USER'S GUIDE TO THE MINGL DATABASE SYSTEM FOR TFTR

Version 4.1

R. M. Wieland

Plasma Physics Laboratory

Princeton University

P. O. Box 451

Princeton, NJ 08544

May 1994

[This page left intentionally blank]

Abstract

The MINGL database system is an integrated set of tools which is used to collect and manage diagnostic data and transport code results from TFTR. An interactive menu-driven format allows users to collect scalar and profile data from various data sources and store them into shareable relational databases. A flexible software tool set makes it easy to browse, select, compare and display the data in any of several formats. Utilities exist for managing the individual datasets and restricting their accessibility, for indexing the tables by author, and for maintaining a table-of-contents for an entire database.

[This page left intentionally blank]

CONTENTS

[This page left intentionally blank]

I. INTRODUCTION

The MINGL database system[1] is an integrated set of tools which is used to collect and manage diagnostic data and transport code results from TFTR. An interactive menu-driven format allows users to collect scalar and profile data from various data sources into shareable relational databases. The various tools provided make it easy to browse, select, compare and display the data in one of several formats. Utilities exist for managing the individual datasets and restricting their accessibility, for indexing the tables by author, and for maintaining a table-of-contents for an entire database. The system currently connects to an INGRES database server and runs on the PPPL VAX cluster under VMS.

The MINGL database system for TFTR is composed of several parts:

1. the MINGL tool set
2. the DBASE table construction set for TFTR
3. the LOCUS tool set
4. the INGRES tool set
5. the MANGL archiving system

MINGL is a shell which allows you to transfer data from various diagnostic and analysis efforts on TFTR to a single database where you are free to form your own data tables and to connect with other tables.

The following features have been designed into the shell to facilitate your access to INGRES public and private database objects:

1. You can define subsets of TFTR data and move them into an INGRES database.

---

[1] MINGL Database Sysytem for TFTR, R.M.Wieland, J.A.Murphy et. al., Rev. Sci. Instrum. **59**, (8) 1768-1770 (August 1988)

2. You can use LOCUS[2]. as the graphics interface to your data, as well as taking advantage of most of the other features it is known for.

3. You can manage your own tables, both in terms of controlling access by other users and in terms of editing, deleting, and appending data.

4. You have access to a "table-of-contents" that tells you what objects are in public databases, and who created them.

5. You can operate on objects in a database without having to learn SQL or QUEL, the INGRES command languages. However, both SQL and QUEL commands, as well as the full range of INGRES utilities, are accessible as needed.

6. You can use this system from your own area on the VAX.

This document describes the overall system in terms of how the various parts interact with objects in the underlying INGRES database structure, what modes of access are available to you as a general user, and what specialized operations are required to administer the public database tables that you create. It then goes on to describe in detail the various mechanisms available for moving TFTR data into data tables.

It is suggested that you first read Appendix 1 before continuing on with this report, in order to have an idea of what kinds of movements and operations are possible within a MINGL form. In addition, on-line help is also available within every form by typing the "Help" (PF2) key followed by "Keys" (Ctrl-E) for key function information and "What-To-Do" (Ctrl-R) for form input information.

---

[2] Cf. “The LOCUS User’s Guide ... A Front-End to Ingres Databases”, by J.A. Murphy and R.M. Wieland (November 1992) PPPP-TM-397R.; also the on-line document accessible through $HELP INGRES.

II. DATABASE STRUCTURES

It is quite natural to wonder about "where" the tables are that you are creating with MINGL and LOCUS. In fact, some people experience a great deal of anxiety when they first discover that they cannot do the things to their data tables that they can do to their regular VMS files. The plain fact is that the MINGL database structure is defined somewhat differently from the standard VAX VMS file format you are accustomed to. The only way to touch and feel the tables you create is either through MINGL or INGRES. What follows is a brief explanation of how the structure is set up.

Databases

Databases are to INGRES what file directories are to VMS. Opening a database is like setting your default area in VMS to a particular file directory on disk. The tables that are contained in a database are like the files contained in a VMS directory. INGRES, however, maintains its own directory structure, and users should never try to modify or delete their tables directly through DCL. Unlike under VMS, however, your scope of operation in INGRES is restricted to the tables in the database you have open. The databases themselves reside on several disks maintained by the INGRES database administrator.

Databases are either public or private, depending on how they are created. If you create a database yourself then it is by definition a private database, and you have complete control over the management and disposition of relations in that database. Because there is a large overhead in allocated disk space for each database, it is best to create only one database, preferably with a name identical to your user name on the VAX.[3] There is usually enough freedom in naming tables that a single database should be sufficient for all your needs. Private databases are usually quotaed, meaning that you are allotted typically 10-20,000 blocks of disk space for your database to expand into.

A database created by special request for the use of a group of users is a public database, and while you will have to use the MINGL shell in

---

[3] To create a private database, you must send email to INGRES and request that a database be created for you. One database per user is the usual rule.

order to manage the tables you create in such a database, you are free to use any other available means (such as INGRES itself) to deal with the data therein. All the tables in a public database are owned by the pseudo-user PUBLICDB. Public databases are not quotaed, and so you will be periodically asked to clean up any tables you are responsible for.

Tables

When you create a table, you must specify the database in which it will exist. Most programs will ask you up front for the name of the database you want to work in. You can place it in almost any existing database you wish, although you will probably want to create it either in your own private database or in a public database such as TFTR.

If you create a table in your own database, then you have complete control over it. The default protection on the table at its inception is {OWNER: all; WORLD: nothing}. To modify this, you can use one of the tools available in MINGL, described in Section III., to give additional access to other users. You can also use an INGRES command directly. The disadvantage of placing the table in a private database is that since INGRES does not allow joins to be made between tables in different databases, you will be unable to "communicate" with tables in the TFTR public database. An advantage, on the other hand, of working in a private database, is that you can avoid the overhead of large system directories that go with large public databases, which can slow some database operations.

The alternative to creating a table in a private database is to create it directly in a public database that contains other tables of interest to you. While MINGL assigns ownership of the table or view that you create to "PUBLICDB", it remembers that you created it and designates you as its "Surrogate Database Administrator" (SDBA). As an SDBA you have full control over the management and disposition of that table as long as you work through the MINGL shell.[4] The default protection assigned to the table is {SDBA: all; WORLD: read-only}, although you can define specific access rights if you wish, even to the extent of designating other users as co-SDBA's. A tool exists to modify the access rights to a given table at a later time.

---

[4] Unfortunately, it is still possible to use INGRES directly to create a table in a public database. You will be unable to access it, however, through either MINGL or LOCUS.

If you first create the table in a private database, then decide later that you would prefer to have it in another database, you can use another tool, described in Section 3, to copy it over.

[This page left intentionally blank]

III. DATABASE MANAGEMENT TOOLS

MINGL provides a centralized menu-driven directory of all the services available in the shell. It allows you to switch from one tool set to another without having to re-initialize the database engine or startup a new executable image as you would otherwise have to do if running the individual utilities. Another benefit of using MINGL is that it displays all the currently available tools in its directory, making it easier for you to find the tool you need in a particular situation. Nonetheless, most of the tools available in MINGL are also available as standalone programs.

All of these tools, as well as the DBASE utilities described in the next chapter, trigger auditing procedures whenever they detect the (intended) creation of new tables in a public database. As part of this special auditing process, these programs will prompt you for an entry describing the contents of the new table. This entry will be appended to **CONTENTS,** a table which functions as a "Table-of-Contents" for the tables in the public database. They will also ask you to define an access control list for the table. The default is SDBA=ALL, WORLD=READONLY.

DESCRIPTION OF MINGL TOOLS BY SUBMENU

MAIN MENU SUBMENUS	FUNCTION
LOCUS	The LOCUS subsystem of programs
INGRES	The INGRES subsystem of programs
UTILITIES	Database management utilities
VMS	Selected VMS utilities [No longer supported]
DBASE	Data collection programs

DATABASE	Change the database under consideration

LOCUS SUBMENUS	FUNCTION
{PLOT, EDIT, etc.}	The standard LOCUS operations

INGRES SUBMENUS	FUNCTION
QUEL	The QUEL line-editor shell
IQUEL	The IQUEL screen-editor shell
SQL	The SQL line-editor command shell
ISQL	The ISQL screen-editor shell
INGMENU	The menu-oriented shell (this level contains a complete range of INGRES programs, such as QBF, ISQL, VIFRED, etc.)

UTILITIES SUBMENUS	FUNCTION
MANAGER	
STRUCTURE	This enables you to modify the indexing structure of your tables.
PERMITS	This enables you to modify the group of users who have access to your tables.
CONTENTS	This enables you to either browse through the table-of-contents table, or to update an entry in it.
DIRECTORY	This enables you to see what tables are available in the database. If the database is a

	public database such as TFTR, it also allows you to see who the **SDBA**'s of a given table are, as well as whether you have READ access to the table.
COPY	This enables you to copy your own table from one database to another, or to copy a view you may have created into the public domain of a public database.
ALTERTABLE	A screen-oriented utility that lets you add, initialize, delete or rename columns in any of your tables. It is also useful for renaming a table.
VIEWBUILDER	This is a screen-oriented utility for creating, deleting and inspecting views.
MANGL	A utility for destroying and/or archiving tables.

DBASE SUBMENUS[5]	FUNCTION
DBASE_TMPL	Generates one of the {GLOBESUM,DIAGCOMP,POWRBAL, SUMREL} template tables from the GLOBE and SNAP databases .
DBASEG	Generates a user defined table from the GLOBE database.

[5]The following entries all refer to kernel programs that are submitted to a designated batch queue by the DBASE shell. The shell will prompt you for all the necessary input information required by the particular kernel. More detailed information on these input files can be found in Section IV.

DBASES	Generates a user defined table from the SNAP database.
DBASEW	Generates a user defined table from waveforms, raw data files, TRANSP scalar data, and 1-d results ufiles.
DBASEU2	Generates a user defined table from 2-d ufiles or from TRANSP scalar or profile data.
DBASE_PRO	Generates a user defined table from SNAP profile data .
DBASET	Generates a user defined radius and time table from TRANSP/RPLOT files.
DBASEU1_PRO	Generates a user defined profile table from 1-d ufiles.
DBASEWT	Generates a user defined time-series table from waveforms, raw data files, TRANSP scalar data, and/or 1-d ufiles.
SNAPSHOTS	A menu driven utility for searching for SNAP shot-tries within a given shot or date range, for browsing the associated comments, and for constructing shot lists for use in some of the DBASE routines.

DATABASE SUBMENU	FUNCTION
Change database	Change database

DESCRIPTION OF MINGL TOOLS BY AVAILABILITY

To summarize the functionality of the MINGL database tool set, the tools are listed below, along with their means of invocation.

Tool:	ALTERTABLE
Function:	Modify parameters in datable; rename table
MINGL:	Utilities:Altertable
Standalone:	$ALTERTABLE

Tool:	CONTENTS
Function:	Browse the table-of-contents (read-only mode)
MINGL:	Utilities:Contents
Standalone:	$CONTENTS

Tool:	COPYTABLE
Function:	Copy a user table
MINGL:	Utilities:Copy
Standalone:	$COPYTABLE

Tool:	COPYVIEW
Function:	Copy a user view [TFTR database only]
MINGL:	Utilities:Copy
Standalone:	$COPYVIEW

Tool:	IFILE_IN
Function:	Read an ASCII data file into a data table
MINGL:	Not available
Standalone:	$IFILE_IN

Tool:	IFILE_OUT
Function:	Export a data table into an ASCII data file
MINGL:	Not available
Standalone:	$IFILE_OUT

Tool:	IITDIR
Function:	Directory of tables and users
MINGL:	Not available
Standalone:	$IITDIR[6]

---

6 Also $IITDIR *database* -t*table* , where table can contain wildcard characters.

Tool: INGRATE
Function: Convert LOCUS/ISAM table to INGRES format
MINGL: Not available
Standalone: $INGRATE

Tool: MANGL
Function: Destroy and/or archive tables
MINGL: Utilities:Manager
Standalone: $MANGL

Tool: MODIFY
Function: Modify table structure
MINGL: Utilities:Manager
Standalone: $MODIFY

Tool: PERMITS
Function: Grant permits to a table
MINGL: Utilities:Manager
Standalone: Not available

Tool: RDIR
Function: Directory of tables and users
MINGL: Utilities:Directory
Standalone: $RDIR

Tool: SNAPSHOTS
Function: Browse the SNAP & GLOBE archives
MINGL: Utilities:Snapshots
Standalone: $SNAPSHOTS

Tool: UCON
Function: Browse the table-of-contents (update mode)
MINGL: Utilities:Contents
Standalone: $UCON

Tool: VIEWBUILDER
Function: Create complex views
MINGL: Utilities:Viewbuilder
Standalone: Not available

DESCRIPTION OF MINGL TOOLS BY FUNCTION

1. To copy a table from one database to another:

   Use **COPYTABLE**. The philosophy behind what can and cannot be copied is simply that you can give any table you own to anyone else, reserving the right to designate yourself as an SDBA of that table if the destination is a public database, or to retain for yourself any privilege if the destination is someone else's database.

2. To rename or make a copy of a table in a database:

   Use **ALTERTABLE**.

3. To make a view of one or more tables in a database:

   LOCUS now has the ability to make multi-table joins. An alternative is to use the **VIEWBUILDER** utility in the **UTILITIES** submenu.

