
HELP VECSPEC	-	how a vector specification must look.

You can get further HELP information by pressing the PF1 key.

Syntax:

HELP <command-name> <,FUNCTION><,SYNTAX> <,OPERANDS><,PRINT>

Required	-	None
Defaults	-	All if FUNCTION, SYNTAX or OPERANDS not specified.
		NOPRINT
Note	-	If HELP is entered without any OPERAND a list of available commands with a short description of each will be displayed.

Operands:

command name	-	Name of the EDITOR command to be explained. Positional Operand.
		NOTE - Must not be less than two characters long.

FUNCTION	-	Function of EDITOR command is to be displayed.
		NOTE - FUNCTION could be abbreviated with character F.
SYNTAX	-	Syntax format is to be displayed.
		NOTE - SYNTAX could be abbreviated with character S.
OPERANDS	-	Operand description is to be displayed.
		NOTE - OPERANDS could be abbreviated with character O.
PRINT	-	Prints HELP information on standard printfile.

Syntax Interpretation:

1. User supplied values are shown in lower case letters.

   A set of two apostrophes means that the value should be supplied within a set of single apostrophes. e.g.: ''value'' should be typed 'value'.

2. Words in upper case letters must be entered as shown.

   In most cases abbreviations are allowed.

3. All commands can be abbreviated up to two letters.

4. Common delimiters in Operandstrings are comma (,) and blank ( ).

   They can be used mutually.

5. Exclusive choices are indicated by a slash (/).

6. Optional Operands are indicated by a set of brackets (< and >).

   For example: <,PRINT> where ',' means any allowed delimiter.

7. Mutually exclusive formats are separated by OR.

8. Positional Operand means, that you must type the operand in the position shown in the Syntax format.

   All other operands can be entered in any order.

9. You can enter the commands in a free format.

10. Continuation of command strings is NOT possible.

6.6.4.8. LOCATE Primary Editor Command

Function:

Locates specified element/row/column of input - output table on the display. Optionally searches MAXIMUM or MINIMUM element in specified row/column or (Sub) matrix.

Syntax:

```
LOCATE <matrix-spec> <MA<XIMUM> | MI<NIMUM>>
```

Alias - None

Required - 'matrix spec' or MAX | MIN

Defaults - matrix spec := (1:nrows,1:ncols)

Note - Must be typed into the PRIMARY command input field (See field (1) in figure 6.6.1.).

Operands:

matrix spec - Matrix specifications of form (row-from: row-to, column-from: column-to). For further information see section 6.6.2.

MAXIMUM - Searches maximum element

a) if no 'matrix-spec' is given: in whole input - output table

b) if 'matrix-spec' is given: in specified submatrix

MINIMUM - Searches minimum element

a) if no 'matrix-spec' is given: in whole input - output table

b) if 'matrix-spec' is given: in specified submatrix

6.6.4.9. RESET Primary Editor Command

Function:

Deletes editor buffer from screen.

Syntax:

RESET

Alias - None

Required - None

Defaults - None

Note - Must be typed into the PRIMARY command input field (see field (1) in figure 6.6.1.).

Operands:

None

6.6.4.10. SDIVIDE Primary Editor Command

Function:

Scalar division of displayed input - output table. If you edit a set of tables do not forget to divide the others of the set.

Syntax:

SDIVIDE number | B<UFFER>(nr) <,matrix-spec>

Alias	-	None
Required	-	Number | BUFFER(nr)
Defaults	-	Whole table if no 'matrix-spec'
Note	-	Must be typed into the PRIMARY command input field (see field (1) in figure 6.6.1.)

Operands:

number	-	Number you want to divide scalar, or:
BUFFER(nr)	-	Takes spec. element of buffer to divide scalar
nr	-	Element nr. of buffer you want to take.
matrix-spec	-	Matrix specification indicating the range of operation: (row-from:row-to,col-from:col-to) (general form) For further information see section 6.6.2..

6.4.11. SMULT Primary Editor Command

Function:

Scalar multiplication of displayed input - output table. If you edit a set of tables do not forget to multiply the others of the set.

Syntax:

```
SMULT  number | B(UFFER)(nr)  <,matrix-spec>
```

Alias	-	None
Required	-	number | BUFFER(nr)
Defaults	-	whole table if no 'matrix-spec'
Note	-	Must be typed into the PRIMARY command input field (see field (1) in figure 6.6.1.).

Operands:

number	-	Number you want to multiply scalar, or:
BUFFER(nr)	-	Takes specified element of buffer to multiply scalar.
nr	-	Element nr. of buffer you want to take.
matrix-spec	-	Matrix specification indicating the range of operation: (row-from:row-to,col-from:col-to) (general form) For further information see section 6.6.2.

6.6.4.12. SUBTRACT Primary Editor Command

Function:

"Vector-wise" Subtraction from the editor buffer. This command handles only the displayed table.

Syntax:

SUBTRACT spec <,spec ...> <BUFFER(nr)>

Alias	-	None
Required	-	Spec
Defaults	-	None
Note	-	Must be typed into the PRIMARY command input field (see field (1) in figure 6.6.1.)

Operands:

spec	-	type matrix-spec
type	-	R<OW> Row-wise subtraction
	-	C<OLUMN> Column-wise subtraction
matrix-spec	-	Matrix specification indicating the range of operation:
		(row-from:row-to,col-from:col-to) (general form)
		For further information see section 6.6.2..
BUFFER(nr)	-	Starts operation at editor buffer element number nr.
nr	-	Number of editor buffer element where operation starts.

6.6.4.13. TEXTMODE Primary Editor Command

Function:

Switches TEXT-mode of EDITOR on or off.. The TEXT-mode allows you to edit row or column texts. For further information type the primary command: HELP TEXTEDIT

Syntax:

TEXTMODE ON | OFF

Alias - None

Required - ON or OFF

Defaults - None

Note - Must be typed into the PRIMARY command input field (see field (1) in figure 6.6.1.)

Operands:

ON - Switch on TEXT-mode of EDITOR.

OFF - Switch off TEXT-mode of EDITOR.

NOTE - This command has no effect in the current version of UNIOP

6.6.5. ROW Editor Commands

The Row editor commands must be typed into a ROW command input field (see field (3) in figure 6.6.1.).

The selection of certain ROW command input field determines the row number of the input - output table where the action starts.

The following ROW commands will be discussed in this section:

D - deletes assigned row
IB - resets editor buffer after assigned row
RB - replaces assigned row by the editor buffer
SB - displays editor buffer after the assigned row

6.6.5.1. D Editor Row Command

Function:

Deletes selected row. If you edit a set of tables the row will be deleted in all tables of the set.

The rows will be renumbered.

Syntax:

D

Note - Must be entered in a ROW command input field (see field (3) in figure 6.6.1.)

This command is invalid for the "zero field" (see field (4) in figure 6.6.1.)

Example:

```
                         ******************************************************************************
                         *                                                                            *
                         * UNIDP EDIT member - BOLIVIA - table nr.  1                                 *
                         * COMMAND ===>                                            SCROLL ===> PAGE   *
                         * ******************************  top of table  *************************** *
                         *  ric                        0000     0001       0002      0003      0004   *
                         *                                   Agricult   Mining    Petoleum  Food Pro  *
Invalid ----------------->0d00                               ure                            cessing   *
                         * 0001 Agriculture                   182575.     4828.        0.    29236.   *
                         * 0002 Mining                             0.        0.        0.       80.   *
Deletes row no. 3 ------->D003 Petroleum                        1768.     7186.    79812.     2497.   *
Deletes row no. 4 ------> d    Food Processing                     0.        0.        0.     9702.   *
                         * 0005 Other Manufacturing             2381.      696.       85.     1039.   *
                         * 0006 Electricity                        0.    20809.      194.     1649.   *
                         * 0007 Transports                       513.    63425.      262.     7296.   *
                         * 0008 Trade and Financing             2312.     2220.    13944.    12919.   *
                         * 0009 Construction                       0.        0.        0.        0.   *
                         * 0010 Other Services                   100.     3670.      220.     2727.   *
                         * ----------------------------------------------------------------------     *
                         * 0011 Agriculture           Imports      0.        0.        0.    11074.   *
                         * 0012 Mining                Imports      0.        0.        0.        0.   *
                         * 0013 Petroleum             Imports    364.     3065.      306.     1643.   *
                         * 0014 Food Processing       Imports      0.        0.        0.    30210.   *
                         * 0015 Other Manufacturing   Imports   4323.    23376.     2763.     2728.   *
                         * 0016 Electricity           Imports      0.        0.        0.        0.   *
                         * 0017 Transports            Imports    759.     4283.      610.     7396.   *
                         *  |                                                                         *
                         **|***************************************************************************
                           |
                           |___ ROW-command input fields
```

Fig. 6.6.7: Example of D - ROW-Command

6.6.5.2. IB Editor Row Command

Function:

Inserts the contents of the editor buffer after the selected row. If you edit a set of tables there will be a row of zeros inserted in all the other tables of the set. You must change them to the desired values.

The rows will be renumbered.

Syntax:

```
IB
```

Note - Must be entered in a ROW command input field (see field (3) in figure 6.6.1.).

If you use the "zero row field" (see field (4) in figure 6.6.1.) the row will be inserted at the top of the table.

Example:

```
                          UNIOP EDIT member - BOLIVIA - table nr.  1
                          COMMAND ===>                                                  SCROLL ===> PAGE
                          ********************************  top of table  *****************************
                           ric                          0000     0001      0002      0003      0004
Insert buffer before                                              Agricult  Mining    Petoleum  Food Pro
row no. 1 ---------------->IB 0                                   ure                           cessing
                          0001 Agriculture                       102575.    4828.        0.    29236.
                          0002 Mining                                 0.       0.        0.       90.
                          0003 Petroleum                           1768.    7186.    79812.     2497.
                          0004 Food Processing                        0.       0.        0.     9707.
                          0005 Other Manufacturing                 2381.     696.       85.     1039.
                          0006 Electricity                            0.   20800.      194.     1649.
Insert buffer after       0007 Transports                           513.   63425.      202.     7296.
row no. 8 ---------------->  Ib Trade and Financing                2312.    2220.    13944.    12919.
                          0009 Construction                           0.       0.        0.        0.
                          0010 Other Services                       100.    3570.      220.     2727.
                          ---------------------------------------------------------------------------
                          0011 Agriculture           Imports          0        0.        0.    11074.
                          0012 Mining                Imports          0.       0.        0.        0.
                          0013 Petroleum             Imports        364.    3068.      386.     1643.
                          0014 Food Processing       Imports          0.       0.        0.    30210.
                          0015 Other Manufacturing   Imports       4323.   23378.     2763.     2728.
Insert buffer after       0016 Electricity           Imports                   0.        0.        0.
row no. 17 --------------->0Ib7 Transports           Imports                 283.      510.     7396.
                           |
                          *|*************************************************************************
                           |
                           |___ ROW-command input fields
```

Fig. 6.6.8: Example of IB - ROW-Command

- The row text of the new inserted row will be filled with apostrophes (see figure 6.6.8.). You can fill in the new text in the TEXTMODE of the editor.
- If you want to insert a row not calculated out of the input - output table CLEAR the editor buffer (see CLEAR primary editor command) insert the buffer after the selected row and type in the figures you want to insert (see also section 6.6.3.).

6.6.5.3. RB Editor Row Command

Function:

Replaces the selected row by the contents of the editor buffer.

If you edit a set of tables only the row in the just displayed table will be changed.

Syntax:

RB

Note - Must be entered in a ROW command input field (see field (3) in figure.6.6.1.)

This command is modified for the "zero row field" (see field (9) in figure 6.6.1.)

