Welcome to Suprlink

Welcome to Suprlink for MPE Version 4.3.05 Suprlink is a program that works with Suprtool to add "multidataset" capability to Suprtool. Rather than take the regular path to multiple datasets -- random retrieval via IMAGE keys -- with its well-known performance problems, we have chosen to follow a different path: fast serial extracts plus a very efficient merge.

Summary of the Suprlink commands:

Before	Input	Reset	*=expression*
Do	Link	Set	*:MPE command*
Exit	LISTREDO	Use	
Form	Output	Verify	
Help	REDO	Xeq	

Keywords

Everything about Suprlink is organized under a few keywords:

KEYWORDS	this list of keywords
MANUAL	printing the manual and using it
NOTATION	common notation in Suprlink commands
ACCESS	how to run Suprlink; different run options
INTRO	purpose of Suprlink
COMMANDS	explanation of all Suprlink commands
FILENAMES	how to adapt built-in file names
INSTALLATION	how to install Suprlink
EXAMPLE OUTPUT	show the fields in a self-describing file
LIMITS	lists maximum record sizes
QUICK	brief syntax reminders
SUPPPORT	how to contact us for product support

Documentation

Printed copies of the Suprtool user manual can be ordered directly from Robelle or one of its distributors. The user documentation for Suprlink assumes you have some previous experience with IMAGE, MPE, and Suprtool. Use the Printdoc program to print the Suprlink user manual. The Printdoc program makes printing user manuals very easy. All you need to do is run the program and answer a few questions about your printer.

```
run printdoc.pub.robelle;info="suprlink.doc.robelle"
```

Printdoc is menu driven, and is very easy to use. Printdoc asks you for information, and if you are not sure of your answer, you can ask for help by typing a question mark (?) and pressing the Return key. The two steps in printing a manual are, first, to choose one of the manuals on the menu; and second, to select a printer. Printdoc supports all types of LaserJet printers and regular line printers.

The user manual covers the commands of Suprlink and answers questions that might arise during Suprlink execution. The user manual is also available to the on-line Suprlink user through the Help command. For instructions, try:

```
+help help
```

Notation

The Suprlink documentation uses a common notation in describing all commands. Here is a sample command definition:

Link *filename* [BY *link-keys* [FROM *input-keys*]] [OPTIONAL]

1. UPPERCASE LETTERS are required elements in the command, and must be typed exactly as they appear.
   Example: BY

2. *Highlighted lowercase letters*, underlined or italic, are "variables" to be filled in by the user. In the help file, underlining and italics are not available and variables will appear simply in lowercase.
   Example: *filename*.

3. [] - Brackets enclose optional fields.
   Example: [FROM *input keys*]

4. {} - Braces enclose comments in examples. Braces are allowed for comments in actual Suprlink commands.
   Example: +output repts temp {produces job-temporary Output}

5. | - Up lines separate alternatives from which you will select. Sometimes, the alternatives are shown listed on several lines.
   Example: [TEMP | ERASE]

6. In examples, there is an implied carriage return at the end of each line.

Installing Suprlink

Suprlink is installed as part of the Suprtool installation process. See the "Installing Suprtool" chapter of the *Suprtool User Manual* for more details of how to install both Suprtool and Suprlink.

Built-In File Names

Suprlink requires an external file for the Help command. Suprlink dynamically changes this file name, but you can use a :File command to override Suprlink's choice.

Default File Names

If you are running Suprlink on MPE V/E or MPE/iX, Suprlink finds the name of the Suprlink program file name (e.g., Suprlink.Pub.Robelle). Suprlink uses this name to determine the name of the other built-in file names. If Suprlink cannot call the procinfo intrinsic (e.g., this intrinsic doesn't exist on MPE V/R), it assumes you are running Suprlink.Pub.Robelle.

Account Name
Group Name
Examples

Account Name

The account name for all built-in files is the same as the one where Suprlink is running (e.g., if you :run Suprlink.Pub.Dev, the help file for Suprlink is Suprlink.Help.Dev).

Group Name

Suprlink examines the group name where Suprlink is running to determine the group name for the built-in file names. If the group name is PUB, Suprlink assumes help files are in the HELP group. The same assumption is made for any group name where the first three letters are not PUB. When Suprlink is run from a group that starts with Pub (e.g., Pub), Suprlink assumes the help files have the same suffix (e.g., Helpnew).

Examples

Here are a few examples of the names that Suprlink would use for the Suprlink help file:

```
Suprlink Program File Name:    Suprlink Help File Name:

Suprlink.Pub.Robelle           Suprlink.Help.Robelle
Suprlink.Pub.Robelle           Suprlink.Help.Robelle
Suprlink.Pub.Account           Suprlink.Help.Account
Suprlink.PubRob.Account        Suprlink.HelpRob.Account
Suprlink.Robelle.Dist          Suprlink.Help.Dist
```

Changing Built-In File Names

You can use a :File command to tell Suprlink where the help file is located. Your :File command must use the file name that Suprlink dynamically assigned to the help file. For example, if Suprlink is called Suprlink.Robelle.Dist, you would use this :File command:

```
:file suprlink.help.dist=linkhelp.robelle.dist
```

Accessing Suprlink

To access Suprlink, type the following command:

```
:run suprlink.pub.robelle
SUPRLINK/Copyright Robelle Solutions Technology Inc. 1988-2001
(Version 4.3.05)
+
```

After a short pause, Suprlink will take over your terminal and print out some identifying information. You will notice that your command prompt has changed to "+", telling you that you have made it into Suprlink. Suprlink expects you to type command lines, ending each one with Return.

Xeq
Suspend
Suprtool
Preventing MPE Commands
Verify Exit
Parm=32
JCW
Batch
Parm Values

How to Xeq a Suprlink Task

Normally, you enter a series of commands. These commands specify the Input file, the Output file, and the Link file name(s). Finally, you enter an Xeq or an Exit command. This begins the actual Suprlink linkage task.

If you entered the Exit command, Suprlink will finish the current task, then return you to the Operating system.

```
+EXIT
End of program
:BYE
```

If you entered the Xeq command, Suprlink will finish the current task, then prompt you for another task. This continues until you enter the Exit command. If you wish to terminate Suprlink immediately (perhaps you are confused), enter Exit Abort. This will terminate the Suprlink program immediately, without attempting any task.

Son Process

If you RUN Suprlink within Qedit or Select, you can retain the Suprlink process and re-activate quickly later.

```
:run qedit.pub.robelle
/:r suprlink.pub.robelle
+...
+exit
SUPRLINK.PUB.ROBELLE is still alive. Okay to HOLD ? Y
/...
/:activate
SUPRLINK.PUB.ROBELLE
+...
+exit
Program Held.  Use :ACTIVATE/:RUN;HOLD to re-run.
/...
```

Suprtool Link Command

You can access Suprlink from within the Suprtool program, using Suprtool's Link command. Suprtool will then run Suprlink for you. All of Suprlink's commands are available to you in Suprtool. When passing Suprlink commands through Suprtool, you must preface each Suprlink command with *Link*. For example,

```
:run suprtool.pub.robelle
>base customer
>get m-customer
>sort custnum
>output invoices,link
>xeq
>link input invoices by custnum
>link link  customer    {send "link customer" to Suprlink}
>link output invcust
>link exit              {exit from Suprlink (not Suprtool)}
```

Refer to the Link command in the "Suprtool Commands" chapter of the Suprtool user manual for more information.

You can also use the Suprtool2 interface and the Link command to "call" Suprlink. See the Suprcall User Manual for a complete example of how to use the interface to pass commands to Suprlink.