4. To transform a TABLE or VIEW into a public object:

   If you create a table or view in a public database while outside of the MINGL shell, you must use the **COPY**{**TABLE|VIEW** } utility to make it accessible from within the MINGL shell. You also need to use this utility if you are copying several tables and an associated view into a public database. [Note: **COPYVIEW** only works from within the TFTR database.]

5. To designate yourself SDBA of a table in a public database:

   a) If you created the table using LOCUS or one of the DBASE utilities, you will automatically become the SDBA of that table.

   b) If you created the table using an INGRES utility (e.g., SQL), within or without MINGL, then you will have to use **COPYTABLE** to transfer ownership to the pseudo username PUBLICDB.

   c) If neither of the above, you can use the **DIRECTORY** command under **UTILITIES** to list the table and determine

who the current SDBA is. That person can add you to the list if necessary.

As an SDBA you can**:**

a) Modify the index structure of the table by choosing **MODIFY**.

b) Update the access control list of the table by using **PERMIT**.

c) Destroy the table or view by using **MANGL**.

6. To delete table access rights:
Use **PERMIT** to

a) See what id-number is associated with the access right (or "permit") you wish to delete;

b) Choose the submenu item "Destroy" and specify that id-number.

7. To give access rights to other users

Use **PERMIT** to grant one or more of the following rights:

a) **ALL** means complete access to the table;

b) **RETRIEVE** means read-only access;

c) Others include {**APPEND**,**DELETE**,**REPLACE**}.

PERMIT will prompt you for two (2) inputs for each rights definition:

i) the name of the right,

ii) the user to whom the right is given {**ALL**, or **username**}

To make someone else an SDBA of the table, enter **ALL** at the first prompt and their VAX username at the second prompt. The resulting INGRES command will be echoed as:

DEFINE PERMIT *ALL* ON table TO *USERNAME* .

To let everyone else have read-only access to the table, first enter **RETRIEVE**, then **ALL**. The corresponding INGRES command will be echoed as:

DEFINE PERMIT *RETRIEVE* ON table TO *ALL* .

You can only do the above if you are an SDBA of the table.

8. To see what tables are available in the public database:

a) choose **DIRECTORY** to see a list of existing tables, the associated **SDBA**'s, the number of rows, and your own read access rights, if any; or

b) choose **CONTENTS** to gain access to the table-of-contents table in the database and browse either sequentially or by a selected search.

c) to scan tables quickly, use the DCL level command
$ iitdir *database* -t*tablespec*
where the italics are filled in by the user. For no table selectivity, omit the -t flag altogether. To see tables beginning with the letter "g", use -tg*.

9. To update an entry in CONTENTS:

Choose **CONTENTS** and specify that you want "update" mode.

10. To modify or add parameters in a table:

Use **ALTERTABLE**.

Creating Tables

Until this point no mention has been made of how you go about creating a new table. There are many ways, among which the most widely used are:

In Public or Private Databases:

1. Use one of the data collection utilities provided in the DBASE section of MINGL and described in Section IV.

2. Convert an existing LOCUS/ISAM database to an INGRES table by using the **INGRATE** tool (for information on its use, see the on-line HELP file associated with the program).

3. Use the EDIT option in **LOCUS**.

4. Use the IFILE_IN program to read in ASCII data files. Documentation is in Appendix 3.

5. Write a FORTRAN program to create a table using the **INGLIB** library routines (cf. **$HELP INGLIB** or the sample programs in **LOCUS$:[INGRES.EXAMPLES]**).

In Private Databases only:

6. Use any one of the INGRES tools (e.g. **ISQL**) .

7. Use the INGRES COPY command directly to read in data from a text or binary file. More information is available by typing $HELP INGRES HOW_TO.

Tables created by method 6 or 7 may be moved to a public database by using the **COPYTABLE** utility.

Information on the Contents of Tables

In order to facilitate the dissemination of information among TFTR database users, there is a table-of-contents table in TFTR called **CONTENTS**. It contains one or more entries for each table and view in the database. Each entry consists of the name of the table, its creator's name, its creation date, and a long description field telling what's in it. Each time you create or copy a new table into TFTR using any of the MINGL utilities, or LOCUS, you will be prompted to provide the necessary information required to complete a new entry in **CONTENTS**. It can be accessed off-line by typing either **$UCON** or **$CONTENTS**.

IV. DATA COLLECTION UTILITIES

Several utilities are available to you in the TFTR version of MINGL for generating tables containing data from the Waveform, Results, Raw Data, Ufile, GLOBE, SNAP and TRANSP archives:

Utility:	**Data Source:**	
DBASE_TMPL:	**SNAP** and **GLOBE** databases (predefined template definitions)	$[f(t_0)]$
DBASEG:	**GLOBE** database	$[f(t_0)]$
DBASES:	**SNAP** database	$[f(t_0)]$

DBASET:	**TRANSP/RPLOT** scalar & profile	[f(t),f(r,t)]
DBASEW:	Waveforms, raw data, 1-d ufiles, TRANSP scalars,	[$f(t_0)$]
DBASEWT:	Waveforms, raw data, 1-d ufiles TRANSP scalars, [f(t)]	
DBASEU1_PRO:	1-d ufile profiles and radial waveforms with optional slice & stack	[f(r), f(R)]
DBASEU2:	2-d ufiles, TRANSP scalar & profile	[f(r,t), f(t,r), or $f(t,r_0)$]
DBASE_PRO:	**SNAP** profiles	[f(r), f(R)]

Collectively, these are referred to as the DBASE set of gateway routines.

These routines require a variety of input files to specify such things as lists of shots, runs, tries, times, averaging times, radii, etc. These parameters drive the two loops implicit in all these

utilities. The outer loop, which determines the rows that appear in the database table, is driven by a **{Shot,|Time|,|Radius|} List,** with one row generated for each entry (or possible combination of entries) in the list. The inner loop, which determines the fields or columns that appear in the database table, is driven by a **Parameter List,** with one column defined for each entry in that list. Each routine has in addition a set of default columns that will always appear (e.g., shot)

MINGL provides a set of INGRES forms which use a common graphical user interface that allows you to specify on one screen all the information necessary to start up one of the gateway jobs in batch. The forms all share some common features:

1. They provide the ability to specify all the usual attributes of a batch job submission.

2. They provide the option of either creating a new table or appending to an existing table in a private or public database.

3. If a new table is created in a public database, they will prompt for an appropriate entry describing the contents of the new table. The entry is appended to a special system maintained table named **CONTENTS** and is accessible by all users who are interested in browsing through the database.

Each of the DBASE routines handles different types of data, and you have to select the one that fits all or part of your needs. It is possible to mix some types of data, both with respect to source and to dimensionality, but not all. It depends on the particular gateway routine. If you do have to make separate tables to accommodate the type of information you are interested in looking at, they can then be related by using the join facilities of LOCUS.

All DBASE routines recognize special keywords, defined by “0000” in the first field of the parameter-list line and the keyword name itself in the third field. They allow you to "carry along" special scalar values in your tables. These values, such as maximum beam power (PBMAX_MW), or beam turn on time (BEAM_ON), are then duplicated in every row of the table associated with the specified shot. Please consult the descriptions of the individual routines for details.

Other keywords appear in some of the shot-list and parameter-list files. If the action associated with a particular keyword is not desired, the user can substitute the NOOP keyword as a place holder in its place. C.f. the DBASEWT.PARAMS example file later in his document for an example.

Your interaction with the DBASE forms:

1. The "Table" input field is common to all forms. You specify here the name of the table in the database that is to receive the specified data items. If the table already exists, you will be informed and given the option of either renaming the table or of continuing in "append" mode, where the new rows generated will be added to the existing table. If the table name you enter is unique, a new table will be created.

2. All of the gateway programs require a *SHOT-LIST FILE* in order to execute. The entries in the shot-file generate one or more rows in your data table. The format of these files is explained in detail on the following pages. Two particular elements of the syntax, known as "time-event keywords" and "flaggable fields", require additional explanation.

a) Time interval specifications:

In those DBASE programs that deal with time, the time interval is specified in the shot-list line. The time center, **toi**, is given either by an absolute number, in seconds, or by a keyword as explained in the next section. A smoothing interval, **dt**, can also be specified, and results in smoothing over the range T [**toi-dt**,**toi+dt**] , with the assumption that the signal is given on a uniform time [T] base.

b) Time Event Keywords

Those DBASE programs that deal with time all recognize a common set of **time-event** keywords in the shot-list line that identify times-of-interest during the plasma discharge. These keywords may be used in place of absolute times, and are listed below.

time-event keyword	description

BEAM_ON	time at which beams turn on
BEAM_OFF	time at which beams turn off
DISRUPTION	time at which plasma disrupts
E_T_MAX	time at which transverse energy reaches a maximum (MD-ET-SL)
FLAT_TOP	time at which I_P reaches flattop
ICRF_ON	time at which ICRF turns on
ICRF_OFF	time at which ICRF turns off
MAX_NEUTRONS	time of maximum neutron count rate (NE-SS-CW)
OHMIC	3.0 sec for true Ohmic discharge; 60 msec before auxiliary power[7] turnon for non-Ohmic discharges
PEL1[-6]	time at which pellet 1[-6] is injected
PWR_ON	time at which auxiliary power turns on
PWR_OFF	time at which auxiliary power turns off
SNAP	SNAP analysis time
TS	time of Thomson TS diagnostic
TV	time of Thomson TV diagnostic

c) Flaggable Fields

[7] By “auxiliary power” we mean the sum of all forms of heating power injected into the plasma.

These special tokens contain a single embedded equals sign ("="), and designate extra fields or columns which you can "flag" yourself by assigning a value right there in the shot-list line. They can take either integer or character values. They are useful for later use in LOCUS, or wherever additional mnemonic or integer values are more useful than value ranges. A complete specification of the format for "Flaggable Fields" can be found in the on-line help file associated with the DBASE menu form.

3. All of the gateway programs, with the exception of **DBASE_TMPL**, also require a *PARAMETER-LIST FILE* in order to execute. The format of these files is explained in detail on the following pages, but certain fundamental rules of syntax are described here. Each line consists of a number of “fields”.

a) Default column names:

Most of the DBASE programs generate a mandatory set of columns, such as "SHOT", "TIME", etc. These parameters are indicated as such in the program descriptions that follow.

b) Rules for specifying the data source in field 1:

Ufiles:

use MB:T.CUR to indicate the ufile MB_ALL:Tnnnnn.CUR, where nnnn= shotno

or use MB_ALL:T.CUR to indicate the ufile MB_ALL:Tnnnnn.CUR

or use USER2:[RTIME.DBASE]T.CUR to indicate the specific Ufile.

In all cases, note the Ufile convention of specifying only the prefix and suffix of the ufile name; i.e., the 5 digit shot number is inserted automatically.

Waveforms:

use YI-NE-SL to indicate a single waveform

or use YM-TE-SL:YM-TM-SL to indicate the YM-TE-SL waveform using the YM-TM-SL waveform as a time base

or use M-POHMIC to indicate a derived waveform.

Raw Data Files:

The general format is [digraph]name[channel], where the digraph name, if not explicitly present, is taken from the 2nd and 3rd characters of the name, and where the channel is required only for scalars.

use DBR1-CHN-040 to indicate the named raw data file

or use [BR]DBW1-CHN-3 to indicate the named raw data file

or use [NE]DNE-SCL-SLW[02] to indicate channel 2 of the DNE-SCL-SLW scalar.

c) Rules for specifying the format in field 2:

The format specification fields are common to almost all DBASE programs. They generally fall into one of 2 categories: "i4" for an integer field and "f4" for a real number field. The data is stored in this format in the data table.

d) Rules for specifying the OPCODES in field 3:

Most DBASE programs recognize a common set of keywords that identify scalars-of-interest during the plasma discharge. These may be used in place of data-source designators. They are used by replacing the data-source designator in field 1 of the parameter-list

by "00000" and the field name designator in field 3 of the parameter-list line by one of the following keywords:

cid=0000 keyword	description
BEAM_ON	time beam turns on [uses NB-BP-SL]
BEAM_OFF	time beam turns off
DISRUPTION	time of disruption
E_T_MAX	time of maximum transverse energy
FLAT_TOP	time I_p reaches flattop
ICRFAVG_MW	average ICRF power delivered
ICRFMAX_MW	maximum ICRF power delivered
ICRFMAX_SQ	maximum squared ICRF power delivered
ICRF_ON	time ICRF turns on
ICRF_OFF	time ICRF turns off
IRAMP	1 for a ramped shot
MAX_NEUTRONS	time of maximum neutron rate
OHMIC	"Ohmic" time (see time-event-keyword, above)
PBAVG_MW	average beam power delivered
PBMAX_MW	maximum beam power delivered
PBMAX_SQ	maximum squared beam power delivered

PEL1[-6]	time the 1st (-6th) pellet is injected
PWRAVG_MW	average auxiliary power delivered
PWRMAX_MW	maximum auxiliary power delivered
PWRMAX_SQ	maximum squared auxiliary power delivered
PWR_ON	time auxiliary power turns on
PWR_OFF	time auxiliary power turns off
SNAP	time SNAP fires
SNAPTRY	the SNAP Try (usually from the Shot List)
TS	time TS fires [uses TS-TE-PR]
TV	time TV fires

e) Rules for specifying EQCODES in field 4 and greater:

These special tokens contain a single embedded equals sign ("="), and designate extra operations that are applied to this particular data element or column only. (A complete specification of the format for "eqcodes" can be found in the on-line help file associated with the DBASE menu form.)

i)	SCALE=f	scale the data by f (any floating point number)
ii)	NODATA=f	store "nodata" as f
iii)	SM=f	specify smoothing time (ms) to be used in DBASEW with opcodes WF_MIN/MAX, WFMIN/MAX_TIME, WFDT_MIN/MAX, WFMIN/MAXDT_TIME.

