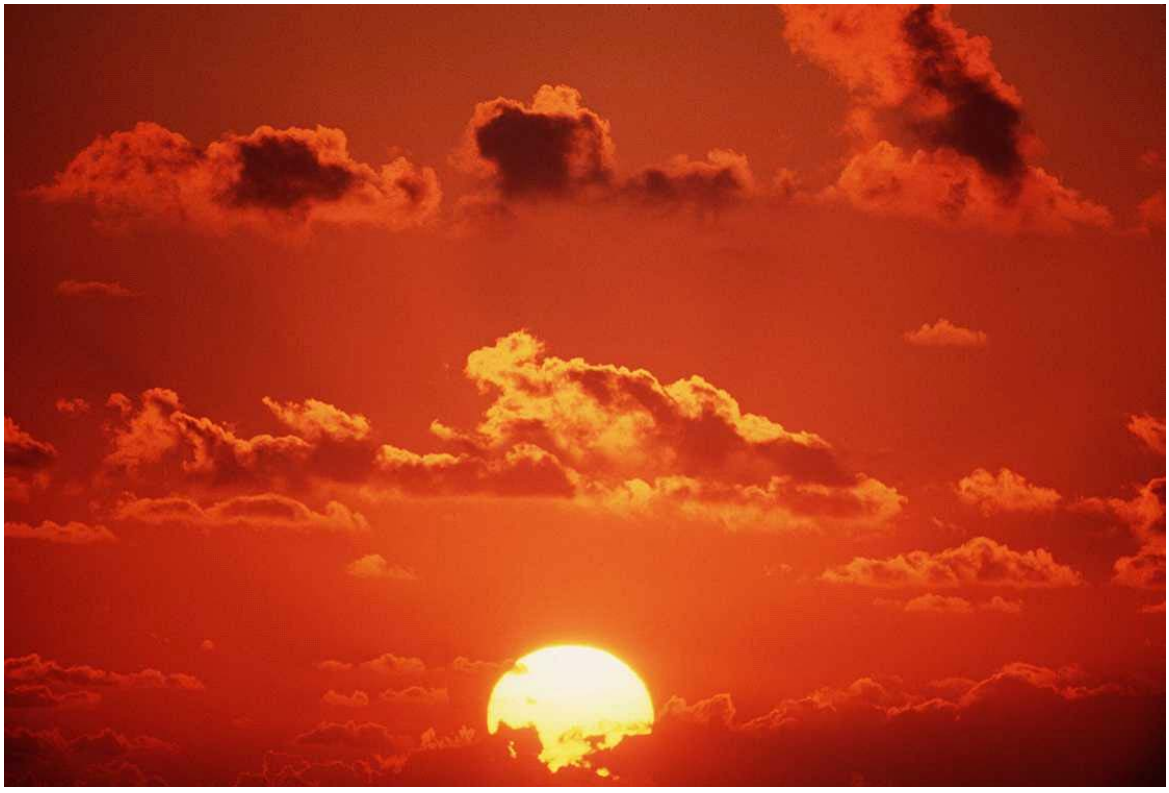
iXSP
Interactive
Extended Set Processor
(XSP)

Programmer's Manual, User's Tutorial and Mathematical Background



Expanding the Horizons of your Databases

Integrated Information Systems

v1.7 Draft

Comments and questions about this manual should be sent to:

iis@umich.edu

Table of Contents

Introduction

This manual has been produced for those interested in an introduction to the Extended Set Processor from IIS and the commands that the processor supports. The manual is broken into three main sections:

- Release Notes
- Installing XSP
- Tutorials
- Command Reference
- Background on the Mathematic Principles of XST

Release Notes

File Names and Paths
For file operations, the default file path is the directory in which the application is running. File names should be in the DOS 8.3 file name format.

Command Names
Commands are not case sensitive in the interactive mode but are case sensitive in batch mode. If you are capturing your interactive sessions to create script files, it is suggested that you type in the command names in upper case.

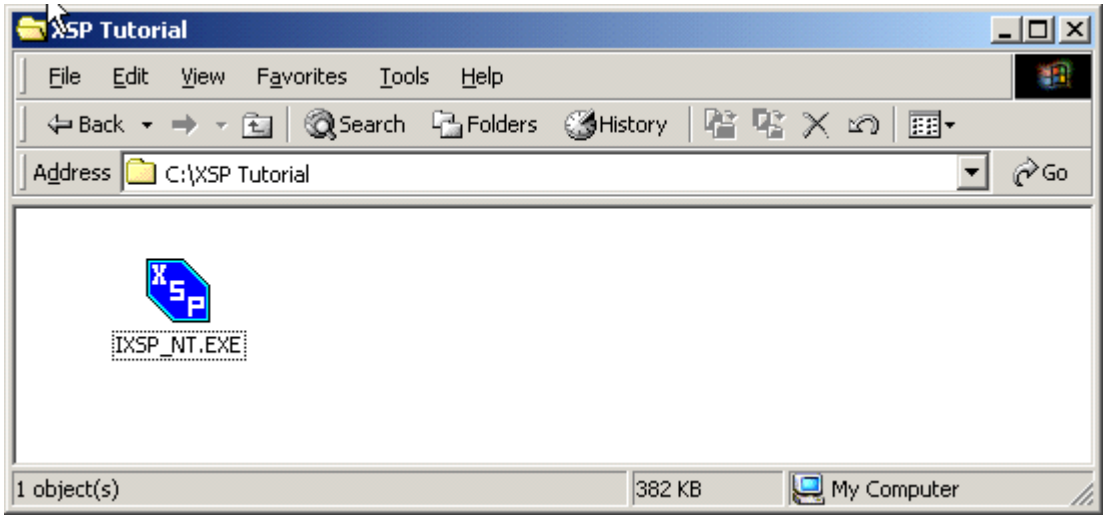
Exiting the Application
The key combination Ctrl C will exit the application.

Command Mode vs. Prompt Mode

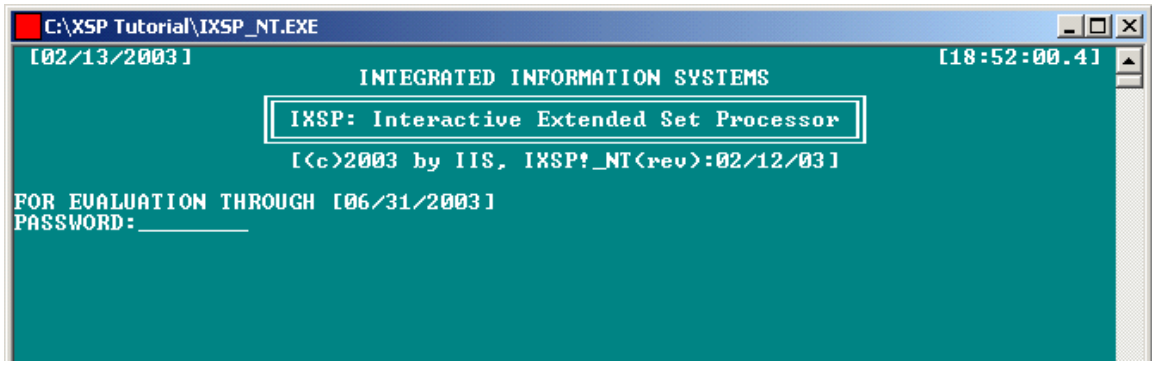
Command Mode prompts you to enter the parameters for a command one parameter at a time. To enter command mode simply press enter at the ">" prompt. To go back to the Command Mode, simply press enter again.