Preventing MPE Commands

If you want to prevent Suprlink from executing any MPE commands (e.g., :Purge), you Run Suprlink with Parm=8192. This feature is automatically invoked by Suprtool's Link command, when Set Limit MPE Off has been specified inside Suprtool.

```
:run suprlink.pub.robelle;parm=8192
```

Exit with Verify

Some users find that they Exit from Suprlink inadvertently. For Suprlink to get user approval on Exit, Run Suprlink with Parm=64.

```
:run suprlink.pub.robelle;parm=64
>e
Okay to exit [no]:
>
```

Preventing Suprlink from Suspending

If you run Suprlink from within HPDesk (and some other programs), Suprlink will suspend on Exit but HPDesk will not notice. The next time you run Suprlink you will get a new copy of Suprlink. Eventually you will have many suspended copies of Suprlink hanging from HPDesk, consuming system resources. Running Suprlink with Parm=32 forces Suprlink to terminate on Exit rather than suspend.

```
:run suprlink.pub.robelle;parm=32
```

Job Control Word

Suprlink sets the system job control word JCW to a fatal state when Suprlink fails in a batch job. Suprlink sets only the high-order bit of the JCW job control word. That is, it adds 32,768 to the existing JCW value. HP subsystems use the other bits of the JCW job control word, so Suprlink does nothing to them.

Using Suprlink in Batch

Suprlink operates in "session" mode or "batch" mode. In batch usage, any "error" message will cause Suprlink to quit, setting the Job Control Word to flush the remainder of the JOB. Warning messages do not cause an abort.

In batch mode, Suprlink does not prompt for missing information as it does in session mode. Instead, it attempts to choose the alternative that has the least chance of destroying valid data. For example, if the Output file is a duplicate file name in batch mode, Suprlink saves the new Output file with a "made up" name (OUTPUTnn, where nn is from 00 to 20), prints a warning message, and aborts.

Summary of Parm= Values

32	don't suspend, terminate completely
64	check with user before Exiting
8192	don't allow MPE commands

Values may be combined by adding them together. For example, Parm = 96 means "check with me before exiting, then when I do actually exit, terminate Suprlink completely instead of suspending".

Introduction to Suprlink

The best way to understand Suprlink is to examine the process of writing a report. Your report program will be written in COBOL, RPG, PowerHouse, or some other language. Imagine that instead of hunting all over the database with DBFIND and DBGET to collect your data, you just read a sorted disc file with a big record containing all the data on a given entity. For example, a sales report might read a disc file whose records consist of sales transactions plus customer information. This file has been sorted by customer number and date. If there are several sales for the same customer, the customer information is just repeated in each record. The report program reads the records, checks for level breaks, and formats and prints the records. Suprlink fits into this model of report programs.

Working from the database to the final flat file, how do we use Suprtool and Suprlink to produce the desired result? Obviously, Suprtool can extract the desired fields from the desired records of the sales detail dataset and put them in a disc file. And Suprtool can extract the desired fields from the customer master dataset and write them to a second disc file. What does Suprlink do?

If Suprtool sorts both files by customer, Suprlink can "link" them together, producing a third file whose composite record consists of the related fields from both files. This file is just what we need to feed into the report program.

The tests that we have performed indicate that this brute force method, which uses serial extracts, sorts, and a merge, is often significantly faster than the "official" IMAGE method of chasing down chains and hash synonyms.

Input Files

Only one file can have repeated records that pass through to the final file. This file is called the primary Input file. If any of the Link files contain duplicate records, Suprlink will select one of them to link to the primary record(s). The Suprlink Output file will have no more records than the Input file.
The Input file and Link files are created with the Output xxx,Link option in Suprtool. These files must be sorted by the same key field in ascending order.

Link Files

You can have up to seven Link files that are combined with the Input file. Suprlink merges the Input file and Link files by comparing the key fields of both files (you can optionally specify a secondary-key). The default is for Suprlink to exclude any Input records that do not have a matching record in all Link files. Specifying the Optional keyword on the Link command will force Suprlink to fill the Output record with default values (spaces and zeros) when it doesn't find a match in a specific Link file. If you want to link the sales transaction to both the customer master and the salesman master, it's probably faster to use traditional methods (e.g., DBFIND and DBGET).

Output Files

The Output file will be a self-describing file, containing data extracted from the Input file and the Link files. Suprlink combines the Input and Link records together in a fixed way, dropping the duplicated key fields and appending the remaining fields of each file in the order specified. You control which fields occur by using the Extract command in Suprtool, but you have no control over their order. Use the Form command to print out the final record format so that you can prepare COBOL COPYLIB or PowerHouse QSCHEMA definitions.

Sort Keys

The Input file and Link files must be sorted by the same key field. Their names do not have to be identical, but they must be the same type and have the same length. Suprlink does not support real- or long-type keys.

Selection Logic

Selection logic can be tricky, since it is distributed over independent Suprtool extract tasks, the Suprlink merge phase, and the final report program.

Suprtool Selection
Suprlink Selection

Suprtool Selection. You can use the If command to select which records you want from each dataset. What you cannot do in Suprtool is check a field in a related dataset. You do have the option to select key values from one dataset, then load them into a Table and use $lookup to select related entries in another dataset. It makes sense to use If on every dataset, since you have another selection possibility when the files are linked. For example, you might select all customers in California and all invoices with an amount greater than $2000.

Suprlink Selection. The Input file limits the scope of the Output file. You cannot have more Output records than you do Input records, but you can have fewer. When you do a Link to another file, you have an implied selection criterion. That is, if Suprlink cannot find a record in the Link file with matching key value(s), the Input record is dropped from the Output file. If you have seven Link files, the Input record must match all seven or be dropped. This is the default selection logic. You can override this for any specific Link file by specifying the OPTIONAL keyword on the Link command. Only do this is you don't care whether that data exists or not, since Suprlink will supply default values for those Link fields.

A Link Example

You want to produce a report of all invoices over $2,000.00 for customers in California. The customer information is in the m-customer dataset, and the invoice information is in the d-invoice dataset. Here are the steps to produce this report:

1. Select and sort the California customers into the file *customer*.

2. Select and sort invoices over $2,000 into the file *invoice*.

3. Because there will often be more than one invoice per customer, specify the invoice file as input to Suprlink.

4. Link in the customer file.

5. Produce your report from the combined records in the output file.

Sort-1
Sort-2
Link
Optional

```
>base sales                  {sales database}
 >get m-customer             {select all customers...}
 >if state = "CA"            {...in California}
 >sort custnum               {sort and link key}
 >output customer,link       {Link output option}
 >xeq
```

We now have a self-describing file with all the customers from California sorted by the customer number. Next we select all invoices over $2,000.00 and sort them into customer number sequence:

```
>get d-invoice              {select all invoices...}
>if amount>200000           {...over $2,000.00}
>sort custnum               {sort and link key again}
>output invoices,link       {remember the link option}
>exit
```

If we specify the cust file as input, the Output file will only contain one invoice per customer. Because we want to produce a report of all the selected invoices, we specify it as the input file:

```
+input invoices          {driving input file, custnum is the key}
+link customer           {combined with customers}
+output invcust          {produces the file we want}
+exit
```

Each record of the invcust file will have both the invoice information and the customer information for each invoice of the Input file (i.e., one record per invoice). What happens if there is no customer record for a specific invoice? In this case, the invoice record does not appear in the Output file. To force Suprlink to include these records, use the optional keyword on the Link command:

```
+input invoices            {sorted by custnum}
+link customer optional    {don't exclude invoices if...}
+output invcust            {the customer information...}
+exit                      {...is missing}
```

Performance Considerations

Select only the records you need, unless the time to load a table of desired key values, plus the time needed to do $lookup for each record, is longer than the time to extract and sort the entire dataset. Use the Sorted and Hold options of the Table command when loading a table. Because of the time needed to search a large table, it is often faster to extract all of the records and let Suprlink skip over the ones it doesn't need.