4. Error checking is done on each input item (e.g., does the indicated file exist, does it have the correct format?) and an error message informing you of the problem is displayed, returning you to the item on the form that contains the problem.

5. Both shot and parameter files allow comments. A comment begins with an "!" in any column, and must be positioned after any other fields in that row. The comments are for use in identifying entries in the file to the user, but are not read by any of the DBASE programs themselves.

6. The "Batch Parameters" field group is common to all the forms. It enables you to specify the particular batch queue you wish to launch your task on, as well as an optional delay in its starting time. In general , TFTR$INGRES is the optimal batch queue to use, although if you are particular, you might want to specify one of the *_NORM queues instead. Use the X batch queue if you want to submit the job yourself (look for the DBASE*.BAT file generated in your default directory). The default starting time is "now". To specify a midnight starting time, enter "tomorrow" in the "After" field. To specify "3 am tomorrow morning", enter "tomorrow+3:00".

7. When you have completed filling in the form, press the ENTER key to launch the batch job.

8. The batch job, and its associated log file, is assigned a name consisting of the program name and a unique 4 digit number (e.g., DBASES_4644, and DBASES_4644.LOG, resp.), which is displayed in the "Job Name" field on the redisplayed form after the job has been successfully launched. The log file will appear in your default login area, unless you choose to redirect it to the area specified in the LOG field. This is a good idea if you are processing a lot of shots, since the log file may grow too large and overflow your disk quota, aborting the job. You will be notified by VMS Mail at the start of the job and again upon its completion.

9. Help files are accessible from each entry form by pressing the PF2 key on your terminal's keypad. These files contain information on the input requirements of the current form ("What to do") as well as information on the keyboard interaction with the screen ("Keys"). Other help information can be found in the MINGL bulletin folder, accessible by typing

   $ BULL MINGL .

10. You can SPAWN into a sub-process by pressing control-U (^U). To return to MINGL, type "$LOGOFF" in the spawned session.

11. You can make a copy of the completed form for your records by pressing the "1" key on your keypad.

12. The SNAP related DBASE forms have fields that refer to “Wait for Shots”, “Rabbit Shots”, and “Warning Shots”. The default for all three is “NO”.

“Wait for Shots”:	If NO, fill currently inaccessible shot-tries with Nulls. If YES, complete the request for retrieval of all shot-tries in the shot list, then abort.
“Rabbit Shots”:	If NO, fill any Rabbit[8] shots with nulls. If YES, process Rabbit shots.
“Warning Shots”:	If NO, fill any Warning shots with nulls. If YES, process Warning shots.

13. Some of the DBASE routines check for the existence of certain logical names, which provide a means whereby the user can modify default behavior.These logical names are listed below, with the DBASE routines they affect. Their use is described in the appropriate DBASE section.

---

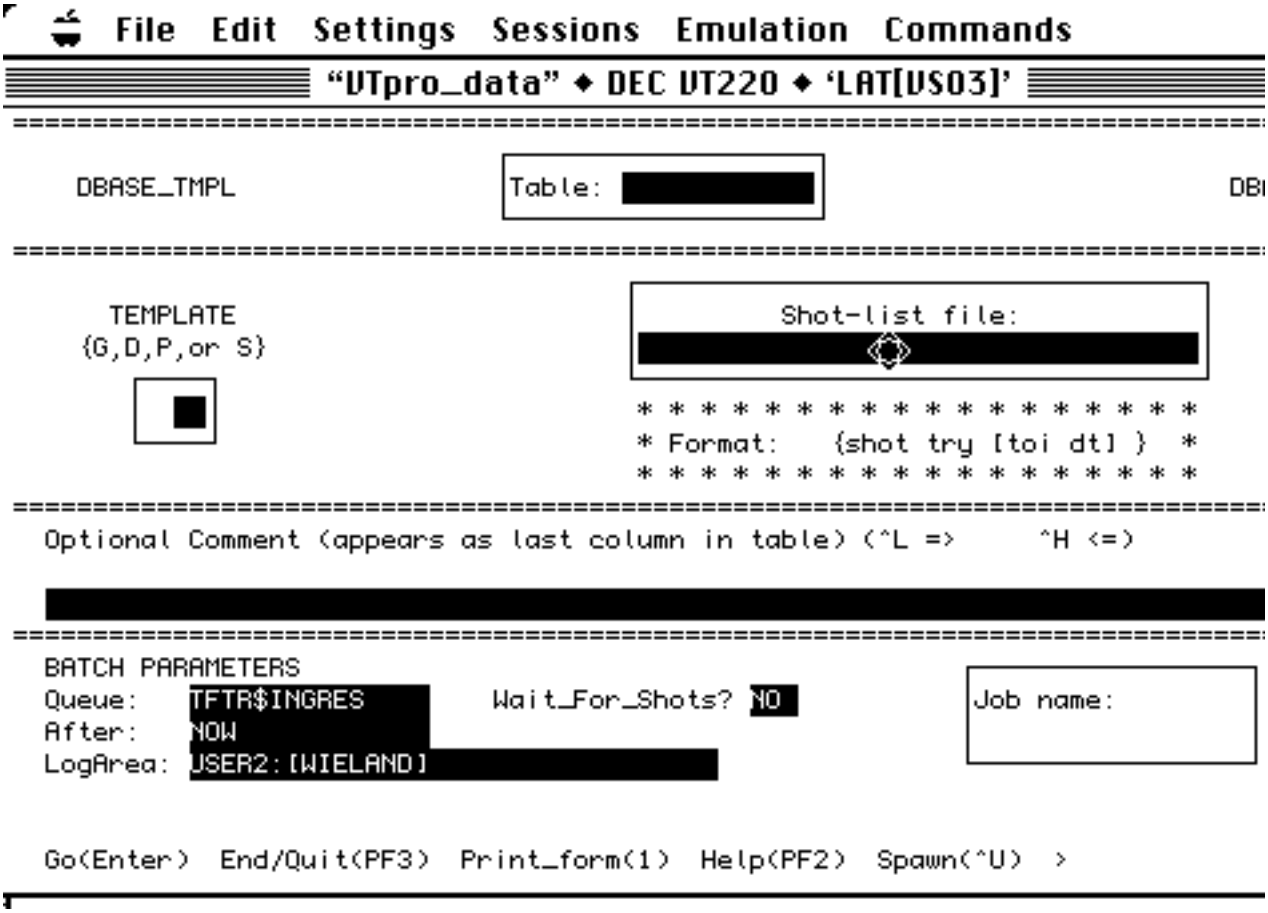
[8] Rabbit shots are special SNAP cases that are usually not of general interest. Warning shots are SNAP cases that have been flagged as requiring special interpretation. For more information, cf. "The SNAP User's Guide, J.A. Murphy et.al., PPPL-TM-393, January 1992.

ING$Hands_Off_Restructuring	All DBASE
ING$Interp_Mode	DBASES, DBASE_PRO
MINGL$Dbaseu2$Grid_Projection	DBASEU2
MINGL$Dbasewt$Grid_Projection	DBASEWT

They are defined either interactively, prior to starting the MINGL session, or by manually editing the *.BAT file that MINGL produces (Cf. the "X" batch queue option in 6, above).

DBASE_TMPL

[Snap & Globe Scalars]



Special Form Fields :

The "Optional Comment" field is a 200 character long text field that is part of every record created by this program. It can be used as an identifier field to distinguish a group of records, such as those that may be created at different times by the DBASE_TEMPLATE program. This can alleviate the need to create several tables. Text fields are recognized by LOCUS/INGRES and so any identifier entered here can be used as a constraint in that program.

Required input files for DBASE_TEMPLATE :

file type: **format:** **description:**

Shot-list	shot try {toi dt}

> shot = shot #
> try = try #
> toi = time of interest in sec (G template only)
> dt = averaging window (G template only)(average over toi +/- dt)

MINGL_FILES:DBASEG.SHOTS

```
18505 01 3.450 .050
18506 01 3.450 .050
18507 01 3.450 .050
18509 01 3.450 .050
```

MINGL_FILES:DBASES.SHOTS

```
24431 01
24431 02
24470 01
24473 01
```

Parameter-list	indicate **G** or **S** or **P** as one of the predefined GLOBE (**GLOBESUM)** or SNAP (**DIAGCOMP, SUMREL, POWRBAL)** templates on the form field "TEMPLATE"

The templates referred to above consist of four separate table structures:

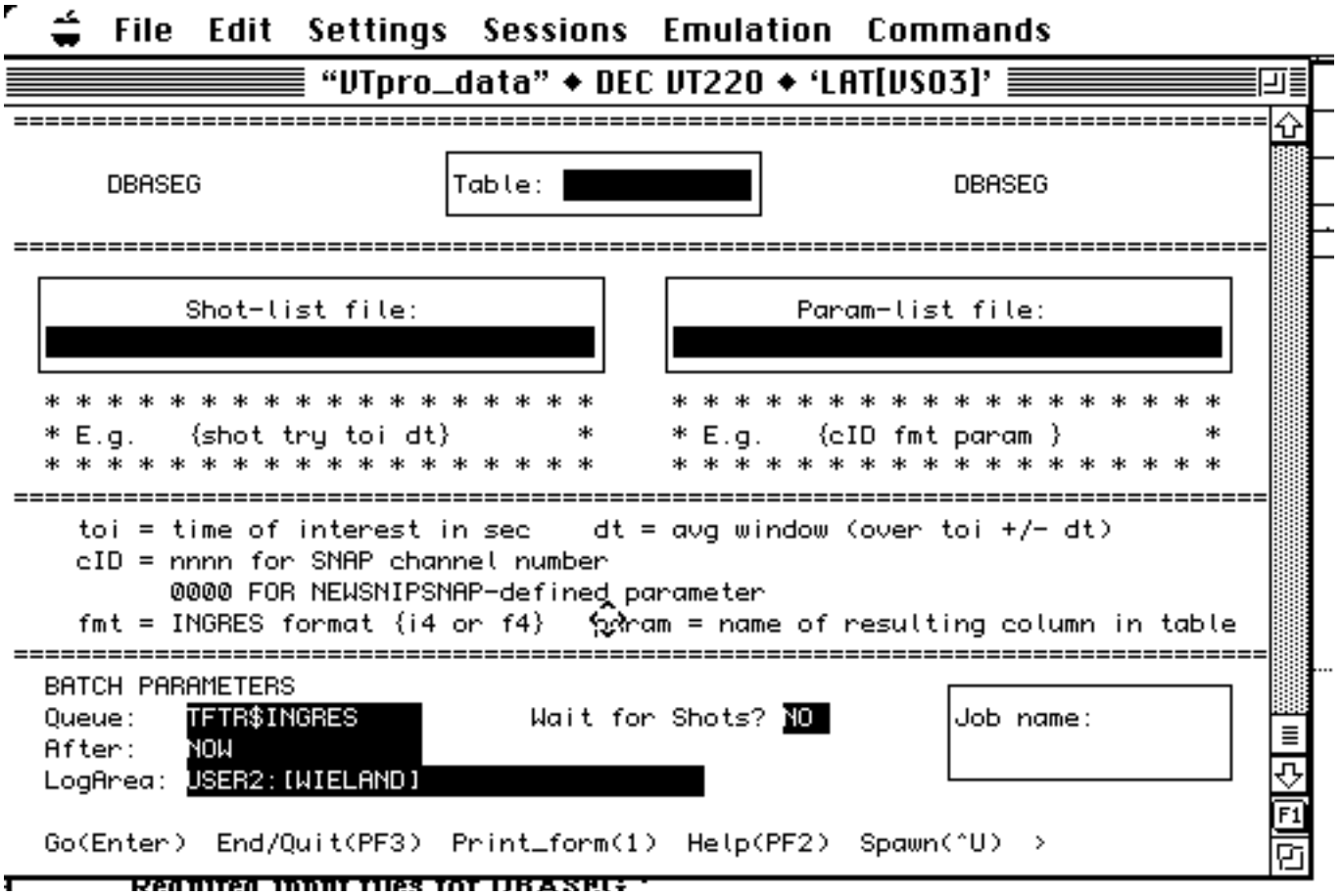
GLOBESUM	**GLOBE**-like parameter set
DIAGCOMP	diagnostic parameter set
POWRBAL	**SNAP-**like parameter set
SUMREL	**SNAP-**like parameter set

A list of the parameters in these templates can be obtained from the file MINGL_FILES:SUMREL-POWRBAL.PARAMS. The HELP submenu associated with **DBASE_TEMPLATE** also contains a list of these parameters.