Installing XSP

Step 1: Copy the XSP program to your local hard drive. You should see the icon displayed below:

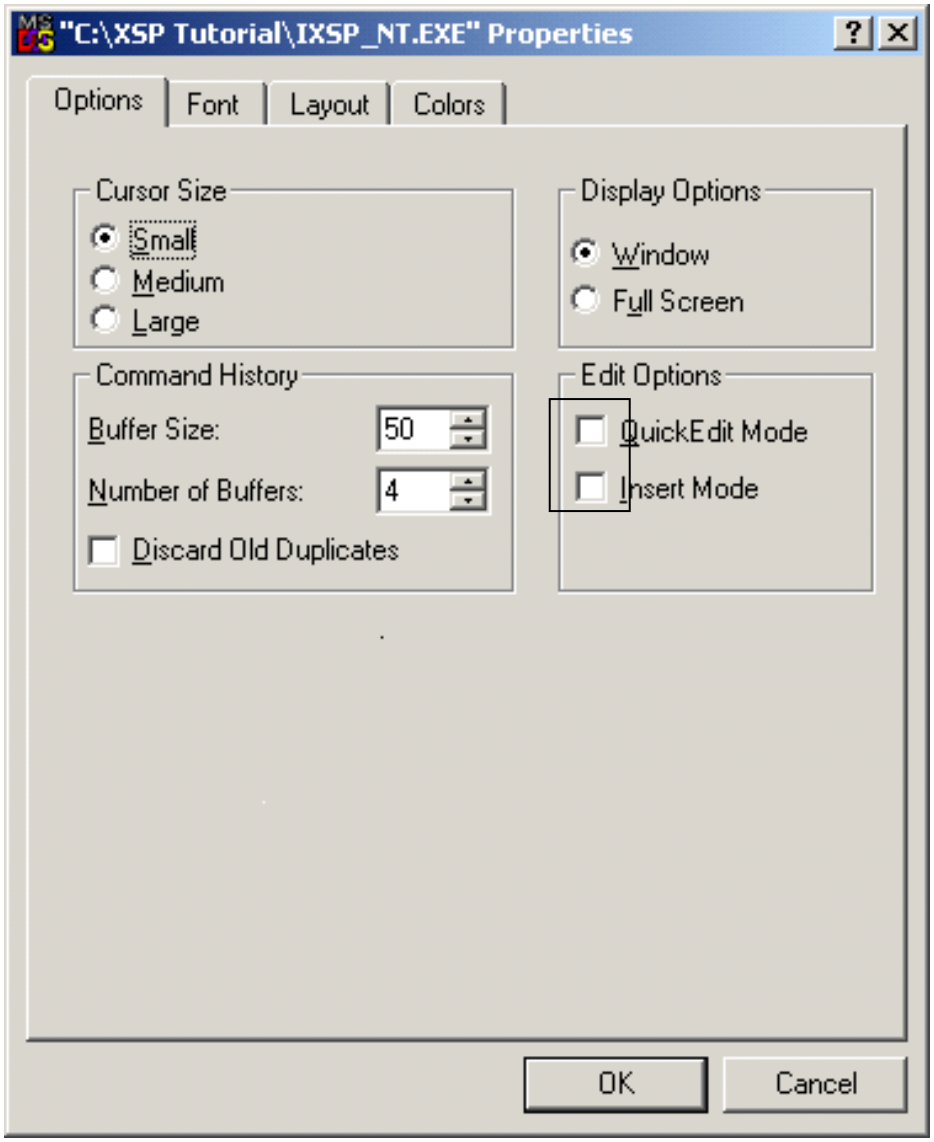


Step 2: Double click the icon to launch the XSP Processor.

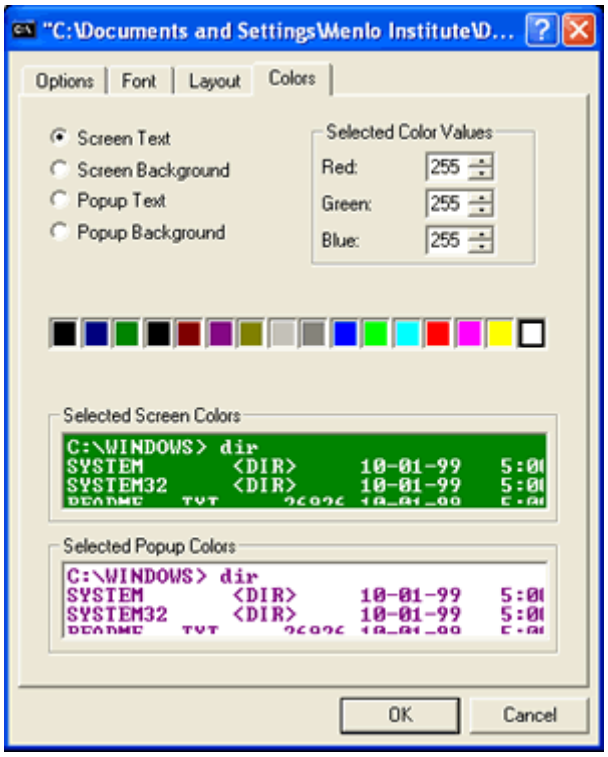


Step 3: Enter the password for the XSP processor. The press [ENTER] after the message "MAXIMUM NUMBER OF MEGABYTES"