Sorts
Disc Space

This method does a lot of sorts. Sort time increases dramatically for large records and large datasets. You should try to use Suprtool's Extract command to reduce the record size. We suspect that the sort time skyrockets if the record size exceeds a 256 bytes.

Suprlink needs enough disc space to invert a significant subset of your database, then link it into an Output file. Although all of the Suprlink files can be job temporary, you still need enough disc space for the original database, the final Output file, the primary Input file, and each of the Link files. One of the tradeoffs with this method is more disc space for faster elapsed time.

Another Example

From the sales records, retrieve all of the sales for October, 2000 and append the customer name, salesman code, and year-to-date sales total to the sales record (these fields are located in the customer records).

```
:run suprtool.pub.robelle
>base mybase
>get sales-detail           {or use Chain command}
>extract customernum,saledate,saleamt,...
>item saledate,date,phdate
>if saledate >= $date(95/10/01) and &
>   saledate <= $date(95/10/31)
>sort customernum
>sort saledate
>output sales,link,temp     {creates temporary SD file}
>xeq

>get customer-master        {get entire dataset}
>extract customernum,name
>extract salesman,ytdsales
>sort customernum
>output custs,link,temp
>exit

:run suprlink.pub.robelle
+input sales                {link sales...}
+link custs                 {...to custs...}
+output repts temp          {...producing REPTS!}
+xeq                        {...run the task}
+form repts                 {fields in repts}
+exit

:purge sales,temp           {these two files...}
:purge custs,temp           {...no longer needed}

:file infile = repts,oldtemp
:file outfile;dev=lp
:run myprog

:purge repts,temp
```

Illegal Digits

Whenever Suprlink is processing files with packed- or zoned-decimal keys, errors can occur because of invalid digits in the keys. Suprlink reports the input and link record numbers with illegal digits and processing stops. You can use Suprtool to examine input and link records, by using record selection with Suprtool's input command. A packed-decimal number consists of nibbles (there are two nibbles in each byte). The last nibble is the sign of the number. The remaining nibbles must each contain a number in the range 0-9. A zoned-decimal number must have a valid digit in each byte and end in "0"-"9", "A"-"R", "{", or "}".

Selecting Non-Matches

Consider a common problem easily solved with Quiz from Cognos: finding all records in a file which have no corresponding records in a related file. For example, to find all records in an invoice lines file with no corresponding invoice master record, the following Quiz code could be written.

```
>access lines link to header optional
>select if not record header exists
>report invoiceno of lines
>go
```

This small amount of code, however, can take a long time to execute, depending on the size of the Lines and Header files. A Quiz program will usually take longer as new links are added, causing the size of the record complex to grow.
Suprlink can provide the same information, possibly in a fraction of the time. The technique as applied to the same problem requires four steps:

1. Sort the Lines file by Invoiceno.

2. Add a new constant field, Linkflag, to the Header file and fill it with "Y". Sort by Invoiceno.

3. Link the two files with Suprlink using the Optional parameter.

4. Select the record complexes where linkflag does not contain a "Y".

Example

```
:run suprtool.pub.robelle
>base invdb
>get lines
>sort invoiceno
>output file1,temp,link
>xeq

>get header
>define linkflag,1,1
>extract invoiceno,linkflag="Y"
>sort invoiceno
>output file2,temp,link
>xeq

>link input file1
>link link file2 optional
>link output file3,temp
>link xeq

>input file3
>if linkflag <> "Y"
>extract invoiceno
>list standard
>exit
```

Any invoice line with a corresponding record in the invoice Header file will have a "Y" in the linkflag field. Records failing the match will contain the default space.

This technique can easily be adapted for use as a command file, to minimize the amount of code added to a job stream. See the "MPE/iX Programming" section of the *Suprtool User Manual* for ideas about command files.

Linking MPE and KSAM Files

Suprlink's input or link files can come from datasets, MPE files or KSAM files. Before linking either an MPE or KSAM file, you must convert the file into a self-describing file. Because MPE and KSAM files have no implied structure, you must use Suprtool's Define and Extract commands to define the structure of the file.

Example

Suprlink's only requirement is that the file contain a sort field and some optional data fields. The sort field does not have to be the first field in the record. The following example demonstrates how to convert an MPE file into a self-describing file:

```
:run suprtool.pub.robelle
>input mpecust           {regular MPE file}
>def first-fields,1,10   {fields before the sort field}
>def sort-field,11,6     {sort field for this file}
>def last-fields,17,20   {fields after the sort field}
>extract first-fields    {remember to extract all of the}
>extract sort-field      {   fields in the correct order}
>extract last-fields
>sort sort-field         {must also remember the sort}
>output sdfile,link      {use the Link option}
>xeq
```

This example would sort the entire MPE file. To select a subset, you would define more fields and use the If command. Treat KSAM files exactly the same way. Suprtool does not read KSAM files in sorted order, so you must remember to specify the sort explicitly when converting the KSAM file to a self-describing file.

Suprlink cannot create nor write to KSAM files. You can use Suprtool to load a KSAM file with the output from Suprlink.

CI Variable Substitution

Suprlink is able to substitute any CI variables from any command line source, whether thru interactive, use file or batch input.

In order to use this feature, first issue the following Set command:

```
set varsub on
```

Variable Substitution is not on by default for backward compatibility.

Batch
Notes

Batch

Since the Streams facility (under default setups) will replace any "!" found in the first column of a job stream. Anytime you want to specify an entire line thru Variable Substitution you will need to leave a space before the variable is specified.

```
!setvar input "i dfile by message-no"
!run suprlink.pub.robelle
+set varsub on
+ !input
```

Notes

For MPE commands some variables will be resolved twice when passed off to MPE, which will give different values for a variable.

Setting variables at the CI level:

```
MPEXL:setvar x 10
MPEXL:setvar y "!!x"
MPEXL:showvar [xy]
X = 10
Y = !x
```

Setting variables within Suprlink

```
+set varsub on
+setvar a 10
+setvar b "!!a"
+showvar [ab]
A = 10
B = 10
```

Because Suprlink does one level of variable substitution prior to the command being passed off to MPE, setting variables in Suprlink that involves other variables will give different results than MPE. We recommend that you set the variables prior to running Suprlink, or that you temporarily turn off variable substitution with the Set Varsub Off command.

```
+set varsub off
+setvar a 10
+setvar b "!!a"
+showvar [ab]
A = 10
B = !a
+set varsub on
```

Suprlink with Quiz/QTP

Quiz and QTP are part of PowerHouse, a popular fourth generation language sold by Cognos. You can use Suprtool and Suprlink to improve the performance of PowerHouse applications. For a complete discussion of how to use Suprtool and Quiz together, refer to the "Suprtool with Quiz/QTP" section of the *Suprtool User Manual*.

Suprlink can write to PowerHouse subfiles that have been created with Quiz or QTP. Subfiles are "self documenting" files that contain a complete description of the file's record structure. This information is stored in *user labels* in the file, and is known as a "mini-dictionary." When you access the subfile in Quiz, its description is read from the mini-dictionary. You must ensure that the PowerHouse subfile description *exactly* matches the record layout of Suprlink's output file. Remember that Suprlink will drop the common "key" fields from the link files.

QTP Subfile
Suprlink Output
Quiz Report
Notes On Subfiles

Step 1: Create the Subfile with QTP

Before running Suprlink, you create an empty subfile with QTP:

```
:purge invcust
:qtp
>access d-invoice link custnum to &
>                     custnum of m-customer
>subfile invcust keep size numrecs include &
>        custnum, invdate, amount, invnum, &
>        name, address
>set input limit 0
>go
```

The subfile must contain all of the fields that Suprlink will produce in the output file, with the same attributes (data-type and length) and in the same order. Use the Include option of QTP's Subfile command to define each of the fields in the correct order.