[This page left intentionally blank]

DBASEG

[GLOBE Scalars]



Required input files for DBASEG :

file type: **format:** **description:**

Shot-list file:	shot try toi dt	flaggable-fields
>	shot =	**GLOBE** shot #
>	try =	**GLOBE** try #
>	toi =	time of interest in sec
>	dt =	averaging window (average over toi +/- dt)

MINGL_FILES:DBASEG.SHOTS
31846 01 3.450 .050 ok=yes mode=1 id=11 31847 01 3.450 .050 ok=no id=12 31849 01 3.450 .050 mode=3

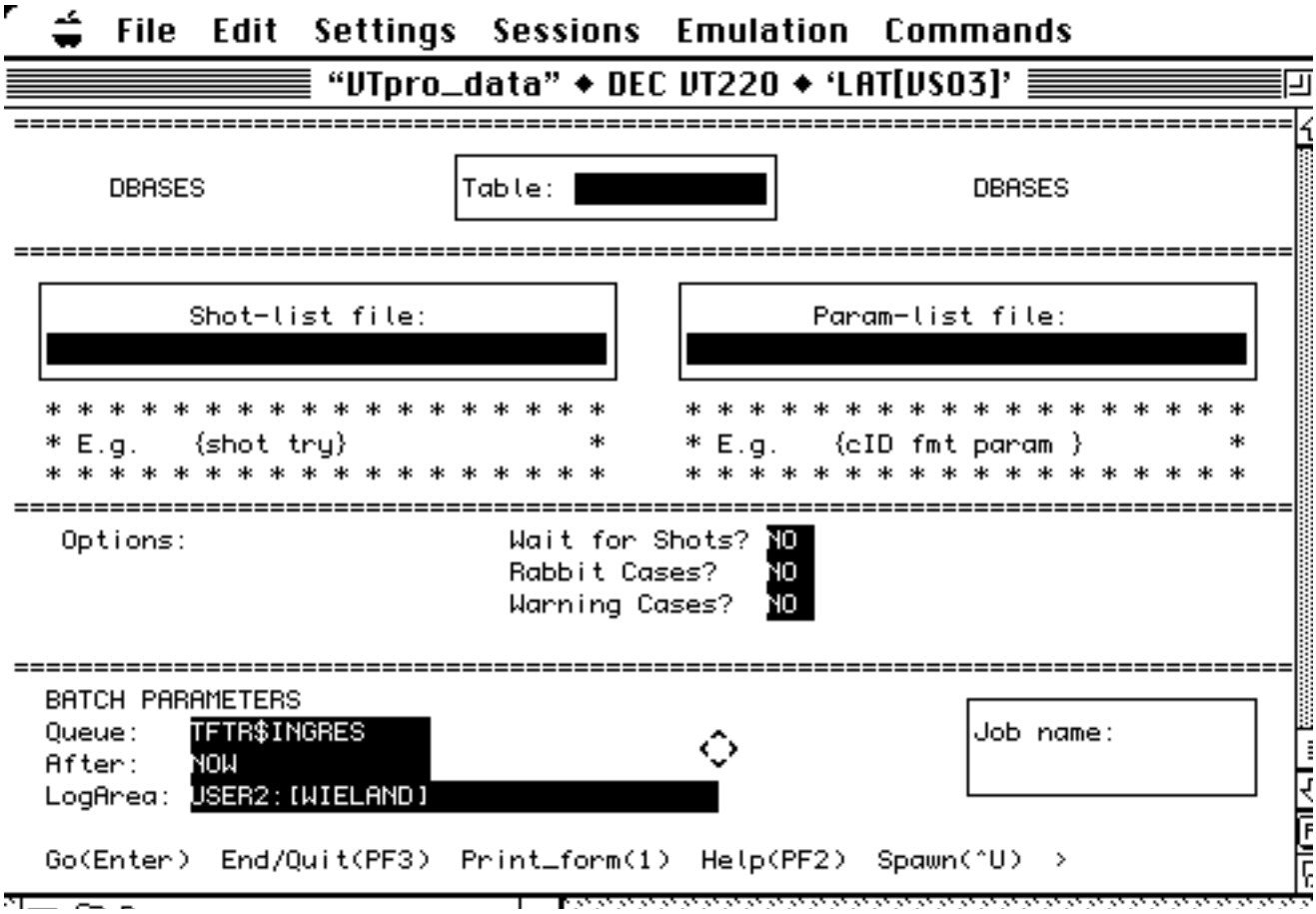
Parameter-list:			cID fmt param
>	cID	=	*nnnn* for **GLOBE** channel ***0000*** for SNIPSNAP keyword
>	fmt	=	INGRES storage format (usually F4 or I4)
>	param	=	keyword used to define the name in the table as well as select the SNIPSNAP parameter when cID=0000.

> example: MINGL_FILES:DBASEG.PARAMS

MINGL_FILES:DBASEG.PARAMS
-default- I4 SHOT !(*) -default- I4 TRY !(*) -default- F4 TIME !(*) -default- F4 DTAV !(*) 0008 F4 GRMJR !Plasma major radius 0009 F4 GRMNR !Plasma minor radius 0044 F4 GBPDI !Globe diamagnetic beta-poloidal 0046 F4 GBTDI !Globe diamagnetic beta-toroidal 0048 F4 GWDIA !Globe diamagnetic stored energy 0067 F4 GTAUD !Globe tauE from diamagnetism 0004 F4 GNEL !Globe line integral density 0051 F4 GVRES !Globe resistive voltage 0080 F4 GTAPS !Globe tau-P
* default entry

DBASES

[SNAP Scalars]



Description:

DBASES extracts SNAP scalars directly, as well as a large selection of other precomputed scalar quantities found in the SNIPSNAP parameter set (cf. Appendix 2). Many of the latter involve profile integrals, which in turn require radial interpolation. The default interpolation method used is cubic spline. The user may substitute linear interpolation by defining the logical name as

$ DEFINE ING$Interp_Mode 1

either interactively, before starting MINGL, or manually, by typing it into the DBASES.BAT file.

Required input files for DBASES :

file type: **format:** **description:**

Shot-list file:	shot	try	flaggable-fields
>	shot	=	**SNAP** shot #
>	try	=	**SNAP** try #

MINGL_FILES:DBASES.SHOTS

```
55806 8    ok=yes mode=6 id=3
45320 1    ok=no  mode=14
55806 3    id=99
45315 3    ok=no
```

Parameter-list:	cID	fmt	param
>	cID	=	*nnnn* for **SNAP** channel *0000* for SNIPSNAP keyword[9]
>	fmt	=	INGRES storage format (usually F4 or I4)
>	param	=	keyword used to define the name in the table as well as select the SNIPSNAP parameter when cID=0000.

MINGL_FILES:DBASES.PARAMS

```
-default- I4 SHOT                !SNAP shot number (*)
-default- I4 TRY                 !SNAP try number  (*)
-default- F4 RUN_TYPE            !SNAP run_type  (*)
-default- F4 WARNING_FLAG    !SNAP warning_flag (*)
PCUR_TOI     F4 IP               !Plasma Current (A)
DNEBAR        F4 NEBAR           !Line average density (#/m**3)
B0TF_TOI     F4 BTOR             !B0 @ R0 (T)
```

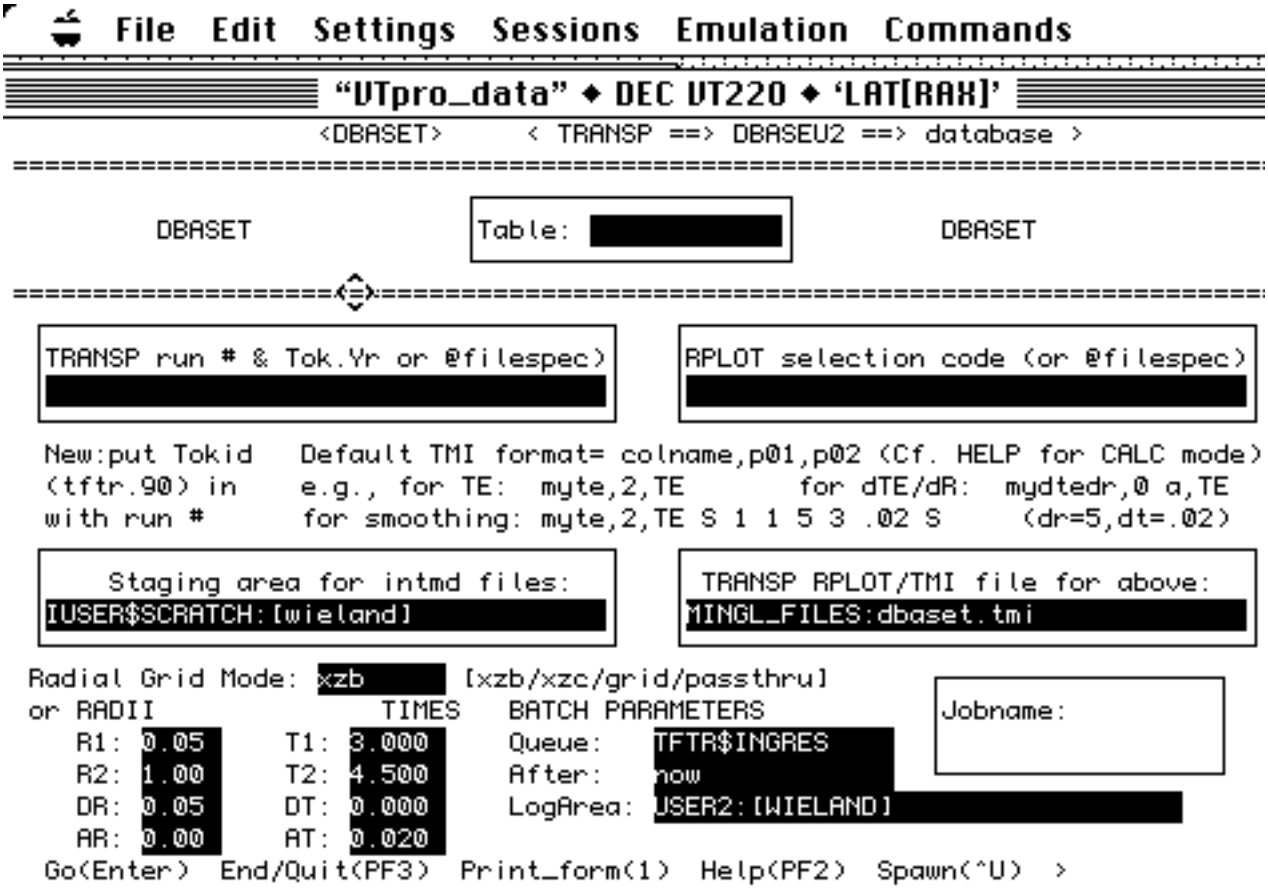
[9] Cf. Appendix 2 for a complete list of the SNIPSNAP keywords available.

RMINOR_TOI F4 RMINR	!Minor radius (m)
RMAJOR_TOI F4 RMAJR	!Major radius (m)
ZEFF_VB F4 ZFFVB	!Zeff deduced from Vis. Br. Signal
4210 F4 IP	!Plasma Current (A)
4073 F4 NEBAR	!Line average density (#/m**3)
4027 F4 BTOR	!B0 @ R0 (T)
4114 F4 RMINR	!Minor radius (m)
4113 F4 RMAJR	!Major radius (m)
4199 F4 ZFFVB	!Zeff deduced from Vis. Br. Signal
4099 F4 POH	!Resistive heating power (w)
4104 F4 QACYL	!q(a), cylindrical
4105 F4 QASHF	!q(a), Shafranov
0000 F4 NEVAV	!SNIPSNAP vol avg ne
HMIX F4 HMIX	!derived scalar channel

(*) a default entry

DBASET

[TRANSP f(x,t) profiles and f(t) scalars]



Description:

DBASET is used to copy 2d profiles from TRANSP data sets into INGRES data tables. It accomplishes this by running the TRANSP utility RPLOT and generating the necessary intermediate 2d-ufiles, and then invoking DBASEU2 to read those ufiles into the specified data table. The DBASET form is documented in the on-line HELP file, obtained by pressing the PF2,Ctrl-R keys on your keypad.

If you want to access 1d RPLOT profiles for inclusion into a database using DBASET, you can take advantage of the underlying

DBASEU2 facility and simply specify the TRANSP scalar as you would for DBASEU2.

If you want to take advantage of RPLOT's calculator mode, or if you want to map TRANSP's non-temporal axis onto the same radial grid no matter what the internal representation in RPLOT (i.e., zone-centered or zone-boundary), then use DBASET.

If you want the intrinsic TRANSP channel radial axis representation, then use DBASEU2 directly.

“Radial Grid Mode” Field:

There are 4 modes available for mapping the TRANSP channels:

1) "XZB" mode: In this mode, which is the default, all TRANSP channels are mapped onto the XZB grid (“XB” in RPLOT).
2) "XZC" mode: In this mode all TRANSP channels are mapped onto the XZC grid (“X” in RPLOT)
3) "GRID" mode: In this mode, all TRANSP channels are mapped onto the user specified radius grid. Modes 1 and 2 are just pre-selected grids in "GRID" mode.
4) "PASSTHRU" mode: In this mode, the TRANSP data channel is passed through as is (i.e., on its intrinsic grids) to DBASEU2. The user can specify a time window by providing values for "T1" and "T2". Dt=Dr=0 is the trigger for this mode. TRANSP channels X, XB, RZON AND RBOUN should be included in the parameter list so that the user can map the data channels onto the appropriate grid in later analysis.

   Note that all these modes automatically take care of the problem in including TRANSP runs from "eras" with different intrinsic radial grid representations.