Example:

```
                        ****************************************************************************
                        *                                                                          *
                        *  UNIDP EDIT member - BOLIVIA - table nr.  1                              *
                        *  COMMAND ===>                                        SCROLL ===> PAGE   *
                        *  ******************************  top of table  ************************* *
                        *   r|c                     0000    0001      0002      0003      0004     *
                        *                                   Agricult  Mining    Petoleum  Food Pro *
Invalid here ------------>0000                              ure                           cessing  *
                        *  0001 Agriculture                 182575.   4828.     0.        29236.   *
                        *  0002 Mining                      0.        0.        0.        80.      *
Replace row no. 4       *  0003 Petroleum                   1768.     7186.     79812.    2497.    *
by buffer --------------->RB   Food Processing              0.        0.        0.        9702.    *
                        *  0005 Other Manufacturing         2381.     696.      85.       1039.    *
                        *  0006 Electricity                 0.        20800.    194.      1649.    *
                        *  0007 Transports                  513.      63425.    202.      7296.    *
                        *  0008 Trade and Financing         2312.     2220.     13944.    12919.   *
                        *  0009 Construction                0.        0.        0.        0.       *
                        *  0010 Other Services              100.      3570.     220.      2727.    *
                        *  ---------------------------------------------------------------------   *
                        *  0011 Agriculture      Imports    0.        0.        0.        11074.   *
Replace row no. 13      *  0012 Mining           Imports    0.        0.        0.        0.       *
by buffer --------------->00rb Petroleum        Imports    364.      3065.     386.      1643.    *
                        *  0014 Food Processing  Imports    0.        0.        0.        30210.   *
                        *  0015 Other Manufacturing Imports 4523.     23376.    2763.     2728.    *
                        *  0016 Electricity      Imports    0.        0.        0.        0.       *
                        *  0017 Transports       Imports    769.      4283.     610.      7396.    *
                        *   |                                                                      *
                        ***|************************************************************************
                           |
                           |___ ROW-command input fields
```

Fig. 6.6.9: Example of RB - ROW-Command

6.6.5.4. SB Editor Row Command

Function:

Displays the contents of the editor buffer as a row after the selected row number. This command does not modify the input - output table.

You can delete the buffer row from the screen via the RESET primary editor command (see section 6.6.4.).

Syntax:

SB

Note - Must be entered in a ROW command input field (see field (3) in figure 6.6.1.). If you use the "zero row field" (see field (4) in figure 6.6.1.) The buffer now will be displayed at the top of the table.

Example:

```
                             ***********************************************************************
                             *                                                                     *
                             * UNIOP EDIT member - BOLIVIA - table nr.  1                          *
                             * COMMAND ===>                                     SCROLL ===> PAGE   *
                             * ******************************  top of table  ******************** *
                             *  r|c                      0000   0001      0002      0003    0004   *
Shows buffer before          *                                  Agricult  Mining   Petoleum Food Pro *
row no. 1  ------------->SB 0                                   ure                      cessing  *
                             * 0001 Agriculture                 182575.    4828.       0.   29236.  *
Show buffer after row        * 0002 Mining                          0.        0.       0.      60.  *
no. 2, used prev. ------>>BUF EDITOR BUFFER                        10.      113.      21.     784.  *
                             * 0003 Petroleum                     1768.     7186.   79812.    2497.  *
                             * 0004 Food Processing                  0.        0.       0.    9702.  *
                             * 0005 Other Manufacturing           2381.      696.      85.    1039.  *
                             * 0006 Electricity                      0.    20800.     194.    1649.  *
                             * 0007 Transports                     513.    63425.     202.    7296.  *
                             * 0008 Trade and Financing           2312.     2220.   13944.   12919.  *
                             * 0009 Construction                     0.        0.       0.       0.  *
                             * 0010 Other Services                 100.     3570.     220.    2727.  *
                             * ------------------------------------------------------------------ *
                             * 0011 Agriculture        Imports       0.        0.       0.   11074.  *
                             * 0012 Mining             Imports       0.        0.       0.       0.  *
                             * 0013 Petroleum          Imports     364.     3065.     386.    1643.  *
                             * 0014 Food Processing    Imports       0.        0.       0.   30210.  *
                             * 0015 Other Manufacturing Imports   4323.    23375.    2763.    2728.  *
                             * 0016 Electricity        Imports       0.        0.       0.       0.  *
                              *  |                                                                 *
                               **|*******************************************************************
                                 |
                                 |___ ROW-command input fields
```

Fig. 6.6.10: Example of SB - ROW-Command

6.6.6. COLUMN Editor Commands

The COLUMN editor commands must be typed into a COLUMN command input field (see field (5) in figure 6.6.1.) The selection of a certain COLUMN command input field determines the column number of the input - output table where the action starts.

The following COLUMN commands will be discussed in this section.

D	-	deletes assigned column
IB	-	inserts editor buffer after assigned column
RB	-	replaces assigned column by the editor buffer
SB	-	displays editor buffer after the assigned column

6.6.6.1. D Editor Column Command

Function:

Deletes selected column. If you edit a set of tables the column will be deleted in all tables of the set.

The columns will be renumbered.

Syntax:

D

Note - Must be entered in a COLUMN command input field (see field (5) in figure 6.6.1.) This command is invalid for the "zero column field" (see field (6) in figure 6.61.)

Example:

```
      Deletes column no. 3 ----------------------------------+
                                                             |
      Invalid here -----------+                              |
                              |                              |
 *****************************|******************************|**********************
*                             |                              |                      *
* UNIOP EDIT member - BOLIVIA | table nr.  1                 |                      *
* COMMAND ===>.               |                              |  SCROLL ===> PAGE    *
* **************************** V *   top of table   ******* V ******************** *
*  r|c                       0000     0001      0002      D         0004   ---------- COLUMN-Command
*                                     Agricult  Mining    Petoleum  Food Pro       *   input fields
* 0000                                ure                           cessing        *
* 0001 Agriculture                     182675.     4828.        0.   29236.        *
* 0002 Min ng                              0.        0.        0.      80.        *
* 0003 Petroleum                        1766.     7186.    79812.    2497.        *
* 0004 Food Processing                     0.        0.        0.    9702.        *
* 0005 Other Manufacturing              2381.      696.       85.    1039.        *
* 0006 Electricity                         0.    20800.      194.    1649.        *
* 0007 Transports                        513.    63425.      202.    7296.        *
* 0008 Trade and Financing              2312.     2220.    13944.   12919.        *
* 0009 Construction                        0.        0.        0.       0.        *
* 0010 Other Services                    100.     3570.      220.    2727.        *
* ---------------------------------------------------------------------------     *
* 0011 Agriculture         Imports         0.        0.        0.   11074.        *
* 0012 Mining              Imports         0.        0.        0.       0.        *
* 0013 Petroleum           Imports       364.     3065.      386.    1643.        *
* 0014 Food Processing     Imports         0.        0.        0.   30210.        *
* 0015 Other Manufacturing Imports      4323.    23375.     2763.    2728.        *
* 0016 Electricity         Imports         0.        0.        0.       0.        *
* 0017 Transports          Imports       759.     4283.      610.    7396.        *
*                                                                                 *
 *********************************************************************************
```

Fig. 6.6.11: Example of D - COLUMN-Command

6.6.6.2. IB Editor Column Command

Function:

Inserts the contents of the editor buffer after the selected column. If you edit a set of tables there will be a column of zeros inserted in all the other tables of the set. You must change them to the desired values.

The columns will be renumbered.

Syntax:

IB

Note - Must be entered in a COLUMN comand input field (see field (5) in figure 6.61.). If you use the "zero column field" (see field (6) in figure 6.6.1.) the column will be inserted at the left margin of the table.

Example:

```
       Insert buffer aft r column no. 3 ----------------+
                                                        |
       Insert buffer before                             |
       column no. 1 ---------+                          |
                             |                          |
 ****************************|**************************|********************
*                            |                          |                    *
*  UNIOP EDIT member - BOLIVIA | table nr. 1            |                    *
*  COMMAND ===>              |                          |   SCROLL ===> PAGE *
*  ************************** V *  top of table  ****** V ****************** *
*   r|c                      0160   0001      0002      IB        0004   --------- COLUMN-Command
*                                   Agricult  Mining    Petoleum  Food Pro    *   input fields
*  0000                             ure                           cessing     *
*  0001 Agriculture                 182575.    4828.       0.     29236.      *
*  0002 Mining                          0.        0.       0.        80.      *
*  0003 Petroleum                    1768.     7186.   79812.      2497.      *
*  0004 Food Processing                 0.        0.       0.      9702.      *
*  0005 Other Manufacturing          2391.      696.      85.      1039.      *
*  0006 Electricity                     0.    20800.     194.      1649.      *
*  0007 Transports                    513.    63425.     202.      7296.      *
*  0008 Trade and Financing          2312.     2220.   13944.     12919.      *
*  0009 Construction                    0.        0.       0.         0.      *
*  0010 Other Services                100.     3570.     220.      2727.      *
*  ----------------------------------------------------------------------     *
*  0011 Agriculture         Imports     0.        0.       0.     11074.      *
*  0012 Mining              Imports     0.        0.       0.         0.      *
*  0013 Petroleum           Imports   364.     3065.     386.      1643.      *
*  0014 Food Processing     Imports     0.        0.       0.     30210.      *
*  0015 Other Manufacturing Imports  4323.    23375.    2763.      2728.      *
*  0016 Electricity         Imports     0.        0.       0.         0.      *
*  0017 Transports          Imports   759.     4283.     610.      7396.      *
*                                                                            *
 ****************************************************************************
```

Fig. 6.6.12: Example of IB - COLUMN-Command

- The column text of the new inserted column will be filled with apostrophes (see figure 6.6.12.).

  You can fill in the new text in the TEXTMODE at the editor.

- If you want to insert a column not calculated out of the imput - output table CLEAR the editor buffer (see CLEAR Primary editor command). Insert the buffer at the selected column and type in the figures you want to insert (see CLEAR section 6.6.3.).

6.6.6.3. RB Editor Column Command

Function:

Replaces the selected column by the contents of the editor buffer.

If you edit a set of tables only the column in the just displayed table will be changed.

Syntax:

Note - Must be entered in a COLUMN command input field (see field (5) in figure 6.6.1.). This command is invalid for the "zero column field" (see field (6) in figure 6.6.1.).

Example:

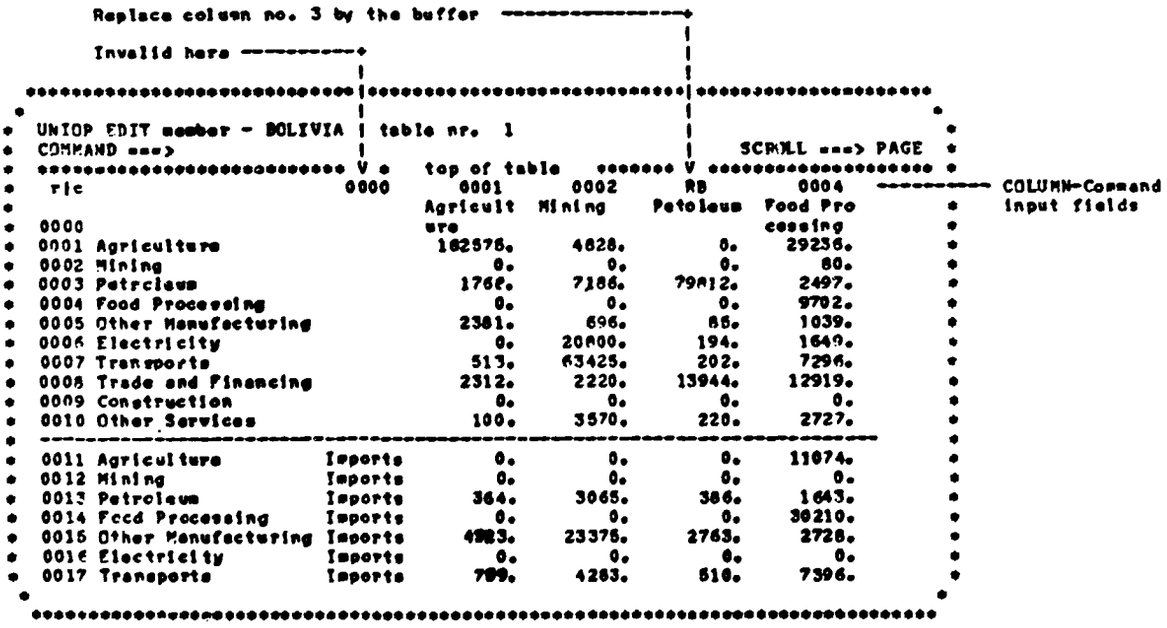


Fig. 6.6.13: Example of RB - COLUMN-Command

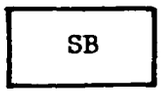
6.6.6.4. SB Editor Column Command

Function:

Displays the contents of the editor buffer as a column after the selected column number. This command does not modify the input - output table.

You can delete the buffer column from the screen via the RESET primary editor command (see section 6.6.4.).