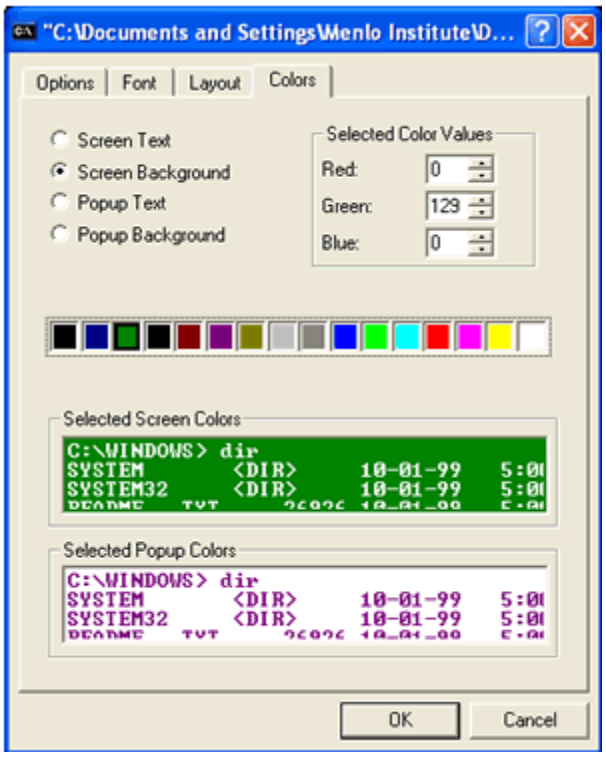
Step 4: Right click on the title bar of the window and select Properties from the pop-up menu. The following dialog is displayed. Deselect “QuickEdit Mode and Insert Mode.



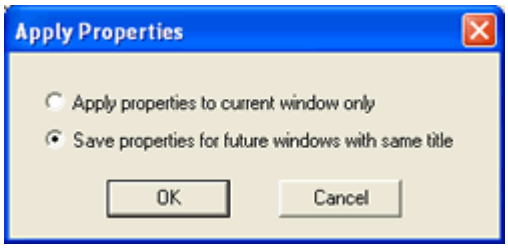
Step 5 (Optional): Select the Colors Tab to select the text color. We selected white text.



Step 6 (Optional): Select the Colors Tab to select the background color. We selected the green.



Step 7: After you have completed your changes select OK. In the dialog displayed you should select the option to "Save properties for future windows with the same title."



You are now ready to use your extended set processor!

Tutorials

Tutorial 1: XSP Introduction

This XSP tutorial teaches an introductory set of XSP commands that will enable you to do a variety of operations using your Extended Set Processor including:

- Creating Sets
- Set Intersection
- Set Union
- Symmetric Difference
- Relative Compliment
- Saving and Loading Extended Sets
- Deleting Sets
- Writing and Running Programs

At the completion of this tutorial you will have a fundamental understanding of how to execute XSP commands and even how to write and execute an XSP program. The commands covered in this tutorial include:

- COMMANDS
- INDEX
- LIST
- IN
- UN
- SD
- RL
- LISTU
- FREE
- PUT
- GET
- QSFILE
- SAVE
- DONE

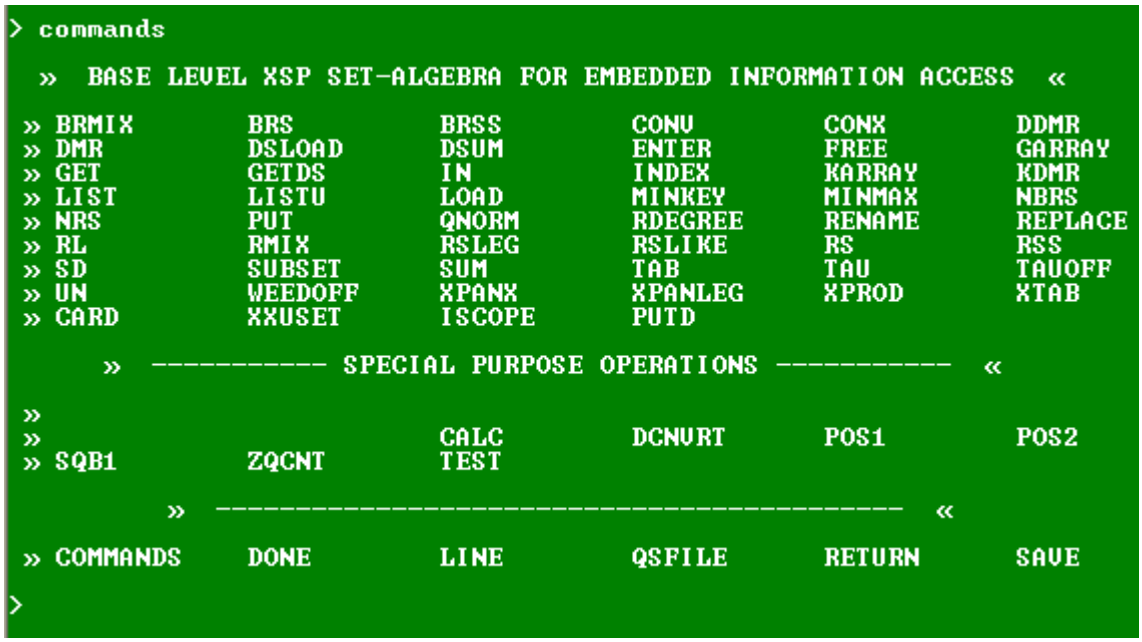
This first tutorial's purpose is simply to give the user some comfort level in understanding the commands of the set processor.

The Supported XSP Command Set

To learn the commands supported by your version of the XSP type the word “commands” after the “>” prompt and press the ENTER key. All values typed into the extended set processor are noted in blue.

>COMMANDS [ENTER]

This causes the XSP processor to list the supported commands. Your command list may look something like the one below:

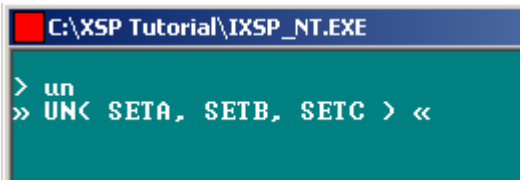


Learning a Command's Parameters

To learn the parameter required by a command simply enter the command without parameters. For example to learn the parameters of the union command "UN" type:

>UN [ENTER]

The parameters for the intersection command are displayed as follows:

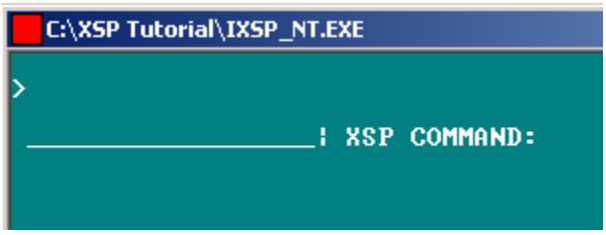


Entering Command Mode

Command Mode prompts you to enter the parameters for a Command one parameter at a time. To enter command mode simply press enter at the ">" prompt.