Required input files for DBASET:

file type:	format:	description:
Shot-list file	run device.yr	flaggable-fields
>	run	= TRANSP run #
>	device.yr	= tokamak.year

MINGL_FILES:DBASET.SHOTS

```
45320a00    tftr.90     ok=yes  type=7  mode=3
45315a00    tftr.90     type=5  ok=no
41442a01    tftr.89
8004        tftr.87     ok=no   mode=6
```

The RPLOT selection code is a collection of RPLOT commands that you would use if you were in RPLOT making the data plots yourself. There is a default setup script that DBASET uses to setup the Auto-RPLOT session; it is MINGL_FILES:DBASET.TMI. There is a field on this form (TRANSP RPLOT/TMI) that lets you substitute your own customizable setup script in its place. In addition, you can use other extended features of RPLOT (such as the calculator mode). Cf. the MINGL folder in $BULLETIN for information on how to do this.

RPLOT Selection Code:
MINGL_FILES:DBASET.PARAMS

```
q, 2, q
dqdr, 0 a, q
pt, 2, ptowb
dptdr, 0 a, ptowb
```

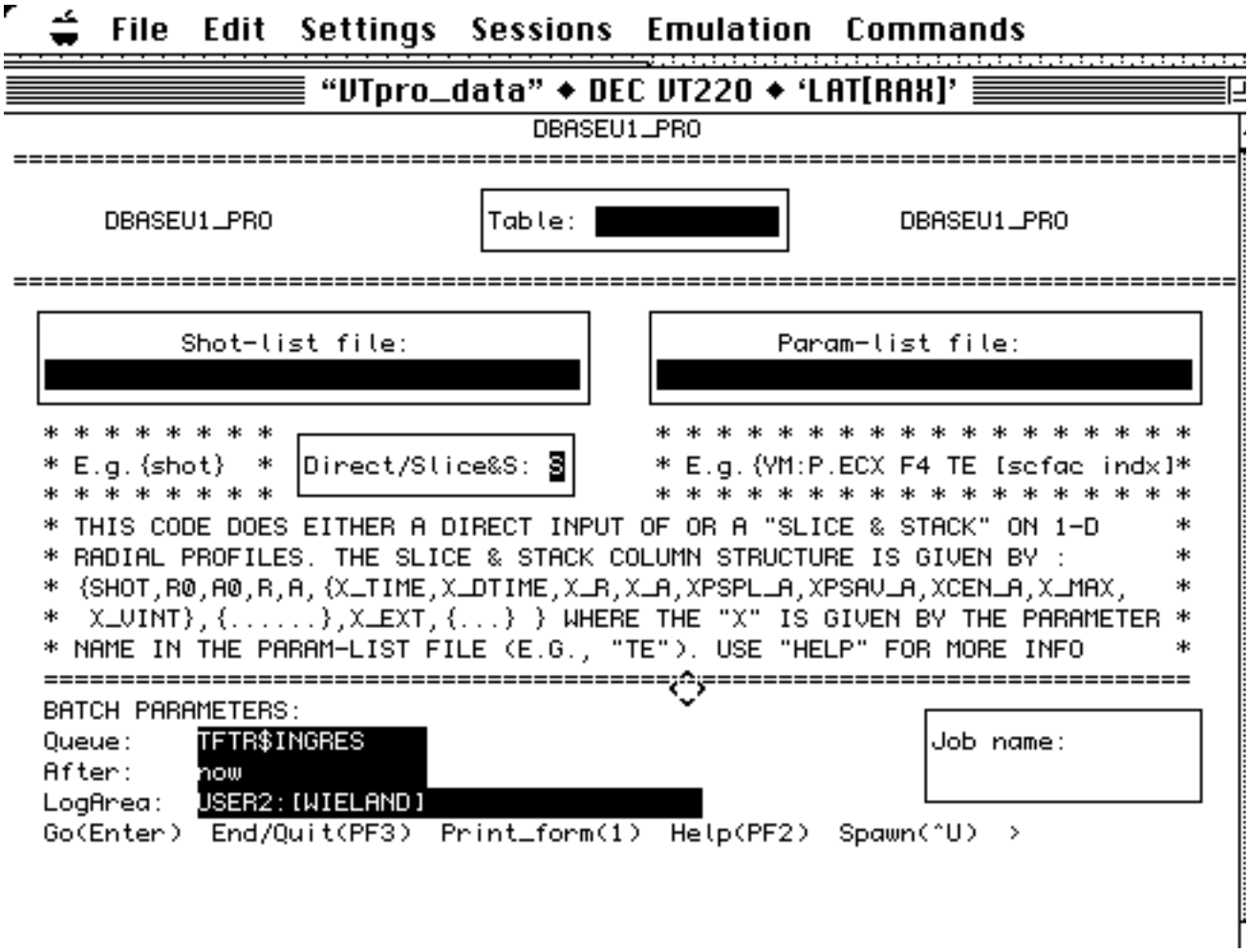
Two kinds of averaging are possible with DBASET. You can tailor your Selection Code file to do RPLOT averaging, or you can specify averaging on the DBASEWT form itself. In the RADII and TIMES fields, you specify the grids upon which you want the data written to the table (R1,R2,DR & T1,T2,DT) as

well as the averaging interval half-width (AR & AT) to be used in the DBASEU2 post-processing the 2d ufiles from RPLOT.

[This page left intentionally blank]

DBASEU1_PRO

[1d Ufiles(r or R), radial waveforms]



Description:

This code is a variant on DBASE_PRO. It takes as its input a parameter list containing the names of 1-d "radial" ufiles or waveforms and a shot list containing the shot numbers and constructs a group of predefined columns for each parameter specified. Two modes of operation are possible:

1. "Direct Input" mode **D**: Profiles are read in on their existing radial grids. The defining grid is determined by the first ufile specified. All the remaining ufiles are interpolated onto that grid.

2. "Slice & Stack" mode **SA** (the default; out to r=a) or **SE** (out to the edge): For each entry in the shot list, there will be a number of rows with values of R ranging from R_0-A_0 to R_0+A_0 and A ranging from 0 to some value <=1, depending on the range of the data. In the column definitions

below, names ending with "_A" are defined on the A (or minor radius) grid. The first character "X" will be replaced by the parameter (or column) name you prescribe in the parameter file. Since A_0 is time dependent, DBASE_PRO uses the time scalar information in the first 1-d ufile to specify the time at which A_0 is evaluated. Therefore, there are two constraints to be considered here: (1) all ufiles and waveforms are considered to be at the time specified in the first ufile, and (2) the first parameter specified must be a ufile containing the time information in its scalar field.

Required input files for DBASEU1_PRO:

file type:	**format:**	**description:**
Shot-list file:	shot	flaggable-fields

> shot = shot #

MINGL_FILES:DBASEU1_PRO.SHOTS

```
42483   ok=yes   mode=6
42393   ok=no    mode=7
22565
```

Parameter-list:	cID fmt param scale index

> cID = ufile ID
> fmt = INGRES format
> param = defines the column name in the table
> scale = scale factor[required]
> index = alternative to a profile retrieve; extract the *n-th* element of the profile
> example: MINGL_FILES:DBASEU1_PRO.PARAMS

MINGL_FILES:DBASEU1_PRO.PARAMS
-default- I4 SHOT !(*) (**) -default- F4 R0 !(*) -default- F4 A0 !(*) -default- F4 R !(** -default- F4 A !(**) YS:P.ECX F4 TE 1000 3 YM:P.ECM F4 TM 1 LOCUS$:INGRES.WIELAND.BENCH]K2_TTTT_.TI F4 TI_SW 1.0 LOCUS$:INGRES.WIELAND.BENCH]K2_TTTT_.TEB F4 DELTI_SW 1.0 FM-NE-T5:FM-XR-T5 F4 NET5 1.0
* default entry for Slice&Stack mode ** default entry for Direct mode

The column definitions for "Slice & Stack" mode are:

Default entries:

SHOT	=	shot number
R0	=	major radius from the waveform MB-RP-SL at time+/-(deltatime/2) of the first ufile specified in the parameter list (cf. x_dtime, below)
A0	=	minor radius from the waveform MB-AP-SL at time+/-(deltatime/2) of the first ufile specified in the parameter list (cf. x_dtime, below)
R	=	(running) major radius on a fixed grid extending from R0-A to R0+A
A	=	(running) NORMALIZED MINOR RADIUS (SNAP-like, zone-centered) extending out to the maximum value extractable by the slice & stack algorithm

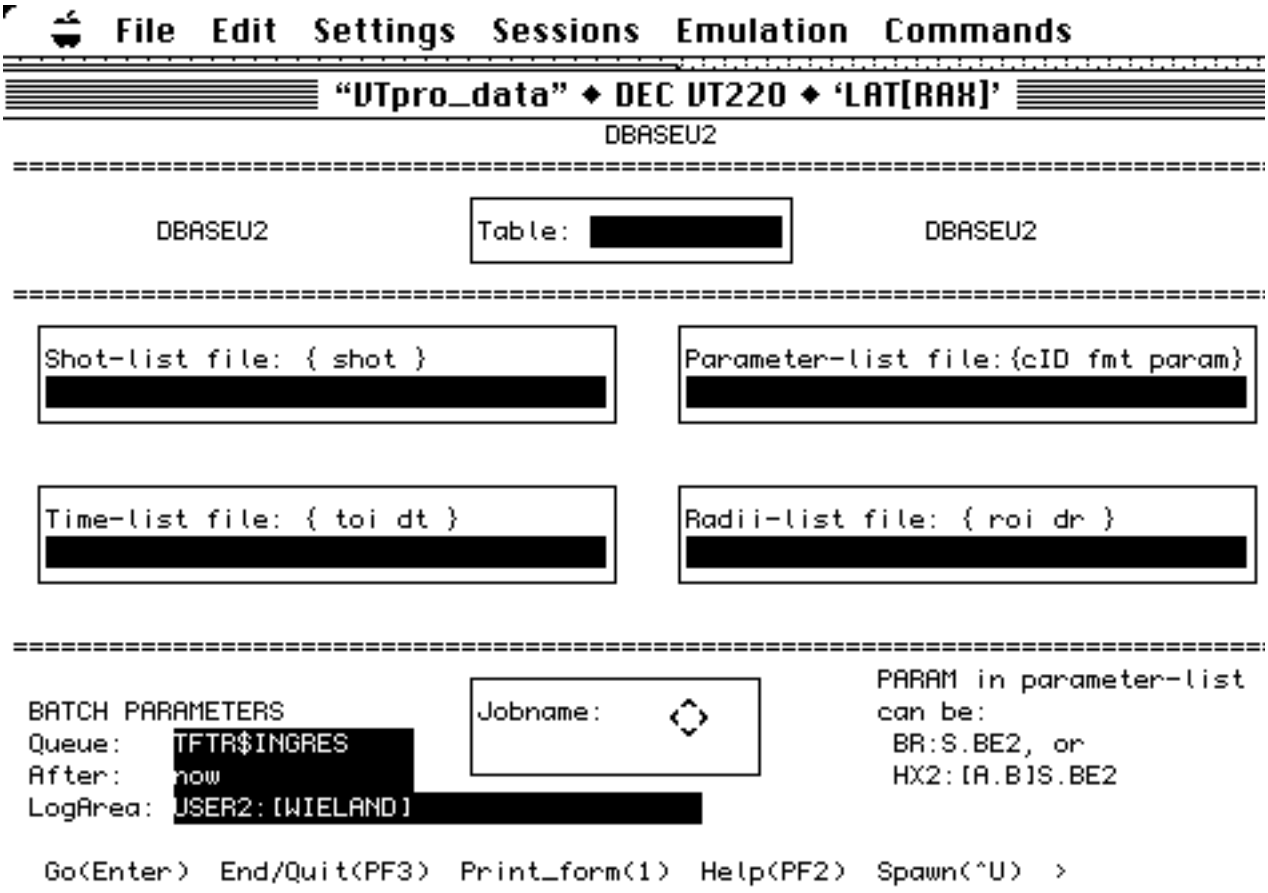
Parameter group entries:

X_TIME	=	the time at which the profile was extracted. this is obtained operationally from the "T0" scalar in the 1-d ufile
X_DTIME	=	the delta time over which the profile has been smoothed. this is obtained operationally from the "T1" and "T2" scalars in the 1-d ufile
X_R	=	the value of the profile parameter on the "R" grid (this is a spline interpolation from the input profile)
X_A	=	the value of the profile parameter on the "A" grid (this is a spline interpolation from the input profile, and the minor radius mapping is a result of the slice & stack algorithm, courtesy of SNAP)
XPSPL_A	=	dX/dA (spline derivative of the slice & stack X_A)
XSAV_A	=	integral of XPSAV_A (cf. below), starting at X_MAX
XPSAV_A	=	surface average derivative of X_R using an algorithm devised by Gary Taylor
XCEN_A	=	midplane centroid position $[R_{outside}(a) + R_{inside}(a)]/2$
X_VINT	=	volume integral of X
X_MAX	=	maximum value of X
X_JET	=	X mapped onto X_A = $(R-R_{max})/(R_0+A_0-R_{max})$; $R \geq R_{max}$
X_JETP	=	d(X_JET)/dA
X_JETI	=	X mapped onto X_A = $(R-R_{max})/(R_0-A_0-R_{max})$; $R \leq R_{max}$

X_JETPI	=	d(X_JETI)/dA
X_nn	=	X_A (nn-th index)
XSAV_nn	=	XSAV_A (nn-th index)
XJET_nn	=	X_JET (nn-th index)
XJETI_nn	=	X_JETI (nn-th index)
X_EXT	=	ufile extension name (e.g., "ECX")

DBASEU2

[2d Ufiles, TRANSP Scalars & Profiles]



Description:

DBASEU2 is designed to read in data sources of up to two dimensions. Typical inputs are 2d ufiles, or TRANSP scalar or profile data. The user can either specify a uniform time and radius grid onto which the data is to be mapped, or can specify that the data is to be "passed through" using both its native grid coordinates.