The *numrecs* parameter must be replaced with the number of records that will be created by the Suprlink run. The default *numrecs* is 1023 when the input limit is set to 0.

Step 2: Output Erase in Suprlink

Once you have created the PowerHouse subfile, use the Erase option of Suprlink's Output command to load the file. This will overwrite any data in the subfile, but it will not touch the PowerHouse mini-dictionary in the user labels:

```
+input invoices              {created by Suprtool}
+link customer               {sorted by custnum}
+output invcust erase        {created by QTP}
+exit
```

Step 3: Report with Quiz

The INVCUST file contains the sorted records for the Quiz report. Quiz knows the structure of this file because of the initial QTP commands that we used to create the file. Now use Quiz to generate the report:

```
quiz
>access *invcust
>report ...
>go
```

Notes on Subfiles

One of the advantages that Suprlink has over the link function in PowerHouse is that Suprlink does not require the "key" field in the link files to be a database key. Because Suprlink uses a serial-merge approach, its files only need to have a common field with the same data-type and length. If you do use Suprlink to link files that do not share a common database key, you need some extra steps to create the PowerHouse subfile.

Since Suprlink cannot currently write to NM Ksam files you cannot directly write to PowerHouse indexed subfiles. You can use Suprtool to load file to the Indexed KSAM file.

Defining Fields
Linking Subfiles

Defining Fields in QTP

In our example above, "custnum" can be used to link the d-invoice and m-customer datasets in QTP because custnum is an IMAGE key in the m-customer dataset. If custnum was not an IMAGE key, you could try declaring the record structure for the subfile with the QTP Define command:

```
:purge invcust
:qtp
>access d-invoice
>define name character size 20 = " "
>define address character size 20 = " "
>subfile invcust keep size numrecs include &
>        custnum, invdate, amount, invnum, &
>        name, address
>set input limit 0
>go
```

You must be careful to ensure that the data definitions of the Defined fields are correct. Note that you cannot assign default display specifications (such as Heading or Picture specifications) for Defined fields in QTP.

Linking Subfiles by Record Number

Another approach, which guarantees that the subfile will contain the correct data definitions and default display characteristics, is to create temporary subfiles with QTP for each dataset, then link them together by record number:

```
:purge invcust
:qtp
>access d-invoice
>subfile invtemp size 1 include &
>        custnum, invdate, amount, invnum
>set input limit 0
>go

>access m-customer
>subfile custtemp size 1 include &
>        name, address
>set input limit 0
>go

>access *invtemp link to record 0 of *custtemp
>subfile invcust keep size numrecs include &
>        custnum, invdate, amount, invnum, &
>        name, address
>set input limit 0
>go
```

Suprlink Commands

When you run Suprlink, it prompts for commands on $stdlist with a "+" character and reads command lines from $stdinx. Suprlink commands contain a command name followed by one or more parameters, and are patterned after the same commands in Suprtool.

In this chapter, we describe the Suprlink commands in alphabetic order. Following each command name in brackets is the minimal abbreviation for the command. For example: [I] for Input and [L] for Link.

Abbreviating

You may shorten the command name to the first letter of the command name.

`+v`	{verify}
`+x`	{xeq}

Uppercase or Lowercase

You may enter the letters in either uppercase or lowercase, because Suprlink upshifts everything in the command line except literal strings within quotes ("abc"). These two commands are identical:

```
+EXIT
+exit
```

Continuation

The maximum *physical* command line is 256 characters. You may enter commands on multiple input lines by putting an "&" continuation character at the end of the line. The maximum total command length is 256 characters. The most common reason for continuing commands is to specify a lengthy Link command with secondary keys.

```
+input students
+link majors by ssn cmaj from &
                ssn curmajor
+output outfile
+exit
```

Comments on Command Lines

Comments may appear at the end of any command line, when they are surrounded by braces. Many of the examples in this manual show comments at the end of each command line. You can enter a comment as the only item in a Suprlink command line. When continuing command lines, the comment can appear before or after the continuation character.

```
+                       {link customer records to invoices. }
+input invoices         {sorted by custnum}
+link customer          {combined with customers}
+output invcust         {produces the file we want}
+exit
```

STREAMX

STREAMX is a product from VESOFT that permits you to build flexible job streams. STREAMX contains a complete programming language with loops, prompts, and parameter substitution. A problem arises when trying to enter comments into a Suprtool batch job that will be submitted with STREAMX. Suprtool uses the {...} pair to delimit comments. STREAMX uses these same characters for expressions.

You cannot change Suprtool's comment character, but you can change the {...} characters in STREAMX. The following example changes the STREAMX expression characters from {...} to ~...~:

```
!job example,user.acct
::setbraces ~~
!run suprlink.pub.robelle
+input invoices    {driving input file, custnum is the key}
+link customer     {combined with customers}
+output invcust    {produces the file we want}
+exit
!tell ~hpjobname~,~hpuser~.~hpaccount~;Example done.
!eoj
```

MPE Commands

Suprlink also accepts MPE commands, with or without a colon.

```
+:comment
+comment
```

For commands that are the same in both Suprlink and MPE, Suprlink only executes the MPE command if you type the colon. For example:

```
+help              {you get Suprlink help}
+:help             {you get MPE help}
```

MPE/iX Commands

MPE/iX Commands

Suprlink/iX will execute any MPE command (e.g., Run), UDCs, and command files. **Caution:** programs that suspend, instead of terminating, are not killed by the HPCICOMMAND intrinsic.

File Names

Suprlink's Input, Link and Output commands accept a file name in either MPE or POSIX format. File names starting with "/", "./", or "../" are treated as POSIX file names. All other file names are assumed to be MPE file names. MPE file names are upshifted and POSIX file names are not. POSIX file names are limited to a maximum of 240 characters. Here are some examples of POSIX file names:

```
:hello david,mgr.dev,david
:CHDIR SUBDIR
:run suprlink.pub.robelle
+input ./file
+verify input
/DEV/DAVID/SUBDIR/file
```

Calculator

Any command line beginning with an equal sign (=) is treated as a calculator expression. This feature can be used to compute blocking factors and do other calculations without the need of an electronic calculator.

You can obtain a short description of the calculator by entering the following:

```
=?        {? gives help}
  {prints a summary of = functions}
```

For a detailed description of the calculator and its options, see the Suprtool manual.

Control-Y

You can interrupt a Suprlink task with the Control-Y key (hold down Control while striking Y). Suprlink responds by telling you how far it has gotten (IN=, OUT=, etc.), and asking if you wish to stop. Hit the Return key to continue or type YES to stop the task.

Before Command [B]

Repeat any combination of the previous 1000 command lines, with or without editing.

BEFORE [*start* [/ *stop*]]
[*string*]
[ALL | @]

(Default: redo previous line)
(BQ=redo without change)

The Before command allows you to modify the commands before it executes them. If you don't need to change them, use BQ or Do.

The Before command uses Qedit-style Control characters for modifying the commands. The default mode is to replace characters. To delete use Control-D, and to insert use Control-B. If you prefer HP-style modify (D, R, I, and U), use the Redo command instead of Before.

Examples
Modify Operators
Persistence

Examples

```
+listf @.soruce    {"source" is not spelled right}
NON-EXISTENT GROUP.  (CIERR 908)
+Before           {redo most recent command}
listf @.soruce     {last command is printed}
         our       {you enter changes to it}
listf @.source     {the edited command is shown}
                    {you press Return}

+listredo -10/
+before 5          {redo 5th command in stack}
+bef 8/10          {redo 8th through 10th}
+b listf           {redo last listf command}
+b listf temp      {redo "listf temp" command}
+b @temp          {redo last containing "temp"}
+before -2         {redo command before previous}
+before -5/-2      {redo by relative lines}
```

Modify Operators

If you wish to change any characters within the line, the modify operators are the regular Control Codes used in Qedit:

Any printing characters replace the ones above.
Control-D plus spaces deletes columns above.
Control-B puts you into "insert before" mode.
Control-A starts appending characters at the end of line.
Control-A, Control-D, plus spaces, deletes from the end.
Control-T ends Insert Mode, allowing movement to a new column.
Control-G recovers the original line.
Control-O specifies "overwrite" mode (needed for spaces).