Syntax:



Note - Must be entered in a COLUMN command input field (see field (5) in figure 6.6.1.). If you use the "zero column field" (see field (6) in figure 6.6.1.), the buffer column will be displayed at the left margin of the table.

Example:

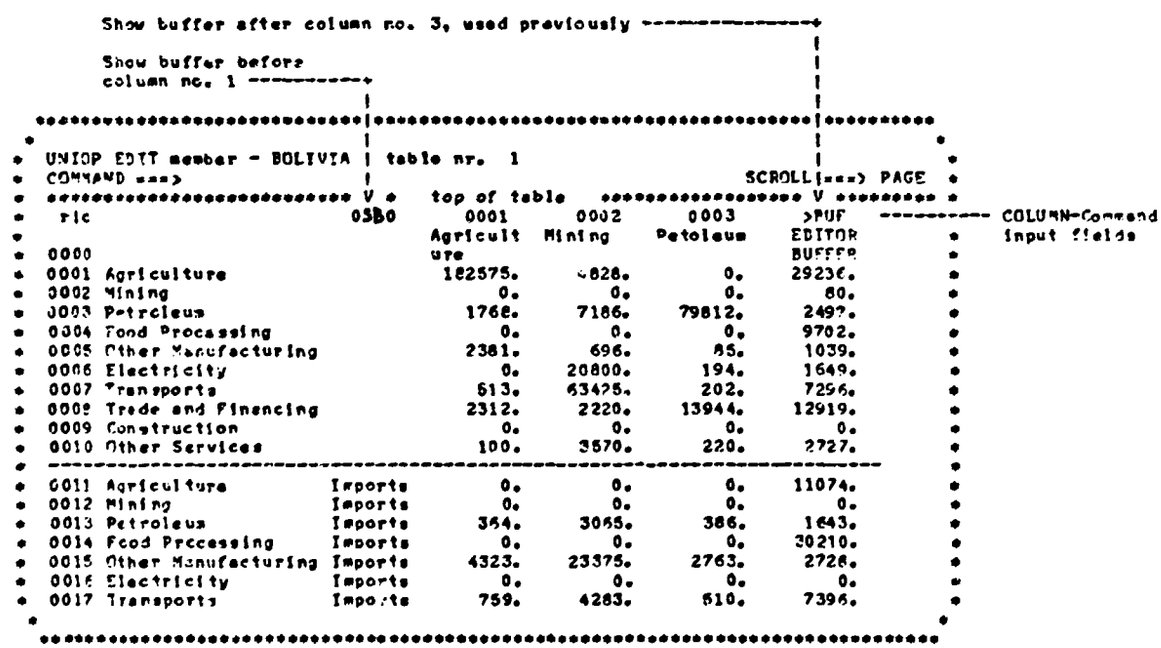


Fig. 6.6.14: Example of SB - COLUMN-Command

6.7. FINISH command

Function:

Terminates an UNIOP Run.
You will get some self-explaining termination messages.

Syntax:

```
FINISH
```

Alias - EXIT

Operands:

None

Supplementary Explanations:

1. By closing the UNIOP run the system informs you of the amount of time used and the number of errors that occurred during the run.

2. If you use the standard PRINT file the system will prompt you to submit the print file to the printer. (See Chapter 7.)

3. If you use the standard PUNCH file the system will prompt you to note messages for cataloging the dataset. (See Chapter 7.)

After processing the FINISH command you enter the closing phase of an UNIOP session. (See also Chapter 7.)

6.8. GET command

Function:

Loads a member from the UNIOP databank into core.

This is the general command to access an input - output table in the UNIOP databank to perform further computations.

Syntax:

GET	member-name

Alias - LOAD

Required - Member-name

Defaults - None

Operands:

Member-name - Name of member you want to get from UNIOP databank. (Syntax see HELP command)

Hints for the databank manager.

The GET command works in the following way:

i) Search directory (= MASTERFILE) for desired member.

If found then:

ii) Reads MASTER record into MASTER common block.

iii) Reads NUDID from TEXT file into TXTR common block.
Access via a pointer in the MASTER record.

iv) Reads data from DATA file. If you have access to a set of tables only the first table will be loaded.

6.9. HELP command

Function:

The HELP command provides function, syntax and operand information on UNIOP commands.

Syntax:

```
HELP  <command-name> <,FUNCTION> <,SYNTAX> <,OPERANDS> <,PRINT>
```

Required - None

Defaults - All if FUNCTION, SYNTAX or OPERANDS not specified; NOPRINT

Note - If HELP is entered without any operand a list of available commands with a short description of each will be displayed.

Operands:

Command-name - Name of the UNIOP command to be explained. Position operand.

NOTE - Must not be less than two characters long.

FUNCTION - Function of UNIOP command is to be displayed.

NOTE - FUNCTION could be abbreviated with character F.

SYNTAX - Syntax format is to be displayed.

NOTE - SYNTAX could be abbreviated with character S.

OPERANDS - OPERAND description is to be displayed.

NOTE - OPERANDS could be abbreviated with character O.

PRINT - Prints information on standard print file.

Syntax Interpretation:

1. User supplied values are shown in lower case letters.

   A set of two apostrophes means that the value should be supplied within a set of single apostrophes. e.g.: ''value'' should be typed 'value'.

2. Words without apostrophes and upper case letters are to be entered as shown. In most cases abbreviations are allowed.

3. All commands can be abbreviated up to two letters.
4. Common delimiters in operandstrings are comma (,) and blank ( ). They can be used mutually.
5. Exclusive choices are indicated by slash (/).
6. Optional operands are indicated by a set of brackets (and).
7. Mutually exclusive formats are separated by 'OR'.
8. 'member-name' means name of member in UNIOP databank.

   General syntax:

   ```
   name-1 /
   name-1.name-2 /
   name-1.name-2.name-3
   ```

   where:

   'name-i' is an user supplied alphanumeric name, max. 8 characters long, starting with an alphameric character.

9. Positional operand means, that you must type the operand in the position shown in the Syntaxformat. All other operands can be entered in any order. In the most cases 'member-name' is a positional operand.
10. You can enter the commands in a free format.
11. Continuation of command-strings is possible. The continuation sign is a semicolon (;) after the last character of the line.

Hints for the Databank Manager.

The HELP information is organized as a sequential dataset named 'UWMR.UNIOP.SYS$.HELP'. A search routine looks for the desired information and displays it on the terminal.

6.10. LIST command

Function:

Displays on terminal:

- Directory of UNIOP databank (all members).
- Supplementary information of specified member.
- Tracing of specified member.

Syntax:

```
LIST <member-name <,TRACE >>
```

Alias - None

Required - None

Defaults - None

Note - Displays UNIOP databank directory if NO operand is specified.

- Displays supplementary information if ONLY member-name is specified.
- Displays tracing of member if member-name AND TRACE is specified.

Operands:

Member-name - Name of member in UNIOP databank. Positional operand. (Syntax see HELP command)

TRACE - Displays tracing of member, i.e. gives you information about predecessors of the specified member.

Note: TRACE could be abbreviated with character 'T'.

Supplementary Explanations:

The LIST command supports the following functions:

1) No option specified:

Displays directory of UNIOP databank. For a set of tables each single table is listed on the screen, (See (1) in figure 6.4.) and numbered in the order they appear in the member. If a member is only a single table one line is displayed. (See (2) in figure 6.4.)

This option informs you also about available space on the UNIOP databank files. (See (3) in figure 6.4.)

```
Directory of Databank                                              (Vers.: 3.04)
                                  UN-          Total   Inventory-
Tablename                         Code Year Dim Row Col   number   Last_use User
-------------------------------------------------------------------------------
ALGERI63.TEST                       12 1963   2  20  23 00001- 000 04-24-81 SYS
ARGNTI63.TEST                       32 1963   2  29  31  0002- 000 04-24-81 SYS
D3IX1.TEST                                    3  20  23 00001- X30 04-24-81 SYS
                    1. Table        16 1963
                    2. Table        20 1964
                    3. Table        12 1965
KENYA67                            404 1967   2  92  33 00038- 000 04-30-81 SYS
AUSTRIA                             40 1964   2 114  61 00004- 000 04-30-81 SYS
AUSTRIA.R8                          40 1964   2  16   9 00004- 000 04-30-81 SYS
ALGER63.TEST.COEF1                  12 1963   2  20  23 00001- 000 04-24-81 SYS
KENYA67.COST                       404 1967   2  88  33 00038- 000 05-07-81 SYS
AUSTRIA.R8C8.COST                   40 1964   2  16   9 00004- 000 05-04-81 SYS
INDIA                              348 1965   2  15  16 00025- 000 05-04-81 SYS
HUNG65MD                           348 1965   2  15  16 00025- 000 05-04-81 SYS
HUNG65MD.COR9                      348 1965   2   9  16 00025- 000 05-04-81 SYS
KENYA67.COEF                       404 1967   2  88  33 00038- 000 05-07-81 SYS

                        End of Directory

        87.200% of MASTER-File is free
        90.505% of TEXT-File   is free
        87.262% of DATA-File   is free
```

(1) (2) (3)

Fig. 6.4: Directory of UNIOP - Databank. (Example for Command 'LIST')

2) **Only member name specified as option:**

Displays some supplementary information about the specified table.

Among others you will get displayed which transaction type starts at which row. (See (1) in figure 6.5.)

Information is also received regarding the amount of space the table uses in the UNIOP databank. (See (2) in figure 6.5.)

```
Description of Databank Teble KENYA67
--------------------------------------

U.N. country code(s)            404
Year(s)                        1967
Inventory number of dataset   '00038- 000'
Version number of dataset        0
Dimensions of dataset            2
   Total number of rows         92
   Total number of columns      33
   Total number of tables        1
Table type(s):
   DOMESTIC-transactions   (Beginning at Row Nr.  26)
   IMPORT  -transactions   (Beginning at Row Nr.  51)
   TOTAL   -transactions   (Beginning at Row Nr.   1)
Implemented  at               04-10-81
Last use     at               04-30-81 by User SYS
Pointers to:
   Masterfile                    7          Continuation Rec.Nr.          0
   Textfile                     75        70 Record(s) (  1.47% of Total)
   Datefile                    173        87 Record(s) (  1.73% of Total)
Systemflags:
   System = 00000002,  History = 00001000,  Typ = 00000002 (Hex)
   Trace Rec.Nr.          0
```

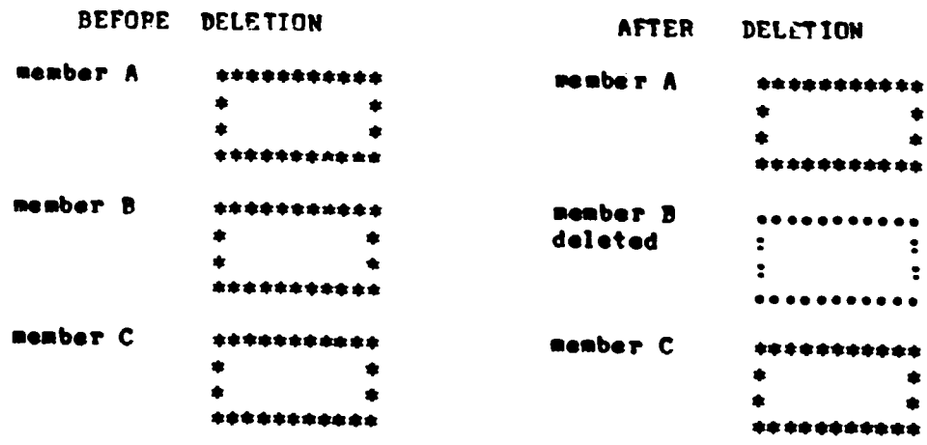
(1) (2)

Fig. 6.5· Additional Information for specified Table.
(Example for Command 'LIST member_name')

3) Trace information (member name and TRACE specified as option):

The UNIOP system keeps track of all predecessors of the specified member. You can, therefore, easily identify the history of a member.

If you delete a member which is a predecessor of the current table, it will also vanish from the tracing list. (See figure 6.6.)



'LIST C,TRACE' displayes:

Table C created out of
1. table B
2. table A

Table C created out of
1. table A

Fig. 6.6: TRACE - Information for specified Table.
(Example for Command 'LIST member_name,TRACE')

Further Improvements:

1) At the moment there is no full screen support implemented for the LIST command. This could easily be achieved.