Ufile "pass through" is limited to one ufile, although multiple entries can appear in the parameter list employing the DDR or DDT

opcodes. This mode is indicated by special entries in the radius- and time-list files, and includes radius and time in combination.

TRANSP "pass through" results in the data being stored on its intrinsic time and radial[10] grids. By specifically including the two kinds of grids in the parameter list, the user is able in programs such as LOCUS to plot the data on the correct grid.

Smoothing is possible in any mode. Three kinds of (time) smoothing are supported:

(1) The default is to use the FILFAS triangular smoother.

(2) Boxcar smoothing on the *raw ufile* time grid is used if the user defines the logical name as
$ DEFINE Mingl$Dbaseu2$Grid_Projection BOXCAR
either interactively, before starting MINGL, or manually, by typing it into the DBASEU2.BAT file.

(3) Boxcar smoothing on the *interpolated* timegrid is used by similarly defining
$DEFINE Mingl$Dbaseu2$Grid_Projection BOXCAR&

Required input files for DBASEU2 :

file type:	**format:**	**description:**

Shot-list file:	shot	try	flaggable-fields

> shot = run or shot #
> try = run id for TRANSP [optional]

MINGL_FILES:DBASEU2.SHOTS
43528 a00 ok=yes mode=5 id=3 !good times

Parameter-list:	cID	fmt	param	opcode	eqcodes

> cID = ufile filespec
[cf. DBASEW or beginning of this section]
> fmt = INGRES storage format (f4 or i4)

[10] The only "radial" types supported are X, XB or 10-channel (eg., DDNINT or NTINT).

>	param =	keyword used to define the name of the field in the table
>	opcode=	DDR (for derivative with respect to R), or DDT (for derivative with respect to TIME),

MINGL_FILES:DBASEU2.PARAMS

-default-	I4	SHOT (*)
-default-	F4	TIME (*)
-default-	F4	RADIUS (*)
-default-	F4	DTAVG (*)
-default-	F4	DRAVG (*)
CX2:[AN.X]DQDR.2D	F4	DQDR SCALE=100 (**)
CX2:[AN.X]NUS.2D	F4	DQDR DDR !radial deriv
CX2:[AN.X]PRAD.2D	F4	DQDT DDT !time deriv
0000	F4	PBMAX_MW
[T]PTOWB	F4	PEQ !TRANSP

(*) default entry
(**) ufile CX2:[AN.X]DQDR07272.2D used as input based on the shot appearing in DBASEU2.SHOTS on the previous page.

Time-list file: toi dt [time_id time_mark dtg]

> toi = time of interest in sec
> dt = averaging window (average over toi +/- dt)
> time_id = determines event time as described in the DBASEW section
> time_mark = {START or STOP}
used to indicate beginning or end of time interval
> dtg = time grid for time interval

MINGL_FILES:DBASEU2.TIMES
(regular mode)

```
3.0000E+00  1.0000E-02
4.0000E+00  1.0000E-02
-.200  .02  BEAM_ON   START  .040
+.100 .02  BEAM_OFF STOP
```

To signal "pass through" mode for a ufile, specify a single time interval using the 2 line Start/Stop syntax, with dtg = 0. In this mode, the user is limited to one time interval. Specify the smoothing time in the "dt" line-1 placeholder in the Times file.

MINGL_FILES:DBASEU2.TIMES
(Pass through mode)

```
-.200          .02     BEAM_ON   START  0.
+.100          .02     BEAM_OFF STOP
               - or -
3.00           .02     NOOP   START
4.50           .02     NOOP   STOP
```

Radii-list file: roi dr

> roi = radius of interest in cm (average over roi +/- dr)
> dr= averaging window

MINGL_FILES:DBASEU2.RADII
(regular mode)

```
3.0000E+01  1.0000E-02
3.5000E+01  1.0000E-02
4.0000E+01  1.0000E-02
```

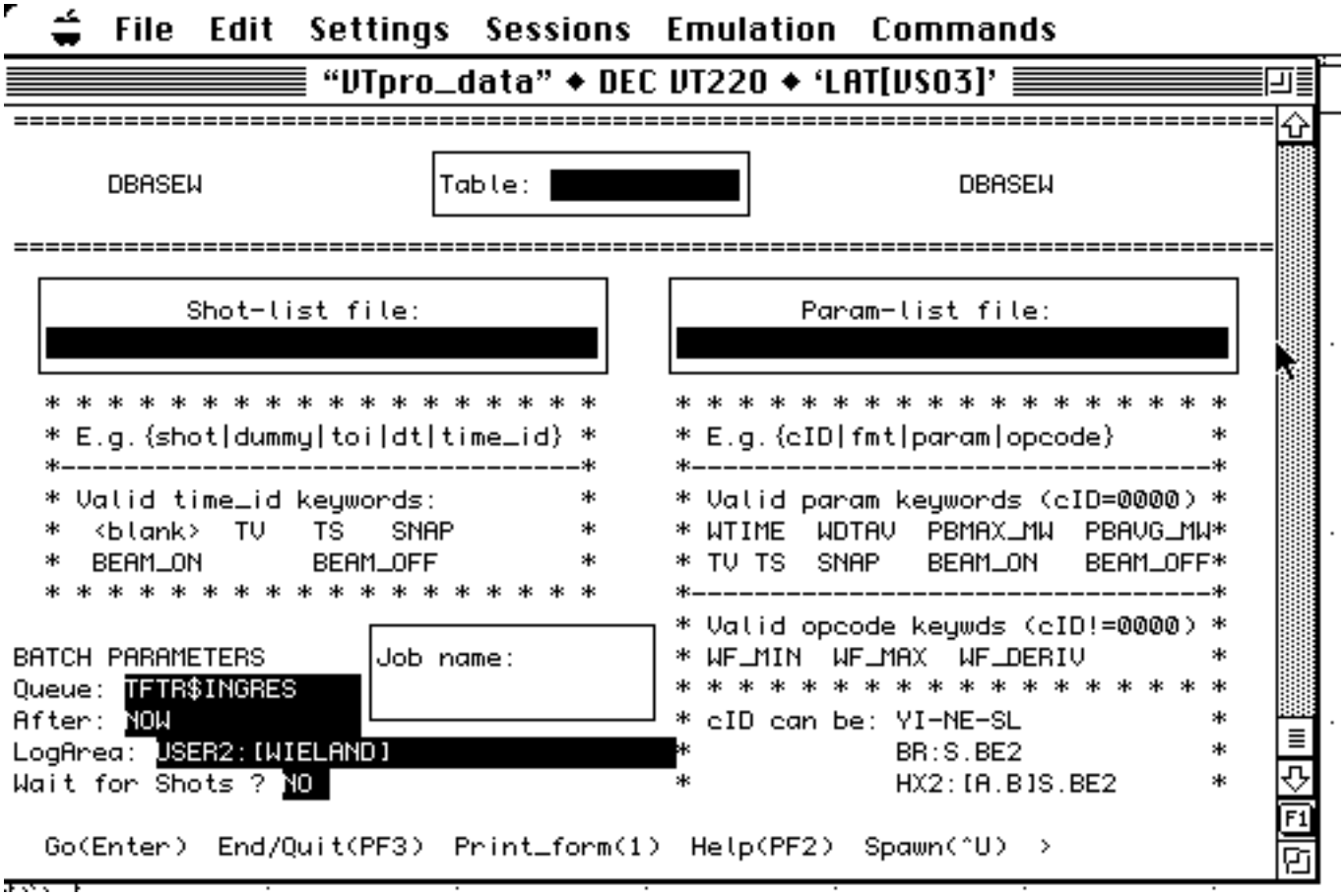
To signal "pass through" mode for a ufile, specify a single radius interval. In this mode the meaning of the two numbers in the single line that appears in the radius file is different; the first number is the start of the radius interval, the second number the end. In this mode, the user is limited to one such radius interval.

MINGL_FILES:DBASEU2.RADII
(pass through mode)

230. 270.

DBASEW

[Scalar Waveform, TRANSP, Ufile and Raw Data Files]



Description :

DBASEW is designed to extract a single, averaged value over a specified time interval from a 1-d data source, which is usually a function of time. The user specifies in the shot-list file both the shot number and an optional try number (for data sources that use a 2nd identifier in the source designation, such as TRANSP) followed by up to 3 time designators that define a time interval over which the data source is to be averaged. The smoothing for many opcode operations is done using a triangular weighting algorithm.[11] Boxcar smoothing is used for the no-opcode parameters.

---

[11] Cf. the OPCODE table on pg. 51 and footnote 12 for details.

Required input files for DBASEW :

file type: **format:** **description:**

Shot-list file:	shot try toi dt time_id flaggable-fields
>	shot = shot # [nnnnn]
>	try = [required for SNAP or TRANSP only, otherwise use 0]
>	toi = [time of interest/offset time in sec]
>	dt = averaging window [except for WF_DIF]
>	time_id = determines event time as follows:

If the "toi" entry is a real number and the "time_id" is blank or NOOP, then the time interval is defined to be [toi-dt, toi+dt]. The "time" column will contain "toi" and the various parameters extracted will be the values evaluated at "toi" after data in the time interval has been smoothed using a triangular weighting algorithm with baseline equal to 2*dt.

If the time_id field is a time-event keyword, then the time interval is defined to be [event-time + toi - dt, event-time + toi + dt], with "toi" playing the role of an offset to the event time..

MINGL_FILES:DBASEW.SHOTS

```
!CASE 1
18505 01 3.450 .050 ! absolute time
18509 01 0.050 .050 TS
18515 01 0.050 .050 BEAM_ON
18523 01 -.050 .050 BEAM_OFF
36220 2029 3.55 0.5                    ! TRANSP run 2020
36220 a01 3.55 0.5                     ! TRANSP run 36220a01
45966   00 -0.120 0.060 BEAM_OFF       itype=3  itemp=5.33 ibozo=nodata
41087   00 -0.120 0.060 BEAM_OFF       itype=5          ! HAS 23
41308   00 -0.120 0.060 BEAM_OFF       itemp=69.01 ibozo=100
```

```
54270   00 -.10   0.05  PWR_ON                    !nb & icrf
50508   00 -.10   0.05  ICRF_ON                   !icrf only
50508   00 +.10   0.05  ICRF_OFF
52435   00 2.70   0.40  ![toi-dt,toi+dt]          !for WF_DIF this translates to
                                                  ! [toi,toi+dt]
```

Parameter-list:	cID|{i:j}|	fmt	param opcode rds_offset eqcodes
>	cID	=	data ID; for TRANSP ID's, prepend with [T]
> or	cID{i:j}	=	data from columns i to j in the waveform header (read as real * format)
>	fmt	=	INGRES storage format (i4 or f4)
>	param	=	column name in the table
>	opcode	=	keyword used to define the operation performed:
>	rds_offset	=	t0 offset (for raw data scalars only); smoothing baseline for WF_DIF.
>	eqcodes	=	cf page 26

Opcode when cid ≠ "0000"

opcode	meaning
WF_MIN[12]:	minimum value over entire time range of data source
WF_MAX:	maximum value over entire time range of data source
WFDT_MIN:	minimum value in time interval toi +/- dt
WFDT_MAX:	maximum value in time interval toi +/- dt
WF_DERIV:	derivative (using slope from Least Squares straight line across toi +/- dt)
WF_DIF:	simple endpoint difference across [toi, toi+dt] with triangular smoothing across a (time) baseline given by rds_offset
WF_INT:	integral (spline integration across entire range = [1,6.5] sec)
WF_INTX:	integral (spline integration across only toi +/- dt)
WFMIN_TIME:	time corresponding to minimum over entire time range of data source
WFMAX_TIME:	time corresponding to maximum over entire time range of data source

continued . . .