> [ENTER]

This causes the XSP process to enter Command Mode displaying the following:



In command mode the parameters for the intersection are displayed one parameter at a time. For example the intersection command will cause the following prompts to be displayed.

```
>
 _____| XSP COMMAND: IN[ENTER]
                                »»»»  IN( A, B, C )
            Input Set [A]:  X  [ENTER]
            Input Set [B]:  Y  [ENTER]
           Output Set [C]:   Z  [ENTER]
```

Command Mode can be helpful as you learn the fundamental commands in the extended set processor.

Seeing the List of Extended Sets

To see the lists of extended sets current in the extended set processor enter the following:

>LISTU [ENTER]

A result close to the following should be displayed:

```
> LISTU
   SetName   M-Key  Deg°     Card       DataSetSize    DataSetName
 ------------ ----- ----- ----------- -------------- --------------------
A                 4    12           2             88 ▮2548787.-XΓ
[Total (was/is)]:       80912       80912
DISK SPACE REMAINING =  2147483647 bytes.
```

Displayed is the set name A created in the previous step. This is currently the only set in the extended set processor. Additional properties displayed about the set include:

- M-Key – M-Key.
- Deg° – Degree.
- Card – The cardinality of the data set.
- DataSetSize – The size in bytes of the data set.
- DataSetName – The filename used to store the set data.

Some of these properties may be different on your system but the SetName, Deg and Card should be the same as those displayed above.

Displaying the Data in an Extended Set

To display the data entered into the extended set A enter the following:

>LIST(A,1,99,4) [ENTER]

Displayed is the data in records 1 to 99 of set A.

```
¦      1,      2,      3,
¦      4,      4,      4,
```

>LIST [ENTER]

A result close to the following should be displayed:

```
> list
» LIST< SETA, F#, L#, F#!<Q> > «
```

The parameters on the LIST commands (SETA, F#, L#, F#:{Q}) are defined as follows:

- SETA – The set the list command is operating on
- F# – The starting record to list (integer)
- L# – The stopping record to list (integer)
- F:{Q} – A format set
  - 0 = Bytes
  - 1 = Characters
  - 4 = 4 Byte Integers

To see the data displayed using a different format change the Format Set in the last parameter as shown below.

Press [ENTER] to get to the “>”

>LIST(A,1,99,0) [ENTER]

```
¦ 00000001, 00000002, 00000003,
¦ 00000004, 00000004, 00000004,
```

Creating a Second Extended Set

Use the Index command to create a second extended set B with data {A,4,5}.

To create the set B with values {1,2,3} enter the following:

>INDEX(B,3) [ENTER]

You will be prompted to enter the data into set B immediately after the command.

>>1,2,3 [ENTER]
>>4,5,6 [ENTER]
>>7,8,9 [ENTER]

Note that the data elements are separated by commas and the records by [ENTER].

>END [ENTER]

Enter “end” to stop entering records into the extended set. To see the data entered into Set B use the list command.

>> LIST(B,1,99,4) [ENTER]

Displayed is the data in records 1 to 99 of set B.

```
¦      1,      2,      3,
¦      4,      5,      6
¦      7,      8,      9

>
```

Finding the Intersection of Set A and Set B

We will use command mode to find the intersection of set A and B. To enter command mode press enter at the ">" prompt.

> [ENTER]

This causes the XSP process to enter Command Mode displaying the following:

>
_____| XSP COMMAND:

In command mode the parameters for the intersection are displayed one parameter at a time. To enter the parameters for Intersection one at a time enter IN at the command prompt..

>
_____| XSP COMMAND: IN[ENTER]
»»»» IN(A, B, C)

Next enter the three sets one a time. The two input sets are A and B and the output set that will be created is set Z.

Input Set [A]: A [ENTER]
Input Set [B]: B [ENTER]
Output Set [C]: Z [ENTER]

Since there is a valid intersection then set Z is created and something like the following is displayed.

SetName	M-Key	Deg°	Card	DataSetSize	DataSetName
Z	12	12	3	100	¦1932589.-XΓ

Enter "END" to stop entering records into the extended set. To see the data entered into Set B use the list command.

Press [ENTER] to get to the ">"

>> LIST(Z,1,99,4)

Displayed is the first record in set Z (the intersection of the two sets A and B).

¦ 1, 2, 3,

Finding the Union of Set A and Set B

To find the union of set A and B first find the parameters for the union command UN. Enter UN at the ">" prompt.

> UN [ENTER]

The parameters for the union command are displayed as follows:

» UN(SETA, SETB, SETC) «

We don't need to enter Command Mode to execute commands we can simply enter the command with all of the parameters. To do the union of set A and set B and place the results in set X enter the following command.

Press [ENTER] to get to the ">"

>UN(A,B,X) [ENTER]

Since there is a valid union the set X is created and something like the following is displayed.

SetName	M-Key	Deg°	Card	DataSetSize	DataSetName
X	8	12	4	112	¦1935054.-XΓ

> LIST(X,1,99,4) [ENTER]

Displayed are the records in set X (the union of the two sets A and B).

1,	2,	3,
4,	4,	4,
4,	5,	6,
7,	8,	9,

Finding the Symmetric Difference between Set A and Set B

To find the symmetric difference between set A and B first find the parameters for the symmetric difference command SD. Enter SD at the ">" prompt.

> SD [ENTER]

The parameters for the symmetric difference command are displayed as follows:

» SD(SETA, SETB, SETC) «

You may have begun to notice a pattern in how the parameters are specified in the extended set processor. To do the symmetric difference between set A and set B and place the results in set W enter the following command.

Press [ENTER] to get to the ">"

>SD(A,B,W) [ENTER]

Since there is a valid set created by the symmetric difference the set X is created and something like the following is displayed.

SetName	M-Key	Deg°	Card	DataSetSize	DataSetName
W	8	12	3	100	¦1935055.-XΓ

> LIST(W,1,99,4) [ENTER]

Displayed are the records in set W (the symmetric difference of the two sets A and B).

```
¦      4,      4,      4,
¦      4,      5,      6,
¦      7,      8,      9,
```

Finding the Relative Compliment of Set A and Set B

To find the relative compliment of set A and set B first find the parameters for the relative compliment command RL. Enter RL at the ">" prompt.

> RL [ENTER]

The parameters for the relative compliment command are displayed as follows:

» RL(SETA, SETB, SETC) «

Press [ENTER] to get to the ">"

To find the relative compliment between set A and set B and place the results in set V enter the following command.

>RL(A,B,V) [ENTER]

Since there is a valid set created by the symmetric difference the set X is created and something like the following is displayed.

SetName	M-Key	Deg°	Card	DataSetSize	DataSetName
V	12	12	1	76	¦1935056.-XΓ

> LIST(V,1,99,4) [ENTER]

Displayed are the records in set V (the relative compliment of the two sets A and B).