2) In any case a feature to print the LIST information on the standard print file would be useful.

3) Implementation of the HISTORY option:
The UNIOP system remembers all commands which change the table.
With the HISTORY option a list of all UNIOP commands which changed the table should be displayed.

4) A feature which displays all assigned standard aggregation schemes types and all available countries, years and versions for each type would be useful.

5) For the users it would be convenient to get the directory listed in alphabetical order. It is also possible to implement tasks to display only ranges of the directory. For example to get all members listed which start with A.B.

6.11. LOOP command

Function:

Reads all assigned input - output tables from external dataset, saves them into UNIOP databank and prints them.

Tables must be in UNIDO standard input - output format.

Syntax:

LOOP

Alias - None

Required - None

Defaults - None

Operands:

None

Notes:

- The system will assign a defaultname out of the first 4 characters of the first countryname and 2 digits of the year. If a set of input - output tables is handled the countryname and year of the first table will be used. If it is not possible to create a name the system will use '$SYSnr' or '$SYS99'.
- In any case the tables will be stored into the UNIOP databank.
- You must use the logical unit number 2C as card inputfile.
- A long listing of the tables will be produced.

Supplementary Explanations:

The LOOP command is equivalent to the UNIOP command sequence:

```
READ *, LUN = 20
PRINT
```

It reads, saves and prints all assigned input - output tables from the standard card input file with logical unit number LUN = 20.

For further explanations see the corresponding command description.

Further Improvements:

In this version there exists only a very restricted LOOP command. The main idea behind this command is to create loops in the UNIOP system to handle a sequence of UNIOP commands for a range of tables.

This feature could be implemented with the UNIOP system but should be discussed in detail.

6.12. PACK command

Function:

Compresses the UNIOP databank. All deleted members will vanish physically from the UNIOP databank files. The purpose of this command is to free storage for the saving of members in the UNIOP databank.

Syntax:

```
PACK
```

Alias - CMPR

Operands:

None

Supplementary Explanations:

Deletion and overwriting of members of the UNIOP databank leaves unused records of the databank files, which could not be used until the UNIOP databank is compressed via the PACK command. This command re-organizes the UNIOP databank and resets all pointers.

Further Improvements:

If a system crash happens during re-organizing the databank, it could happen that the re-organization is not closed. In this case an erroneous access to the tables would be possible, because pointers are wrong.

It would be useful to implement a backup copy feature to keep a save copy of the UNIOP databank which will be deleted if the re-organization procedure is successful. If you enter the UNIOP system after a system crash and the re-organization was not successfully

closed, the old status of the UNIOP databank should be automatically restored.

This feature could be implemented with medium effort.

6.13. PRINT command

Function:

Prints tables. Tables will be printed to a standard printfile named:

'prefix.user-id.UNIOP#.OUTLIST'

When you terminate your session you can route this printfile to the printer.

Syntax:

```
PRINT <SHORT><,EDIT><,LINESZ=Chars><,PAGESZ=lines>
```

Alias - None

Required - None

Defaults - NOEDIT,LONG,LINESZ=132,PAGESZ=66

Note - NOEDIT and LONG should not be used as operands.

Operands:

SHORT - No printing of "Numerical Data's Index and Description" (NUDID).

EDIT - Edits tables in most compressed printformat.

LINESZ - Keyword to specify the linesize of the printout,

chars - number of characters in line. This must be in the range of 80 - 132 characters.

PAGESZ - Keyword to specify the page size of the printout.

lines - number of lines to be printed on a single page. This must be greater than 10 lines.

Supplementary Explanations:

The PRINT command writes the currently active member on the standard print file. If required, the system will submit this print file as a batch job to the printer after closing an UNIOP run. (See Chapter 7.)

6.14. PUT command

Function:

Saves the current active member from core into the UNIOP databank.

Syntax:

PUT	<member-name>

Alias - SAVE

Required - None

Defaults - None

Note - If you omit 'member-name' the currently used name will be used.

Operands:

Member-name - Name you want to give the member in UNIOP databank. (Syntax see HELP command)

Supplementary Explanations:

1. The UNIOP system takes care of saving the tables. You will be prompted by the system when the saving of members is necessary. (See figure 6.9.)

```
WARNING! Table not saved,  XENYA.TEST.R8
Do you want to continue the current command?
TYPE 'Y' or 'N':
```

Fig. 6.9: Prompt Message for Saving members.

2. Because in this version of UNIOP there exists no copy command for copying members, you can use the PUT command to create copies of the current member.

The following options for editing the printout are implemented:

1. No option (default)

   Printing of:

   - Front page
   - Numerical data's index and description (NUDID)
   - Tables

2. SHORT option

   Only tables will be printed (no NUDID)

3. EDIT option

   The tables will be printed in a compressed form. In this case it could happen that the column texts will be truncated.

4. LINESZ and PAGESZ option

   You can change the default page format with this option (if you use another print form at the printer or for easier xerox copying, etc.)

   The Default values are:

   PAGESZ=66 line/page

   LINESZ=132 characters/line

6.15. READ command

Function:

Reads input - output tables from external dataset and saves them into the UNIOP databank. Tables must be in UNIDO standard input - output format.

Syntax:

READ member-name <,BYPASS><LIST><PRINT> <,NOSAVE> <,LUN=log-unit>

Alias - None

Required - 'member-name' / '*'

Defaults - LUN=20
NOBYPASS
NOLIST
NOPRINT
SAVE

Note - NOBYPASS, NOLIST, NOPRINT and SAVE should not be used as operands.

Operands:

Member-name - User supplied membername for UNIOP databank, or:

'*' - Defaultname. Positional operand.

BYPASS - Use if you want to skip reading a table.

NOSAVE - Does not save member in UNIOP databank. If you read in set of tables you should not use this operand.

LIST - Displays NUDID on Terminal.

PRINT - Prints NUDID on Standard Printfile.

LUN - Keyword to specify another logical filenumber for cardinput.

log-unit - Logical filenumber in the range of 20 ... 24.

Supplementary Explanations:

1. The READ command is the general UNIOP command to transfer one member (i.e. a single input - output table or set of input - output tables) from external card input dataset to the UNIOP databank.

2. The input - output tables you want to handle with the UNIOP program must be in UNIDO standard format (See Chapter 2.)

3. In the preparation phase of an UNIOP session (See Chapter 7.) you will be prompted by the system for allocation of card image file-members of the standard master input - output table library. By default the logical unit number will be LUN = 2Ø (and therefore the file name FT2ØFØØ1).

4. LUN = log-unit option: You have the possibility to allocate different card input data sets before calling UNIOP to handle input - output tables which are not on the standard master input - output table library.

Example:

Additional to members from the standard master input - output table library you want to read:

a) from your sequential data set called 'UWM.IOCARD' and,
b) two members MEMBA and MEMBB from a partitioned dataset called 'UWM.IOCARD.LIB'

To read in these tables you have to type the following statements before execution of UNIOP:

```
ALLOCATE DSN('UWM.IOCARD') FILE(FT21FØØ1) SHR              (1)
ALLOCATE DSN('UWM.IOCARD.LIB(MEMBA)',-
            'UWM.IOCARD.LIB(MEMBB)') FILE(FT22FØØ1) SHR    (2)
EXEC UNIOP                                                 (3)
```

(1) Allocates the logical unit number LUN = 21 to the data set 'UWM.IOCARD'

(2) Allocates the logical unit number LUN = 22 to the two members of the partitioned data set.

(3) Calls the UNIOP program and enters preparation phase of an UNIOP session (See Chapter 7.)

If you have entered the UNIOP program you can now handle these tables by using the following UNIOP commands:

```
READ NAME1, LUN = 21, PROMPT
READ NAME2, LUN = 22
```

5. BYPASS option. If you have allocated different input - output tables as card input files you can use this option to be prompted by the UNIOP system, if you want to skip the table just entered into processing. (See figure 6.10.)

```
Following Table(s) enter into processing:
    KENYA        (U.N.Code=404, Year=1967, Version= 0)
  Nr. of Dimensions          =   2
  Nr. of Tables              =   1
  Nr. of Rows per Table      =  92
  Nr. of Columns per Table   =  33
  Figures are punched   COLUMN  wise
  Inputformat           =(A4,I3,I3,7F10.0)
  Formatindicator       = 1 0 1 0 0 1 0 0 0 7 0 0
DO YOU WANT HANDEL THIS TABLE AS DB-TABLE KENYA.TEST
TYPE 'Y' OR 'N':
y
   WARNING: Alpha Identificator not checked
```

Fig. 6.10: Prompt Message from READ-Command. (Example of 'READ ...,BYPASS,...')

6. NOSAVE option. By default all tables read in by the READ command will be automatically saved in the UNIOP databank. Using the NOSAVE option prevents this automatical saving. Therefore, you can process this table, for example, via the EDITOR and save only this revised version in the UNIOP databank.

Since in this version of UNIOP there is no temporary saving implemented you should not use this option for a set of tables. In this case if you do not want to save the original table permanently you can use the DELETE command after saving the revised version.

7. LIST option. Displays one to one copy while reading original NUDID on terminal.

8. PRINT option. Prints one to one copy of original NUDID on standard print file.

9. Default name '*'. If you use the character '*' for member name, the system will generate a default name for this member in the UNIOP databank.

The default name consists of:

a) max. six characters of country name (created out of U.N. code entry in NUDID). If the system could not find a country name it will assign the characters 'SYS' instead

b) the last two digits of the year entry in NUDID ('99' will be taken if no such entry was specified in NUDID)

Example:

1) A table entered with U.N. CODE = 404 and YEAR = 1967 entries in NUDID then the system will assign a default name called 'KENYA67'

2) A table entered with no U.N. code and no year entry in NUDID specified then the system will assign our default name called 'SYS99'.

10. Many different input formats could be handled with UNIOP. (See Chapter 2.)

Hints for the Databank Manager:

1. Members of a partitioned cardinput dataset will be treated as a concatenated dataset. In this version there is no skipping to another input channel implemented when reading the EOF mark.

2. Following procedure will be performed when using the READ command:

a) NUDID will be read. Decoding and saving in TXTR common block.

b) If PROMPT option specified the system prompts you for skipping the table. If skipp was specified then the current table will be skipped until the start of the next table or until EOF mark of dataset.

c) The formatindicator will be decoded and completed by the other dataformat entries of NUDID. The table will read according to the format specified. A test for combatibility of specifications will be done.

d) If a set of input - output tables is specified, the tables will be read in sequence and stored in the UNIOP databank before the next table is read.

3. If only a single table is to be handled saving in UNIOP databank will occur by default unless you specify NOSAVE option.

Further Improvements:

1) In the PRINT option the printing also of the input - output tables could easily be implemented.

2) If a temporary saving is implemented the NOSAVE option could be expanded also to sets of tables.

3) Implementation of a stack for different record types. In this version only one record type could be handled for a single member.

6.16. RWIND command

Function:

Rewinds specified dataset.

The dataset must NOT be a DIRECTACCESS-FILE!

Syntax:

```
RWIND  log-unit
```

Alias - None

Required - log-unit

Defaults - None

Operands:

log-unit - 2-digit logical-unit-number of dataset you want to rewind.

NOTE: The logical unit number xx corresponds to the Filename FTxxFyyy in the to UNIOP external environment.

Supplementary Explanations:

The specified logical unit number 'log unit' must not be an direct access file. UNIOP will prompt you if you specify such a file. You can, for example, rewind the standard print or punch file if you want to erase the records written so far.

Another application of the RWIND command is, if you want to reread the card input file. (See READ command.)

6.17. SYSTEM command

Function:

Handles systemfunctions for UNIOP databank.

1) Initializes UNIOP databank files
2) Resets version number of UNIOP databank files

Syntax:

```
SYSTEM INITIAL / VERSION
```

Alias - None
Required - INITIAL / VERSION
Defaults - None

Operands:

INITIAL - Initializes UNIOP databank files.
Attention! You will destroy all previously stored members.

VERSION - Resets versionnumber of UNIOP databank files to current versionnumber.

Supplementary Explanations for the Databank Manager

1. The SYSTEM command should only be used by the Databank manager.

2. The option INITIAL. Initialize all UNIOP databank files. The system will prompt you with a warning message that you will destroy all members in the UNIOP databank and asks you for continuation.

3. The option VERSION. Resets version numbers of UNIOP databank files to the current version of UNIOP. If you do this, be aware of the same record structure of the "old" UNIOP databank file.