Persistent Redo

Redo commands can be saved in a permanent file and can therefore be used from another session. You can use the **Set redo** command to specify a filename to save your redo commands. Please see the Set Redo command for details.

Do Command [DO]

The Do command will repeat (without changes) any of the previous 1000 commands.

DO [*start* [/ *stop*]]
[*string*]
[ALL | @]

(Default: repeat the previous command)

Commands are numbered sequentially from 1 as entered and the last 1000 of them are retained. Use the :Listredo command to display the previous commands. You can repeat a single command (`do 5`), a range of commands (`do 5/10`) or the most recent command whose name matches a string (`do list`). If you want to modify the commands before executing them, use Redo or Before.

Examples

```
+listredo
+do                 {do previous command again}
+do 39              {do command line 39 again}
+do 5/8             {do command lines 5 to 8 again}
+do link            {do most recent Link command}
+do show            {do last starting with "show"}
+do showjob job     {do last "showjob job" command}
+do @job            {do last containing "job"}
+do -2              {do command before previous}
+do -7/-5           {do by relative line number}
+do 5/              {do command lines 5 to last}
```

Notes

The Do command cannot be abbreviated.

Persistent Redo

Redo commands can be saved in a permanent file and can therefore be used from another session. You can use the **Set redo** command to specify a filename to save your redo commands. Please see the Set Redo command for details.

Exit Command [E]

Exit Suprlink in one of three ways. By default, perform the current linkage task, if any, then leave Suprlink. Users are often frustrated when they exit Suprlink after specifying part of a task and Suprlink starts processing the task. Use the Abort or Suspend options to exit Suprlink conveniently without executing the current task.

EXIT [ABORT | SUSPEND | XEQ]

Typing Exit with no parameters means Exit Xeq. Suprlink recognizes special command names which specify both the Exit command and an exit option (e.g., ES means Exit Suspend).

Abort
Suspend
Xeq

Exit Abort [EA]

Cancels the current operation and terminates Suprlink. The Exit command without parameters always attempts to perform the task currently specified, while Exit Abort cancels the task and terminates immediately. Should Suprlink be executed as a son process, Exit will only suspend Suprlink, while Exit Abort will actually terminate the process.

Examples

Examples

```
+:comment.  You began to specify a linkage, stopped for
+:comment.  coffee, and decided to cancel the task
+:comment.  upon your return.
+input invoices
... coffee break ...
+exit abort              {cancel linkage and terminate}
```

Exit Suspend [ES]

When running Suprlink as a son process (e.g., from Qedit), it would be nice to suspend Suprlink without executing the current task. Exit Suspend does this. After returning to Suprlink, all specifications for the current task are still in effect.

Examples

Examples

```
:run qedit.pub.robelle
/:run suprlink.pub.robelle
+input invoices
+link customer            {start specifying options}
+exit suspend             {return to Qedit without an Xeq}
SUPRLINK.PUB.ROBELLE is still alive. Okay to HOLD ? Y
/...
/:activate suprlink
SUPRLINK.PUB.ROBELLE
+output invcust           {continue specifications}
+xeq                      {execute the current task}
```

Exit Xeq [EX]

To perform the current linkage task, you can either use Xeq (which leaves you inside Suprlink, ready to define another task) or Exit Xeq (which leaves Suprlink when done with the task).
After the Suprlink task completes, Suprlink either terminates, or suspends and awakens a father process (i.e., :RUN from within Qedit). Exit Xeq is the default option (i.e., specifying exit starts execution of the current task).

Examples

Examples

```
:run suprlink.pub.robelle
+exit                    {no input was specified}
End Of Program
:run suprlink.pub.robelle
+input invoices
+link customer
+output invcust
+exit                    {link and stop}
```

Form Command [F]

Display the fields in a self-describing file.

FORM [*filename*]

If no file name is specified, the fields in the input file are displayed. The display shows the field type and field length in IMAGE notation. An I1-field is a single integer. Packed-fields show the number of nibbles (subtract one to obtain the number of digits). Byte and zoned-decimal fields show the byte length.

When showing the form of a self-describing file, Suprlink shows the byte offset of each field after the subcount, type, and sublength. The first field always appears at offset one.

There are two types of self-describing files. One type is produced with Suprtool's Query output option. You produce the other type with the Link output option. The Form command shows the internal self-describing version number, enabling you to tell the difference.

Query
Link
Formout File

A.00.00 - Query Output Option

Compound fields have a question mark for the type, and the length is the number of bytes in the field. Sort information about the file is missing. Here is an example form listing:

Example

```
+form custfile

 File: CUSTFILE.EXAMPLE.ROBELLE   (SD Version A.00.00)
    Entry:                Offset
       CHARACTER         X5    1 {length is five bytes}
       ZONED             Z5    6 {room for five digits}
       INTEGER           I1   11 {single integer}
       DOUBLE            I2   13 {double integer}
       PACKED            P6   17 {room for five digits}
       QUAD              I4   20 {eight-byte integer}
       REPEATINT         ?6   28 {compound field}
       LOGICAL           K1   34 {single logical}
       DBLLOG            K2   36 {double logical}
 Limit: 10000  EOF: 15  Entry Length: 44  Blocking: 64
```

B.00.00 - Link Output Option

These self-describing files contain information about how the file is sorted. Compound fields are handled correctly, so the Form command shows compound fields just as you would see them in IMAGE. The Item command in Suprtool identifies the date format of an item. The Link output option saves the date format and any decimals as part of the field description:

```
+form datafile

File: DATAFILE.EXAMPLE.ROBELLE   (SD Version B.00.00)
   Entry:                Offset
      CHARACTER          X5    1  <<Sort #1 >>
      REPEATINT         3I1    6  {compound field}
      DATE               J2   12  <<YYYYMMDD>>
      DOLLAR             P6   16  << .2 >>
Limit: 10000  EOF: 15  Entry Length: 16  Blocking: 64
```

Formout File

The Form command writes all output to the file Formout. This file defaults to $stdlist. You can redirect this file to a line printer or a disc drive. If you redirect the Formout file to a disc file, Suprtool assumes a temporary file by default.

```
>:file formout;dev=lp
>form custfile          {writes to line printer}
>:file formout;dev=disc
>form invfile           {writes to temporary file}
```

Help Command [H]

Show what commands and options are available in Suprlink.

HELP [*command* | *keyword* [,*option*]]

(Default: browse through the entire help file)

Command Help

If you specify any parameters, Help first assumes that you want help on a specific Suprlink command. If you know the structure of the help file, you can specify one of the keywords under the command name.

`+help link`	{help on the Link command}
`+help link,notes`	{notes section of the Link command}

Keyword Help

If we cannot find any help in the "Commands" section of the help file, we assume that you specified one of the outer-level keywords in the help file. To see this list of keywords, type help with no parameters. You will see a short introduction to Suprlink and then a list of keywords. You can specify any of these keywords on the Help command. You can also specify a subkeyword.

```
+help before,example
```
{example section of Before command}

Quick Help - HQ

HQ asks Suprlink to look under the keyword QUICK in the help file. QUICK contains the text from the Suprlink Quick Reference Guide, offering the experienced user a quick review of the syntax of any command.

`+hq input`	{quick description of Input}
`+hq commands`	{quick list of command names}

Notes

If no parameters are specified, Help allows you to browse through the help file, Suprlink.Help.Robelle. The Help command uses the Qhelp subsystem from the QLIB. For "help in help", type "?" when you see the Qhelp prompt character ("?"). The help file is organized into levels. To go back to the previous level, press Return. Press F8 to exit the Qhelp subsystem and return to Suprlink.