¦ 4, 4, 4,

Of course, unlike the Intersection, Union and Symmetric Difference the Relative Compliment changes based on the ordering of the parameter sets as you can see if you create the following set Y. This is a result of the definition of Relative Compliment. Relative Compliment is relative to the first set.

>RL(B,A,Y) [ENTER]
>LIST(Y,1,99,4) [ENTER]

¦ 4, 5, 6,
¦ 7, 8, 9,

Displaying a List of All Sets in the Processor

To display all of the extended sets in the processor enter the LISTU command. This lists the universe of sets known by the processor:

>LISTU [ENTER]

Your listing should look something like the following:.

SetName	M-Key	Deg°	Card	DataSetSize	DataSetName
A	4	12	2	88	¦1935051.-XΓ
B	4	12	3	100	¦1935052.-XΓ
V	12	12	1	76	¦1935056.-XΓ
W	8	12	3	100	¦1935055.-XΓ
X	8	12	4	112	¦1935054.-XΓ
Z	12	12	1	76	¦1935053.-XΓ

Deleting a Set in the Processor

To delete a set in the extended set processor enter the FREE command. Enter FREE at the ">" prompt to see the command parameters.

> FREE [ENTER]

The parameters for the FREE command are displayed as follows:

» FREE(S(1),…S(20))

Press [ENTER] to get to the ">"

The parameters are the SetNames you intend to free. Free the set named Y by entering:

>FREE(Y) [ENTER]

If you enter the following command you will notice that the set named Y is no longer listed in the universe.

>LISTU [ENTER]

Your listing should look something like the following:.

SetName	M-Key	Deg°	Card	DataSetSize	DataSetName
A	4	12	2	88	¦1935051.-XΓ
B	4	12	3	100	¦1935052.-XΓ
W	8	12	3	100	¦1935055.-XΓ
X	8	12	4	112	¦1935054.-XΓ
Z	12	12	1	76	¦1935053.-XΓ

Saving a Set for Later Use

Unless you specifically save a set for later use the sets currently listed in the set processor will be deleted when you exit the set processor. To save a set for later use the set processor provides a PUT command. Enter PUT at the ">" prompt to see the command parameters.

> PUT [ENTER]

The parameters for PUT are displayed as follows:

Press [ENTER] to get to the ">"

» PUT(SetA, XSN)

The parameters are the SetNames you intend to save and the path and file name to be saved. Save the set Z to the file "TutSetZ"

>PUT(Z,tutsetz) [ENTER]

You should see a result like the following:.

SetName	M-Key	Deg°	Card	DataSetSize	DataSetName
Z	12	12	1	76	TUTSETZ

Set Z is not longer stored in a temporary file it is stored in a permanent file named TUTSETZ. Since we didn't specify a path the file is located in the same directory as the extended set processor.

If you free set Z, the file will not be deleted since it is now called "TUTSETZ". It is preserved for later use. Use the FREE command to free set Z.

>FREE(Z) [ENTER]

Confirm that the file has not been deleted by using your file system. Use the LISTU command to confirm that the set is no longer loaded in the extended set processor.

>LISTU [ENTER]

You should see that the file has not been deleted.

Getting a Saved Set

To get a specifically saved XSP set back into the set processor use the GET command. Enter GET at the ">" prompt to see the command parameters.

>GET [ENTER]

The following should be displayed:

» GET(SetA, XSN)

The parameters are the Set Name you intend to give the saved set when it is loaded into the set processor and the path and file name to be loaded. Get the set Z back into the set processer from the file "TutSetZ"

>GET(Z,tutsetz) [ENTER]

You should see a result like the following:.

SetName	M-Key	Deg°	Card	DataSetSize	DataSetName
Z	12	12	1	76	TUTSETZ

Set Z is not longer stored in a temporary file it is stored in a permanent file named TUTSETZ. Since we didn't specify a path the file is located in the same directory as the extended set processor. Use the list command to confirm the data has been properly reloaded.

>LIST(Z,1,99,4) [ENTER]

You should see a result like the following:.

```
¦       1,       2,       3,
>
```

Writing a Program

All of the commands can be written in or copied to a file and run as a batch program. Note: Currently the commands in the batch file must be upper case. Using Notepad (or another word processor that will not embed a lot of hidden characters) to type in the following text.

<< Save the text below in a file named TUTPROG1.QSF in the same directory as the set processor. >>

```
! This is a Comment: Comments that begin with ! will not be displayed while this batch file is running.
! Run this batch file with the command QSFILE
* This is also a comment. This comment will print our while the batch file is running

INDEX(M,3)
1,2,3
3,4,5
4,5,6
end

LIST(M,1,99,4)

INDEX(N,3)
7,8,9
10,11,12
13,14,15
end

LIST(N,1,99,4)

LISTU

UN(M,N,O)

LIST(O,1,99,4)

FREE(M,N,O)

<<STOP SAVING>>
```

You have now saved an XSP program that can be run in batch mode in the future.

Running a Program

To run a program that has been previously saved use the QSFILE command. The parameter for the QSFILE command is the name of the batch file to run which in this case is QSFILE(TUTPROG1.QSF)

The commands are run in sequence and the results are displayed on the screen.

>QSFILE(TUTPROG1.QSF) [ENTER]

Congratulations, you have written and executed a basic XSP program. You should see a result displayed like the following:

_____| ENTERING QSFILE INTERPRETER * DSN = TUTPROG1.QSF |_____

* This is also a comment. This comment will print our while the batch file is run

>INDEX(M,3)

```
»        1        2        3
»        3        4        5
»        4        5        6
» END!
```

```
SetName   M-Key  Deg°     Card      DataSetSize    DataSetName
---------- ----- ----- ---------- -------------- --------------------
 M             4    12          3          100 ¦1951199.-XΓ
```

>LIST(M,1,99,4)

```
¦       1,       2,       3,
¦       3,       4,       5,
¦       4,       5,       6,
```

>INDEX(N,3)

```
»        7,       8,        9
»       10,      11,       12
»       13,      14,       15
» END!
```

```
 SetName   M-Key  Deg°    Card   DataSetSize  DataSetName
 ---------- -----  ------ ---------- -------------  --------------------
 N             4         12          3          100        ¦1951200.-XΓ
```

>LIST(N,1,99,4)

```
¦       7,       8,       9,
¦      10,      11,      12,
¦      13,      14,      15,

>LISTU

 SetName   M-Key  Deg°     Card      DataSetSize    DataSetName
 ------------ ----- ----- ---------- -------------- --------------------
 M             4    12         3          100 ¦1951199.-XΓ
 N             4    12         3          100 ¦1951200.-XΓ
 [Total (was/is)]:      80912      80912
 DISK SPACE REMAINING =  2147483647 bytes.

>UN(M,N,O)

 SetName   M-Key  Deg°  Card   DataSetSize    DataSetName
 ------------ -----      ----- ---------- --------------   --------------------
 O             4        12         6          136      ¦1951201.-XΓ

>LIST(O,1,99,4)


¦       1,       2,       3,
¦       3,       4,       5,
¦       4,       5,       6,
¦       7,       8,       9,
¦      10,      11,      12,
¦      13,      14,      15,

>FREE(M,N,O)

 [QSFILE Elapse Time :    0.07 sec.,   0.00 min.]

 _______| EXITING QSFILE INTERPRETER * DSN = TUTPROG1.QSF       |______
```

Exiting the Set Processor

To exit the set processor you may enter the SAVE or the DONE commands.