6.18. UPDATE command

Function:

Overwrites an existing member in the UNIOP databank.

If it does not exist, it will create a new member in the UNIOP databank.

Syntax:

UPDATE <member-name>

Alias - None

Required - None

Defaults - None

Note - If you omit member-name the currently used name will be used.

Hints for the Databank Manager

If the space of the updated member exceeds the already occupied space on the UNIOP databank files a new record will be created. The old space will remain physically. Re-organization of these gaps will be possible by the PACK command.

6.19. WRITE command

Function:

Writes member from UNIOP databank to an external dataset.

Tables will be punched to a standard punchfile named:

'prefix.user-id.UNIOP#.PUNCH'

When you terminate your session you can catalogue and rename this punchfile.

Syntax:

```
WRITE <SHORT>/<STANDARD>
```

Alias - PNCH

Required - None

Defaults - STANDARD

Operands:

STANDARD - Writes NUDID and Data in Standard UNIDO format.

SHORT - Writes 1. Record (Header) consisting of:

U.N.-Country Code, Year, Nr. of Rows, Nr. of Columns and Format in FORTRAN-format (4I4,12A4)

and starting from 2. Record:

Data in Row-major-form (using the FORTRAN-format 'Format' from the Header)

Supplementary Explanations:

The main purpose of the WRITE command is to create a link to the UNIOP external environment. The WRITE command uses a sequential

standard punch file name 'prefix.user-identifier.UNIOP#.PUNCH' with a logical record size of 80 bytes and fixed block format. After termination of an UNIOP session the system will prompt you for cataloguing and eventually renaming this punch file. (See Chapter 7.) Two formats are implemented:

1. STANDARD option (Default): Writes the tables in standard UNIDO input - output table format. The data figures will be punched in the most comprehensive form.

2. SHORT option: Writes the tables in a short form which is easy to handle for other programs. If a set of tables are punched, every single table will be punched separately in sequential order.

The format of this option is:

1. Record (header record) Writeformat

U.N. CODE	Col. 1-4	I4
YEAR	Col. 5-8	I4
Nr. of ROWS	Col. 9.12	I4
Nr. of COLUMNS	Col.13-16	I4
PUNCH FORMAT in which the table is written	Col.17-64	12A4

Starting from 2 Record:

The data items in row major form, starting from the first row columnwise to the last row. All zero data will also be punched.

The PUNCH format will be determined by the UNIOP system, so that the data could be punched in the most compressed form.

For example:

If a table has a size of Nr. of ROWS x Nr. of COLUMNS = 10 x 15 = 150 items, and the maximal item size if F7.2 (7 print positions for one item with 2 decimal fractions) than

$\lfloor \frac{80}{7} \rfloor$ = 11 items will be punched in one record [1]

and

$\lceil \frac{150}{11} \rceil$ = 14 records will be punched [2] for this table.

The PUNCH format will be in this case look like (11F7.2)

Further Improvements

1. Only indexed items will be punched in the STANDARD option. In the majority of cases it is the most comprehensive way of storing data. For dense occupied tables there could be a non-indexed form (Row or column wise) which is more efficient. The implementation of such a feature could easily be carried out.

2. It could be useful to create other interfaces with the UNIOP system. (For example special links for the World model, SAS or IDIOM.) Such features could be implemented into the UNIOP system.

1/ $\lfloor X \rfloor$ means least integer number of X
2/ $\lceil X \rceil$ means greatest integer number of X

7. How to Start an UNIOP Session:

An UNIOP session could be distinguished by three phases:

1. The preparation phase.
2. The execution phase.
3. The closing phase.

The preparation phase and closing phase is commanded by a CLIST procedure.

The preparation phase allocates all files necessary for the UNIOP session and ends with the call of the UNIOP program. (STEP (7) ÷ STEP (11), see later.)

The execution phase executes the UNIOP program (See later STEP (12) ÷ STEP (13)). It ends if you type the UNIOP command FIN.

The closing phase frees all files used in the UNIOP session and prepares a job to submit the standard print file, if it was used, to printing. (See later STEP (13) ÷ STEP (15)).

The following paragraphs discusses these phases in detail:

Before you enter the UNIOP system do the following steps:

STEP (1) If you are already in TSO go to step (3)

STEP (2) Initialize your TSO session by the command
LOGON user-id/password PROC(LOGON7ØD) ACCT(WI) SIZE(6ØØ)

STEP (3) If you do not want to allocate other cardimage files than the standard cardimage file 'UMW.ORIGINAL. IOTABLE' go to step (5)

STEP (4) Allocate supplementary cardimage files in one of the following ways:

i) members of a partitioned dataset by concatenation:

```
ALLOCATE DSN(datasetname(member-1), -
                .....
             datasetname(member-u)),FILE(FTxxFØØ1),SHR
```

ii) members of a partitioned dataset by selecting alternative channels:

ALLOCATE DSN(datasetname(member-1)),FILE(FTxxFØØ1),SHR

ALLOCATE DSN(datasetname(member-2)),FILE(FTxxFØØ2),SHR

ALLOCATE DSN(datasetname(member-n)),FILE(FTxxFnnn),SHR

iii) sequential dataset

ALLOCATE DSN(datasetname),FILE(FTxxFØØ1),SHR

Where:

xx means 2 digit logical file number not in the set {1-15,20,25}

and

nnn means 3 digit channel number.

Note - Remember the logical file numbers xx because you have to address these files in the UNIOP program with these members. (See Chapter 6: READ command.)

STEP (5) If you do not want to allocate other aggregation scheme files (SAF) go to step (7)

STEP (6) Allocate supplementary standard aggregation scheme file (SAF). (See step (4) : i) - iii))

Here xx means a 2 digit logical file number not in the set {1-15,20,25}

Note - Remember the logical file numbers xx because you have to address these files in the UNIOP program. (See Chapter 6: AGGREGATE command.)

STEP (7) Now you are ready to enter the UNIOP system with the command:

EX 'UWMR.UNIOP.CLIST' (if you do not use the prefix 'UWMR')

or

EX UNIOP (if you use the prefix 'UWMR')

Now you enter the preparation phase of the UNIOP session:

The system prompts you with the message:-

YOUR CARD INPUT DATASET IS 'UWM.ORIGINAL.IOTABLE.MASTER'

TYPE NEW DATASETNAME OR PRESS 'ENTER'.

STEP (8) Pressing the 'ENTER' key on the terminal should be the choice unless a new dataset containing input - output table data will be used. (The default logical unit number will be chosen. See READ command, Chapter 6.)

If you want to change it to another dataset, type the new dataset name enclosed in apostrophes (') when it has not your prefix.

STEP (9) Now the system prompts you with the message:

TYPE MEMBER OR '?' OR PRESS 'ENTER':

If you want to allocate card image members carry out step (10) or go on to step (11).

STEP (10) If the user has no notion of the used member names on the cardimage dataset the entry of sign ? brings the list of members (commence rerolling by hitting the 'ENTER' key until the required table name appears on the screen).

The 'PA1' key (program attention key) could terminate the list of member names.

The message:

TYPE MEMBER OF '?' OR PRESS 'ENTER'

will appear again.

Now enter the member name, (e.g. GHANA60)

Repeat step (10) until you do not want any member allocated.

STEP (11) Press the 'ENTER' key. This terminates the selection of members.

Now the system prompts you with the message:

YOUR AGGREGATION SCHEME DATASET IS

'UWM.ORIGINAL.IOTABLES.AGGREGA'

TYPE NEW DATASETNAME OR PRESS'ENTER':

Repeat step (8).

Now the system enters the execution phase of UNIOP program. Wait until the UNIOP program informs you to enter a command with the message:

COMMAND:

STEP (12) Now you can enter all UNIOP commands from Chapter 6.

If you want information about the commands use the HELP processor by typing:

HELP (abbreviation HE)

You will get the list of available commands.

The system informs you about a special UNIOP command by typing:

HELP command

You will also receive more information by typing:

HELP HELP

Note: The GET command (See Chapter 6.) must be used to make a member available before it can be browsed, edited, or otherwise manipulated.

STEP (13) If you want to finish your UNIOP session type the UNIOP command:

FIN

The system ends the execution phase and enters the closing phase.

If you did not use the standard printfile, the system will continue with step (14).

If you have used the standard printfile the system will prompt you with the message:

ROUTING OF STANDARD PRINT DATASET

'prefix.user-id.UNIOP#.OUTLIST'

TYPE NR. OF COPIES:

If you do not want to route the printfile to the printer type Ø (zero). The system will continue with step (14).

If you want to route the printfile to the printer then type the number of copies you require, e.g.1.

Now the system prepares a job for printing the log-file and the standard printfile at the printer, continuing with step (15).

STEP (14) The system will prompt you with the message:

ROUTING OF LOG-FILE 'prefix.user-id.UNIOP#.LOG'

TYPE NR. OF COPIES:

If you do not want to route the log-file to the printer type Ø (zero). The system will then continue with step (16).

If you want to route the printfile to the printer, then type the number of copies you want.

Now the system prepares a job for printing the log file only.

STEP (15) The system prompts you with the message:

TYPE DESTINATION OR '?':

If you type the character ? you will get displayed a list of options. Use one of them.

The system now closes the preparation of the job and sends you the following message:

JOB jobname(job-id.) SUBMITTED

Remember the jobname (job-id.). You will get your printout under this identification.

STEP (16) If you have used the standard PUNCH file in your UNIOP session the system will prompt you with the message:

DO YOU WANT TO SAVE STANDARD PUNCH
DATASET 'prefix.user-id.UNIOP#.PUNCH'
TYPE 'Y' or 'N':

If you do not want to catalogue the file type the character N (for no). The system will then delete this file and continue with step (14).

If you want to catalogue the file type the character Y (for yes). The system will prompt you:

DO YOU WANT TO RENAME STANDARD PUNCH
DATASET 'prefix.user-id.UNIOP#.PUNCH'.TYPE
'Y' or 'N':

If you want to rename the dataset type the character Y. (for yes). The system will then prompt you with the message:

TYPE NEW DATASET NAME:

Type the dataset name you want to assign to the standard punch file. You can now use the dataset with the new name for further processing.

STEP (17) The system will now finish the closing phase with the message:

READY

Now you can proceed in the TSO environment.

NOTE: If you have allocated supplementary datasets to the UNIOP session (See STEP (4) and STEP (6)) this dataset will remain allocated. To return them to the system, type:

FREE DSN(dataset, ...,dataset).

Further Improvements

Further improvements could be easily carried out to make allocation of supplementary datasets (See STEP (4) and STEP (5)) easier for the user. This could be done by including STEP (3) ÷ STEP (6) into the CLIST procedure.

8. The Implementation of the UNIOP System

The implementation of the UNIOP system will be discussed in the following Chapter:

8.1. Implementation of UNIOP program
8.2. Implementation of UNIOP databank
8.3. Temporary help files used by the UNIOP system
8.4. Other dataset used
8.5. Further improvements

8.1. Implementation of the UNIOP program

Source Modules of UNIOP Program

All source modules of the UNIOP program are stored on the two partitioned dataset:

'UWMR.UNIOP.FORT'
'UWMR.UNIOP.ASM'

The main program is the member called AMAIN. All other modules used, have the prefix 'M'. (See figure 8.1. and figure 8.2.)

There are some other modules, not with a prefix M, which were not used in this version.

Object Modules of UNIOP Program

The compiled source modules of the UNIOP program are on a library (partitioned dataset) called: 'UWMR.UNIOP.OBJ'. Figure 8.3. shows the member list and file description of this dataset. The object modules have the same name as the source modules.

Link data Modules of UNIOP Program

Some of the object modules are prepared by the linkage editor to build a load library. The corresponding linkage editor statements are stored in a partitioned dataset called: 'UWMR.UNIOP.LKED'.

Figure 8.4. shows the member list and file description of this dataset. The link editor modules have the same name as the corresponding source modules.

The TEST member creates the current load modul (i.e. executable program). It is the only module which must not use the NCAL option of the linkage editor. The other members must use the NCAL option to prevent the linkage editor to resolve weak entries. This is the input dataset for the linkage editor.

Load Modules of UNIOP Program

This is the load library of the UNIOP program. The library (portioned dataset) is called: 'UWMR.UNIOP.LOAD'.

Figure 8.5. shows the member list and file description of this dataset.