Input Command [I]

Specifies the primary input source and the name of the key field by which it is sorted.

INPUT *filename* [BY *key-field*]

There can be only one Input file per linkage task, but up to seven Link files. The Input file should be created by Suprtool using the Output-Link option and must be sorted by *key-field*. The key field can be any type, except for Real or Long. The primary Input file may have more than one record per key value, and each record may appear in the Output file.

It is best to have Suprtool Extract only the fields you will actually need, since if any of the Suprtool extracts result in enormous Output files, the time to do the sort may be prohibitive.

The BY-clause is only necessary when the Input file has been created using the Suprtool Output-Query option instead of the Output-Link option. Output-Link adds the sort field information to the self-describing file, so that you do not have to specify it in a BY clause.

Link Command [L]

Link the Input file to another Link file, maximum of seven input files.

LINK *filename* [BY *link-keys* [FROM *input-keys*]]
[OPTIONAL | REQUIRED]

File name
Primary Key
Secondary Key
Input Key
Options

File Name

The Link file should be created by Suprtool with the Output,Link option; it should only contain the fields that you actually need in the final report, plus any sort fields. If you do a :LISTF of the file, it should show "SD" as the CODE; this means it is self-describing. It contains a description of its own record structure in some special user labels; this allows you to refer to the same field names as in the original database and Suprlink can compute where they occur in the record. For example:

```
+input sales              {Sales is sorted by custno}
+link custfile            {key is custno}
+link addrfile
+output custsale          {link three files...}
+exit                     {...into custsale}
```

Link Keys

Suprlink allows files to be linked by up to two keys, a primary and a secondary key field.

By default, Suprlink assumes that the key field to the Link file is the same key field specified for the Input file. If the Link key field is different from the Input key field, use the BY-clause to specify the correct key field:

```
+input customer          {key-name is custnum}
+link sales by custno    {new name for the same field}
```

You would also use the BY-clause if the Link file was created using the Suprtool Output,Query option instead of Output,Link.

Secondary Keys

Suppose that you are linking a master to a detail and the detail can have several entries for each master. Suprlink has an option that allows you to select which link record you want by matching a second key field in the master.

LINK *filename* BY *primary-key secondary-key*

This option forces Suprlink to compare both the primary-key and the secondary-key when comparing an input record to a link record. For example,

```
+input students              {key-name is ssn}
+link majors by ssn cmaj     {Students contains cmaj}
```

This example says that the file Majors is sorted by ssn and may contain more than one record per student. To select the desired record for each student, Suprlink matches the students' cmaj against the cmaj in the link record.

Secondary Input Key

It is possible that the second key field has a different name in the input file and the Link file. The FROM-clause lets you handle this case:

```
+input students                {key-name is ssn}
+link majors by ssn cmaj from ssn curmajor
```

Note that you must specify the Input file key field as part of the FROM-clause. This example is identical to the previous secondary key example, but in this case the current major field is called "curmajor" in the students file and "cmaj" in the majors file.

Optional Linkage

If there is more than one link record with the same key value, Suprlink will select the first one it finds. You can sort by another value such as date-time to force a certain record to be first. Please note that this is unlike Quiz, which does a hierarchical expansion to include every record accessed. If there are no link records for a given key value of the input file, that input record is dropped from the output file (this is the default option, REQUIRED).

To make the linkage optional, specify the OPTIONAL keyword:

LINK *filename* OPTIONAL

When you use OPTIONAL, and Suprlink does not find a matching link record in the file, Suprlink fills in the linked fields with default values. The default for byte-type fields is spaces, for zoned-type the default is ASCII zeros "0", and for all other types the default is binary zeros. For example,

```
+input custfile            {key-name is custno}
+link addrfile optional    {don't drop customers...}
+output custaddr           {...if there is no address}
+exit
```

Listredo Command [LISTREDO]

The Listredo command will display any of the previous 1000 commands.

LISTREDO	[*start* [/ *stop*]]	[;ABS] [;OUT=*file*]
	[*string*]	[;REL]
	[ALL | @]	[;UNN]

(Default: display previous 20 commands)
(BJ and ,, are short for LISTREDO)

Commands are numbered sequentially from 1 as entered and the last 1000 are retained. You can display a single command, a range of commands, all 1000, or all the commands whose name matches the string. You can print the commands with ABSolute line numbers (the default), RELative line numbers (-5/-4), or UNNumbered. You can write the commands to your terminal or OUT to a temporary file. If you want to redo any of these commands, see Do, Redo, and Before.

Examples

```
+listredo 5
+listredo 5/10
+listredo help      {print all Help commands}
+listredo -10       {print last ten commands}
+listredo ALL       {print entire redo stack}
+listredo purge     {print all Purge commands}
+listredo purge xx  {print all "purge xx" commands}
+listredo @purge    {print all with "purge" anywhere}
+listredo @;rel     {print ALL, relative numbers}
+listredo 1/10;out=*lp  {dump commands to printer}
+listredo @;unn;out=save  {write commands to a file}
```

Saving to a File

Saving the Listredo commands to a file is not currently available in Suprlink/UX.

Notes

The Listredo command cannot be abbreviated, but BJ is accepted as a short form.

Persistent Redo

Redo commands can be saved in a permanent file and can therefore be used from another session. You can use the **Set redo** command to specify a filename to save your redo commands. Please see the Set Redo command for details.

Output Command [O]

Specify the name of the output file and whether it is temporary.

OUTPUT *filename* [TEMP] [ERASE] [DATA] [LINK]

By default, the name of the Output-file is Output. The output file is a permanent, self-describing file, containing data extracted from the input file and the Link files. A few application tools may not read self-describing files. In this case, use the Data option to make the output file a standard MPE file with a filecode of zero and no user labels.

There are two different types of self-describing files. The first type is created with Suprtool's Output Query option. A superior form of self-describing file is produced with Suprtool's Output Link option. Suprlink creates the output self-describing file in the same format as the input file. We recommend that you use the same type of self-describing file for all input and link files.

Record Format
Quiz
Size

Output Record Format

The record structure is determined by Suprlink, but is relatively easy to anticipate. Suprlink starts with all of the fields of the input file, in order. For each Link file, it appends the fields of the Link-file to the Output record, in order. Suprlink drops the key fields from the Link records, since they always contain duplicated data.

If a field name (other than one of the two explicit keys) is duplicated in several datasets, it will end up duplicated in the final output file. An example would be a Timestamp field that occurs in every dataset. Workaround: use the Extract command from Suprtool to take out only the fields you want, or to rename duplicate fields.

You can verify the format of the Output-file using the Form command. It shows the field names, length, and structure, in order. From this display, you can generate an appropriate COPYLIB or QSCHEMA definition.

Quiz Subfiles

The Erase option is provided for Quiz users who create an empty subfile using QTP or Quiz before running Suprtool and Suprlink. See the *Suprlink with Quiz/QTP* section for details.

Since Suprlink cannot currently write to NM Ksam files you cannot directly write to PowerHouse indexed subfiles. You can use Suprtool to load file to the Indexed KSAM file.

Size of the Output File

The output file is built with the same capacity as the input file. You can reduce the size of the output file using a :File command.

```
+:file allsales; disc=10000
+output allsales
```

Redo Command [REDO]

Enables you to modify and repeat any of the previous 1000 command lines.

```
REDO    [ start [ / stop ] ]
        [ string ]
        [ ALL | @ ]
```

(Default: redo the previous command)

The Redo command allows you to modify the commands before it executes them. If you don't need to change them, use the Do command. Commands are numbered sequentially from 1 as entered and the last 1000 are retained. Use the :Listredo command to display the previous commands. You can redo a single command, a range of commands, or the most recent command whose name matches a string.