[12] Smoothing for WF_MIN, WF_MAX, WFMIN_TIME, WFMAX_TIME, WFDT_MIN, WFDT_MAX, WFMINDT_TIME, WFMAXDT_TIME is set by using the “SM=“ opcode.

opcode (cont.)	meaning
WFMINDT_TIME:	time corresponding to minimum over toi +/-dt
WFMAXDT_TIME:	time corresponding to maximum over toi +/-dt
WF_M2[13]:	a "2nd moment" type operator over 1.0-6.5 sec range
WF_SUM:	straight sum over 1.0-6.5 sec range

MINGL_FILES:DBASEW.PARAMS

```
-default-                I4  SHOT  (*)
-default-                F4  TIME  (*)
-default-                F4  DTAV  (*)
0000                     F4  BEAM_ON
0000                     F4  BEAM_OFF
MB-IP-SL                 F4  WIP_MAX   WF_MAX   SM=.010
MB-IP-SL                 F4  WIP_MIN   WF_MIN   SM=0.
MV-VS-SL                 F4  WVSUR       ! Loop Voltage (volts)
MB-LM-SL                 F4  WLAMD       ! Lambda
NB-BP-SL                 F4  WPINJ       ! Injected beam power (watts)
0000                     F4  PBMAX_MW    ! cid=0000 keyword
0000                     F4  PBAVG_MW
0000                     F4  PBMAX_SQ
[T]TFLUX                 F4  TFLUX       ! TRANSP scalar function
0000                     F4  TV
0000                     F4  TS
0000                     F4  SNAP
0000                     I4  SNAPTRY
0000                     F4  PBAVG_MW      ! neutral beam power
0000                     F4  PBMAX_MW
0000                     F4  PBMAX_SQ
0000                     F4  BEAM_ON
0000                     F4  BEAM_OFF
0000                     F4  ICRFAVG_MW                  ! icrf power
```

[13] $WF\_M2 = \sqrt{\frac{\sum y(t)*(t-\bar{t})^2}{\sum y(t)}}$ where $\bar{t} = \frac{\sum y(t)*t}{\sum y(t)}$

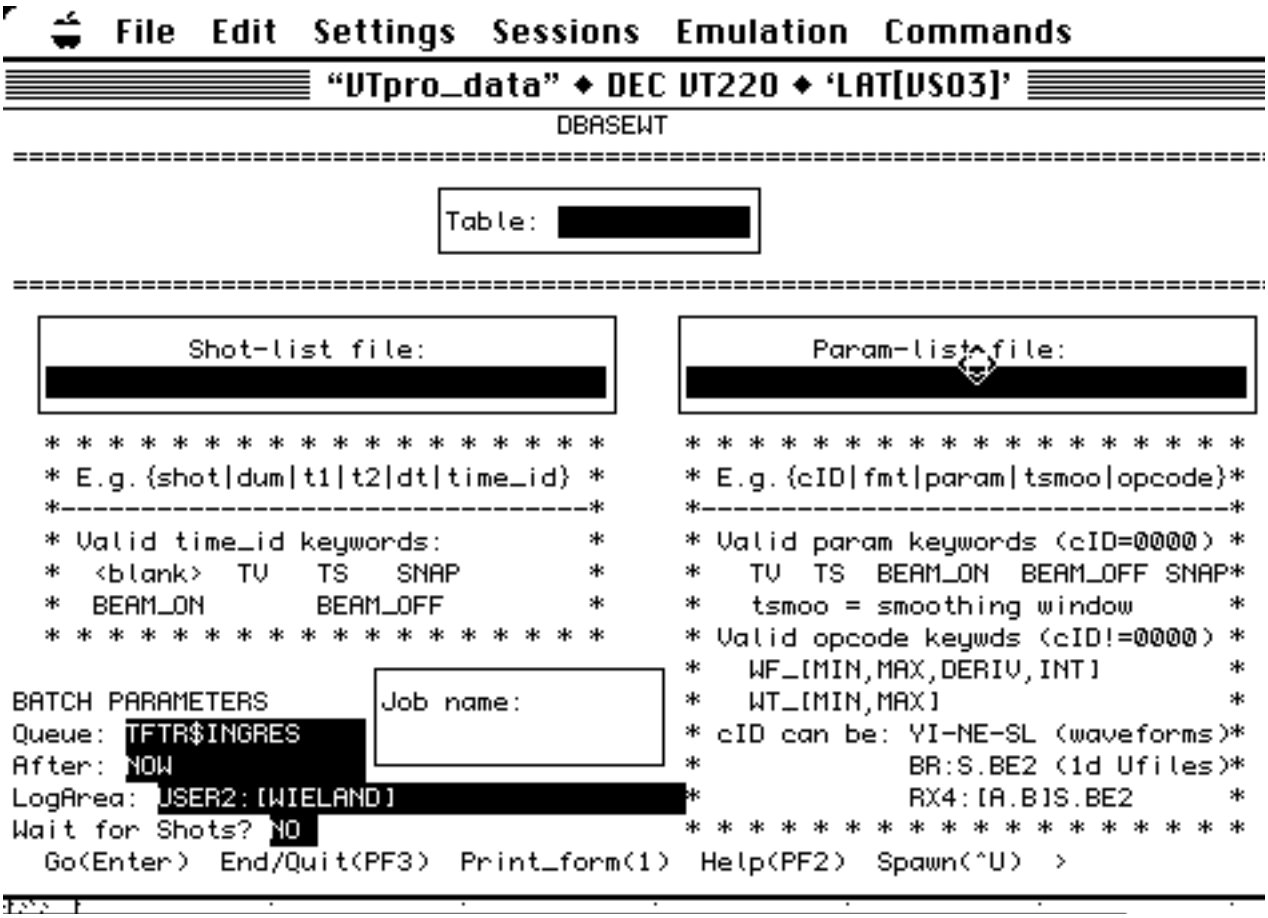
```
0000                                F4  ICRFMAX_MW
0000                                F4  ICRFMAX_SQ
0000                                F4  ICRF_ON
0000                                F4  ICRF_OFF
0000                                F4  PWRAVG_MW       ! auxiliary power
0000                                F4  PWRMAX_MW
0000                                F4  PWRMAX_SQ
0000                                F4  PWR_ON
0000                                F4  PWR_OFF
0000                                F4  DISRUPTION
0000                                F4  FLAT_TOP
0000                                F4  MAX_NEUTRONS
0000                                F4  E_T_MAX
0000                                F4  OHMIC
0000                                I4  IRAMP
[BR]DBW1-CHN-3                      F4  BW  scale=100. ! Raw Data file - digraph BR
DBR1-CHN-040                        F4  BR0             ! Digraph defaults to BR
[NE]DNE-SCL-SLW[02]                 F4  NE0             ! Raw Data Scalar file: chan=02
MB-IP-SL                            F4  WIP             !Plasma Current (amps)
MB-IP-SL                            F4  WIPDOT WF_DERIV     !Plasma Current
MV-VS-SL                            F4  WVSUR           !Loop Voltage (volts)
YM-TE-SL:YM-TM-SL                   F4  WTEPM           !peak Te from Michelson
XC-VR-SL:XC-TM-SL                   F4  WXCVR           !time base is in a separate file
XC-TI-SL:XC-TM-SL                   F4  WXCTI           !time base is in a separate file
M-LIO2                              F4  LIO2            ! derived waveform
M-POHMIC                            F4  POHMIC          WF_DIF      .040
D-PTOT                              F4  PTOT
L-MODEG-NE                          F4  TAUE_LM_NE
L-MODEG-NE                          F4  TAUE_LM_NEDT WF_DERIV
M-ETOT                              F4  D_ETOT_DT  WF_DERIV  !d/dt of TOTENERGY
FM-YI-NE                            F4  NEL_MIRI        !MIRI's SIMULATION OF YI-NEL
FM-YI-NE{12:15}                     F4  NEL_MIRI        !scalar header cols 12-15
WIELAND$:[DBASE]U.EXT               F4  MUFP  SCALE=1.67            !Ufile example
```

(*) default parameter

[This page left intentionally blank]

DBASEWT

[Time Series Waveforms, Raw Data Files & TRANSP Scalars]



Description :

DBASEWT is used to build time series in a database table. The limits of the time series as well as the time grid onto which the data is mapped are specified by the user in the shot-list file. There is no “pass through” mode. Smoothing may be specified separately for each parameter by an entry in the parameter-list file. The smoothing is done using a triangular weighting algorithm. Boxcar smoothing is possible as an alternative if the user defines the logical name as

$ DEFINE Mingl$Dbasewt$Grid_Projection BOXCAR

either interactively, before starting MINGL, or manually, by typing it into the DBASEWT.BAT file.

Required input files for DBASEWT:

file type: **format:** **description:**

Shot-list file:	shot try t1 t2 dt time_id1 time_id2 flaggable-fields

> shot = shot #
> try = TRANSP try or 00

> t1 = start time

> t2 = stop time

> dt = uniform time grid increment for interpolation grid.

> time_id1 = determines event time associated with t1 (cf. below)

> time_id2 = determines event time associated with t2 (cf. below). If not present, time_id1 is applied to both t1 and t2.

The time interval is formed as follows:

time_id[1,2] entry:	event time interval:
blank ,blank	[t1,t2]
TS, blank	[TS time + t1, TS time + t2]
BEAM_ON, BEAM_OFF	[time 1st beam ON + t1,time last beam OFF + t2]

MINGL_FILES:DBASEWT.SHOTS

```
20208 1 3. 4.9 .005
20322 1 -.2 +.2  .05        BEAM_ON  BEAM_OFF
45966 1 1.5 5.5 .010        itype=3  itemp=5.33 ibozo=nodata
40770 1 -1. .50 .010        MAX_NEUTRONS   BEAM_OFF itype=5
40779 1 -.5 +.1 .010        E_T_MAX
```

```
41196 1 -.2 +.3 .010          BEAM_ON  BEAM_OFF itemp=69.01 ibozo=100
41876 1 3. 4.9 .005           itemp=77.03 itype=-1
41911 1 3. 4.9 .005
41978 1 -.5 +.5  .010         OHMIC
41979 1 -.5 -.01 .010         E_T_MAX
41969 1 0. 3. .005
41979 1 0. 3. .005
36220 2029 3.0 4.9 .005  ! TRANSP entry using user time grid
36220 a01 3.0 4.9            ! TRANSP entry using inherent time grid
```

Parameter-list: cID fmt param smoothing-time opcode rds_offset rds_scale eqcodes

> cID = waveform ID
> fmt = INGRES storage format (i4 or f4)
> param = keyword used to define the column name in the table
> smoothing-time = baseline for triangular smoothing
> opcode= keyword used to define the additional operation, if any, performed (smoothing is always done before any of these operations is performed):

opcode	meaning
WF_DERIV:	derivative (by spline interpolation over time interval)
WF_INT:	integral (by spline interpolation over time interval)
WF_MIN:	minimum value in time interval
WF_MAX:	maximum value in time interval
WF_M2[14]:	a "2nd moment" type operator over time interval
WF_SUM:	running sum over time interval
WT_MIN:	time corresponding to minimum in interval
WT_MAX:	time corresponding to maximum in interval

> rds_offset = t0 offset (raw data scalars only)
> rds_scale = scale factor (raw data scalars only)

[14] Cf. DBASEW.

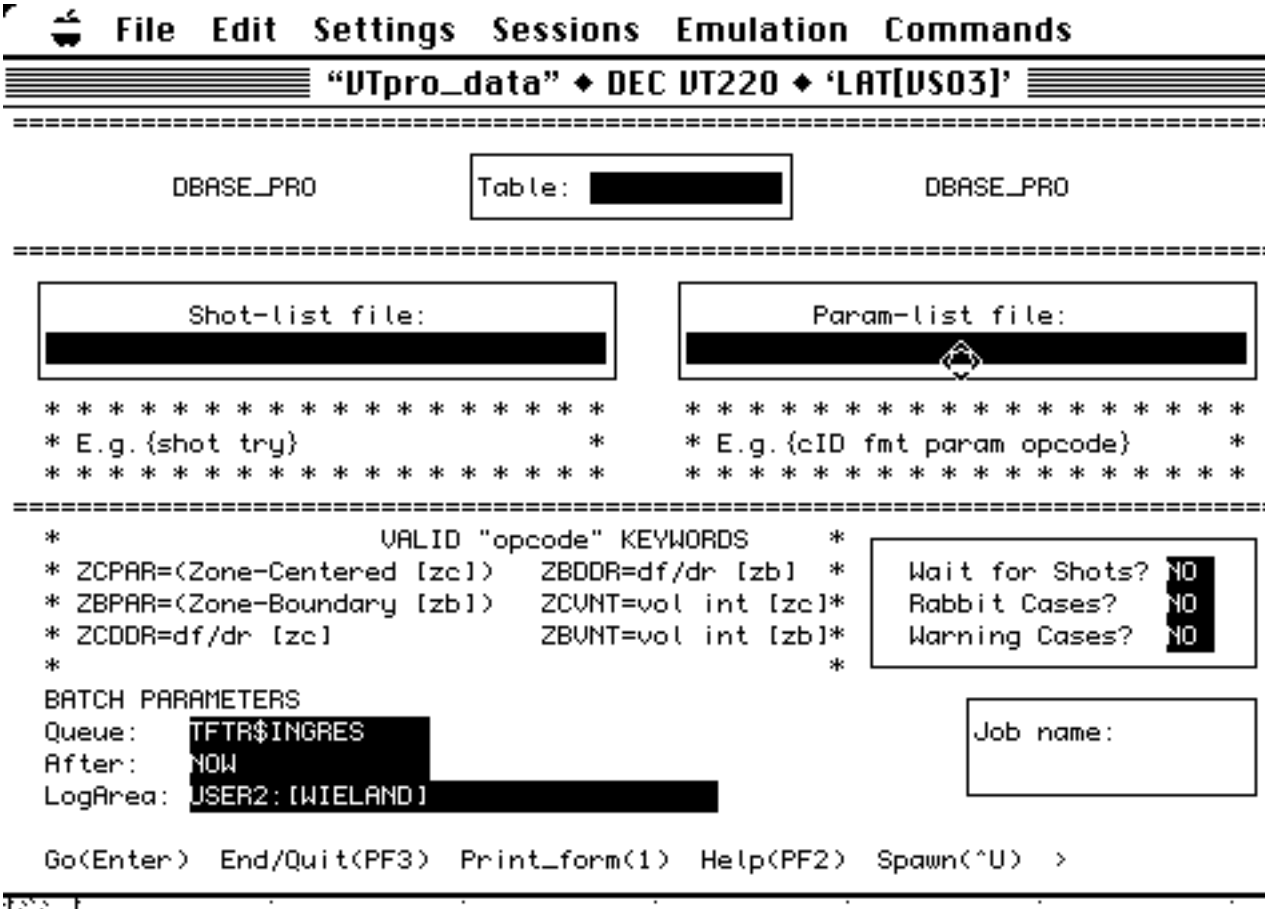
MINGL_FILES:DBASEWT.PARAMS

-default-	I4	SHOT	!(*)
-default-	F4	TIME	!(*)
BR:DBW1-CHN-3	F4	BW13	.025 scale=100.
DBR1-CHN-040	F4	BR030	.025
DBR2-CHN-040	F4	BR040	.025
NE:DNE-SCL-SLW:02	F4	NE02	.025 NOOP 0 2
BONZO-ALLEGR	F4	Bonzo	.025 ! this will surely fail
L-MODEG-NE	F4	LMG	.025
MB-IP-SL	F4	WIP_MIN	.025 WT_MIN !PC (amps)
NB-3B-SP	F4	WNB3B	.025 3b !injected power
FM-LD-04	F4	WNEL4	.025 !line integrated density
SH-VB-SL	F4	WVB	.025 !VB intensity
XI-DB-SL	F4	WXIDB	.025 !X-ray Wave detector B
NE-SP-SL	F4	NMIN	.025 scale=1.67 WT_MIN
M-LIO2	F4	LIO2	.025
M-TAUE-D	F4	TAUE_D	.025
NB-TCOFR	F4	TCOFR	.025
NB-EINJ	F4	EINJ	.025
0000	F4	PBMAX_MW	

(*) default entry

DBASE_PRO

[SNAP Radial (r or R) Profiles]



Description :

DBASE_PRO loads SNAP 1d profiles into data tables. It can accommodate minor radius profiles (2000 series channel numbers) and major radius profiles (1000 series channel numbers), but not in the same table. It can also load in SNAP scalar channels, repeating them for each radius entry associated with the given shot.