ALIAS names refer to entries in the corresponding LOAD module for the automatical search feature of the linkage editor or loader. The module UNIOP is the currently released version of UNIOP program. The module TEST is the current test version of the UNIOP program.

8.2. Implementation of the UNIOP databank

Currently used databank files are:

MASTER FILE
DSN = 'UWMR.UNIOP.SYSØ.MASTER'
Organization = PS
Record format = FT
Record Length = 120
Block Size = 120
Used tracks = 2 Max. nr. of log.records = 125

TEXT FILE

DSN = 'UWMR.UNIOP.SYS$ TEXT'
Organization = PS
Record Format = FT
Record Length = 80
Block size = 80
Used tracks = 66 Max. nr. of log. records - 4750

DATA FILE

DSN = 'UWMR.UNIOP.SYS$.DATAS'
Organization = PS
Record format = FT
Record length = 80
Block size = 80
Used tracks = 70 Max. nr. of log. records = 5040

The UNIOP databank files are direct access files. They should be catalogued.

The databank file could be enlarged, but this should be done only by the databank manager, because you have to recompile a routine and link again the UNIOP program. (This is a disadvantage of the FORTRAN compiler, which must know the D.A. file definitions at compile time.)

Further Improvements

For frequent use of the UNIOP-system it would be useful to use either an own version of UNIOP-databank files for each user or some more distinct session. This could be achieved by a minor change in the CLIST procedure calling the UNIOP program. (See Chapter 7.) In any case this should be carried out by the databank manager.

8.3. Non-temporary Help Files used by the UNIOP System

Following supplementary data sets were used by the UNIOP system.

1. HELP file where all help information is stored.

   DSN = 'UWMR.UNIOP.SYS$.HELP'

   Organization = PS

   Record format = FB

   Record length = 81

   Block size = 810

   Used tracks = 3

2. NUDID file where specification for numerical data index and description = (NUDID) access are stored (Direct access file).

   DSN = 'UWMR.UNIOP.SYS$.NUDID'

   Organization = PS

   Record format = FB

   Record Length = 93

   Block size = 93

   Used tracks = 3

3. UTIL1 file is a direct access file for storing large tables. Because the FORTRAN opening of DA. files uses time when first used (DISP=NEW), this utility file is catalogued. This is not a very efficient solution and could probably be improved in later versions.

   DSN = 'UWMC.UNIOP,SYS$.UTIL1'

   Organization = PS

   Record format = FT

   Record length = 4000

   Block size = 4000

4. U.N. Country Code file is the standard Country Code file.

   DSN = 'UWM.UN.COUNTRY.CODE'

   Organization = PS

   Record format = FB

   Record length = 80

   Block size = 800

   Used tracks = 2

8.4. Other Data Sets used by the UNIOP System

The following data set were part of the UNIDO input - output databank.

1. Standard 'Cardimage' file

DSN = 'UWM.ORIGINAL.IOTABLE'
Organization = PO
Record format = FB
Record length = 80
Block size = 6160
Used tracks = 269
Used director blocks = 13
Number of members = 80

2. Standard Aggregation Scheme File

DSN = 'UWM.ORIGINAL.IOTABLES.AGGREGA'
Organization = PS
Record format = PB
Record length = 80
Block size = 6160
Used tracks = 12

8.5. Further Improvements

It would be desirable to minimize the core requirements. This could be done by creating an OVERLAY structure. Because the UNIOP program is structured in a way which makes OVERLAY easy, this could be done. However, some effort is necessary to select the optimal way between core requirements and access time.

Another improvement concerns the size of the UTIL1 direct access file. (See above Chapter 8.3.)

PROJECT: UWMR
LIBRARY: UNIOP
TYPE: FORT

DATE: 82/02/10
TIME: 20:08
PAGE: 001

GENERAL DATA:
VOLUME SERIAL: IAEA01
DEVICE TYPE: 3350
ORGANIZATION: PO
CREATION DATE: 81/10/14

GENERAL DATA:
RECORD FORMAT: FB
RECORD LENGTH: 80
BLOCK SIZE: 6,160
1ST EXTENT SIZE: 94
SECONDARY QUAN: 10

CURRENT ALLOCATION:
94 TRACKS
1 EXTENT
20 DIRECTORY BLOCKS

CURRENT UTILIZATION:
94 TRACKS
1 EXTENT
12 DIRECTORY BLOCKS
73 MEMBERS

MEMBER NAME	VERS.MOD LEVEL	CREATION DATE	DATE AND TIME LAST MODIFIED		CURRENT NO. LINES	INITIAL NO. LINES	MODIFIED NO. LINES	USER ID
AMAIN	01.08	81/09/23	81/12/28	18:21	331	311	57	UR
COMDBF	03.00	81/05/04	81/05/04	18:11	5	5	0	UR
COMIOTAB	03.00	81/05/04	81/05/04	18:11	6	6	0	UR
COMLOG	01.00	81/09/23	81/09/23	12:12	5	5	0	UR
COMMSTF	03.00	81/05/04	81/05/04	18:11	5	5	0	UR
COMMSTR	03.00	81/05/04	81/05/04	18:11	19	19	0	UR
COMNUDF	03.00	81/05/04	81/05/04	18:11	5	5	0	UR
COMOUTL	01.02	81/11/25	81/11/25	13:43	7	6	2	UR
COMSYS	03.01	81/05/04	81/12/28	16:40	8	7	4	UR
COMTFORM	03.00	81/05/04	81/05/04	18:11	9	9	0	UR
COMTXTF	03.00	81/05/04	81/05/04	18:11	5	5	0	UR
COMTXTR	03.00	81/05/04	81/05/04	18:11	13	13	0	UR
COMUTIL1	03.00	81/05/04	81/05/04	18:11	9	9	0	UR
COMVDUB	03.00	81/05/04	81/05/04	18:11	6	6	0	UR
MAGREG	01.05	81/11/11	81/12/28	17:24	1424	1476	41	UR
MBIT	03.00	81/05/05	81/05/05	20:34	162	162	0	UR
MBROWSE	01.04	82/01/07	82/01/22	20:00	581	578	64	UR
MCHANGE	03.00	81/04/30	81/04/30	16:57	80	80	0	UR
MCOPA	03.04	81/04/30	81/11/19	17:48	187	157	30	UR
MCOPF	03.04	81/04/28	81/12/28	17:57	348	327	34	UR
MDELETE								
MED	01.03	82/01/22	82/01/22	21:10	697	696	13	UR
MEIPCMD	01.03	82/01/22	82/01/22	21:18	750	750	6	UR
MEDSCMD								
MEISCREN								
MEDUTIL								
MERMSG	03.07	81/05/05	81/12/02	16:57	708	670	177	UR
MFIN	03.02	81/04/30	81/12/28	18:03	28	26	11	UR
MGET								
MHELP	01.00	82/02/04	82/02/04	16:35	278	278	0	UP
MLIST	03.02	81/04/30	81/05/13	20:23	281	276	6	UP
MLOG	01.07	81/09/23	81/12/04	16:48	108	98	28	UP
MLOOP	03.03	81/04/30	81/12/15	21:22	59	59	3	UR

Fig. 8.1: FORTRAN Source Modules for UNIOP-Programm.

PROJECT: UWMP
LIBRARY: UNIOP
TYPE: FORT

MEMBER NAME	VERS.MOD LEVEL	CREATION DATE	DATE AND TIME LAST MODIFIED
MMSG	01.03	81/12/03	81/12/04 16:49
MNUDR	03.02	81/05/04	81/11/11 17:53
MNUDUT	03.00	81/05/04	81/05/04 18:11
MNUDW	03.00	81/04/30	81/04/30 16:57
MOUTL	01.06	81/11/25	81/12/15 21:02
MPACK	03.10	81/05/04	82/01/22 22:15
MPDSO			
MPRINT	01.04	81/08/27	82/02/04 16:39
MPUT			
MQUALIF	03.00	81/05/05	81/05/05 20:34
MREAD	03.02	81/04/28	81/12/28 17:01
MRWIND	03.00	81/04/30	81/04/30 16:57
MSCAN	01.04	81/12/10	81/12/10 19:24
MSCREEN	01.02	81/12/16	82/02/04 20:17
MSTRING	01.04	81/12/29	81/12/29 19:59
MSYST	03.01	81/04/28	81/12/28 17:50
MUPD	01.02	82/01/22	82/01/22 20:37
MUTILA	01.00	82/01/22	82/01/22 20:51
MUTILB	03.00	81/05/05	81/05/05 20:34
MVDU	03.10	81/04/28	81/09/08 15:30
MVDUTIL	03.12	81/05/04	82/01/07 17:19
MWRITE	03.01	81/04/27	81/05/13 13:13

END OF MEMBER LIST

Fig. 8.1: FORTRAN Source Modules

DATE: 82/02/10
TIME: 20:08
PAGE: 002

CURRENT NO. LINES	INITIAL NO. LINES	MODIFIED NO. LINES	USER ID
35	30	13	UR
1101	1101	18	UR
249	249	0	UR
529	529	0	UR
67	98	65	UR
242	202	55	UR
653	624	106	UR
238	238	0	UR
906	897	38	UR
42	42	0	UR
409	375	36	UR
492	490	4	UR
1098	1104	2	UR
161	160	26	UR
561	537	32	UR
659	659	0	UR
78	78	0	UR
262	234	62	UR
237	186	78	UR
396	399	0	UR

for UNIOP-Programm.

PROJECT: UUMR
LIBRARY: UNIOP
TYPE: ASM

DATE: 81/05/18
TIME: 20:47
PAGE: 001

GENERAL DATA:
VOLUME SERIAL: IAFA16
DEVICE TYPE: 3350
ORGANIZATION: PO
CREATION DATE: 81/05/19

GENERAL DATA:
RECORD FORMAT: FB
RECORD LENGTH: 80
BLOCK SIZE: 6,160
1ST EXTENT SIZE: 1
SECONDARY QUAN: 1

CURRENT ALLOCATION:
2 TRACKS
2 EXTENTS
2 DIRECTORY BLOCKS

CURRENT UTILIZATION:
2 TRACKS
2 EXTENTS
2 DIRECTORY BLOCKS
6 MEMBERS

MEMBER NAME	VERS.MOD LEVEL	CREATION DATE	DATE AND TIME LAST MODIFIED	CURRENT NO. LINES	INITIAL NO. LINES	MODIFIED NO. LINES	USER ID
ALPS	01.00	81/05/19	81/05/19 12:12	118	118	0	UR
AND	01.00	81/05/19	81/05/19 12:12	55	55	0	UR
OR	01.00	81/05/19	81/05/19 12:12	55	55	0	UR
VDUGET	01.00	81/05/19	81/05/19 12:12	38	38	0	UR
VDUPUT	01.00	81/05/19	81/05/19 12:12	49	49	0	UR
XOR	01.00	81/05/19	81/05/19 12:12	71	71	0	UR
MAXIMUMS:	01.00	81/05/19	81/05/19 12:12	118	118	0	
TOTALS:				386	386	0	

END OF MEMBER LIST

Fig. 8.2: ASSEMBLER Source Modules for UNIOP-Program.

PROJECT: UWMR
LIBRARY: UNIOP
TYPE: OBJ

DATE: 82/02/10
TIME: 20:08
PAGE: 001

GENERAL DATA:		GENERAL DATA:		CURRENT ALLOCATION:	CURRENT UTILIZATION:
VOLUME SERIAL:	IAFA03	RECORD FORMAT:	FB	65 TRACKS	65 TRACKS
DEVICE TYPE:	3350	RECORD LENGTH:	80	1 EXTENT	1 EXTENT
ORGANIZATION:	PO	BLOCK SIZE:	3,200	10 DIRECTORY BLOCKS	3 DIRECTORY BLOCKS
CREATION DATE:	81/05/19	1ST EXTENT SIZE:	65		54 MEMBERS
		SECONDARY QUAN:	7		

ALRS
AMAIN
AND
MAGREG
MBIT
MBROWSE
MCHANGE
MCODA
MCOEF
MDELETE
MED
MEDPCMD
MELSCMD
MFISCREN
MEDUTIL
MFRMSG
MFIN
MGET
MHELP
MLIST
MLOG
MLOOP
MMSG
MNUDR
MNUDUT
MNUDW
MOUTL
MPACK
MPDS
MPDSO
MPRINT
MPUT
MQUALIF
MREAD
MRWIND

Fig. 8.3: OBJECT Modules for UNIOP-Programm.