The Redo command uses MPE-style editing logic (D, I, R, U and >). The default mode is to replace characters. To delete, type DDDD under the characters to be removed. To insert, type I under the insertion spot, then the new characters. To undo your changes, type U. To append to the end of the line, use >xxx. To delete from the end of the line, use >DD. To replace at the end of the line, use >Rxxx. And to erase the rest of the line, use D>. If you prefer Qedit-style editing (Control-D, etc.), use the Before command instead of the Redo command.

Examples

```
+listf @.soruce    {"source" is not spelled right}
NON-EXISTENT GROUP.  (CIERR 908)
+redo             {redo most recent command}
listf @.soruce     {last command is printed}
          our    {you enter changes to it}
listf @.source    {edited command is shown}
           {you press Return}

+listredo all
+redo 5         {redo 5th command in stack}
+redo           {redo previous command}
+redo -2        {redo command before previous}
+redo 8/10      {redo 8th through 10th}
+redo -10/      {redo -10 through last}
+redo purge     {redo last Purge command}
+redo purge temp   {redo last "purge temp"}
+redo @temp   {redo last containing "temp"}
```

Notes

The Redo command cannot be abbreviated. To save more commands, use a :File command on the file LINKREDO before you run Suprlink:

```
file linkredo;disc=5000
run suprlink.pub.robelle
```

Persistent Redo

Redo commands can be saved in a permanent file and can therefore be used from another session. You can use the **Set redo** command to specify a filename to save your redo commands. Please see the Set Redo command for details.

Reset Command [R]

Cancel the current linkage task.

RESET

Reset closes the current Input-file and any Link files, then resets the output file name to Output. This is actually a Reset All command; you cannot reset particular commands as you can do in Suprtool. If you try to reset an individual command, Suprlink prints a warning.

Set Command [S]

Enables or disables certain operating options within Suprlink. These options are not reset by Xeq or Reset commands.

```
SET   [MAPPED        ]   ON|OFF
      [STATISTICS    ]   ON|OFF
      [VARSUB        ]   ON|OFF
```

Mapped
Redo
Statistics
Variable Substitution

SET MAPPED ON (Default: OFF)

MAPPED forces Suprlink/iX to read the input and link files using mapped file access. Specifying this option is an error in Suprlink/V. If the input and link files are not in memory, the wall time performance is worse with Set Mapped On, but CPU time performance is better. You must Set Mapped On before specifying the input file.

Set Redo [*filename* **]**

(Default: none)
(Initially: temporary file)

Commands entered at the Suprlink prompt are saved in something called the redo stack. You can recall commands from the redo stack by using other commands such as Before, Do and Redo. By default, the redo stack is stored in a temporary file and discarded as soon as you exit. This temporary stack is not preserved across Suprlink invocations.

The new Set Redo command assigns a permanent file as the redo stack, allowing the stack to become available for future Suprlink invocations. For example, to assign the Myredo file as a persistent redo stack, enter

```
+Set Redo Myredo
```

If the file does not exist, Suprlink creates it. Otherwise, Suprlink uses the existing file. All subsequent commands are written to the persistent redo stack. The setting is valid for the duration of the Suprlink session. As soon as you exit Suprlink, the setting is discarded. Next time you run Suprlink, you will get the temporary stack.

If the file name is not qualified, the redo stack is created in the logon group and account. This may be desirable if you want to have separate stacks. If you want to always use the same persistent stacks, you should qualify the name.

The Verify command shows which stack is currently in use. If it shows <temporary>, it means Suprlink is using the default stack. Anything else is the name of the file used on the Set Redo command.

Concurrency
Qedit

Concurrency

When Suprlink uses the default temporary stack, it is only accessible to that particular instance of Suprlink. You can run as many Suprlink instances as you need and each one gets its own redo stack. With temporary stacks, you will never get into concurrency problems.

If you start using a persistent redo stack, however, you might start running into concurrency problems. A persistent redo stack can only be used by one Qedit instance at a time. If you try to use a persistent redo stack that is already in use, you will get the following message:

```
+Set Redo Myredo
EXCLUSIVE VIOLATION: FILE BEING ACCESSED  (FSERR 90)
Unable to open file for REDO stack
```

In this situation, Suprlink continues to use the redo stack active at the time and lets you continue working as normal.

Qedit can also have permanent redo stacks. To prevent products from writing to each other's stack, it is advisable to have separate stacks for each product by giving them different file names.

For example, if you use

```
set redo myredo
```

you will have a redo stack called Myredo for your Suprlink commands. If you exit Suprlink, then run Qedit and supply the same Set Redo command, your Qedit commands will be written to the same file that was used for your Suprlink commands.

SET STATISTICS ON (Default: OFF)

STATISTICS causes Suprlink to print statistics at the end of each task.

SET VARSUB ON (Default: OFF)

Setting Variable Substitution causes Suprlink to resolve any CI variables in a command before processing.

Use Command [U]

Specifies a file of commands to be executed as a group.

USE[Q] *filename*

Examples

A usefile makes your task easier by allowing common commands to be specified once in an external file. For example, the following usefile contains all the commands for creating the invcust file:

```
>use usecust
input invoices          {sorted by custnum}
link customer           {combined with customers}
output invcust          {produces the file we want}
exit
```

Suprlink prints the lines in the usefile, including the comment lines. This allows you to include instructions and reminders in the usefile. In the example above, there were no commands for the user to enter.

Notes

Usefiles cannot be nested in Suprlink. The usefile may be any unnumbered text file or a Qedit workfile, but no more than 256 characters per record will be processed.

By default, Suprlink displays the commands in a usefile as they are executed. Suprlink can execute commands *quietly* using the Useq command. For compatibility with Qedit, Useq can be abbreviated to UQ.

Verify Command [V]

Print the definition of the current linkage task.

```
VERIFY
```

Verify prints the current Input, Link, and Output files; in other words, it is a Verify All command.

Xeq Command [X]

Perform the current linkage task.

XEQ

Xeq checks that you have specified an input file and at least one Link file. Then it performs the linkage and creates the output file. Finally, it closes the files and resets, ready for you to specify another linkage task or Exit. If you also wish to leave Suprlink after completing the linkage task, use Exit instead of Xeq.

Example Suprlink Output

The Form command displays the fields in a self-describing file. This information is stored in the user labels of an MPE file and is not accessible with other tools. Use the Form command to obtain the record layout of Suprlink output files.

The following example shows the Form command listing for an input file, a Link file, and the resulting output file. We start with an input file of invoices.

Invoice File
Customer File
Linking
Output File

```
+form invoices

 File: INVOICES.TEST.DATA   (SD Version B.00.00)
    Entry:                     Offset
       CUSTNUM              X8      1  <<Sort #1 >>
       DELIVERED            I2      9
       PRODUCTNUM           Z8     13
       PRICE                I2     21
       PURCHASED            I2     25
       QTY                  I1     29
       TAX                  I2     31
       TOTAL                I2     35
 Limit: 100  EOF: 100  Entry Length: 38  Blocking: 107
```

Suprtool produced both the invoice and the customer file by using the Get, Extract, and Sort commands. The invoice file was produced with Suprtool's Output Link option. If you had used Suprtool's Output Query option, the Form command would not have printed any information about the key fields. The next listing is the customer file.