For the 2000 series, all channels are interpolated onto the zone-centered (ZC) radius array, channel 2001. This interpolation, as well as the interpolation associated with the opcodes described later in this section (cf. page 61) use a default cubic spline interpolation. The user may substitute linear interpolation by defining the logical name as

$ DEFINE ING$Interp_Mode 1

either interactively, before starting MINGL, or manually, by typing it into the DBASE_PRO.BAT file.

For the 1000 series, the user must specify the major radius channel onto which the data channel is to be mapped by following the data channel with a colon followed by the radius channel (e.g., 1076:1073 f4 mychan). The grid read for the 1st such parameter then becomes the grid onto which all subsequent channels are mapped onto. The user should exercise caution, because even for one parameter the grid may vary from shot to shot, in which case an erroneous mapping will ensue. The channel profiles are mapped to one of two grids, depending on how you reply to the form prompt for the interpolant grid parameters RMIN, RMAX, and DELTA-R. If you provide non-zero values, all the profiles are interpolated onto the specified radial grid; if you specify zeroes, all the profiles are interpolated onto the radial grid associated with the first channel specified.

DBASE_PRO recognizes either the old style 4 digit channel identifier or the new style alpha mnemonic.

Required input files for DBASE_PRO :

file type:	**format:**	**description:**

Shot-list file:	shot try	flaggable-fields

MINGL_FILES:DBASE_PRO.SHOTS

```
37266  4   itype=3  itemp=5.33 ibozo=nodata !this is a test
37264  3   itype=5  !this is a test
37262  4   !this is a test
37261  2   itemp=69.01 ibozo=100    !this is a test
37259  5   itemp=77.03 itype=-1 !this is a test
37258  4   !this is a test
37257  3   !this is a test
37255  3   !this is a test
37253  3   !this is a test
37249  4   !this is a test
```

Parameter-list:	cID	fmt	param	opcode

>	cID	=	*nnnn* for **SNAP** channel integer; *name* for **SNAP** channel name; *0000* for SNIPSNAP definition[15]
>	fmt	=	INGRES storage format (f4 or i4)
>	param	=	keyword used to define the name in the table as well as select the SNIPSNAP parameter when cID=0000.
>	opcode	=	SNIPSNAP_PRO opcode :

opcode[16]	meaning
PARn	zc/b quantity smoothed (**)
DDR	zc/b: df/dr [by finite-differencing]
DDRn	zc/b: df/dr [by finite-differencing;smoothed(**)]
VNT	zc/b: vol int from r=0 (***)
VNPn	zc/b: vol int from r=0 of $s(r)^n$ (***)
PARAX	zc/b: value nearest r=0
PARMX	zc/b: max value of quantity
PARMN	zc/b: min value of quantity
PARnn	zc/b: value at nn-th index
X	repeated entry for scalars

(**) using a triangular weighted smoothing algorithm; n is the number of points used in the triangular baseline

(***) all integrals return ZC values

---

[15] Cf. Appendix 2.

[16] The ZC and ZB prefixes used in earlier MINGL versions (e.g., ZCPAR) are no longer required to distinguish zone-centered from zone-boundary channels.

MINGL_FILES:DBASE_PRO.PARAMS
2000 series channels
minor radius

```
-default-      I4      SHOT   !(*)
-default-      I4      TRY    !(*)
-default-      F4      RUN_TYPE        !(*)
-default-      F4      WARNING_FLAG!(*)
-default-      F4      RADIUS   !(*)
2009 F4 te par
2009 F4 tesq vnp2
2009 F4 tesq1 vnp2.1
2009 F4 tesqrt vnp.5
2009 F4 teint vnt
2009 F4 dtedr ddr
2070 F4 qneo par
2070 F4 dqdr ddr
4098 F4 pheat x
ufp_all  F4  ufp_all  par  !derived vector
```

(*) default entry

MINGL_FILES:DBASE_PRO.PARAMS
1000 series channels
major radius

```
-default-      I4      SHOT      !(*)
-default-      I4      TRY   !(*)
-default-      F4      RUN_TYPE        !(*)
-default-      F4      WARNING_FLAG!(*)
-default-      F4      RADIUS   !(*)
1076:1073 F4 te par
```

(*) default entry

V. DATA ARCHIVING PROCEDURES

MANGL is the new utility which lets you archive, destroy, or restore databases and tables. It is available on all nodes of the cluster, and works in much the same way as MINGL:DBASE does, in that you compose an operation or sequence of operations that are then submitted to a batch MINGL queue for execution. MANGL is invoked either through the UTILITY level menu in MINGL or by typing "MANGL" at the DCL dollar-sign prompt. All documentation is provided in-line through the forms help facility, which is initiated by typing PF2-^R on any form you're in at the time. The archive medium can be either Jukebox (default) or Tape.

*** -------- To DESTROY Tables --------- ***

Note that MANGL is a much improved facility for "destroying" tables. You mark the tables you want destroyed, and hit "ENTER" to send off the request to be completed asynchronously in batch.

*** --------- --------- ***

In the current implementation of this facility, archiving and restoring are initiated only when you manually invoke the MANGL utility. In a future version, tables of a certain "age" will be automatically moved off-line to the archival storage medium. Each database will have an index table that logs each archive operation, as well as each restore operation that you perform later to bring the table back. The "age" of a table will be determined by the time interval since the table was last "touched" by certain predefined MINGL operations, and any tables whose "age" exceeds the latency period defined for the local database will be liable to be archived.

The predefined operations referred to above will probably be <Open> operations performed by programs like LOCUS or ALTERTABLE, i.e., operations which indicate that the table is being used by someone on the system. The latency period will probably start out at something like "2 months", and will be subject to change depending on disk usage.

By having access to a long term storage facility, you will have more flexibility

in managing your data.

[This page left intentionally blank]

APPENDIX 1.
Moving Through MINGL Forms

To initialize your VAX session for MINGL, use the following command:

$ SETUP LOCUS INGRES

To open a database using MINGL, type

$ MINGL <database>

Full screen forms are the backbone of MINGL. To use the system you have to be able to navigate through a form. There are just a few key things you need to know:

1. TAB moves you from one field to the next.

2. Key PF3 drops you back to the previous menu without any other action. On the MAC keyboard with keypad, the top 4 buttons on the keypad are known as keys PF1, PF2, PF3, and PF4, respectively.

3. CTRL-A toggles you between INSERT and OVERSTRIKE mode.

4. In a scrollable field, COMMA (or "+") on the keypad scrolls down, and MINUS scrolls up.

5. Numbered items in the submenus refer to keypad number keys, not keyboard number keys. The same holds true for numbers on the menu line, at the bottom of the screen.

6. To reach the functions on the menu line at the bottom of the screen, hit the PF1 key; to return to the main part of the form, enter RETURN.

7. Extensive help documentation is available by hitting PF2, then

CTRL-R for program information, or CTRL-E for key definitions.

8. To temporarily drop down to the DCL level, use CTRL-U. To return to MINGL later, type LOGOFF.

9. CTRL-W refreshes the screen if overwritten by system broadcasts, like MAIL

Most of the tools described here assume you are using a terminal set to VT100 emulation mode and with a numeric keypad.

[This page left intentionally blank]

APPENDIX 2.
SNAP Opcodes

A special set of OPCODES is available from DBASES and DBASE_PRO. These opcodes generally return various quantities of interest from SNAP data sets, where the results involve some integral or derivative of fundamental plasma variables evaluated at pre-selected minor radii. Two kinds of variants exist; examples are given below.

opcode variant	**definition**
PBE*x*	where *x* is {blank,0,A,A2,A3, or2A2}
PBE or PBE*A*	total beam power deposited in plasma
PBE*0*	beam power deposited on axis
PBE*A3*	beam power deposited out to r=a/3
PBE*A2*	beam power deposited out to r=a/2
PBE*2A3*	beam power deposited out to r=2a/3
DUBD*q*	where *q* is {Q11,Q21,Q31, or Q32}
DUBD*Q11*	evaluated at the q=1/1 surface
DUBD*Q21*	evaluated at the q=2/1 surface
DUBD*Q31*	evaluated at the q=3/1 surface
DUBD*Q32*	evaluated at the q=3/2 surface

* *

The SNAP opcodes are listed on the next page. Opcode definitions have to be obtained from the source code directly; cf. MINGL_FILES:SNIPSNAP.FOR .

List of SNAP Opcodes follows on the next page . . .

List of SNAP OPCODES

APLAS
CHIE
CHIET
CHIETx
CHIEx
CHII
CHIIx
DNEDR
DNEDRx
DQq
DTEDR
DTEDRx
DTIDR
DTIDRx
DUBDq
DUEDq
DUIDq
HIO2
HOFR
HOFRx
LIOV2
NDEN
NDENx
NDONE
NDVAV
NE
NEMAX
NETOT
NEVAV
NEx
NHVAV
NI
NIx
NUSTE
NUSTI
PBE
PBEx
PBI
PBIx
PBTH
PBTHx
PCNDE
PCNDEx
PCNDI
PCNDIx
PCNVE
PCNVEx
PCNVI
PCNVIx
PCXTH
PCXTHx
PIE
PIEx
POH
POHx
PRAD
PRADx
PS
PT
SPRDA
SPRDH
STRY
TE
TEDAV
TEDAV2
TEMAX
TEVAV
TEx
TI
TIMAX
TIVAV
TIx
UB
UBDq
UBx/q
UE
UEDq
UEx/q
UI
UIDq
UIx/q

APPENDIX 3.
Transporting Data from ASCII Files To Data Tables and Vice-Versa

IFILE_IN and **IFILE_OUT** are two programs that are recommended for copying ASCII file data into and out of data tables, respectively. The programs are simple to use, and examples of their use are given below.

IFILE_IN

Description:

Reads an ASCII data file and converts it into a data table. If the table name is new, a new table is created. If the table name already exists, the user is given the option of appending to the table.

Operation:

1. Type $IFILE_IN
2. Program asks for name of database.
3. Program asks for name of table.
4. Program asks for name of ASCII data file.

Data File Format:

line 1. Column names separated by blanks

line 2. INGRES formats (i4,f4, or text(n)) separated by blanks

lines 3... The data separated by blanks (no quotes around text; text data cannot contain embedded blanks)

Example:

```
VS03$ create x.x

shot   time   comment

i4        f4        text(12)

100     1.2        good

200     1.8        bad

^Z

VS03$ ifile_in

Please answer the following questions:

Database:     ingwork

Checking... This may take a minute.

Table Name:     testif

Data File:     x.x

Table testif is being created and filled.

        2 data rows have been processed.

Would you like to enter a <N>ew table name or <Q>uit.

Type N or Q:    q

>> Exiting IFILE_IN ...


VS03$ report  ingwork  testif
```

```
INGRES REPORT -- Copyright (c) 1981, 1992 Ingres Corporation

Setting up default report . . .

Retrieving data . . .

                    Report Name: testif

18-jan-1993                 09:25:34

      Report on Table: testif

      ------ -- -----  ------

      Shot        Time     Comment

      ----      -----  -------

       100       1.200   good

       200       1.800   bad
```

IFILE_OUT

Description:

Reads a data table and converts it into an ASCII data file.

Operation:

1. Type $IFILE_OUT
2. Program asks for name of database.
3. Program asks for name of table.

4. A form appears showing the names of the columns in the table. The user “tags” the columns to be written out, then hits the “Enter” key.

5. The user selects the delimiter for the output file, TAB or SPACE.