> SAVE [ENTER]

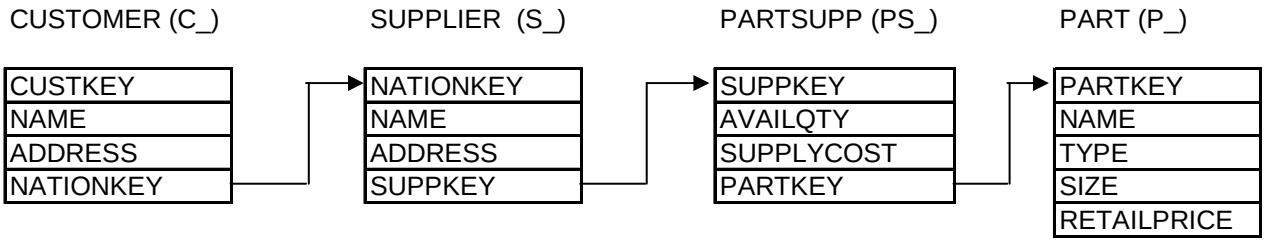
Exits the set processor but does not delete all of the temporary files. The temporary files are **not** automatically associated with a set when the set processor is restarted.

> DONE [ENTER]

Exits the set process and deletes all of the temporary files. The temporary file data and solution sets created in this session will be lost unless they have been specifically saved using the PUT command.

Tutorial 2: Loading Data to be Analyzed

Before data can be loaded into the set processor it is necessary to understand its various relationships. The following is the schema of four files that will be loaded into the set processor. To obtain a copy of the files, contact IIS.



The table layouts for the four files are stated below. Note: See the XSP Datatype Conversion Key in the appendix for information on how the XSP Datatypes were derived.

customer.tbl

Column Name	**Datatype Requirements**	**XSP Datatype**	**Comments**
C_CUSTKEY	variable text, size 4	304	
C_NAME	variable text, size 12	312	
C_ADDRESS	variable text, size 16	316	
C_NATIONKEY	variable text, size 12	312	Foreign key to S_NATIONKEY

supplier.tbl

Column Name	**Datatype Requirements**	**XSP Datatype**	**Comment**
S_ NATIONKEY	identifier	308	Foreign key to C_NATIONKEY
S_ NAME	variable text, size 25	308	
S_ ADDRESS	variable text, size 40	320	
S_ SUPPKEY	identifier	304	Foreign key to PS_SUPPKEY

partsupp.tbl

Column Name	Datatype Requirements	XSP Datatype	Comment
PS_ SUPPKEY	identifier	304	Foreign key to S_SUPPKEY
PS_AVAILQTY	integer	304	
PS_SUPPLYCOST	decimal	204	
PS_PARTKEY	identifier	308	Foreign key to P_PARTKEY

part.tbl

Column Name	Datatype Requirements	XSP Datatype	Comment
P_PARTKEY	identifier	308	Foreign key to PS_PARTKEY
P_NAME	variable text, size 55	316	
P_TYPE	variable text, size 25	304	
P_SIZE	integer	304	
P_RETAILPRICE	decimal	204	

The header information for these four file are stored in the following files:

- customer.tbh
- supplier.tbh
- partsupp.tbh
- part.tbh

Place the following eight files in the same directory as the set processor. Only the tbl files will be used for Tutorial 2. The tbh files will be referenced in a future tutorial.

customer.tbl	customer.tbh
supplier.tbl	supplier.tbh
partsupp.tbl	partsupp.tbh
part.tbl	part.tbh

Before the data can be loaded one must first complete a "Scope Setup" for each file that defines the number of columns of the *scope* and then the nature of the values in each column of the table. Enter the following information in blue into the set processor. Note that the values for the second column of the table come from the XSP Datatype column from the table layout above.

To create a scope set for the customer file, enter the following to the set processor:

>ISCOPE(Scope_C, 2) [ENTER]
<1, 304 [ENTER]
<2, 312 [ENTER]
<3, 316 [ENTER]
<4, 312 [ENTER]
< END! [ENTER]

The following is similar to what should be displayed:

```
> ISCOPE(Scope_C,2)
<1, 304)
<2, 312)
<3, 316)
<4, 312)
<end
 END!
    SetName    M-Key  Deg°      Card       DataSetSize     DataSetName
 ------------ ----- ----- ---------- -------------- --------------------
 SCOPE_C            4     8          4             96 ▮1753032.-XΓ
 [Total (was/is)]:       16392       80912
```

The next step is to create a dataset load for the customer file:

>DSLOAD(9, Scope_C, 0, Set_C, 0, 0, 0, CUSTOMER.TBL) [ENTER]

The following is similar to what should be displayed:

```
> DSLOAD(9,Scope_C,0,Set_C,0,0,0,Customer.tbl)
[DSN]Customer.tbl
[File Size]           0         160
[BUFFSZ]          160
[KC_out, KD_out =           5           0]
    SetName    M-Key  Deg°      Card       DataSetSize     DataSetName
 ------------ ----- ----- ---------- -------------- --------------------
 SET_C              1    44          5            284 ▮1753033.-XΓ
```

Now remove the data from the scope that was just created by entering the following.

>FREE(Scope_C) [ENTER]

Now create a scope set for the part file:

>ISCOPE(Scope_P, 2) [ENTER]
<1, 308 [ENTER]
<2, 316 [ENTER]
<3, 304 [ENTER]
<4, 304 [ENTER]
<5, 204 [ENTER]
END [ENTER]

The following is similar to what should be displayed:

```
> ISCOPE(Scope_P,2)
<1, 308)
<2, 316)
<3, 304)
<4, 304)
<5, 204)
<END
 END!
   SetName   M-Key  Deg°     Card       DataSetSize    DataSetName
 ------------ ----- ----- ---------- -------------- --------------------
 SCOPE_P          4     8          5            104 ▮1753035.-XΓ
```

The next step is to create a dataset load for the part file:

>DSLOAD(9, Scope_P, 0, Set_P, 0, 0, 0, PART.TBL) [ENTER]

The following is similar to what should be displayed:

```
> DSLOAD(9,Scope_P,0,Set_P,0,0,0,PART.TBL)
[DSN]PART.TBL
[File Size]          0         123
[BUFFSZ]         123
[KC_out, KD_out =          4           0]
   SetName   M-Key  Deg°     Card       DataSetSize    DataSetName
 ------------ ----- ----- ---------- -------------- --------------------
 SET_P            6    36          4            208 ▮1753036.-XΓ
```

Now remove the data from the scope that was just created by entering the following.