Example

```
+form cust

 File: CUST.TEST.DATA   (SD Version B.00.00)
    Entry:                     Offset
       CITY                 X12     1
       RATING               I2     13
       CUSTNUM              X8     17  <<Sort #1 >>
       STATUS               X2     25
       FIRSTNAME            X10    27
       LASTNAME             X16    37
       STATE                X2     53
       ADDRESS             2X25    55
       ZIPCODE              X6    105
 Limit: 50  EOF: 50  Entry Length: 110  Blocking: 37
```

The street address is a compound-field. If you had used Suprtool's Output Query option, the field would have appeared with a question mark for the data-type. In that case, you cannot use the field as a key-field in Suprlink, but the actual data in the field will be processed and linked correctly. Your final report should be able to read this data just as if it came from the database. We use Suprlink to combine the invoice and cust files into one Output-file:

```
:run suprlink.pub.robelle
+i invoices by custnum
+l cust
+o invcust
+e
```

The final Form command shows the record layout of the Output-file. You would use this file as input to your report program.

Example

```
+form invcust

 File: INVCUST.TEST.DATA   (SD Version B.00.00)
    Entry:                    Offset
       CUSTNUM              X8      1  <<Sort #1 >>
       DELIVERED            I2      9
       PRODUCTNUM           Z8     13
       PRICE                I2     21
       PURCHASED            I2     25
       QTY                  I1     29
       TAX                  I2     31
       TOTAL                I2     35
       CITY                 X12    39
       RATING               I2     51
       STATUS               X2     55
       FIRSTNAME            X10    57
       LASTNAME             X16    67
       STATE                X2     83
       ADDRESS              2X25   85
       ZIPCODE              X6    135
 Limit: 100  EOF: 100  Entry Length: 140  Blocking: 29
```

Limits Within Suprlink

Input File
Link File
Output File

Input File - **Maximum Record Size** - **2048 Words**

This is also the maximum size of an IMAGE entry. We recommend that you use Suprtool's Extract command to minimize the input record size.

Input File - **Maximum Block Size** - **4096 Words**

By default, Suprtool restricts the maximum block size to 2,048 words. You can use the Set Blocksize command to increase this size up to 8192 words. If you increase the maximum block size, it is likely that Suprtool will produce an output file that Suprlink cannot read.

Input File - **Maximum Fields** - **255**

Suprlink restricts the number of fields per file to be 255. If you must have more fields, use Suprtool's Define and Extract commands to extract several fields as one contiguous series of bytes.

Link File - **Maximum Record Size** - **2048 Words**

As with the input file, you should use Suprtool's Extract command to minimize the link record size.

Link File - **Maximum Block Size** - **2048 Words**

See the description of the maximum input block size.

Link File - **Maximum Fields** - **255**

See the description of the maximum number of input fields.

Link File - **Maximum Number** - **Seven**

Suprlink will link one input file with up to seven Link files.

Output File - **Maximum Record Size** - **4096 Words**

When linking many files together, it is easy to produce large output records. Once again, using the Extract command to minimize the size of the input and link records will avoid large output records.

Output File - **Maximum Fields** - **1023**

Internal Suprlink tables that keep track of the output fields are restricted to 1023 entries.

Suprlink

Suprlink can access multiple files, produce a sorted disc file with a composite record of related fields from up to 7 files, and feed the merged file into your report program. Suprlink accepts only one command per line, but commands can be continued on the next line with an ampersand (&). The prompt character is "+", not ">."

```
:run Suprlink.Pub.Robelle
```

Before
Do
Exit
Form
Help
Input
Link
Listredo
Output
Redo
Reset
Set
Use
Verify
Xeq

Before see Suprtool section

DO see Suprtool section

Exit [**ABORT** | **SUSPEND** | **XEQ**]

Perform task specified, and return to MPE or parent process. (:Run Suprlink with Parm=64 to verify on exit.)

```
+exit           {default = Xeq}
+exit suspend   {stop without executing}
```

Form [**filename**]

Display the fields in a self-describing file.

```
+form          {default = show fields in the input file}
+form customer
```

Help [**command-name** | **keyword** [**,section**]]

Access the on-line user manual (Suprlink.Help.Robelle).

`+help`	{default is browse all}
`+help input`	{explain Input command}

Input **filename** [**BY key-field**]

Select the primary input source. There is an option to specify the **key-field** by which the file is sorted.

```
+input invoices by cust-no
```

Link **filename** [**BY link-keys** [**FROM input-keys**]]
[**REQUIRED** | **OPTIONAL**]

Link the input file to another link file, maximum of seven.

```
+link custfile
+link sales by customer-no
```

LISTREDO see Suprtool section

Output **filename** [**TEMP**][**ERASE**]
[**DATA**][**LINK**]

Specify output file name and whether it is temporary.

`+output custsale` {default file name = Output}

REDO see Suprtool section

Reset

Cancel the current task.

```
+reset          {reset everything}
```

Set [**option-name value**] [**ON** | **OFF**]

Set configurable options.

```
+set mapped on
+set statistics on
+set varsub on
```
{MPE/iX only}

Use[Q] **filename**

Execute commands from a Text or Qedit file.

```
+use cap2.infile
```

Verify

Show current specifications.

`+verify` {show everything}

Xeq

Perform the current Suprlink task.

```
+xeq
```

How to Contact Robelle

In the United States, in Canada, and in places not listed below, contact us at the following address:

Robelle Solutions Technology Inc.
Suite 201, 15399-102A Ave.
Surrey, B.C. Canada V3R 7K1

Toll-free: 1.888.robelle
: (1.888.762.3553)
Phone : 604.582.1700
Fax : 604.582.1799

E-mail : solutions@robelle.com
E-mail : support@robelle.com
Web : www.robelle.com

For our international distributors listing, note that the phone and fax numbers shown are for out-of-country dialing.

Europe
Africa
Asia and Australia
North America

Europe

France, Belgium

ARES
Attention: Renee Belegou
Phone: 33 1 69 86 60 24
Fax: 33 1 69 28 19 18
E-mail: rbelegou@ares.fr
Web: www.ares.fr

Germany

SWS SoftWare Systems GmbH
Attention: Renate Pfund
Phone: 49 7621 689 190
Fax: 41 31 981 32 63
E-mail: info@sws.ch
Web: www.sws.ch

The Netherlands, Belgium

Samco Automation b.v.
Attention: Marius Schild
Phone: 31 13 5215655
Fax: 31 13 5288815
E-mail: marius@samco.nl
Web: www.samco.nl

Nordic Countries

Ole Nord AB
Attention: Ole Nord
Phone: 46 8 623 00 50
Fax: 46 8 35 42 45
E-mail: info@olenordab.se
Web: www.olenordab.se

Switzerland, Austria

SWS SoftWare Systems AG
Attention: Renate Pfund
Phone: 41 31 981 06 66
Fax: 41 31 981 32 63
E-mail: info@sws.ch
Web: www.sws.ch

United Kingdom, Ireland

Robelle Consulting
Attention: Clive Oldfield
Phone: +44 20 7473 2558
Fax: +44 20 7473 2558
E-mail: robelle_oldfield@email.msn.com

Africa

South Africa

Synergy Computing (Pty) Ltd
Attention: Paul Howard
Phone: 27 21 685 7809
Fax: 27 21 685 7927
E-mail: synergy@synergy.co.za

Asia and Australia

Australia, New Zealand

MRFM Pty. Ltd.
Attention: Michael Redmond
Phone: +61 3 9629 8633
Fax: +61 3 9629 8062
E-mail: mredmond@mrfm.com.au
Web: www.mrfm.com.au

Hong Kong

SCS Computer Systems Ltd.
Attention: Steven Lai
Phone: 852 2609 1338
Fax: 852 2607 3042

Singapore, Malaysia

Singapore Computer Systems Ltd.
Attention: Toh Tiau Hong
Phone: 65 441 2688
Fax: 65 441 2811
E-mail: tohth@scs.com.sg
Web: www.scs.com.sg

North America

Mexico

Infosistemas Financieros SA de CV
Attention: Anita De Urquijo
Phone: 52 5 813 1325
Fax: 52 5 813 3026
E-mail: adeurquijo@if.com.mx
Web: www.if.com.mx