PROJECT: UWMR
LIBRARY: UNIOP
TYPE: OBJ

DATE: 82/02/10
TIME: 20:05
PAGE: 002

MSCAN
MSCREEN
MSTRING
MSYST
MUPD
MUTILA
MUTILB
MVDU
MVDUTIL
MWRITE
OP
XOR

END OF MEMBER LIST

Fig. 8.3: OBJECT Modules for UNIOP-Programm.

PROJECT: UWMR
LIBRARY: UNIOP
TYPE: LKED

DATE: 81/05/18
TIME: 20:47
PAGE: 001

GENERAL DATA:
VOLUME SERIAL: IAEA1S
DEVICE TYPE: 3350
ORGANIZATION: PO
CREATION DATE: 79/12/03

GENERAL DATA:
RECORD FORMAT: FB
RECORD LENGTH: 80
BLOCK SIZE: 800
1ST EXTENT SIZE: 1
SECONDARY QUAN: 1

CURRENT ALLOCATION:
3 TRACKS
3 EXTENTS
10 DIRECTORY BLOCKS

CURRENT UTILIZATION:
3 TRACKS
3 EXTENTS
5 DIRECTORY BLOCKS
26 MEMBERS

MEMBER NAME	VERS.MOD LEVEL	CREATION DATE	DATE AND TIME LAST MODIFIED	CURRENT NO. LINES	INITIAL NO. LINES	MODIFIED NO. LINES	USER ID
MAGREG	01.01	81/04/30	81/04/30 17:48	5	4	1	UR
MBIT	01.01	81/05/05	81/05/05 15:54	9	5	4	UR
MBROWSE	01.01	81/04/30	81/05/05 17:05	5	5	1	UR
MCHANGE	01.00	81/04/30	81/04/30 19:43	5	5	0	UR
MCOEF	01.01	81/04/30	81/04/30 17:49	5	4	1	UR
MDELETE	01.00	81/04/30	81/04/30 19:45	5	5	0	UR
MEDIT	01.00	81/05/05	81/05/05 17:52	5	5	0	UR
MERMSG	01.00	81/05/05	81/05/05 17:45	5	5	0	UR
MFIN	01.00	81/04/30	81/04/30 19:42	5	5	0	UR
MGET	01.01	81/04/30	81/04/30 17:54	5	5	1	UR
MHELP	01.01	81/04/30	81/04/30 17:55	5	5	1	UR
MLIST	01.01	81/04/30	81/04/30 17:55	5	4	1	UR
MLOOP	01.00	81/04/30	81/04/30 19:20	5	5	0	UR
MNUDR	01.00	81/05/05	81/05/05 15:50	5	5	0	UR
MNUEW	01.01	81/04/30	81/04/30 19:11	5	4	1	UR
MPACK	01.00	81/05/04	81/05/04 15:58	5	5	0	UR
MPRINT	01.00	81/04/30	81/04/30 19:14	5	5	0	UR
MPUT	01.01	81/04/30	81/04/30 17:47	5	5	1	UR
MQUALIF	01.00	81/05/05	81/05/05 19:57	5	5	0	UR
MREAD	01.01	81/04/30	81/04/30 17:58	5	4	1	UR
MRWIND	01.00	81/04/30	81/04/30 19:42	5	5	0	UR
MSYST	01.01	81/04/30	81/04/30 17:58	5	4	1	UR
MUPD	01.01	81/05/05	81/05/13 13:46	5	5	1	UR
MUTILS	01.00	81/05/05	81/05/05 19:55	5	5	0	UR
MWRITE	01.01	81/04/30	81/04/30 17:59	5	4	1	UR
TEST	01.02	81/05/05	81/05/05 19:55	9	4	5	UR
MAXIMUMS:	01.02	81/05/05	81/05/13 13:46	9	5	5	
TOTALS:				138	122	21	

END OF MEMBER LIST

Fig. 8.4: LINKAGE-EDITOR Modules for UNIOP-Programm

PROJECT: UWMR
LIBRARY: UNIOP
TYPE: LOAD

DATE: 82/02/10
TIME: 20:09
PAGE: 001

GENERAL DATA:
VOLUME SERIAL: IAEA15
DEVICE TYPE: 3350
ORGANIZATION: PO
CREATION DATE: 81/05/19

GENERAL DATA:
RECORD FORMAT: U
RECORD LENGTH: 0
BLOCK SIZE: 13,030
1ST EXTENT SIZE: 120
SECONDARY QUAN: 19

CURRENT ALLOCATION:
177 TRACKS
4 EXTENTS
20 DIRECTORY BLOCKS

CURRENT UTILIZATION:
160 TRACKS
4 EXTENTS
16 DIRECTORY BLOCKS
88 MEMBERS

MEMBER NAME	LOAD MODULE SIZE (HEX)	(DEC)	ENTRY POINT (HEX)	TTR (HEX)	ALIAS NAME	AUTH CODE	MODULE ATTRIBUTES
ACO	00F768	63,336	000000	004410	MCOBF	00	F-ONLY
AG	018438	99,384	000000	004008	MAGREG	00	F-ONLY
BROWSE	0117D8	71,640	000000	004717	MBROWSE	00	F-ONLY
BROWSN	0117D8	71,640	000428	004717	MBROWSE	00	F-ONLY
BROWST	0117D8	71,640	001180	004717	MBROWSE	00	F-ONLY
CHG	0006C0	1,728	000000	000108	MCHANGE	00	F-ONLY
DECION	001760	5,984	000328	00040A	MQUALIF	00	F-ONLY
DELIO	000E18	3,608	000000	00470A	MDELETE	00	F-ONLY
DELMEM	000E18	3,608	000830	00470A	MDELETE	00	F-ONLY
ED	016538	91,448	000000	004916	MED	00	F-ONLY
ERMSG	005808	22,536	000000	003808	MERMSG	00	F-ONLY
ERMSGI	005808	22,536	000058	003808	MERMSG	00	F-ONLY
ERMSGR	005808	22,536	000080	003808	MERMSG	00	F-ONLY
ERMSGO	005808	22,536	000030	003808	MERMSG	00	F-ONLY
FIN	000308	776	000000	00450F	MFIN	00	F-ONLY
FNDIO	000650	1,616	000230	002808	MUTILB	00	F-ONLY
GET	0105F8	67,064	000000	003F05	MGET	00	F-ONLY
GETIOT	0105F8	67,064	0003F8	003F05	MGET	00	F-ONLY
HLP	002970	10,608	000000	004C13	MHELP	00	F-ONLY
HLPS	002970	10,608	000030	004C13	MHELP	00	F-ONLY
HLPSCR	002970	10,608	0007E8	004C13	MHELP	00	F-ONLY
IDAY	000108	264	000000	060016		NONE	F-ONLY
INTRCD	000110	272	000014	00001C		NONE	F-ONLY
KBSET	0015C8	5,576	000C50	000022	MBIT	00	F-ONLY
KBTST	0015C8	5,576	000F30	000022	MBIT	00	F-ONLY
LI	001B78	7,032	000000	000206	MLIST	00	F-ONLY
LOOP	000458	1,112	000000	003C05	MLOOP	00	F-ONLY
MAGREG	018438	99,384	000000	004008		00	F-ONLY
MBIT	0015C8	5,576	000000	000022		00	F-ONLY
MBROWSE	0117D8	71,640	000000	004717		00	F-ONLY
MCHANGE	0006C0	1,728	000000	000108		00	F-ONLY
MCOBF	00F768	63,336	000000	004410		00	F-ONLY
MDELETE	000E18	3,608	000000	00470A		00	F-ONLY

Fig. 8.5: LOAD Modules for UNIOP-Programm.

PROJECT: UWMP
LIBRARY: UNICP
TYPE: LOAD

MEMBER NAME	LOAD MODULE SIZE (HEX)	(DEC)	ENTRY POINT (HEX)	TTR (HEX)	ALIAS NAME	AUTH CODE
MCD	016538	91,448	000000	004916		00
MERMSG	005808	22,536	000000	003808		00
MFIN	000308	776	000000	00450E		00
MGET	0105F8	67,064	000000	003F05		00
MHELP	002970	10,608	000000	004C13		00
MLIST	001B78	7,032	000000	000206		00
MLIOP	000458	1,112	000000	003C05		00
MNUDR	006AC8	27,336	000000	002F05		00
MNUDW	0118C8	71,880	000000	000301		00
MPACK	0023F8	9,208	000000	004C07		00
MPRINT	011920	71,968	000000	004D0A		00
MPUT	011A70	72,304	000000	004515		00
MQUALIF	001760	5,984	000000	00040A		00
MREAD	0131A0	78,240	000000	003C0C		00
MRWIND	000350	848	000000	000505		00
MSYST	000D48	3,400	000000	004406		00
MUPD	0103D8	66,520	000000	004811		00
MUTILB	000650	1,616	000000	002808		00
MWRITE	00F6B0	63,152	000000	00050C		00
NUDR	006AC8	27,336	000000	002F05	MNUDR	00
NUDW1	0118C8	71,880	000000	000301	MNUDW	00
PAK	0023F8	9,208	000000	004C07	MPACK	00
PRI	011920	71,968	000000	004D0A	MPRINT	00
PRIOT	011920	71,968	000610	004D0A	MPRINT	00
PU	00F6B0	63,152	000000	00050C	MWRITE	00
PUT	011A70	72,304	000000	004515	MPUT	00
PUTIOT	011A70	72,304	000768	004515	MPUT	00
QUAL	001760	5,984	000000	00040A	MQUALIF	00
RDC	0131A0	78,240	000000	003C0C	MREAD	00
RDCST	0131A0	78,240	000AC8	003C0C	MREAD	00
RWIND	000350	848	000000	000505	MRWIND	00
SYSER	005808	22,536	0055F0	003808	MERMSG	00
SYSTEM	000D48	3,400	000000	004406	MSYST	00
TIME	000078	120	000000	000605		NONE
TSTIOS	000650	1,616	000000	002808	MUTILB	00
UPD	0103D8	66,520	000000	004811	MUPD	00
UPDIOT	0103D8	66,520	0007D8	004811	MUPD	00

Fig. 8.5: LOAD Modules for UNIOP-

DATE: 82/02/10
TIME: 20:09
PAGE: 002

MODULE
ATTRIBUTES

F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY
F-ONLY

-Programm.

PROJECT: UWMR
LIBRARY: UNIOP
TYPE: LOAD

DATE: 82/02/10
TIME: 20:09
PAGE: 003

MEMBER NAME	LOAD MODULE SIZE (HEX)	(DEC)	ENTRY POINT (HEX)	TTR (HEX)	ALIAS NAME	AUTH CODE	MODULE ATTRIBUTES
UPDMST	010318	66,520	001D00	004F11	MJPD	00	F-ONLY
VDUGET	0000A8	168	000000	00060B		00	F-ONLY
VDUPUT	000C78	3,192	000000	000612		00	F-ONLY
XTEST	0878C8	555,208	000000	00770A		00	F-ONLY

END OF MEMBER LIST

MODULE ATTRIBUTE CODES:

F-ONLY	PROCESS ONLY BY F LEVEL LINKAGE EDITOR
NON-EXEC	NOT EXECUTABLE
OL	ONLY LOADABLE
OVLY	IN OVERLAY STRUCTURE
REFR	REFRESHABLE
RENT	REENTERABLE
REUS	REUSABLE
SCTR	SCATTER FORMAT
TEST	MODULE TO BE TESTED

Fig. 8.5: LOAD Modules for UNIOP-Programm.

9. Summary of Proposals for Further Improvements

The following table will summarize the proposals for further improvements in the previous chapters.

Chapter	Improvements
4.2.2.	Improvement on the MASTER file
4.3.	Improvements on the whole UNIOP databank organization
5.3.	General improvements on the UNIOP program
6.1.	Improvements on the AGGREGATE command
6.4.	Improvements on the COEF command
6.5.	Improvements on the DELETE command
6.9.	Improvements on the HELP command
6.10.	Improvements on the LIST command
6.11.	Improvements on the LOOP command
6.12.	Improvements on the PACK command
6.15.	Improvements on the READ command
6.19.	Improvements on the WRITE command
7.	Improvements in the UNIOP CLIST procedure for executing an UNIOP session
8.2.	Improvements in the implementation of the UNIOP databank
8.5.	General improvements on the implementation of UNIOP system

APPENDIX A

List of available original input/output
tables in dataset: 'UWM.ORIGINAL.IOTABLE'

Memberlisting of Dataset: UWM.