>FREE(Scope_P) [ENTER]

Now create a scope set for the partsupp file:

>ISCOPE(Scope_Q, 2) [ENTER]
<1, 304 [ENTER]
<2, 304 [ENTER]
<3, 204 [ENTER]
<4, 308 [ENTER]
<END! [ENTER]

The following is similar to what should be displayed:

```
> ISCOPE(Scope_Q,2)
<1, 304)
<2, 304)
<3, 204)
<4, 308)
<END
 END!
   SetName   M-Key  Deg°     Card       DataSetSize    DataSetName
 ------------ ----- ----- ---------- -------------- --------------------
 SCOPE_Q          4     8          4             96 ▮1832948.-XΓ
```

The next step is to create a dataset load for the partsupp file:

>DSLOAD(9, Scope_Q, 0, Set_Q, 0, 0, 0, PARTSUPP.TBL) [ENTER]

The following is similar to what should be displayed:

```
> DSLOAD(9,Scope_Q,0,Set_Q,0,0,0,PARTSUPP.TBL)
[DSN]PARTSUPP.TBL
[File Size]          0        1365
[BUFFSZ]        1365
[KC_out, KD_out =          69           0]
   SetName   M-Key  Deg°     Card       DataSetSize    DataSetName
 ___________ _____ _____ __________ _______________ ____________________
 SET_Q          14    20          69            1444 ▮1832949.-XΓ
```

Now remove the data from the scope that was just created.

>FREE(Scope_Q) [ENTER]

Now create a scope set for the supplier file:

>ISCOPE(Scope_S, 2) [ENTER]
<1, 308 [ENTER]
<2, 308 [ENTER]
<3, 320 [ENTER]
<4, 304 [ENTER]
< END![ENTER]

The following is similar to what should be displayed:

```
> ISCOPE(Scope_S,2)
<1, 308)
<2, 308)
<3, 320)
<4, 304)
<END
 END!
   SetName   M-Key  Deg°     Card       DataSetSize    DataSetName
 ___________ _____ _____ __________ _______________ ____________________
 SCOPE_S          4     8          4              96 ▮1832950.-XΓ
```

The next step is to create a dataset load for the supplier file:

>DSLOAD(9, Scope_S, 0, Set_S, 0, 0, 0, SUPPLIER.TBL) [ENTER]

The following is similar to what should be displayed:

```
> DSLOAD(9,Scope_S,0,Set_S,0,0,0,Supplier.tbl)
[DSN]Supplier.tbl
[File Size]          0         755
[BUFFSZ]         755
[KC_out, KD_out =          23           0]
    SetName    M-Key  Deg°      Card       DataSetSize     DataSetName
 ------------ ----- ----- ---------- -------------- --------------------
 SET_S            9    40          23            984 ▮1832951.-XΓ
```

Now remove the data from the scope that was just created:

>FREE(Scope_S) [ENTER]

Tutorial 3: The Join Commands: Left, Right, Inner and Outer Joins

Coming Soon

Tutorial 4: Scope Sets and Mix Commands

Coming Soon

Tutorial 5: Reading Non-XSP Data Files

Coming Soon

XSP Command Set Reference

COMMANDS

Display all commands.

CONX

Left, Right, Inner and Outer Join.
CONX (I, SETA, [0|1|2a], SETB, [0|1|2a], [SETX], SETC)

DMR

Domain Restriction
Restrict the domain of the set following the rules specified.

DMR (I, ILEN, SETA, SETC)
I: Start byte to include.
ILEN: Number of Bytes to take.
SETA: Set to restrict.
SETC: Result restricted set.

DONE

Quits the applications and deletes the temporary files.

DSLOAD

DSLOAD(V, SETQ, [SETX], SETC, [[SETY], SETD], [KEY], DSN)
Data set load.
V = a delimiter in ascii format. For example, a “9” would represent a “stroke”or “|”
SETQ = Set name
SETX = Set name
SETC = Set name
SETY = Set name
SETD = Set name
KEY = Key
DSN = A dataset name or file that contains the data.

FREE

Free(S(1),..,S(20))

Examples
Free(*)
Delete all of the sets.

Free (A, B)
Delete set "A" and set "B".

GET

Get (SETA, DSN)
SETA: Name we give the set.
DSN: “File Name” to be associated with the extended set name.

IN

Provides the intersection of two sets.
IN(SETA, SETB, SETC)
Place the intersection of SETA and SETB into SETC.

INDEX

Build an index file.
Index(SETX, NumI)
SETX: Set name
NumI: Number of elements in a record

Example
index (x,3)
» 1, 2, 1
» 2, 1, 1
» end <-- Stop adding records to the set.

LIST

List the contents of a set.
LIST (SETA, F#, L#, F#:{Q})
List the contents specified of SETA.
F#: The starting record to list (integer)
L#: The stopping record to list (integer)
F#:{Q}: A format set
0 = Bytes
1 = Characters
4 = 4 Byte Integers

LISTU

List the universe of sets.

PUT

Write a set out to a file
PUT(SETA, XSN)
SETA: Set to write
XSN: File name to write

QNORM

Determines the nature of a dataset

The syntax can be expressed in three ways:

QNORM(Ic, , SETQ, DSN)
QNORM(SETQ, Ic, , SETC, DSN)
QNORM(SETQ, Ic, SETX, SETC, DSN)

QSFILE

QS File(ScriptFileName)
Runs the script in the file specified.

RL

Provides the relative complement of the two sets.
RL(SETA, SETB, SETC)
Makes SETC equal to the relative complement of SETA and SETB.

RMIX

Record Mix
RMIX(I, SETA, [SETB:V], SETX, SETC)
I: Offset into record
SETA: Set to mix.
[SETB:V]: Restriction set.
SETX: Mix Set.
SETC: Result Set.

RSLIKE

RSLIKE(I, J, SETA, SETB:V, [SETX], SETC, SETD)
I: Start byte
J: Stop byte
SETA:
SETB:V:
SETC:
[SETX]: Mix Set

SAVE

Quits the application and doesn't delete the temporary files.

SD

Provides the symmetric difference of SETA and SETB into SETC.
SD(SETA, SETB, SETC)
Symmetric Difference
Makes SETC equal to the symmetric difference of SETA and SETB
All of the items not in the intersection of SETA and SETB.

SUBSET

Take a subset of a set.
SUBSET(SETA, First, Inc, Last, SETC)
SETA: Source Set to make into a subset.
First: Start
Inc: Step by
Last: Inclusive
SETC: Result Set.

UN

Provides the union of two sets.
UN(SETA, SETB, SETC)
Makes SETC equal to the union of SETA and SETB

XPANX

Join two sets.
XPANX (I, SETA, SETB, [SETX], SETC)

XSP Datatype Conversion key

102 Integer(2)	2-Bytes
104 Integer(4)	4-Bytes
204 Real(4)	4-Bytes
208 Real(8)	8-Bytes
3XX Char(XX)	XX-Bytes

Example
For a character field with a length of 24 would be converted as “324”

Background on the Mathematic Principles of XST

XST Set Notation

$Q = \{x^y: P(x,y)\}$: Q is a set containing all elements x with scope y such that the predicate conditional P(x,y) is true.

$x \varepsilon_y Q$: x is a y-element of Q.
$x -\varepsilon_y Q$: x is NOT a y-element of Q.

$x \varepsilon Q$: There exists a y such that x is a y-element of Q.
$x -\varepsilon Q$: There exists NO y such that x is a y-element of Q.

Examples

NESTED SET: $Q = \{A^a, \{x^s, b^t\}^L, z^{\{a^b\}}\}$

CST SET: { a, b, c } = $\{a^{NULL}, b^{NULL}, c^{NULL}\}$

TUPLE: <a,b,c> = $\{a^1, b^2, c^3\}$

LABELED TUPLE: $<a^A,b^B,c^C> = <a,b,c>_{<a,b,c>}$

Note: "Tuples" are a notational convenience for sets with certain properties.
< a[A], b[B], c[C] > is identical to < {a[A]}, {b[B]}, {c[C]} > =
$< \{a^A\}, \{b^B\}, \{c^C\}>$

Definitions

SUB(A,B): TRUE, iff A is a subset of B.
NESUB(A,B): TRUE, iff A is a non-empty subset of B.

Q = NNUL: TRUE, iff Q is non-empty.
Q = NULL: TRUE, iff Q is empty.

(Ex,y,z)... "There exists" an "x" and a "y" and a "z" such that ...

(Vx,y,z)... "For all" "x" and "Y" and "z" ...

Basic Scope Definitions

SCOPE SET: $S(A) = \{y^y: (\exists x)\ (x \in_y A)\ \}$

ELEMENT SET: $\mathcal{E}(A) = \{x^X: (\exists y)\ (x \in_y A)\ \}$

NESTED SCOPE SET: $Sc(A) = \{\ y^y: (\exists z,s)(\ z \in_s A\ \&\ y \in_y S(z)\ )\ \}$

MEMBERSHIP INVERSION: $\hat{A} = \{y^x: x \in_y A\ \}$

SCOPE RESTRICTION: $A^{(\sigma)} = \{\ x^s: x \in_s A\ \&\ \nu \in_s (\sigma\}$

SCOPE TRANSFORMATION: $A^{\prec\sigma\succ} = \{\ x^v: (\exists s)(\ x \in_s A\ \&\ \nu \in_s \sigma)\ \}$

Basic Set Operations

UNION:
$C = A.\cup.B = \{\ x^y: (x \in_y A \text{ or } x \in_y B\ )\ \}$

INTERSECTION:
$C = A.\cap B = \{\ x^y: (x \in_y A \text{ and } x \in_y B\ )\ \}$

RELATIVE COMPLIMENT:
$C = A.\sim.B = \{\ x^y: (x \in_y A \text{ and } x \notin_y B\ )\ \}$

SYMMETRIC DIFFERENCE:
$C = A.\Delta.B = \{\ x^y: (x \in_y A \text{ and } x \notin_y B\ )\ \}$ or $\{\ x^y: (x \in_y B \text{and } x \notin_y A\ )\ \}$

DOMAIN EXTRACTION:
$D\sigma(,Q) = \{\ x^S: (\exists z)(\ z \in_S Q)\ \&\ x = z^\sigma \neq \varnothing$

RESCOPE:
$R\ t(Q) = \{\ y^s: (\exists z)(\ z \in_S Q)\ \&\ (\ y = z^{\prec t\succ} \neq \varnothing)\ \}$

SET RESTRICTION:
$Q|_\sigma A = \{z^s: (\exists a)(\ a \in_s A\ \&\ z \in_s Q\ \&\ \varnothing \neq a^{\prec\sigma\succ} \subseteq z)\}$

IMAGE:
$Q^A{}_{<\sigma,\ t>} = \{\ y^s: (\exists a,z)(\ a \in_s A\ \&\ z \in_s Q\ \&\ \varnothing \neq a^{\prec\sigma\succ} \subseteq z \rightarrow y = z^{\prec t\succ} \neq \varnothing\ )\ \}$

Notice: $Q^A{}_{<\sigma,\ t>} = R_t\ (Q|_\sigma A)$

Generic XSP Functions

Coming Soon

Familiar Set Operations

UNION: UN(SET_A, SET_B, SET_C)
where: C = A.∪ B

INTERSECTION: IN(SET_A, SET_B, SET_C)
where: C = A.∩ B

RELATIVE COMPLIMENT: RL(SET_A, SET_B, SET_C)
where: C = A.∼ B

SYMMETRIC DIFFERENCE: SD(SET_A, SET_B, SET_C)
where: C = A ∆ B

Extended Set Operations

For any set A, there exists a set B such that the scopes of A are the elements of B: (VA)(EB)(SS(A)=ES(B)). Note: SS(B) is defined, but not specified.

SCOPE SET OF: GET_SCOPE(SET_B, SET_A) where ES(B)=SS(A).

META Functions

OPEN_UNIV(U)

CLOSE_UNIV(U)

XSP & Data

XSP Functions are independent of, and in no way related to, data models, databases, or data structurings of any kind. However, all representations of data compatible with being represented in a digital computer environment have an inherent mathematical identity that can be expressed in terms of XST membership conditions. Thus "loading" data into an XSP can assume many forms, depending on the representation of the data to be loaded, and the membership condition chosen to reflect its mathematical identity.

All data load functions must provide all of the element and scope information that is required to define set membership for the data to be loaded. A simple generic load function could be of the form:

LOAD_DATA(Set_A, File_Name.---), or of the form:

LOAD_DATA(Scope_Q, Set_A, File_Name.---)
For XML data files:

LOAD_XML(Set_A, File_Name.XML)

For character delimited text files:

LOAD_CDT(c, Scope_Q, Set_A, File_Name.CDT) ["c" is a single character]

For further information about these topics please visit the IIS website at: xsp.xegesis.org