GENERAL DATA:
VOLUME SERIAL: IAEA15
DEVICE TYPE: 3350
ORGANIZATION: PO
CREATION DATE: 80/06/11

GENERAL DATA:
RECORD FORMAT: FB
RECORD LENGTH: 80
BLOCK SIZE: 6,160
1ST EXTENT SIZE: 274
SECONDARY QUAN: 34

MEMBER NAME	VERS.MOD LEVEL	CREATION DATE	DATE AND TIME LAST MODIFIED
A	01.38	80/05/12	82/01/20 15:02
ALGER63	01.04	80/08/25	80/09/24 16:11
ARGEN63	01.14	80/07/30	81/11/19 17:48
AUSTA74	01.08	80/08/01	82/02/01 13:44
AUST864	01.04	80/08/01	80/09/24 16:12
BANGL862	01.14	80/08/01	81/11/18 16:48
BELG65	01.03	80/08/01	80/09/24 16:14
BOLG70	01.09	80/08/01	81/02/05 17:28
BOLIVIA	01.18	80/07/30	81/12/02 15:33
BRAZ70AB	01.03	80/08/01	80/09/24 16:15
BRAZ70MA	01.06	80/08/01	81/06/24 15:11
CAMBOD65	01.03	80/08/25	80/09/26 16:30
CAMEROON	01.02	81/07/16	81/11/24 13:04
CHIL562	01.03	80/08/01	80/09/24 16:17
COLUM870	01.03	80/08/01	80/09/24 16:18
CONGO67	01.06	80/08/04	81/07/15 15:14
COSTCA66	01.03	80/08/04	80/09/24 16:19
CSSR67MD	01.05	80/10/29	80/11/07 12:14
DENM70	01.04	80/08/06	81/02/05 17:04
ECUAD63	01.03	80/08/06	80/09/24 16:23
FRANCE65	01.02	80/08/06	80/09/24 16:24
FRANCE70	01.04	80/08/06	81/02/05 17:11
GERMAN65	01.02	80/08/0C	80/09/25 13:07
GERMAN70	01.03	80/08/06	81/02/05 17:15
GHANA60	01.03	80/08/06	80/09/26 16:23
GHANA68	01.01	80/08/26	80/09/25 13:08
GREECE70	01.04	80/08/26	81/07/01 16:40
HUNG65MD	01.09	80/10/31	81/02/25 13:04
HUNG70GP	01.10	80/10/22	81/02/25 13:04
HUNG75GP	01.12	80/10/30	81/02/25 13:05
HUNG76	01.04	81/02/24	81/02/25 15:09
INDIA68	01.02	80/08/26	81/02/25 13:05
INDO75	01.01	80/08/26	80/09/25 13:10
IRAN73	01.01	80/08/26	80/09/25 13:10
IRAQ63	01.04	80/08/26	80/09/26 16:26
IRELD69	01.01	80/08/26	80/09/25 13:11
ISRAEL72	01.01	80/08/26	80/09/25 13:12
ITALY65	01.01	80/08/26	80/09/25 13:12

ORIGINAL.IOTABLE DATE: 82/02/10

CURRENT ALLOCATION:
274 TRACKS
1 EXTENT
14 DIRECTORY BLOCKS

CURRENT UTILIZATION:
274 TRACKS
1 EXTENT
14 DIRECTORY BLOCKS
82 MEMBERS

CURRENT NO. LINES	INITIAL NO. LINES	MODIFIED NO. LINES	USER ID
192	8	0	U&F
280	275	0	UQ
397	331	0	UQ
1329	1220	0	U&F
673	669	0	UQ
383	278	0	UQ
657	653	0	UQ
1024	823	0	UR
483	216	0	UQ
1285	1281	0	UQ
394	390	0	UQ
229	225	0	UQ
253	250	0	UQ
531	527	0	UQ
332	328	0	UQ
348	307	0	UQ
394	392	0	UQ
275	276	0	UV
952	761	0	UR
239	237	0	UQ
701	699	0	UQ
944	746	0	UR
794	792	0	UQ
1063	865	0	UR
278	274	0	UQ
310	309	0	UQ
648	550	0	UQ
275	271	0	UQ
286	274	0	UQ
285	277	0	UQ
2334	2217	0	UV
594	593	0	UQ
850	849	0	UQ
546	545	0	UQ
408	403	0	UQ
484	483	0	UQ
537	536	0	UQ
632	631	0	UQ

(cnt.) Memberlisting of Dataset: UWM.ORIGINAL.TOTABLE

MEMBER NAME	VERS.MOD LEVEL	CREATION DATE	DATE AND TIME LAST MODIFIED	CURRENT NO. LINES
ITALY70	01.02	80/08/26	81/02/05 17:18	1059
IVORY60	01.01	80/08/26	80/09/25 13:13	317
IVORY72	01.01	80/08/26	80/09/25 13:14	323
IVORY76	01.01	80/08/26	80/09/25 13:14	405
JAPAN70	01.01	80/08/26	80/09/25 13:14	674
JORDAN68	01.02	80/08/26	80/09/26 16:28	412
KENYA67	01.03	80/09/26	81/02/26 12:37	789
KENYA71	01.53	80/05/12	81/11/18 19:14	709
KOREA66	01.02	80/09/26	81/07/09 15:32	736
KOREA70	01.05	80/09/26	81/11/17 13:41	1291
MADAG66	01.00	80/09/26	80/09/26 17:20	440
MADAG73	01.00	80/09/30	80/09/30 13:52	547
MALAY65	01.01	80/09/30	81/02/26 12:38	664
MALI59	01.01	80/09/30	80/09/30 14:34	258
MEXICO70	01.07	80/09/30	80/10/01 15:05	1273
MOROC64	01.00	80/10/01	80/10/01 14:49	467
NETHER65	01.00	80/10/01	80/10/01 14:54	603
NETHER70	01.01	80/10/02	81/02/05 17:21	1010
NIGRIA60	01.00	81/03/17	81/03/17 15:07	313
NZEAL65	01.02	81/03/17	81/04/01 13:59	886
PAKIST62	01.08	81/04/01	81/11/18 17:09	410
PAPUA72	01.00	81/06/02	81/06/02 15:02	1046
PERU65	01.00	81/06/02	81/06/02 15:23	798
PERU740	01.17	80/05/09	80/07/08 12:52	1652
PHILIP69	01.16	82/01/27	82/01/29 16:44	871
POL67MD	01.06	80/11/03	80/11/07 12:28	275
RHODES65	01.00	81/06/23	81/06/23 14:30	786
SENEG59	01.03	81/06/23	81/06/25 13:57	564
SINCA73	01.02	81/06/25	82/01/27 18:29	1027
SPAIN70				
TAIWAN66				
TAIWAN69	01.00	80/05/22	80/05/22 16:08	1272
TAIWAN71	01.00	80/05/22	80/05/22 15:32	1181
TANZAN70	01.03	80/05/23	80/06/02 11:11	573
TUNES68	01.00	80/05/22	80/05/22 12:57	874
UK70	01.01	80/05/23	80/05/30 15:42	1237
URUG61	01.04	80/05/09	80/05/22 12:59	269
USASERIE	01.25	81/06/16	81/11/19 10:46	779
USA63				
USA67	01.00	80/05/22	80/05/22 13:06	1363
USA72ABS	01.01	80/05/22	80/05/22 18:10	1415
USA72MAK				
USSR66RC	01.08	80/10/24	80/10/30 11:21	262
ZAMBIA65	01.02	80/10/03	80/10/03 16:03	782
MAXIMUMS:	01.53	82/01/27	82/02/01 13:44	2,334
TOTALS:				53,430

END OF MEMBER LIST

DATE: 82/02/10

INITIAL NO. LINES	MODIFIED NO. LINES	USER ID
962	0	UP
316	0	UQ
322	0	UQ
404	0	UQ
673	0	UQ
409	0	UQ
659	0	UQ
8	0	UQ
736	0	UQ
1189	0	UQ
[illegible]	0	UQ
847	0	UQ
658	0	UQ
258	0	UQ
1272	0	UQ
467	0	UQ
603	0	UQ
814	0	UP
313	0	UQ
875	0	UQ
339	0	UQ
1046	0	UQ
798	0	UQ
1625	0	UP
731	0	U$F
241	0	UV
766	0	UQ
497	0	UQ
1027	0	U$F
1272	0	UQ
1191	0	UQ
570	0	UQ
874	0	UQ
1237	0	UT
257	0	UQ
198	0	UQ
1363	0	UQ
1415	0	UQ
34	0	UQ
782	0	UR
2,217	0	
49,027	0	

APPENDIX B

Glossary

Aggregation type - Type of aggregation on standard aggregation file (Chapter 3.) e.g. UNITADR8,UNITADC8.

Aggregation type indicator - A 2 digit number in the range of 1 ... 32 in a one to one relation to all defined aggregation types. (Chapter 3.)

Aggregation scheme - Specification to create an aggregated table out of an existing one. (Chapter 3.)

Standard aggregation file (SAF) - External dataset where aggregation schemes are defined in a standardized way. (Chapter 3.)

For the original tables there already exists a data set called 'UWM.ORIGINAL.IOTABLES.AGGREGA'

SAF - See standard aggregation file.

Standard print file - Print file associated with each UNIOP session. There all print information will be prepared for printing at the printer. In the closing phase of an UNIOP session it will perhaps be routed to the printer.

The system will associate a dataset name like: 'qualifier. user-id.UNIOP#.OUTLIST' with this standard print file. (See Chapter 7.)

Standard punch file - For the standard punch file a dataset name will be associated by UNIOP looking like: 'prefix.user-id.UNIOP#.PUNCH' (See Chapter 7.)

UNIOP session - is divided into 3 parts:

a) Preparation phase

b) Execution phase

c) Closing phase (See Chapter 7.)

UNIOP databank file - 3 direct access files for UNIOP databank

a) MASTER file

b) TEXT file

c) DATA file (See Chapter 4.2.)

SAF Directory - Directory on standard aggregation scheme file SAF. (See Chapter 3.)

SAF Type Block - All aggregation information for an aggregation type on standard aggregation scheme file (SAF). (See Chapter 3.)

SAF Country Block - Country records and aggregation scheme for country records on SAF. (See Chapter 3.)

External Environment - All features not in the UNIDO input - output databank. (e.g. TSO, Operating systems etc.)

UNIDO INPUT/OUTPUT DATABANK - Cardimage master file plus standard aggregation scheme file plus UNIOP system. (See Chapter 1.)

UNIOP System - UNIOP program plus UNIOP databank. (See Chapter 4 and Chapter 5.)

UNIOP Program - Program which controls all handling with the UNIDO - input/output databank. (See Chapter 5.)

UNIOP Databank - Databank for storing original and generations of input/output tables. Part of the UNIOP system. (See Chapter 4.2.)

MASTER File - Master direct access file of UNIOP databank. (See Chapter 4.2.2.)

TEXT File - Part of UNIOP databank. Stores NUDID in most compressed form. (See Chapter 4.2.3.)

DATA File - Part of UNIOP databank. Stores proper data of input/output tables. (See Chapter 4.2.4.

CARDIMAGE MASTER File - Saves original input/output tables. (See Chapter 1, Chapter 2 and Appendix A.)

MEMBER (of the UNIOP Databank) - (See Chapter 2 and Chapter 4.2.)

Single Table	-	Individual input/output table in UNIDO standard input/output table format. (See Chapter 2.1.)
Set of Tables	-	Set of standardized individual input/output tables in UNIDO standard input/output table format. (See Chapter 2.1.)
NUDID	-	Numerical data's index and description. Part of the UNIDO standard input/output table format. (See Chapter 2.2.)
Active Table	-	Table currently actively used by the UNIOP program. Must either be loaded by the GET command. (See Chapter 6.) or newly created by the READ command. (See Chapter 5 and Chapter 6.)
Directory of UNIOP Databank	-	Table of contents of all members in the UNIOP databank. (See LIST command in Chapter 6.10.)
LOG file	-	Log book of the UNIOP-session. Each transaction will be reported. In the closing phase of our UNIOP session you route this log book. The dataset name will be: 'qualifier.user-id.UNIOP#. LOG'.